AA203 Final Report: Drone Perching with Deep Reinforcement Learning

Yicheng Wang and Teo Ren

{wycheng, tianao}@stanford.edu

June 5, 2024

Abstract

In this project, we explore the development of a deep reinforcement learning (DRL) algorithm for training drones to execute precise perching maneuvers using the NVIDIA Isaac Gym simulator. Perching, the action of landing and stabilizing on a target surface, is crucial for applications where drones require stationary operation to conserve energy or perform tasks. Our approach leverages the high-performance capabilities of Isaac Gym for simulating complex interactions and behaviors in a controlled environment. By integrating a passive gripper mechanism and designing tailored reward functions, we have enabled the drone to adjust its position and orientation effectively to achieve successful perching on designated targets. The reinforcement learning environment is formulated as a Markov Decision Process, with a focus on optimizing the drone's control strategy through the Proximal Policy Optimization (PPO) algorithm. Preliminary results indicate significant improvements in the drone's perching accuracy and stability over training iterations. This work demonstrates the feasibility of simulation-based drone training for intricate behaviors. We are happy to share our report and video with current and future students. [Presentation Video] [Code]

1 Introduction

The rapid advancements in drone technology have opened up numerous applications across various fields, including surveillance, environmental monitoring, and delivery services. One of the critical challenges in the deployment of drones in complex environments is the ability to perform precise perching behavior. Perching, or the act of landing on a specific surface or object, is essential for tasks that require drones to conserve energy, gather data from stationary positions, or interact with objects in their environment.

To address this challenge, we propose to employ Isaac Gym, NVIDIA's high-performance simulator for robotics, to train a drone to perform perching behavior. Isaac Gym provides a scalable and efficient platform for simulating and training robotic systems using reinforcement learning. The simulator's ability to handle large-scale parallel training makes it an ideal tool for developing complex behaviors such as perching.

By training the drone within Isaac Gym, we aim to develop robust perching algorithms that can be transferred to real-world scenarios. The use of simulation allows us to explore a wide range of environmental conditions and perching strategies, accelerating the development process and reducing the need for extensive physical testing.

2 Related Work

In recent years, research into drone perching and autonomous landing has advanced significantly. One notable innovation is the passive, mechanically intelligent gripper designed by Hsiao[1]. This gripper operates without active control mechanisms, simplifying the control algorithms needed for successful perching. This aligns with our goal to develop a deep reinforcement learning (DRL)-based control system that is effective, energy-efficient, and robust in various environments.

DRL has proven effective in training drones for complex behaviors like navigating obstacles, autonomous landing, and aerial maneuvers. Our work focuses on using Isaac Gym for training drones to perch. Isaac Gym[2], developed by Viktor Makoviychuk and colleagues at NVIDIA, allows for rapid training of robotic policies on GPUs, bypassing CPU limitations. This platform accelerates training times significantly, supporting various environments and running thousands of environments in parallel on a single GPU. It uses NVIDIA's PhysX for high-fidelity simulation and integrates with machine learning frameworks like PyTorch, making it highly efficient and scalable.

Additionally, research by Guido et al. demonstrates the use of DRL to optimize gliding and perching strategies for an elliptical body[3]. Their study shows that DRL can find energy-efficient and time-efficient gliding paths without explicit physics knowledge. The results highlight two main flight patterns—bounding and tumbling—based on the shape and weight of the ellipse, offering insights into bio-inspired robotic design. The study suggests further exploration in three-dimensional contexts and real-world applications.

3 Problem Statement

The primary objective of this project is to develop a deep reinforcement learning algorithm capable of training a drone to perform precise perching behavior. The drone is equipped with a passive gripper at its base, which it must use to securely perch on a designated target.In most scenarios, we expect the drone to actively adjust its position during perching so that the passive gripper attached beneath it faces the target point. This way, when the trigger point of the gripper comes into contact with the target, the passive gripper immediately releases stored energy and completes the perching maneuver almost instantaneously through its mechanical structure. The key challenges include:

- Enabling the drone to learn how to adjust its position through reinforcement learning (RL). Therefore, we need to establish a new orientation reward method to incentivize the drone's position adjustments.
- The open-source drone environment we are using does not include a gripper [4]. Consequently, we need to design a gripper and simplify the perching task as much as possible, allowing the drone to learn to control the gripper to approach the target point through RL.
- To train the drone, equipped with the gripper, to successfully perch on an object within the environment. This requires us to add more reward functions, such as rewards for collision forces, to ensure the drone can effectively combine orientation control with the perching task.

4 Approach

In this section, we present the methodology employed for designing the perching task within the NVIDIA Isaac Gym simulator. The task involves a quadrotor drone equipped with a passive gripper

navigating to a designated bar and securely perching, establishing a challenging reinforcement learning (RL) environment.

4.1 Quadrotor and Gripper Design

We developed a new task for the NVIDIA Isaac Gym simulator, aligned with the VecTask class structure. This task leverages the parallelization capabilities of Isaac Gym, enabling efficient simulation of multiple environments. The environment is represented by a box with dimensions of 5x5x2.5 meters. Within this environment, there are two primary actors: the quadrotor drone and the target bar.

4.1.1 Drone Model

The quadrotor drone model is designed using a URDF file, based on standard quadrotor specifications with adjustments to the physical configuration to suit the perching task. The drone has four rotors, each capable of generating thrust and torque to control the drone's position and orientation. The drone's dynamics are modeled to reflect real-world flight characteristics, governed by the Newton-Euler equations:

$$m\frac{d\mathbf{v}}{dt} = \mathbf{F}_{thrust} + \mathbf{F}_{gravity} + \mathbf{F}_{aero} \tag{1}$$

$$I\frac{d\omega}{dt} = \tau_{thrust} + \tau_{aero} \tag{2}$$

where m is the mass of the drone, $\mathbf{v}$ is the linear velocity, $\mathbf{F}_{thrust}$ is the total thrust force, $\mathbf{F}_{gravity}$ is the gravitational force, $\mathbf{F}_{aero}$ represents aerodynamic forces, I is the moment of inertia, ω is the angular velocity, τ_{thrust} is the torque generated by the rotors, and τ_{aero} represents aerodynamic torques.

4.1.2 Gripper Design

As we mentioned earlier, in most cases, the gripper used for perching on drones is designed to operate passively. When the trigger point of the gripper underneath the drone contacts the target point, the passive gripper immediately releases stored energy and completes the perching maneuver almost instantaneously through its mechanical structure.

In our reinforcement learning environment, to simplify computations and increase training feasibility, we have completely fixed the joints of the gripper, maintaining it at an appropriate open angle without allowing it to adjust. This way, during perching training, the task is simplified to having a specific point on the gripper reach a designated point on a hoop in the environment, at which point the task is considered complete. In the real world, Upon the gripper center touching the target bar, the passive gripping mechanism engages, stabilizing the drone without additional control inputs. The specific design of the drone with the gripper and its interactive environment is illustrated in the figure 4.1.2. This design simplifies the control strategy, allowing the RL algorithm to focus on the approach and stabilization maneuvers.

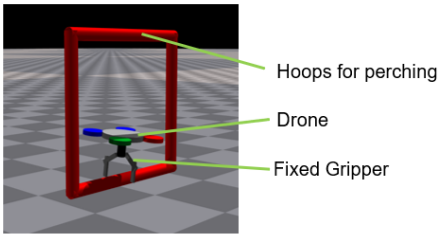

Figure 1: Drone with Gripper

4.2 Reinforcement Learning Task

The task is modeled as a Markov Decision Process (MDP) defined by the tuple (S, A, P, R, γ):

- **State Space (S):** The state space includes the drone's position, orientation, linear velocities, and angular velocities. Each state $s_t \in S$ represents the drone's current status at time t.
- **Action Space (A):** The action space consists of thrust commands for each of the four rotors, bounded between -1 and 1 for normalization.
- **State Transition Probability (P):** The probability $P(s_{t+1}|s_t, a_t)$ of transitioning from state s_t to state s_{t+1} under action a_t.
- **Reward Function (R):** The reward function provides immediate rewards for actions that lead to successful perching.
- **Discount Factor (γ):** A discount factor balancing the importance of future rewards.

The goal is to find an optimal policy π^* that maximizes the expected cumulative reward:

$$\pi^* = \arg\max_{\pi} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \mid \pi\right] \tag{3}$$

4.3 Reward Function

The reward function is designed to guide the drone to the target bar and execute a successful perch. It is composed of several components:

- **Position Reward** $r_{position}$ encourages the drone to move closer to the target, inversely proportional to the distance between the drone and the target:

$$r_{position} = \frac{1.0}{1.0 + d_{target}^2}, \quad d_{target} \text{ is the Euclidean distance} \tag{4}$$

- **Pitch Angle Reward** r_{pitch} encourages the drone to perch with a certain pitch angle:

$$r_{pitch} = \frac{1.0}{1.0 + \theta_{pitch}^2}, \quad \theta_{pitch} \text{ is the pitch angle difference from reference} \tag{5}$$

- **Uprightness Reward** r_{up} encourages the drone to maintain a stable pose near the target:

$$r_{up} = \frac{1.0}{1.0 + \theta_{up}^2}, \quad \theta_{up} \text{ is the angle from vertical} \tag{6}$$

- **Orientation Reward** r_{orien} encourages the drone to adjust its pose:

$$r_{orien} = \frac{1.0}{1.0 + \theta_{head}^2}, \quad \theta_{head} \text{ is the heading angle difference from reference} \tag{7}$$

- **Velocity Reward** r_{lin_vel} Encourages fast success:

$$r_{lin\_vel} = \frac{1.0}{1.0 + \|v_{lin}\|}, \quad v_{lin} \text{ is the linear velocity} \tag{8}$$

- **Spinnage Reward** r_{ang_vel} encourages the drone to adjust the gripper orientation:

$$r_{ang\_vel} = \frac{1.0}{1.0 + \|v_{ang}\|}, \quad v_{ang} \text{ is the angular velocity} \tag{9}$$

- **Success Reward** $r_{success}$ provides a significant reward for successful perching

$$r_{success} = \begin{cases} 400.0 & \text{if perching is successful} \\ 0.0 & \text{otherwise} \end{cases} \tag{10}$$

- **Collision Penalty** $r_{collision}$ penalizes the drone for collisions with the environment:

$$r_{collision} = \begin{cases} -2.0 & \text{if collision detected} \\ 0.0 & \text{otherwise} \end{cases} \tag{11}$$

The overall reward function is:

$$R(s_t, a_t) = \omega_{position} r_{position}(1 + \omega_{up} r_{up} + r_{ang\_vel}) + \omega_{orien} r_{orien} + r_{lin\_vel} + r_{success} + r_{collision} \tag{12}$$

where the weights are $\omega_{position} = 2.0$ and $\omega_{up} = 4.0$.

The reset conditions are triggered when the drone reaches an unhealthy state or successfully perches. These conditions include excessive distance from the target, low altitude, and abnormal tilt angles.

4.4 Observation Space

The observation space consists of 14 values representing the state of the drone:

- **Relative Position:** The drone's position relative to the target (3 values).
- **Orientation:** The drone's orientation represented by quaternions (4 values).
- **Linear Velocities:** The drone's linear velocities (3 values).
- **Angular Velocities:** The drone's angular velocities (3 values).
- **Orientation Difference:** The normalized angular difference between the drone's orientation and the target's bottom bar orientation (1 value).

4.5 Policy Training

Using Isaac Gym, we perform policy rollouts on 4096 environments in parallel, significantly speeding up data collection. The Proximal Policy Optimization (PPO) algorithm is utilized for training, which is well-suited for optimizing continuous control tasks. We utilized the PPO algorithm from the rl-game library, which comes with the Isaac Gym simulator. The PPO algorithm iteratively updates the policy based on interactions with the environment to maximize the cumulative reward.

The network architecture for the PPO algorithm is configured to capture the complexity of the task, ensuring effective and efficient learning of the perching behavior. The training setup involves tuning hyperparameters such as learning rate, discount factor, and entropy coefficient to achieve optimal performance.

The training process is monitored through key metrics, including training reward and episode length, to evaluate the learning progress and policy performance.

4.5.1 Objective Functions

PPO uses a clipped surrogate objective function to prevent large policy updates, ensuring stability:

$$L^{\text{CLIP}}(\theta) = \hat{\mathbb{E}}_t \left[\min \left(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t \right) \right] \tag{13}$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ is the probability ratio between the new and old policies, $\hat{A}_t$ is the advantage estimate, and ϵ is the clipping parameter.

The value function loss optimizes the value function, typically using the mean squared error between predicted values and actual returns:

$$L^{\text{VF}}(\theta) = \hat{\mathbb{E}}_t \left[\left(V_\theta(s_t) - R_t \right)^2 \right] \tag{14}$$

where $V_\theta(s_t)$ is the value function estimate, and R_t is the actual return.

To encourage exploration, an entropy bonus is added, penalizing low-entropy policies to promote a diverse range of actions:

$$L^{\text{ENT}}(\theta) = \hat{\mathbb{E}}_t \left[\mathcal{H}(\pi_\theta(\cdot|s_t)) \right] \tag{15}$$

where $\mathcal{H}(\pi_\theta(\cdot|s_t))$ is the policy entropy.

The total objective function combines these components:

$$L(\theta) = L^{\text{CLIP}}(\theta) - c_1 L^{\text{VF}}(\theta) + c_2 L^{\text{ENT}}(\theta) \tag{16}$$

where c_1 and c_2 balance the contributions of the value function loss and entropy bonus, respectively.

4.5.2 Hyperparameters

The Proximal Policy Optimization (PPO) algorithm used for training the drone includes key hyperparameters as follows: a learning rate of 1×10^{-3}, a clip parameter (ϵ) of 0.2, a discount factor (γ) of 0.985, and a GAE parameter (λ) of 0.95. The value function coefficient is set to 2, with a batch size of 16384, and training occurs over 8 epochs with a minibatch size of 64. The horizon length is 16, and the KL threshold is 0.016. Training is capped at 800 epochs, with models saved

every 50 epochs. Gradient norm is limited to 1.0, the sequence length is 4, and the bounds loss coefficient is 0.0001.

5 Experiments

We conducted a series of experiments to evaluate the performance of our approach, involving training the drone in the simulation environment with the designed observation space, reward function, and PPO hyperparameters. Our preliminary results indicated that the drone was capable of learning the perching behavior within the simulation environment, with the PPO algorithm demonstrating stability and efficiency in training the control policy. The success rate of perching attempts improved significantly over the training period, and the trained policy exhibited robustness to variations in the environment. The experiments has two parts:

5.1 Drone Orientation

We had trained a model for drone halting at target point as our preliminary results. As is illustrated in Figure 5.1, we introduced a pitch reward to encourage the drone perching at specific point with desired landing flare angle, and successfully achieved better imitations of perching behaviour. After we attached our designed gripper to the drone, however, we observed large overshooting in the newly trained model. An explanation is that the additional mass and inertia makes the quadrotor difficult to maneuver, causing the thrust exceeding the limit, thus unable for the drone to fully explore the space. We tried to transfer the previous model trained with drones without gripper to this new setup and it works well.

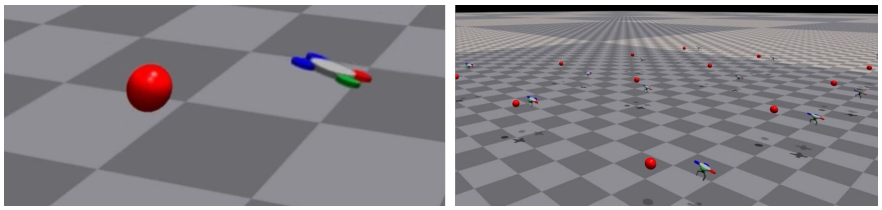

Figure 2: Drone and Drone with gripper with orientation reward

5.1.1 Drone Perching

Next, we integrated collision property to our environment. With extensive reward function engineering, and fine-tuning the PPO hyperparameters, we accomplished the drone perching on the bottom bar of the hoop (Figure 5.1.1

5.2 Training Results

Figure 5.2 demonstrates the rewards per iteration when training our drone with gripper halting and perching. The curve shows that the rewards of drone orientation and drone perching experiment stabilize at about 13k after 1k steps and 11k after 2k steps respectively.

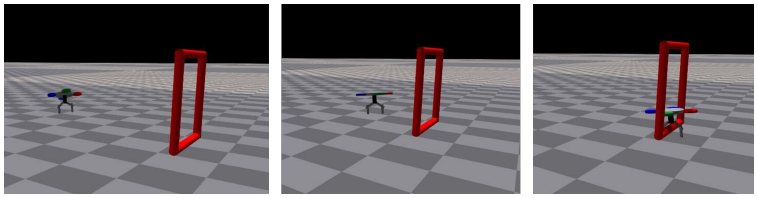

Figure 3: Successfully perching on the Hoops

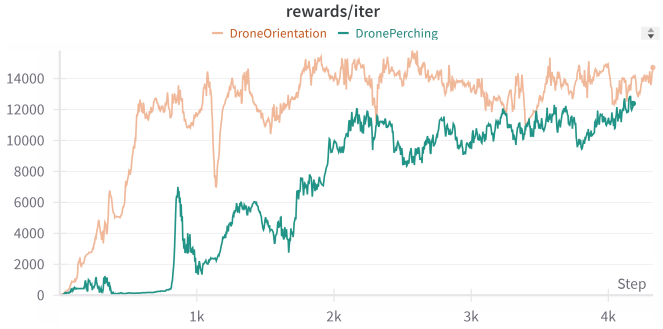

Figure 4: Training Data of Rewards

6 Future Work and Limitation

In this report, we presented our approach to training a drone for perching behavior using deep reinforcement learning and Isaac Gym. Our results demonstrate the feasibility of using simulation-based training to develop complex behaviors for drones. Future work will focus on transferring the trained policy to real-world drones and further refining the simulation environment to capture more realistic scenarios. Additionally, we aim to explore other reinforcement learning algorithms and hybrid approaches to enhance the learning process and improve the overall performance of the perching behavior.

References

[1] HaoTse Hsiao et al. "A mechanically intelligent and passive gripper for aerial perching and grasping". In: *IEEE/ASME Transactions on Mechatronics* 27.6 (2022), pp. 5243–5253.

[2] Viktor Makoviychuk et al. "Isaac gym: High performance gpu-based physics simulation for robot learning". In: *arXiv preprint arXiv:2108.10470* (2021).

[3] Guido Novati, Lakshminarayanan Mahadevan, and Petros Koumoutsakos. "Controlled gliding and perching through deep-reinforcement-learning". In: *Physical Review Fluids* 4.9 (2019), p. 093902.

[4] Rolando Esquivel-Sancho. "AUTONOMOUS DRONE FLIGHT THROUGH HOOPS, REINFORCEMENT LEARNING". In: (2023). URL: `https://github.com/esquivelrs/DTU-RL-Isaac-Gym-Drone-Env/blob/main/report/RL_for_Drone_Navigation_through_Hoops_in_Isaac_Gym.pdf`.