**DOI:** https://doi.org/10.30837/IST-2024.P1.043

# Quadrotor Simulation Packages for Reinforcement Learning

Vadym Honcharenko[1] та Valentyn Yesilevskyi[1]

[1] *Kharkiv National University of Radio Electronics, Kharkiv, Ukraine*

**Abstract**
This publication reviews the most popular quadrotor simulations applied in reinforcement learning for UAV control, obstacle avoidance, path planning, and swarm behavior. We emphasize the importance of high-quality simulations that integrate visual and physical models to mimic real-world dynamics and sensor data accurately. Several popular quadrotor simulation platforms - AirSim, gym-pybullet-drones, Flightmare, Aerial Gym, RotorPy, and RotorS, are evaluated for their different capabilities and supporting reinforcement learning applications. Each package is analyzed for its strengths in rendering, physical accuracy, parallelization, and compatibility with reinforcement learning frameworks. These simulators provide valuable tools for developing and testing reinforcement learning algorithms.

**Keywords**
quadrotor; reinforcement learning; simulation AirSim; gym-pybullet-drones; Flightmare; Aerial Gym; RotorPy; RotorS

## 1. Introduction

Over the last decade, quadrotors have become valuable in many practical applications in agriculture, civil engineering, environmental monitoring [1], search and rescue operations, the military sphere, and many other fields.

Reinforcement learning, a fundamental machine learning component, is the cornerstone of quadrotor control. It involves iterative training of agents to make decisions by interacting with an environment via a feedback loop with rewards and penalties as a loss function.

For the last decade, reinforcement learning achieved significant advancements for UAV in many fields of its autonomy [9]: UAV control, obstacle avoidance, path planning, and flocking behaviors. UAV control optimizes flight stability and responsiveness. Obstacle avoidance helps dynamically navigate through space to avoid collisions with objects. Path planning algorithms enable drones to find the most efficient routes subject to the conditions, adjusting to real-time environmental changes. Flocking is applied to control multiple UAVs in cooperative scenarios, allowing swarms of quadrotors to fly in formation, coordinate on missions, and avoid inter-agent collisions.

## 2. Quadrotor simulation in reinforcement learning

The training of reinforcement learning agents uses numerical simulation, which necessitates a robust mathematical model of interaction between an agent and an environment and a reward function. The quality of these elements is pivotal in transferring agent policymaking from simulation to the real world. Over the last decade, numerous works have been dedicated to solving quadrotor control problems with reinforcement learning [2].

Simulations are used to imitate real-world agent-environment feedback in reinforcement learning. Quadrotor simulation has two main parts - visual and physical. Visual simulation

vadym.honcharenko@nure.ua (V. Honcharenko); valentyn.yesilevskyi@nure.ua (V. Yesilevskyi)
0009-0002-0370-3361 (V. Honcharenko); 0000-0002-5935-1505 (V. Yesilevskyi)

imitates optical-flow data (visual RGB – input, segmentation, depth maps, etc.) for sensors like cameras and LiDARs. They differ from rendering engines (Unreal Engine, Gazebo, Unity, and others) and provide data quality (resolution, world realism) and quantity (RGB, segmentation maps, depth maps, etc.). Visual simulation is essential for obstacle avoidance, object tracking, and visual navigation. Physical simulation imitates quadrotor kinematics, interactions with other objects, and different space conditions (wind, air pressure, collisions). They differ with several implemented effects (like some not commonly implemented effects – downwash, parasitic aerial drug, wind, and others), physics simplifications, and assumptions (linearized motion equations, quadratic drag and torque, constant mass distribution, and others). Physical simulations are essential for UAV control, navigation, and path planning.

## 3. Simulation packages

Many simulation packages have been developed and published since the increasing adoption of machine learning-based techniques for UAVs in the mid-2010s. They help to make fast, precise, and realistic UAV simulations that are needed to test UAVs and train machine learning and control algorithms. Besides physical and visual fidelity, simulation packages have other significant properties: speed, compatibility with programming languages and software (like different ROS and libraries like OpenAI Gym), licensing, maintenance status, community support, and development entrance level. [9]

### 3.1. AirSim

AirSim [3] is a Microsoft UAV simulation product built on the Unreal rendering engine that utilizes Fast Physics and PhysX physics engines developed in C++ and C#. It is focused on visual fidelity; because of this, it is used mainly for autonomous navigation and computer vision. AirSim offers a large number of sensors and compatibility with open-source controllers (like PX4 and ArduPilot), which can be helpful for pilot training applications.

AirSim uses NVIDIA's physics engine, PhysX, which is not specialized for UAVs, and it is tightly coupled with the rendering engine to allow simulating environment dynamics; for this reason, it can achieve only limited simulation speeds. This limitation makes it difficult to use for reinforcement learning tasks in control and navigation areas.

### 3.2. gym-pybullet-drones

gym-pybullet-drones [4] is an open-source package for simulating multiple quadrotors with PyBullet physical and visual engines suited for Python development. It is mostly used for swarm-controlling tasks like line trajectory tracking, takeoff, and hovering.

Gym-pybullet-drones is based on the open-source Bullet Physics engine, which is notable for its realistic collisions and aerodynamic effects (like a drag, ground effect, and downwash). PyBullet engine also allows the extraction of synchronized rendering and kinematics. Both CPU- and GPU-based rendering are also available. Notably, it is compatible with the Unified Robot Description Format (URDF).

This package has interfaces for multi-agent and vision-based reinforcement learning applications utilizing the Gymnasium APIs. It includes examples of reinforcement learning workflows for single-agent and multi-agent scenarios.

### 3.3. Flightmare

Flightmare [5] is an open-source simulator based on the Unity rendering engine and supports different physical engines suited for both Python and C++ development. Flightmare is used for machine learning applications like path-planning in a complex 3D environment and visual-inertial odometry and is a popular tool for autonomous drone racing. The rendering engine can generate realistic visual information and simulate sensor noise, lens distortions, and extract the 3D point-cloud of the scene.

Flightmare supports three physical engines: a Gazebo-based quadrotor dynamics, real-world dynamics, and a parallelized implementation of classical quadrotor dynamics. Gazebo-based dynamics are slower but more realistic thanks to high-fidelity physics engines, e.g., Bullet. That allows users to control robot dynamics, ranging from simplified UAV models to advanced ones with friction and rotor drag.

In contrast to AirSim, we decouple the rendering module from the physics engine, which offers fast and accurate physics simulation when rendering is not required.

### 3.4. Aerial Gym

Aerial Gym [6] is an open-source simulator extension to Isaac Gym developed by NVIDIA that can simulate millions of multirotor vehicles parallelly, taking advantage of GPU parallelization. It provides customizable obstacle randomization functionality for navigation tasks, uses Universal Robot Description Format (URDF) files with an extensive set of objects in the environment, and interfaces for efficiently managing the environment. Aerial Gym provides sample environments containing robots equipped with simulated cameras capable of capturing RGB, depth, segmentation, and optical flow data at thousands of frames per second. It also integrates with PX4 and ROS 2 and additional sensors (magnetometer, GPS, and barometer).

### 3.5. RotorPy

RotorPy [7] is a lightweight open-source Python simulator focused on providing a comprehensive quadrotor model. The simulator's quadrotor model is extensively detailed, including 6-DoF dynamics, aerodynamic wrenches, actuator dynamics, sensors, and wind models. RotorPy includes a Gymnasium environment for RL applications. However, despite having an accurate quadrotor physics model and aerodynamic effects, it lacks complicated environments.

### 3.6. RotorS

RotorS [10] is built on top of Gazebo Classic and offers a modular framework for designing UAVs and developing control algorithms, mainly focusing on simulating vehicle dynamics. It provides several multi-rotor models. RotorS has been extensively used in robotics to develop algorithms on MAVs, such as drone racing, exploration, path-planning, or mapping. Nevertheless, the Gazebo engine has limited rendering capabilities and is not designed for efficient parallel dynamics simulation, which makes it challenging to use in reinforcement learning. Also, it does not come with ready-to-use reinforcement learning interfaces, and its dependency on Gazebo makes it ill-advised for parallel execution or vision-based learning applications.

## 4. Conclusions

While reinforcement learning has significantly advanced quadrotor autonomy in many areas, the choice of simulation environment plays a critical role in the success of these applications. The simulation packages reviewed in this paper offer distinct strengths and limitations, making it essential to select a simulator that best suits the specific application carefully. For example, AirSim excels in photorealistic visual environments but may need more speed for large-scale reinforcement training. At the same time, gym-pybullet-drones prioritize fast, multi-agent simulations with realistic physics but more straightforward visual rendering. Flightmare offers a balance with decoupled physics and rendering, providing flexibility for high-speed simulations, whereas Aerial Gym specializes in large-scale, GPU-accelerated simulations for multi-agent learning. RotorPy, on the other hand, is lightweight and focuses on detailed quadrotor dynamics but lacks complex environments.

Each simulator has trade-offs regarding computational speed, physical fidelity, visual realism, and compatibility with RL frameworks. Therefore, it is crucial to interrogate the pros and cons of each package relative to the specific requirements of the task - whether the focus is on visual navigation, high-fidelity physics, or large-scale multi-agent coordination -before applying it. Users can maximize efficiency, realism, and effectiveness in their reinforcement learning-based quadrotor applications by aligning the simulation environment with the research or development project goals.

## References

[1] Haeri, S. P., Razi, A., Khoshdel, S., Afghah, F., Coen, J., Fule, P., Watts, A., Kokolakis, N.-M., & Vamvoudakis, K. G. (2024). A comprehensive survey of research towards AI-enabled unmanned aerial systems in pre-, active-, and post-wildfire management. https://doi.org/10.48550/arXiv.2401.02456

[2] Buşoniu, Lucian, et al. "Reinforcement learning for control: Performance, stability, and deep approximators." *Annual Reviews in Control* 46 (2018): 8-28.

[3] Shah, Shital, et al. "Airsim: High-fidelity visual and physical simulation for autonomous vehicles." *Field and Service Robotics: Results of the 11th International Conference*. Springer International Publishing, 2018.

[4] Panerati, Jacopo, et al. "Learning to fly—a gym environment with pybullet physics for reinforcement learning of multi-agent quadcopter control." *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021.

[5] Song, Yunlong, et al. "Flightmare: A flexible quadrotor simulator." *Conference on Robot Learning*. PMLR, 2021.

[6] Kulkarni, Mihir, Theodor JL Forgaard, and Kostas Alexis. "Aerial Gym--Isaac Gym Simulator for Aerial Robots." *arXiv preprint arXiv:2305.16510* (2023).

[7] Folk, Spencer, James Paulos, and Vijay Kumar. "Rotorpy: A python-based multirotor simulator with aerodynamics for education and research." *arXiv preprint arXiv:2306.04485* (2023).

[8] Dimmig, Cora A., et al. Survey of Simulators for Aerial Robots: An Overview and In-Depth Systematic Comparisons. *IEEE Robotics & Automation Magazine*, 2024.

[9] AlMahamid, Fadi, and Katarina Grolinger. "Autonomous unmanned aerial vehicle navigation using reinforcement learning: A systematic review." *Engineering Applications of Artificial Intelligence* 115 (2022): 105321.

[10] Furrer, Fadri, et al. "Rotors—a modular gazebo mav simulator framework." *Robot Operating System (ROS) The Complete Reference (Volume 1)* (2016): 595-625.