A LiDAR-Based Deep Reinforcement Learning Autonomous Navigation System in Unknown Environments

Yuki Onda, Abner Asignation Jr., and Satoshi Suzuki*
Chiba University, Japan

http://www.imavs.org/

Abstract

This study develops an autonomous, UAV navigation system based on an end-to-end deep reinforcement learning algorithm. While conventional autonomous flight systems rely on constructing environmental maps or prior knowledge, the proposed method directly utilizes obstacle range data from a LiDAR sensor, along with the UAV's own position and velocity information, as inputs to the Proximal Policy Optimization (PPO) algorithm. The policy is trained to derive optimal flight paths based on real-time environmental perception, enabling navigation to target locations without building any environmental maps. To evaluate the effectiveness of the proposed method, flight simulations were conducted in a physics-based simulator featuring obstacle environments. Furthermore, indoor real-world experiments confirmed that the trained policy demonstrated high adaptability and robustness in actual environments. This study contributes to the realization of autonomous flight in more realistic and complex environments without the need for environmental maps, laying the groundwork for future practical applications.

1 Introduction

In recent years, Unmanned Aerial Vehicle (UAV) technology has advanced significantly. UAVs offer several advantages over manned aircraft, including smaller size, lower cost, and ease of operation. Moreover, UAVs are particularly well-suited for complex mission environments such as forest search and rescue operations [1] and bridge inspections [2]. However, such environments often present challenges, including scattered obstacles and unreliable communication or satellite signals [3]. In particular, efficient obstacle avoidance and rapid target acquisition are critical in search and rescue missions. Therefore, navigation capabilities such as autonomous navigation and obstacle avoidance are essential for next-generation UAV systems [4, 5].

Conventional non-learning-based methods suffer from limitations such as the need for explicit path planning and inefficient use of computational resources and memory [6, 7]. To address these issues, a previous study proposed a novel learning-based and model-free approach to UAV navigation in unknown and complex environments [8]. This approach utilizes reinforcement learning, which enables the UAV to learn from patterns and make informed decisions based on data [9]. As a result, the UAV can learn optimal control policies from flight data and reach its destination quickly and safely while avoiding various obstacles. However, in that study, both the UAV and obstacle positions were predefined in simulation. The real-world experiments simply replicated the simulated environment by placing obstacles at the same coordinates as in the simulation. Therefore, the experiments were limited in scope and did not demonstrate the ability to generalize to unknown real-world environments.

In this study, we envision future applications in complex environments such as forests, where visibility is poor and the use of GNSS or vision sensors is difficult. To this end, we extend previous models by incorporating a LiDAR sensor and develop an end-to-end navigation system that enables a drone to reach target locations while avoiding unknown obstacles through reinforcement learning and LiDAR. Furthermore, we verify the effectiveness of the proposed system through both simulations and real-world experiments, while evaluating its generalization performance in environments with unseen obstacles

The organization of this paper is as follows. Section 2 describes an overview of the system developed in this study. Section 3 explains various aspects of the reinforcement learning design used in the simulation. Section 4 presents the simulation environment, experimental setup, and discusses the results and their analysis. Finally, Section 5 concludes the paper and outlines future work.

2 Navigation system using deep reinforcement learning

As shown in 1, this study proposes a guidance framework based on direct policy search and optimization through reinforcement learning. The aim of this framework is to enable obstacle-avoidance navigation for UAVs to reach target destinations in unknown environments. In this framework, low-level angular velocity, attitude, and speed are controlled by a well-tuned cascading PID, while high-level path planning and position control are integrated into a control policy.

*Email address(es): suzuki-s@chiba-u.jp

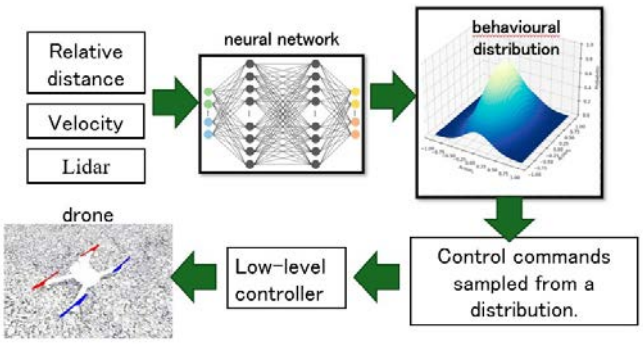


Figure 1: RL controller for learning

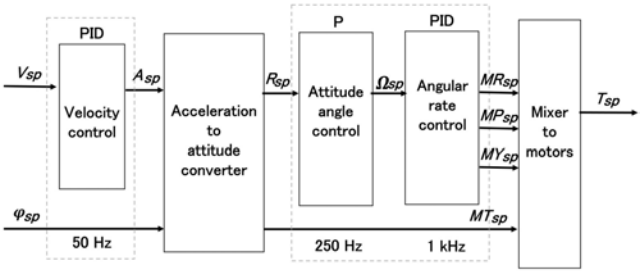


Figure 2: Low level controller

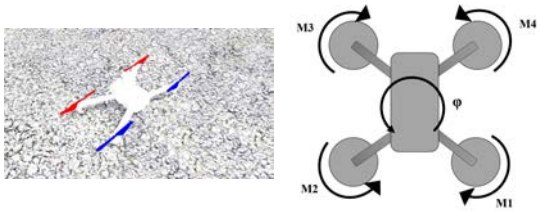


Figure 3: Drone model

This control policy is implemented as a fully connected neural network that takes as input raw sensor measurements from the LiDAR, together with additional state information such as the current UAV velocity, the relative position to the target, and the previous action. The network outputs an action distribution that represents the probability distribution over target speed commands. Specifically, the actor network produces the mean vector of a Gaussian distribution, while the covariance matrix is fixed to a constant value ($0.6I$). Thus, the policy can be expressed as

$$a \sim \pi_\theta(a \mid s) = \mathcal{N}\big(\mu_\theta(s),\, 0.6I\big) \tag{1}$$

This design allows stochastic sampling during training to ensure exploration, although the variance remains constant throughout learning. Finally, the actual action is sampled from this distribution and sent to the low-level controller, enabling both exploration during training and robust execution during deployment.

2.1 Low-Level Controller

In this study, a cascaded control architecture combining P and PID controllers operating at different frequencies is employed as the low-level control system to accurately execute the velocity commands from the high-level RL controller. As shown in Figure 2, the outer loop computes the desired acceleration from the target velocity, which is then converted into the desired attitude angles in the intermediate layer. The inner loop subsequently controls the attitude angles and angular rates. The thrust commands in each direction are finally transformed into motor rotational speeds, enabling stable and responsive flight control. Each control loop operates at a different frequency according to its responsiveness: 50 Hz for velocity control, 250 Hz for attitude control, and 1 kHz for angular rate control.

2.2 High-Level Controller

This study aims to address the problem of high latency in environment recognition and to maximize the agile maneuverability of UAVs through path planning in complex environments. To achieve this, we utilize the Proximal Policy Optimization (PPO) algorithm [10], a type of reinforcement learning method [11], to process environmental and UAV state information collected in parallel from both onboard and external sensors. In the proposed system, the PPO algorithm generates control commands (i.e., velocity commands) based on inputs such as the relative position to the target point and the UAV's velocity, obtained from either the simulator or the motion capture system, as well as obstacle range data from the LiDAR sensor. These commands are then directly supplied to the low-level control system in an end-to-end manner.

3 Simulation

3.1 Problem Setup

The Gazebo simulator, Robot Operating System (ROS), and reinforcement learning algorithm run on a desktop computer running Ubuntu 20.04. The main hardware configuration is an Intel i9-14900K CPU, Nvidia 4080 GPU, and 32GB RAM. In this study, we used the model (Figure 3) from [12], which was designed for simulation in ROS and Gazebo and a 2D LiDAR sensor was used to obtain a point cloud of the forward range except for the backward 90°at a distance of up to 5m from the UAV.

3.2 Hyperparameter

The hyperparameters in Table 1 were used to learn the control policies in all designed learning environments.

3.3 State, Action

The input state to reinforcement learning is $s = [s^{dis}, s^{vel}, s^{lidar}, act_{(t-1)}]$. Here, s^{dis} denotes the relative

http://www.imavs.org/

http://www.imavs.org/

position to the target point, represented by the differences in the x and y coordinates $(\Delta x, \Delta y)$, expressed in the global frame (i.e., world coordinates). This requires global localization, which is provided by the OptiTrack system in real-world experiments and by the simulator during training. s^{vel} is the velocity of the UAV in the x and y directions, and s^{lidar} is the range data from the LiDAR sensor, consisting of 417 points. In addition, act, the output of the network one step before, is added as a state quantity to stabilize the output of the network. Here, the network output, action act, is a velocity command to the low-level controller in the x- and y-directions. In order to ensure stable learning, the state variables S fed into the network are normalized within the range of -1 to 1.

3.4 Neural Network

For the actor network and the critic network, two neural networks with a three-layer network structure with the same number of parameters and structure shown in Figure 4 were prepared. Both neural networks use the LeakyReLU function as the activation function and are connected between the input and hidden layers and between the hidden and output layers.The input layer takes S in Section 3.3 as input in both networks. The difference between the actor network and the critic network is the output layer: the output layer of the actor network has $q = 1$, while the output layer of the critic network has $q = 2$.

3.5 Rewards

The reward function is a measure of the quality of one step of an action, and both the value function and the action value function are built on this basis. Therefore, the design of the reward function must be able to enable the control policy to avoid obstacles and fly to the destination; the PPO algorithm uses a clip function, and the size of one policy update is constrained by the hyperparameters. Therefore, to guarantee stable convergence and avoid local optima, the reward function should be continuous and differentiable. Based on these considerations, we designed a synthetic reward function with

Parameter Name	Value
Control period	0.02s
Max steps	2×10^8
Max step per episode	800
Dimension of states	423
Dimension of actions	2
Timesteps per batch	4056
Discount rate of reward	0.99
Parameter update times per iteration	20
Learning rate	3×10^{-3}
ϵ of clip function	0.2
Covariance matrix element value	0.6

Table 1: Hyperparameter

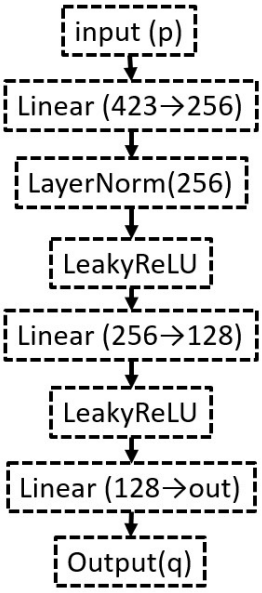


Figure 4: Network structure

the following five components and their weights.

$$R_{\text{total}} = R_{\text{dis}} + R_{\text{v}} + R_{\text{a}} + R_{\text{goal}} + R_{\text{ng}} \tag{2}$$

The first term R_{dis} is defined by Equation 3 based on the distance Dis_t between the UAV and the target location at the current step. This term rewards the UAV based on the amount of change in the distance between the UAV and the destination, and a positive reward is given when the UAV approaches the target.

$$R_{\text{dis}} = \begin{cases} k_{\text{dis}}(\text{Dis}_{t-1} - \text{Dis}_t), & \text{if } \text{Dis}_{t-1} - \text{Dis}_t > 0 \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

R_{v} is a constraint on the horizontal speed, denoted by Equation 4, which provides a negative reward if the horizontal speed of the UAV v_t exceeds $\text{speed}_{\text{max}}$. In this study, we limit it to 3m/sec.

$$R_{\text{v}} = \begin{cases} 0, & \text{if } v_t \leq \text{speed}_{\text{max}} \\ k_v(|v_t - \text{speed}_{\text{max}}|), & \text{if } v_t > \text{speed}_{\text{max}} \end{cases} \tag{4}$$

R_{a} is a penalty based on the UAV's velocity change, as expressed in Equation 5. It is defined as the sum of the absolute differences between the UAV's velocities in the X and Y axes at time t and those at time $t-1$. This term suppresses abrupt velocity changes and promotes smooth and stable motion, while contributing to reduced energy consumption and improved learning stability.

$$R_{\mathrm{a}} = k_a(|v_{x(t)} - v_{x(t-1)}| + |v_{y(t)} - v_{y(t-1)}|) \quad (5)$$

The R_{goal} is represented by Equation 6, which gives an additional positive reward if the target point is reached within a radius of 1 meter from the target point and a negative reward if the target point is moved away from after the target point is reached. This reward formulation is designed to encourage the agent to not only reach the target point but also to remain at the target location.

$$R_{\mathrm{goal}} = \begin{cases} k_{goal} & \text{if reach the destination} \\ k_{leave} & \text{if leave the destination} \end{cases} \quad (6)$$

R_{ng} provides a large negative reward as punishment if the UAV collides with an obstacle or reaches the maximum number of time steps in an episode without reaching the target point, as expressed in Equation 7. The episode is terminated when these conditions are reached in order to increase learning stability and efficiency.

$$R_{\mathrm{ng}} = \begin{cases} k_{obs} & \text{if collide with obstacle} \\ k_{step} & \text{if max step} \end{cases} \quad (7)$$

The weights of each reward function are as shown in Table 2.

4 EXPERIMENT

4.1 Experiment Environment

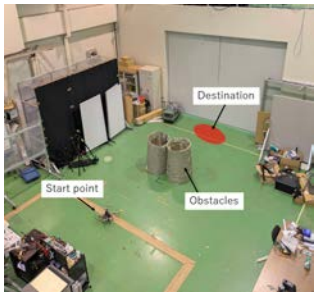
Experiments were conducted in an indoor environment, as shown in Figure 5. The UAV used in the experiment was the one shown in Figure 6. The size of the UAV is about $45\,\mathrm{cm} \times 45\,\mathrm{cm}$ and it is equipped with a LiDAR sensor and an on-board computer. The onboard computer was a Jetson Orin Nano from NVIDIA Corporation. Motion capture by OptiTrack was used instead of GNSS to acquire self-position.

4.2 Learning

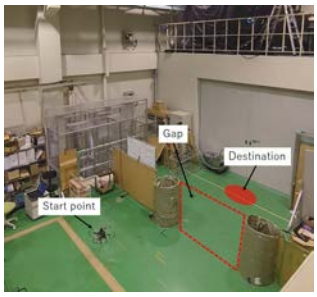
In this study, since the validation experiments using the actual UAV were conducted indoors, we first examined the impact of environmental elements such as walls, excluding obstacles, on the learned policy. For this purpose, we constructed four types of learning environments that simulate indoor conditions. These included enclosed environments with

k_{dis}	2.5
k_v	0.1
k_a	0.01
k_{goal}	10
k_{leave}	-3
k_{obs}	-5
k_{step}	-3

Table 2: Weights of reward function



(a) env1



(b) env2∼env4

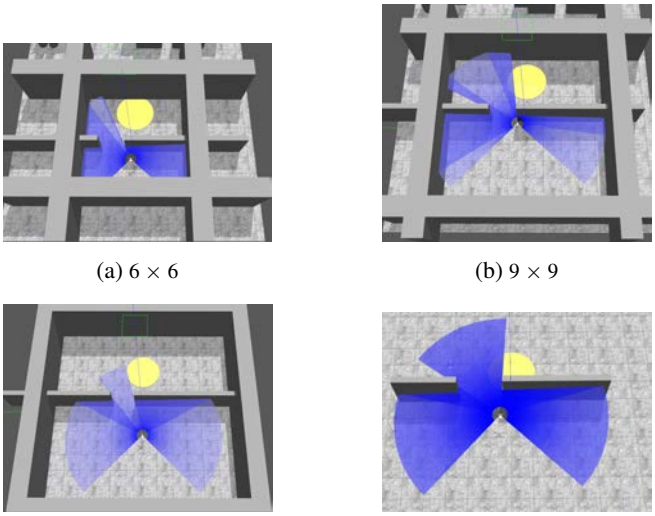
Figure 5: Experiment environment

Figure 6: UAV for experiments

dimensions of 6m × 6m, 9m × 9m, and 12m × 12m, as well as an open environment without any walls, as illustrated in Figure 7. The position of the wall opening was fixed at $x = 1.0$ and $y = 0.5$ to 2.0. The effective utilization rates of the LiDAR sensor in each environment were approximately 100 percent in the 6m × 6m environment, between 70 and 90 percent in the 9m × 9m environment, and around 50 percent in the 12m × 12m environment, as shown in Figure 8. To assess the generalization performance, we conducted 100 trials under each training condition using the random environment shown in Figure 10. The success rates were 65 percent for the policy trained in the 6m × 6m environment, 33 percent for the 9m × 9m environment, and 11 percent for the 12m × 12m environment. In addition, the policy trained in the open environment was able to reach the target when tested under the same open conditions, but failed to do so in any of the indoor environments. These findings indicate that a higher utilization rate of LiDAR data during training is associated with improved generalization performance in unfamiliar environments.

Based on these results, for the verification experiment with the actual device, training was conducted in a Gazebo environment like that shown in Figure 9, where the wall distance, start position, target location, and gap position were

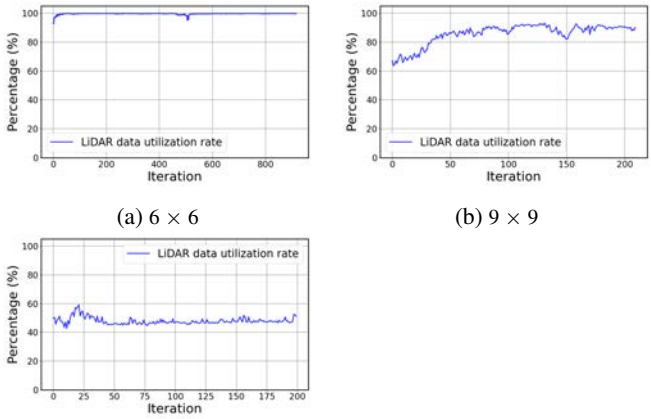
http://www.imavs.org/



(a) 6 × 6 (b) 9 × 9

(c) 12 × 12 (d) open space

Figure 7: Comparison by wall distance



(a) 6 × 6 (b) 9 × 9

(c) 12 × 12

Figure 8: Number of valid data for lidar data in the study

randomly varied as shown in Figure 10.

4.3 Result

To validate the performance of the proposed algorithm in a real-world environment, we designed the tasks of avoiding two cylinders and moving through three different gaps between two walls, as shown in Table 3. The UAV performs the tasks of avoiding obstacles or passing through gaps, reaching the destination, and hovering. The low-level controller runs on Pixhawk hardware, while the high-level control policy runs on a Jetson Orin Nano. The data nodes of the motion capture system run in ROS and communicate via a network interface. The UAV is then controlled to perform its mission in the real environment using the policies learned in the Figure 10 environment. An overview of the system used in this experiment is shown in Figure 11.

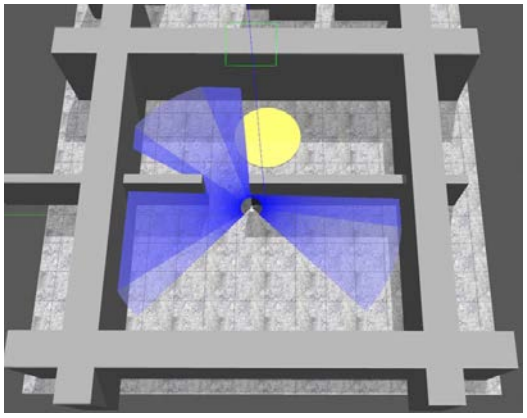


Figure 9: Simulation environment

The results of the position and velocity are shown in Figure 12 and Figure 13, respectively. The actual trajectories of the UAV are shown in Figure 14, and all experiments were conducted using the same policy. Results showed that the UAV was able to avoid obstacles and hover at the target point in all four environments using the same policy. This experiment demonstrates that in indoor environments, our method can control the UAV to complete the task of navigating complex real-world environments while sensing the environment using LiDAR sensors. In addition, by learning in a space surrounded by walls on all four sides with randomized wall distances and obstacle locations, we were able to obtain some generalization performance against obstacles, which was confirmed in an actual UAV. In particular, the experimental results of env1 showed that the robot was able to avoid obstacles with configurations and shapes that were not present in the learning process, confirming its generalization performance that can cope with unknown environments.

5 Conclusion

In this study, we applied deep reinforcement learning with end-to-end mapless navigation using LiDAR sensors in environments with unknown obstacles. While previous studies faced challenges in generalizing to new obstacle layouts,

	start	goal(r=1.0)	obs(r=0.3)	gap
env1	x = -2.0 y = 0	x = 2.0 y = 0	x = 0.25 y = 0, 0.6	\
env2	x = -1.5 y = -1.0	x = 2.5 y = 0	\	x = 0.5 y = -0.5 ∼ 1.0
env3	x = -1.5 y = 1.0	x = 2.5 y = 0	\	x = 0.5 y = -0.5 ∼ 1.0
env4	x = -1.5 y = 1.0	x = 2.0 y = 0	\	x = 0.25 y = -1.5 ∼ 0

Table 3: Experiment environment

http://www.imavs.org/

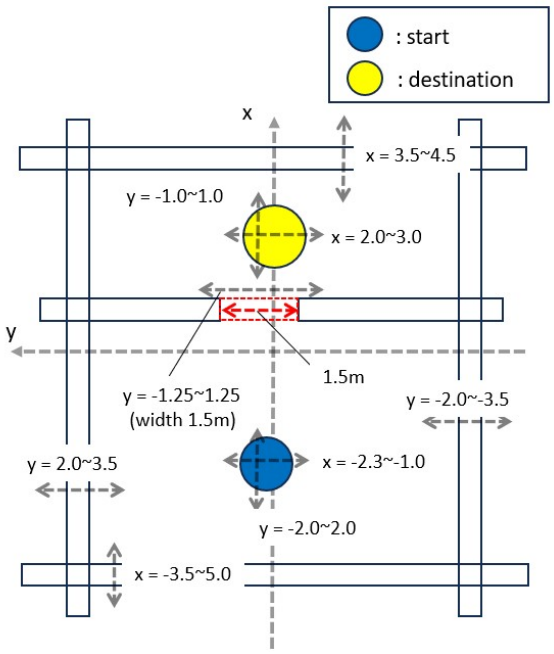


Figure 10: Randomized environment

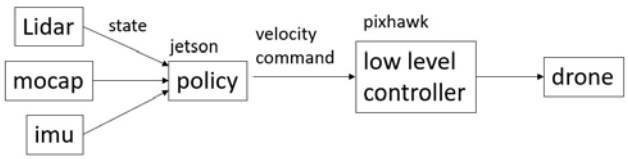
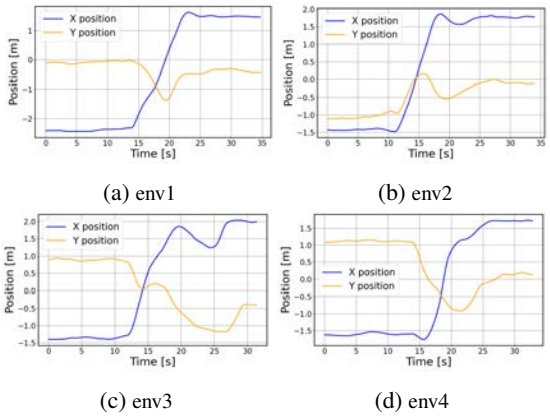


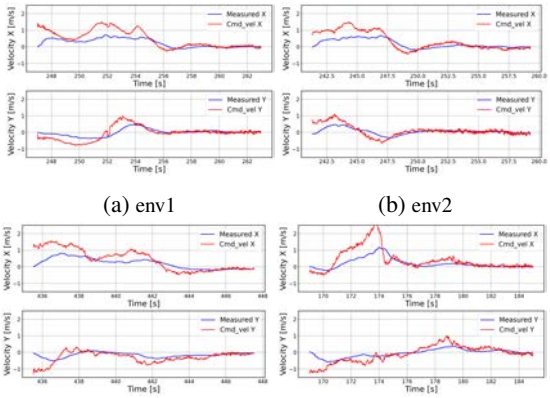
Figure 11: Real drone system



(a) env1 (b) env2

(c) env3 (d) env4

Figure 12: Position results for each environment



(a) env1 (b) env2

(c) env3 (d) env4

Figure 13: Speed results for each environment

our approach successfully achieved navigation adaptable to different obstacle placements, as confirmed through simulations and real-world experiments. We also investigated the impact of the effective data ratio from LiDAR on policy generalization. However, several issues remain. In real environments, elements such as luggage not present in simulations act as noise and degrade performance. Additionally, since LiDAR operates in two dimensions, high-speed navigation limits ground detection, reducing speed. The current experiments were based on simple obstacle layouts, so future work will explore more complex scenarios. Moreover, as the UAV currently relies on external sensors for localization, we plan to implement internal sensor-based estimation for onboard-only operation. Future evaluations will also include outdoor environments such as forests.

REFERENCES

[1] David C. Schedl, Indrajit Kurmi, and Oliver Bimber. An autonomous drone for search and rescue in forests using airborne optical sectioning. CoRR, abs/2105.04328, 2021.

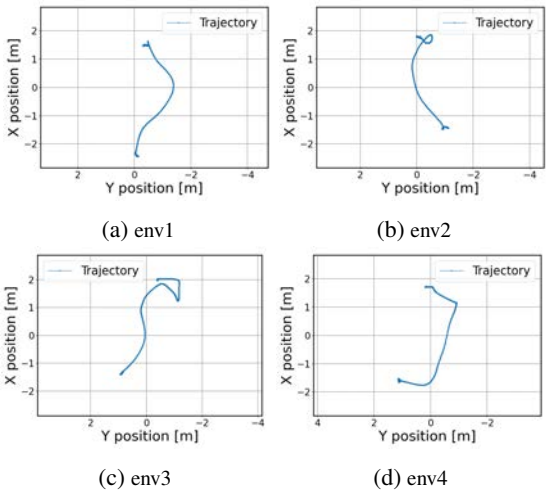
[2] Junwon Seo, Luis Duque, and Jim Wacker. Drone-enabled bridge inspection methodology and application. Automation in Construction, 94:112–126, 2018.

[3] Sebastien Boiteau, Fernando Vanegas Alvarez, Juan Sandino, Luis Gonzalez, and Julian Galvez. Autonomous uav navigation for target detection in visually degraded and gps denied environments. In Proceedings of the IEEE Aerospace Conference (AERO), pages 1–10, March 2023.

[4] S. A. H. Mohsan, N. Q. H. Othman, Y. Li, M. H. Alsharif, and M. A. Khan. Unmanned aerial vehicles (uavs): practical aspects, applications, open challenges, security issues, and future trends. Intelligent Service Robotics, 16(1):109–137, 2023.

[5] Benjamin Rivière, Wolfgang Hönig, Yisong Yue, and Soon-Jo Chung. Glas: Global-to-local safe autonomy synthesis for multi-robot motion planning with end-to-

http://www.imavs.org/



(a) env1 (b) env2

(c) env3 (d) env4

Figure 14: Trajectory results for each environment

end learning. IEEE Robotics and Automation Letters, 5(3):4249–4256, 2020.

[6] Yiming Mao, Ming Chen, Xianglin Wei, and Bing Chen. Obstacle recognition and avoidance for uavs under resource-constrained environments. IEEE Access, PP:1–1, 08 2020.

[7] Yiming Mao, Ming Chen, Xianglin Wei, and Bing Chen. Obstacle recognition and avoidance for uavs under resource-constrained environments. IEEE Access, 8:169408–169422, 2020.

[8] Hongxun Liu and Satoshi Suzuki. Model-free guidance method for drones in complex environments using direct policy exploration and optimization. Drones, 7(8), 2023.

[9] Christos Chronis, Georgios Anagnostopoulos, Elena Politi, George Dimitrakopoulos, and Iraklis Varlamis. Dynamic navigation in unconstrained environments using reinforcement learning algorithms. IEEE Access, 11:117984–118001, 2023.

[10] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. CoRR, abs/1707.06347, 2017.

[11] Tengteng Zhang and Hongwei Mo. Reinforcement learning for robot research: A comprehensive review and open issues. International Journal of Advanced Robotic Systems, 18(3), 2021.

[12] Fadri Furrer, Michael Burri, Markus Achtelik, and Roland Siegwart. RotorS—A Modular Gazebo MAV Simulator Framework, pages 595–625. Springer International Publishing, Cham, 2016.

http://www.imavs.org/