Vision Is All You Need: A Vision-Only Approach to Dynamics Estimation in Autonomous Navigation

Lazar Milikic (353622), Ahmad Jarrar Khan (353435),
Said Gurbuz (369141), Marko Mekjavic (359986)
CS-503 Project Progress Report {1,2}

I. Milestone Progress

Initially, we defined the project's primary goal as developing an autonomous drone system that utilizes vision to accurately estimate environmental dynamics, a critical capability in many real-world scenarios. Inspired by recent studies in autonomous drone racing by Song et al. [1] and Fu et al. [2], we tasked our drone with swiftly navigating through sequential checkpoints while evading projectiles. Contrary to these studies where privileged information aids policy learning as a teacher, our approach—inspired by the RMA framework [3]—directly focuses on predicting privileged information from past observations.

A. Environment Preparation

To establish a baseline for performance with privileged information, as supported by prior research, our initial efforts were directed at configuring environmental parameters like gate positions, projectile firing rates and velocities, and the reward structure. Using the *Gym Pybullet Drones environment* [4], known for its customization flexibility in Python and offering sufficient realism, to decide these parameters, we ensured that our "blind" agent, equipped with privileged information, could navigate and evade projectiles effectively.

1) Gates Generation: During training, three gates are randomly placed at different heights and aligned along the y-axis. The maximum angle between consecutive gates is $60°$ to facilitate manageable turns. The drone's performance is evaluated in three environments, each varying in difficulty.

2) Projectiles: Fired towards the drone with a probability of 0.02 per time step, ensuring the drone's camera can capture them to accommodate the simulation's single-camera limitation. A maximum of two projectiles may be airborne simultaneously to maintain balanced challenge levels.

The projectiles' dynamics utilize privileged information about the drone's current position and velocity. Random arrival times between $0.4s$ and $0.9s$ are assigned to each projectile, introducing unique dynamics that the drone must predict. The projectiles' firing velocities are calculated based on the drone's position and velocity at launch, adjusted for gravitational effects, ensuring realistic behavior.

3) Reward Function: Originally, we proposed a straightforward reward function that incrementally rewarded the drone for closing the distance to gates and provided high rewards for passing through them, while penalties were applied for failures. This approach proved overly challenging, requiring the drone to master multiple complex tasks: self-control, navigating through gates, and dodging projectiles. Thus, we segmented the learning process into three sequential curriculum phases:

1) The first phase focuses on basic stabilization and learning to hover towards gates, where high rewards are given for maintaining stability, and penalties are imposed for low flying or excessive tilting.
2) The second phase emphasizes navigation. Rewards increase as the drone decreases the distance to gates and successfully navigates through them. The reward system gradually shifts to encourage faster navigation.
3) The final phase introduces projectile dodging. Early on, hitting a projectile does not incur a penalty to avoid detrimental "suicide" strategies (to avoid joint penalties for crashing and being hit); instead, we enhance penalties for crashes. Initially, there's one projectile with a small survival reward. As training advances, more projectiles are added, and the main rewards focus on efficient gate navigation.

B. Drone Policy Model

1) Observation Space: The drone's observation comprises various self-information metrics, including velocity, angular velocity, quaternion rotation, and orientation parameters. These metrics allow faster and more stable learning. In the privileged setting, the drone receives information about the relative positions and velocities of gates and projectiles from its camera, allowing for enhanced predictability. Gates and projectiles are always generated within the camera's field of view for visibility. Optionally, the drone's front-facing camera provides a 64×48 RGBA signal, which can be integrated with the other data streams.

2) Action Space: The action space for the drone involves controlling the thrust levels of its four rotors.

3) Policy Network: In this milestone, we developed a policy model for a "blind" agent with privileged information, serving as a baseline for further work. We explored various architectures but found that a simple MLP performed best. The policy is trained using PPO from *Stable Baselines3* [5].

II. Discussion & Next Steps

A. Challenges

The primary objective of our first milestone was to create a functional environment and structure that supports learning with privileged information. This task proved more challenging and time-consuming than anticipated, primarily due to difficulties ensuring the RL agent's learning.

We invested considerable effort in developing the curriculum steps and optimizing the reward structure. Initial attempts to have drones immediately navigate through gates were unsuccessful, highlighting the model's sensitivity to even minor adjustments in reward values or penalties. For example, slightly increasing the penalty for being hit by a projectile led the drone to adopt a counterproductive strategy of deliberately crashing to avoid hits, diverging from the intended behavior of evading projectiles.

This sensitivity may stem from the slow data collection process and limited diversity in environment interactions, as we could only use 16 parallel rollouts, whereas previous studies used up to 100 [1]. Attempts to increase the number of rollouts significantly slowed training, necessitating a compromise on computational resources and parallelization.

In response, we explored the SAC [6] algorithm, an off-policy method that relies less on parallel rollouts and more on the experience replay buffer. While SAC accelerated learning in simpler curriculum phases, it led to unstable training and poorer performance in more complex scenarios. This experience reaffirmed our use of the PPO algorithm, which proved more effective when carefully tuning the reward structure for each curriculum step.

Defining the policy model posed a challenge because of the varying input features based on the number of airborne projectiles. Transformer and GNN architectures showed potential but needed significantly more data to converge efficiently. However, a simpler MLP with zero padding for missing projectiles performed well, converging in about 10 million steps, likely due to its simplicity and smaller size.

Subsequent experiments revealed instances of unstable behavior even when policies converged. Further investigation showed that large positional values of projectiles or gates led to unstable predictions. Implementing feature normalization using tools from *Stable Baselines3* [5] helped stabilize training and enable more consistent inference.

B. Baseline policies

1) Blind Agent Equipped with Privileged Information: In our baseline setup, we trained a blind agent equipped with privileged information, designed to handle up to two projectiles simultaneously. The agent uses an MLP with three hidden layers of widths 128, 64, and 32, and GELU activation [7]. This choice was made despite *Stable Baselines3*'s recommendation for Tanh activation [5], as Tanh led to slower learning and non-convergence of the agent.

We evaluated this policy across 100 different seeds in three environments, each with five gates. On average, the drone under projectile fire successfully navigated through 3.4 out of 5 gates. Limiting the projectiles to one at a time increased the average success rate to 4.1 out of 5 gates, indicating less challenge. However, with three projectiles, the success rate dropped to 2.1 out of 5, proving too difficult. These results led us to conclude that two projectiles provide a realistic and appropriately challenging scenario.

2) RGB-Directly Learning: As an additional baseline, we experimented with training an agent solely on visual signals and self-information about the drone, omitting privileged information. Following approaches from prior studies [2], we employed a Yolo-based model [8] to encode images, which were then combined with the drone's self-information to inform MLP-driven action predictions. Initial training efforts were unsuccessful, likely hindered by insufficient data and suboptimal initialization strategies.

Although we still hope to develop the visual-only baseline for comparison, given the challenges in deploying RL algorithms and setting up environments, our primary focus will shift to privileged-based methods and adapting them for use in non-privileged contexts, as these approaches are more effective for training drone agents in complex scenarios.

C. Next Steps

Now that we have established a functional environment and a successful baseline with the Blind Agent equipped with privileged information, our next objective is to adapt this model for scenarios where privileged information is unavailable. We plan to develop capabilities for the drone to navigate and avoid projectiles using two distinct methods.

1) Predictive Dynamics Module: We intend to train a dynamics module that can predict the privileged information provided to the policy using past observations, which include the RGB signal, actions taken, and drone self-information. We propose using encoder-decoder architectures like TubeR [9] and TransVOD [10], known for their effectiveness in video understanding and object detection, to adapt these technologies to our specific requirements.

2) Implicit Learning via a Student Policy: Alternatively, we will explore an implicit learning strategy where a student policy learns to predict actions from the Blind Agent's policy using recent observations. This approach seeks to reproduce successful strategies from prior studies [2] and assess their effectiveness in complex drone operation scenarios, providing a comparison with our dynamics module.

While we initially proposed additional baselines, such as a vision-based drone learning without privileged information, our focus is currently on refining and successfully implementing these two primary methods. Additional baselines may be revisited later, depending on time availability, to broaden our analysis and enhance the drone's functional robustness.

III. Author Contribution Statement

Ahmad Jarrar worked on creating the custom gym environment for the project. Lazar Milikic worked on creating the reward function, curriculum learning design, and training the baseline model. Lazar and Ahmad created the algorithm for the parabolic projectiles. Said Gurbuz worked on the vision-based drone and models. Marko Mekjavic kept track of the project and prepared the material for report writing. Lazar and Marko wrote the report. Said worked on polishing the final version of the report. Lazar Milikic conducted experiments/tests for blind agents with privileged information, and Said Gurbuz conducted experiments/tests for vision-based agents without privileged information.

References

[1] Y. Song, M. Steinweg, E. Kaufmann, and D. Scaramuzza, "Autonomous drone racing with deep reinforcement learning," 2021.

[2] J. Fu, Y. Song, Y. Wu, F. Yu, and D. Scaramuzza, "Learning deep sensorimotor policies for vision-based autonomous drone racing," 2022.

[3] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "Rma: Rapid motor adaptation for legged robots," 2021.

[4] J. Panerati, H. Zheng, S. Zhou, J. Xu, A. Prorok, and A. P. Schoellig, "Learning to fly – a gym environment with pybullet physics for reinforcement learning of multi-agent quadcopter control," 2021.

[5] A. Hill, A. Raffin, M. Ernestus, A. Gleave, A. Kanervisto, R. Traore, P. Dhariwal, C. Hesse, O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Sidor, and Y. Wu, "Stable baselines," https://github.com/hill-a/stable-baselines, 2018.

[6] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," 2018.

[7] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," 2023.

[8] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," 2016.

[9] J. Zhao, Y. Zhang, X. Li, H. Chen, B. Shuai, M. Xu, C. Liu, K. Kundu, Y. Xiong, D. Modolo *et al.*, "Tuber: Tubelet transformer for video action detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 13 598–13 607.

[10] Q. Zhou, X. Li, L. He, Y. Yang, G. Cheng, Y. Tong, L. Ma, and D. Tao, "Transvod: End-to-end video object detection with spatial-temporal transformers," *arXiv preprint arXiv:2201.05047*, 2022.