Manipulator Grasping Based on Deep Reinforcement Learning

Yufei Liu†

†School of Data Science, The Chinese University of Hong Kong, Shenzhen, China

1. Introduction

Robotic grasping is a challenging task that involves perception, planning, and control. In practice, it is difficult to handle the large variety of object shapes, materials, and environmental conditions, as well as the uncertainty in sensing and actuation. Because of these issues, enabling autonomous learning becomes important for improving the reliability and usability of manipulators in different work settings [1, 2].

This report reproduces the Franka cube stack task provided in the official Isaac Gym examples (`https://github.com/isaac-sim/IsaacGymEnvs`). This task uses the PPO algorithm [3] to enable the Franka Emika Panda arm [4] to stack two blocks placed at random positions on table.

The report is organized as follows. Section 2 defines the cube stacking task and presents the core theoretical foundations. Section 3 describes the simulation setup in Isaac Gym and the PPO training method. Section 4 presents the experimental results, including training curves and stacking performance. Section 5 concludes the report by summarizing its main content.

2. Problem definition

The objective of this work is to develop a robust policy for a 7-DoF Franka Emika Panda arm to sequentially grasp a 50 mm cube (Cube A) and precisely stack it on top of a 70 mm cube (Cube B) from randomized initial positions. The policy is trained end-to-end using Proximal Policy Optimization (PPO)[3], with the implementation provided by `rl-games`. To support stable long-horizon manipulation, the task adopts Operational Space Control (OSC) [5]: the policy outputs 6-DoF end-effector pose increments and a binary gripper command, which are converted into joint torques by an inner-loop PD controller. This low-level controller simplifies learning and leads to smoother end-effector motion in this stacking task.

In this task, the robot arm must handle the uncertainty in the initial positions of both cubes and execute precise grasping and stacking motions. The success of the policy is measured by its ability to consistently grasp Cube A and place it accurately on top of Cube B without causing the cubes to fall or be misplaced. The manipulation process requires coordination between position control of the end-effector and the gripper, making it a challenging long-horizon task. By designing appropriate reward functions and using PPO for end-to-end policy learning, the arm can gradually acquire robust and reliable stacking behavior.

3. Method

Figure 1 shows the simulation pipeline of Isaac Gym and the reward design used in this task. As shown in Figure 1(a), the main steps before the actual training are to initialize the simulator and create the environments. In this task, a simple plane is added as the ground, and the required assets-including the Franka manipulator, the table, and the two cubes are loaded into the scene. The training process then starts by resetting the environments, as illustrated in Figure 1(a). After this setup, the training is executed by calling the external script `train.py`, which runs the PPO optimization loop and updates the policy throughout the training process. The critical parameters of the PPO algorithm are shown in Table 1. It should be noted that domain randomization [6] and the contact friction are not included during training in order to obtain the desired results more quickly.

Table 1: Training parameters of PPO

Parameter	Value
Episode length of each iteration	500
Maximize epochs	10000
Horizon length	32
Learning rate	$5 \cdot 10^{-4}$
Number of environments	2048
Value loss coefficient	4
Target KL	0.008
PPO clip	0.2

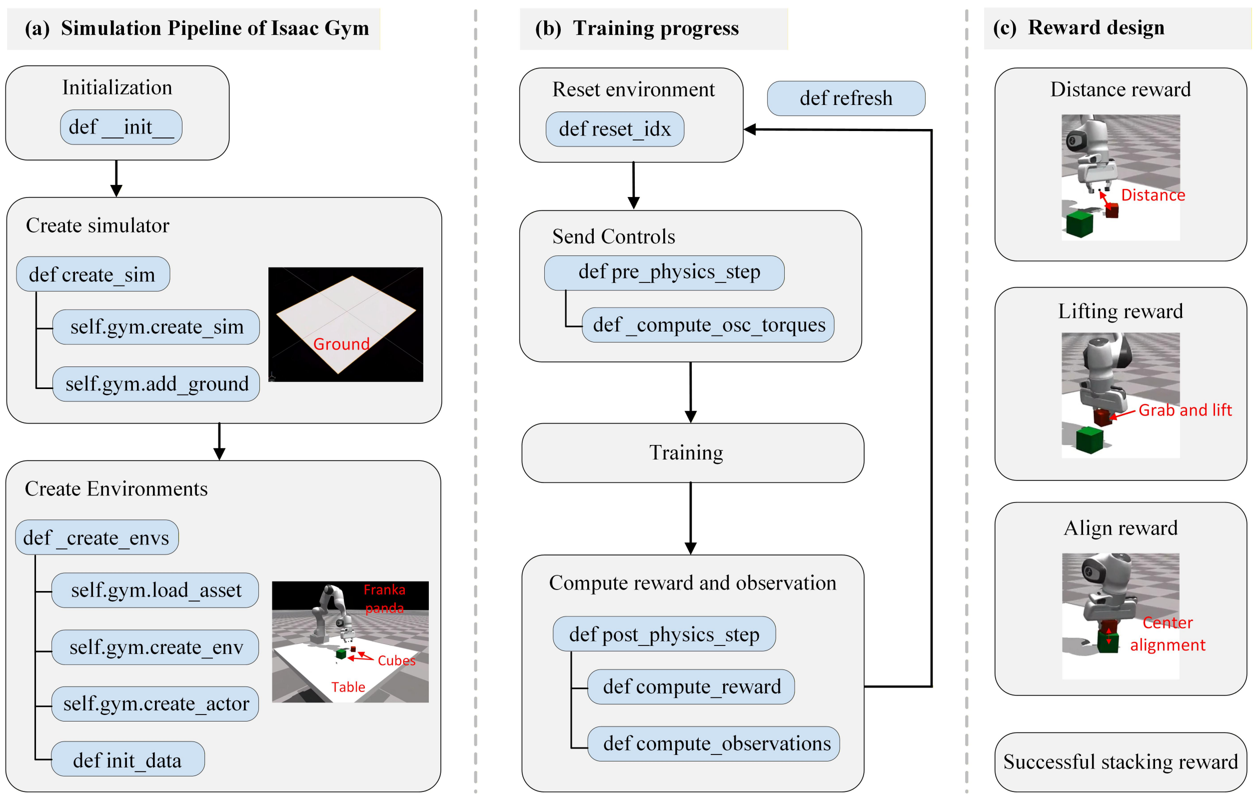

Figure 1: Simulation pipeline and reward design

The reward design is the key to ensuring the desired behavior, and four rewards are defined in this task to guide the policy toward the final objective. As shown in Figure 1(a), a distance reward is first applied to encourage the gripper to move closer to Cube A. Next, a lifting reward is added to encourage the gripper to grasp Cube A and lift it. Third, an alignment reward drives Cube A toward Cube B and encourages reaching the target stacking position. Finally, a successful stacking reward is given when Cube A is correctly placed on Cube B and the gripper releases it. The magnitude and conditions of each reward are summarized in Table 2. It should be noted that if the successful stacking condition is met, a large sparse bonus of 16.0 is awarded and the other three small reward are ignored.

The symbols used in Table 2 are defined as follows: d_{G} denotes the distance between the robot gripper and cube A; d_{L} and d_{R} represent the distances from the left and right grippers to cube A, respectively; $d_{\text{A-B}}$ is the distance between the centers of cube A and cube B, used for computing the alignment reward; $d_{\text{A-hand}}$ indicates the distance from the gripper to cube A, which is used to check whether the hand is away when stacking is completed; h_{A} is the height of cube A above the table surface; w_{A} is the height of cube A itself; and $p_{\mathrm{A}}^{\mathrm{xy}}$ and $p_{\mathrm{B}}^{\mathrm{xy}}$ denote the positions of cube A and cube B projected onto the XY plane, respectively.

Table 2: Reward setting

Reward type	Compute value	Max value	Reward condition (mm)
Distance reward	$1 - \tanh\left(10 \cdot \frac{d_{\mathrm{G}}+d_{\mathrm{L}}+d_{\mathrm{R}}}{3}\right)$	0.1	Always
Lifting reward	$h_{\mathrm{A}} - w_{\mathrm{A}}$	1.5	$h_{\mathrm{A}} - w_{\mathrm{A}} > 40$
Align reward	$1 - \tanh(10 \cdot d_{\text{A-B}})$	2.0	$h_{\mathrm{A}} - w_{\mathrm{A}} > 40$
Successful stacking reward	16.0	16.0	$\left\|p_{\mathrm{A}}^{\mathrm{xy}} - p_{\mathrm{B}}^{\mathrm{xy}}\right\|_2 < 20$ $h_{\mathrm{A}} - w_{\mathrm{A}} > 40$ $d_{\text{A-hand}} > 40$

4. Results

The training curves are presented in Figure 2. Figure 2(a) shows the raw training reward. It can be seen that the reward gradually increases and stabilizes as the number of epochs grows, reaching near the full reward (which is 16.0) around 1200 training epochs. This indicates that the four designed reward components were effectively learned, demonstrating good learning performance. Figures 2(b)-(d) illustrate the changes in learning rate and loss over the training epochs. The adaptive learning rate decreases automatically from 5×10^{-4} to 4×10^{-5}, while both actor and critic losses converge smoothly, indicating stable and sample-efficient training.

Figure 3 illustrates the performance of the policy after 3800 training epochs. It can be observed that in both test1 and test2, the robot arm is able to successfully grasp randomly initialized cube A and stack it on cube B, demonstrating the effectiveness of the training. A display video is also provided, see Section Appendix A.

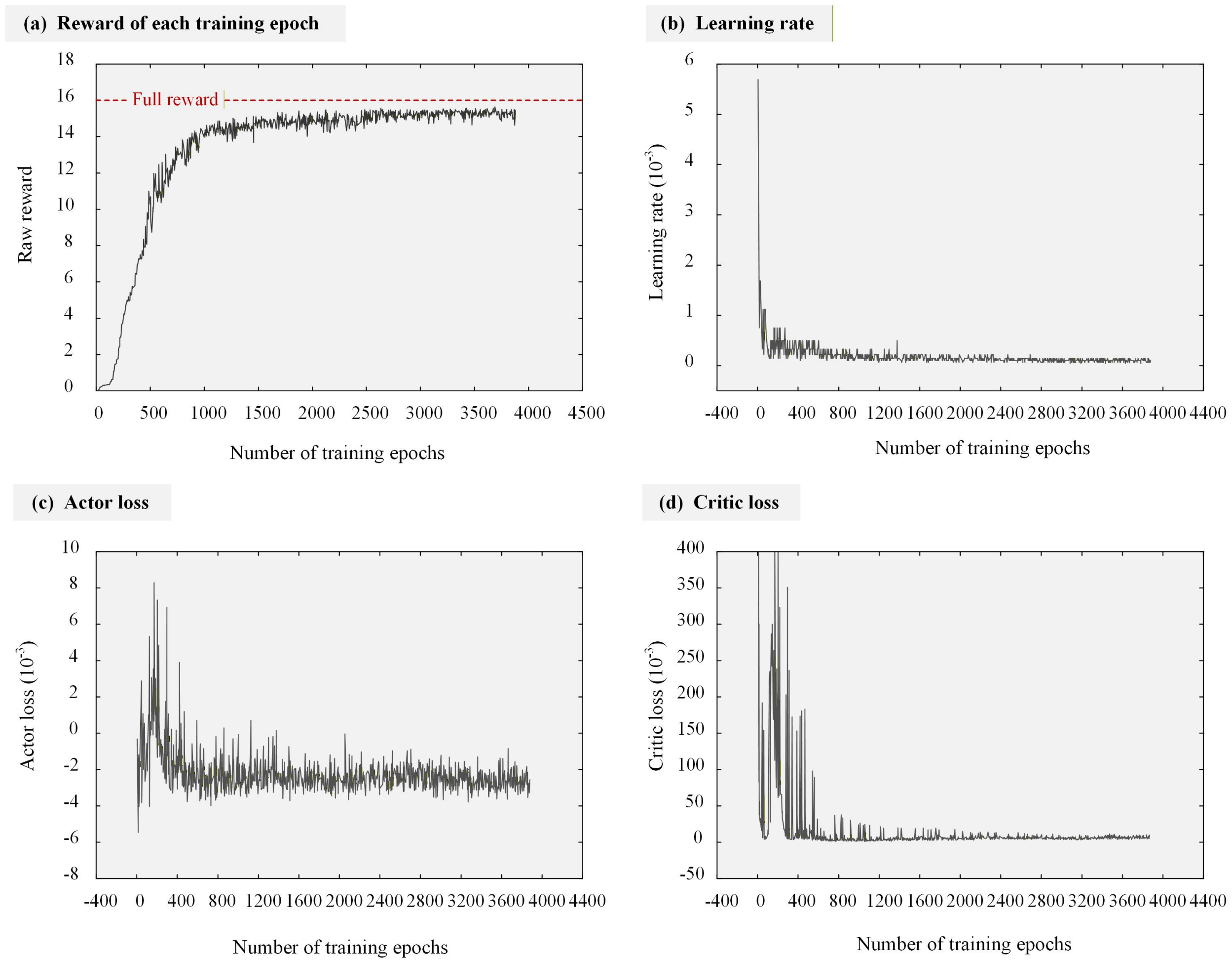

Figure 2: Training curves.

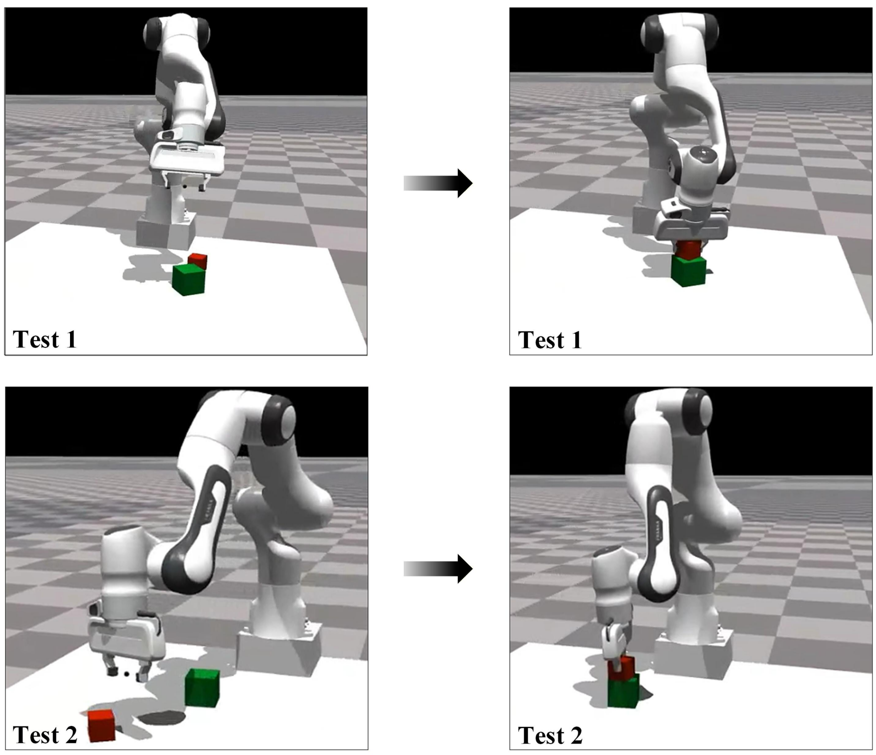

Figure 3: Training curves.

5. Conclusion

(1) This report reproduces the Franka Panda cube stacking task in Isaac Gym. The goal is to develop a policy that allows the robot arm to grasp cube A and stack it on cube B from randomly generated positions. The simulation setup, PPO algorithm, and task implementation are presented.

(2) Four rewards are defined in PPO training: a distance reward to encourage approaching cube A, a lifting reward to encourage grasping and lifting, an alignment reward to guide cube A toward cube B, and a sparse stacking reward for successfully placing cube A on cube B.

(3) The trained policy shows effective performance. The robot arm can successfully grasp randomly placed cube A and stack it on cube B. Training curves indicate stable learning and smooth convergence, demonstrating the effectiveness of the reward design and the PPO training process.

Acknowledgement

We drew inspiration from the course materials and design ideas of the AIR5023 course, taught by Prof. Xiaoqiang Ji, which provided valuable guidance for structuring our content and presentation.

Appendix A. Trained Policy and raw data

The project code and the trained policy checkpoint, as well as a video demonstrating the performance of the trained policy, are provided at the following links: `https://github.com/YFLiu-Robotic/Result-of-Franka-cube-stack-task`

References

[1] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, D. Quillen, Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection, The International Journal of Robotics Research 37 (4-5) (2018) 421–436. doi:10.1177/0278364917710318.

[2] J. Bohg, A. Morales, T. Asfour, D. Kragic, Data-driven grasp synthesis—a survey, IEEE Transactions on Robotics 30 (2) (2014) 289–309. doi:10.1109/TRO.2013.2289018.

[3] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal policy optimization algorithms, ArXiv abs/1707.06347 (2017).
URL https://api.semanticscholar.org/CorpusID:28695052

[4] Franka emika gmbh, franka emika robots' instruction handbook (2021).

[5] O. Khatib, A unified approach for motion and force control of robot manipulators: The operational space formulation, IEEE Journal on Robotics and Automation 3 (1) (1987) 43–53. doi:10.1109/JRA.1987.1087068.

[6] J. Ibarz, J. Tan, C. Finn, M. Kalakrishnan, P. Pastor, S. Levine, How to train your robot with deep reinforcement learning: lessons we have learned, The International Journal of Robotics Research 40 (2021) 698 – 721.
URL https://api.semanticscholar.org/CorpusID:231839855