International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 02 Issue: 06 June 2024
Page No: 1665- 1668
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2024.0229

Interactive Robotic Arm Simulation

Dipali Ghatge[1], Pratham Patil[2], Atharva Algude[3], Shubhangi Chikane[4], Atharv Dhotre[5]
[1,2,3,4,5]Karmaveer Bhaurao Patil College of Engineering, Satara, India.
Emails: *dipali.ghatge@kbpcoes.edu.in[1], pratham.patil.work@gmail.com[2], algudeatharva007@gmail.com[3], shubhangichikane86@gmail.com[4], atharvadhotre396@gmail.com[5]*

Abstract

In the dynamic landscape of robotics and artificial intelligence, this research pioneers a groundbreaking fusion of simulation technology and advanced machine learning, specifically reinforcement learning, to enhance robotic arm capabilities. The focus centers on the utilization of a cutting-edge simulator, powered by the PyBullet physics engine, to faithfully replicate the intricate dynamics of a robotic arm within a digital environment. Serving as an experimental ground, the simulator enables the robotic arm to navigate, manipulate objects, and dynamically engage with its surroundings. Through a symbiotic relationship between simulation technology and reinforcement learning, this research focuses on an adaptive learning approach. This approach accelerates the robotic arm's skill acquisition, refining critical aspects such as precision, speed, and adaptability. The project contributes to the evolution of robotic arm capabilities, paving the way for more autonomous, versatile, and adept robotic systems in the realm of artificial intelligence and robotics.

Keywords: *Reinforcement Learning; Simulation, Robotic Arm; Pybullet; Artificial Intelligence; Proximal Policy Optimization; Deep Deterministic Policy Gradient; Actor-Critic; Open AI Gym; Keras-RL.*

1. Introduction

In the rapidly evolving landscape of robotics and artificial intelligence, this research project converges simulation technology with advanced machine learning, particularly in the context of robotic arms. The aim is to revolutionize the capabilities of robotic arms through the integration of a cutting-edge simulator, driven by the PyBullet physics engine, and sophisticated reinforcement learning algorithms facilitated by Keras-RL and OpenAI Gym. Robotic arms play a pivotal role in diverse applications, from manufacturing to healthcare. However, the challenge lies in enabling these robotic arms to autonomously adapt and optimize their actions over time. The proposed solution involves creating a digital environment within the PyBullet-powered simulator, allowing the robotic arm to navigate, manipulate objects, and dynamically engage with its surroundings. What distinguishes this project is the infusion of reinforcement learning algorithms, empowering the robotic arm to refine its performance iteratively in the virtual domain (Figure 1). This adaptive learning approach not only accelerates skill acquisition but also equips the robotic arm to master diverse tasks efficiently, enhancing precision, speed, and adaptability.

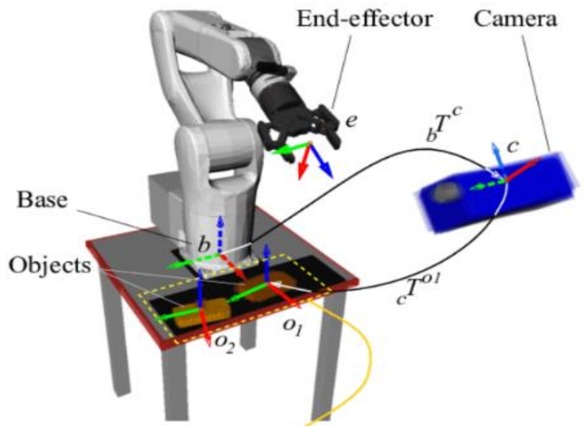

Figure 1 Robotic Arm

2. Method

The **Actor-Critic** method is a popular reinforcement learning algorithm that involves two distinct components within an agent: the actor and the critic [1,2]. As the agent interacts with its environment, it learns to effectively map observed states to two crucial outputs, influencing its decision-making process.

2.1.Actor

Responsibility: The actor is tasked with providing a probability distribution over all possible actions in the action space based on the observed state of the environment (Refer Figures 2 & 3).

Output: For each action available to the agent, the actor outputs a probability value, indicating the likelihood of selecting that particular action given the current state.

Example: In a robotic arm manipulation task, the actor might output a probability distribution over actions such as "move left," "move right," "grasp," etc., depending on the current configuration of the arm.

2.2.Critic

Responsibility: The critic is responsible for estimating the cumulative rewards the agent expects to receive in the future based on its current state and actions.

Output: The critic outputs the expected sum of rewards, which serves as a measure of the potential long-term benefits of the agent's actions.

Example: In a game scenario, the critic evaluates the current state and the actions taken by the agent, providing an estimate of the total future rewards the agent anticipates, considering the consequences of its decisions.

2.3.Learning Process

The actor and critic work collaboratively to maximize the expected cumulative rewards. The actor's recommended actions are influenced by the feedback from the critic, aiming to select actions that lead to higher expected rewards. [11-14]

2.4.Example Scenario

Consider a self-driving car as an example. The actor determines the probability distribution of possible actions, such as accelerating, braking, or turning, based on the current traffic conditions and road state. Simultaneously, the critic evaluates the chosen actions, estimating the cumulative rewards associated with the car's decisions over time. If the critic identifies that a certain driving behavior tends to lead to positive outcomes (e.g., reaching the destination quickly and safely), the actor adjusts its probabilities to favor that action more frequently. In this way, the Actor-Critic method combines the strengths of both components, with the actor exploring and exploiting actions in the environment, guided by the critic's assessment of the expected long-term rewards. This collaborative learning approach contributes to the agent's ability to make informed decisions that maximize its cumulative rewards over time.

3. Proximal Policy Optimization [3]

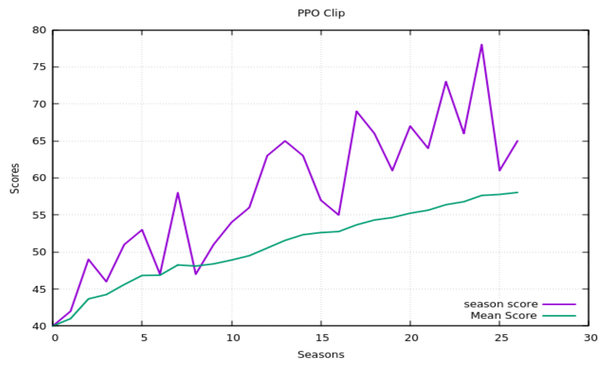

Figure 2 Performance of PPO Algorithm

Proximal Policy Optimization (PPO) is a technique in reinforcement learning that falls under the category of policy gradient methods. [5-8] It is versatile and applicable to environments with discrete or continuous action spaces. PPO adopts an on-policy training approach, meaning it learns directly from experiences collected while following the current policy. A key aspect of PPO is its utilization of the actor-critic method. Here, the actor component maps observations to actions, while the critic estimates the expected rewards based on those actions. The training process begins with the collection of trajectories, which are sequences of states and actions, for each training epoch. These trajectories are obtained by sampling from the current version of the stochastic policy. Subsequently, PPO computes rewards-to-go and advantage estimates. These metrics are crucial for updating both the policy and fitting the value function. Policy updates are performed using a stochastic gradient ascent optimizer, while the value function is adjusted through a gradient descent algorithm. This iterative procedure continues over multiple epochs until the agent successfully solves the environment, achieving the desired task or goal.

3.1.PPO Network Architecture

Feature Network: A Convolutional Neural Network shared by both Actor and critic networks

Actor Network: (Image Input) --> Feature Net --> Fully Connected network --> (Action Output)
Critic Network: (Image Input) --> Feature Net --> Fully Connected Network --> (Value Output)

3.2.Deep Deterministic Policy Gradient [4]

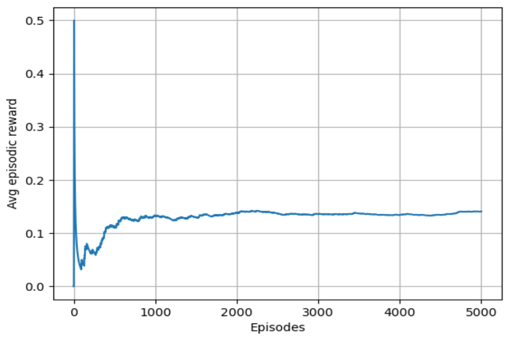

Figure 3 Performance of DDPG Algorithm

Deep Deterministic Policy Gradient (DDPG) stands as a model-free, off-policy algorithm designed for learning continuous action spaces.[9] It amalgamates concepts from both Deterministic Policy Gradient (DPG) and Deep Q-Networks (DQN). Drawing from DQN, DDPG incorporates Experience Replay and slow-learning target networks. At its core, it operates on the principles of DPG, enabling it to navigate continuous action spaces.

For example

The challenge at hand is to tackle the classic Inverted Pendulum control problem, where the agent is limited to two actions: swinging left or swinging right. What distinguishes this problem is the continuous nature of the action space. Unlike discrete actions such as -1 or +1, here, the agent must choose from an infinite range of actions spanning from -2 to +2. The theoretical framework of DDPG involves two networks:

- **Actor**: This network proposes an action based on the current state.
- **Critic**: This network evaluates the proposed action's quality, assigning positive values for favorable actions and negative values for unfavorable ones.

DDPG introduces two additional techniques not found in the original DQN. Firstly, it employs two Target networks. Why? These networks contribute to training stability. By learning from estimated targets and updating the Target networks slowly, DDPG ensures the stability of its estimated targets. Conceptually, this approach can be likened to saying, "I have a strategy that seems promising; I'll stick with it for a while until I discover a better one," as opposed to continually re-learning the game after every move.

4. Results

In the dynamic intersection of simulation technology and advanced machine learning, this research pioneers a novel approach to enhancing robotic arm capabilities (Figure 4). We have used Pybullet physics engine as our simulator and algorithms like Proximal Policy Optimization and Deep Deterministic Policy Gradient. There performances are as follows:

4.1.Algorithm Performance

DDPG algorithm didn't work as intended the performance didn't rise even after 2k episodes. [10]

4.2.Results with PPO 'clip' Method

Rewards and Scores: Each season involves 1000 environment steps. The score for each season is the total reward. obtained during these 1000 steps. Mean score is the average of season scores over last 100 seasons. The score increases as training progresses.

4.3.Results with PPO 'penalty' Method

Updating beta slows down the learning process. The following results are produced with fixed beta (= 0.5).

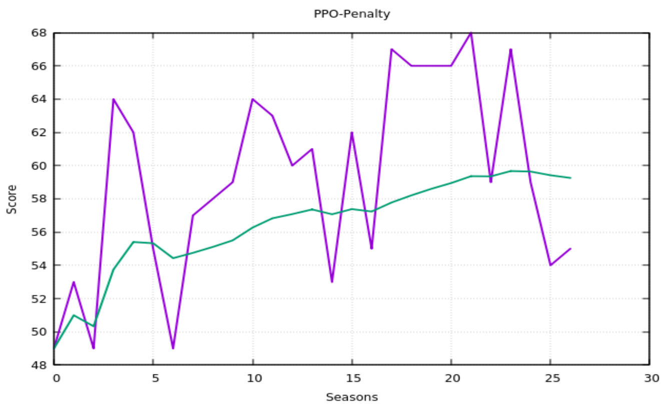

Figure 4 Results

Conclusion

In conclusion, this research contributes to the evolution of robotic arm capabilities, paving the way for more autonomous, versatile, and adept robotic systems, ultimately advancing the field of artificial intelligence and robotics.

References

[1]. Kiran Kumbhar B M. International Conference on Circuits, Controls, Communications and Computing, October 2016.
[2]. Jignesh Vinod Patel. Robotic Manipulator (Robotic Arm), International Journal of Engineering Research & Technology, September 9, 2021.
[3]. Proximal Policy Optimization Algorithms, Schulman et al.2017.
[4]. Deterministic Policy Gradient Algorithms, Silver et al.2014.
[5]. Continuous Control with Deep Reinforcement Learning, Lillicrap, et al.2016.
[6]. Mark Rowland, Robert Dadashi, Saurabh Kumar, Remi Munos, Marc G. Bellemare, Will Dabney. Proceedings of the 36th International Conference on Machine Learning, PMLR 97:5528-5536, 2019.
[7]. Deep Reinforcement Learning with Double Q-Learning. H. van Hasselt, A. Guez, D. Silver. AAAI 2016.
[8]. Natural Actor-Critic Algorithms Shalabh Bhatnagar, Richard Sutton, Mohammad Ghavamzadeh, Mark Lee.
[9]. Proximal Policy Optimization Algorithms John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, Oleg Klimov.
[10]. PyBullet Quickstart Guide Erwin Coumans, Yunfei Bai, 2016-2023.
[11]. "Deep Reinforcement Learning", Springer Science and Business Media LLC, 2020
[12]. Shivaram Kalyanakrishnan, Peter Stone. "Characterizing reinforcement learning methods through parameterized learning problems", Machine Learning, 2011.
[13]. Xiao Zhang, Zhi Wu, Qirun Sun, Wei Gu, Shu Zheng, Jingtao Zhao. "Application and progress of artificial intelligence technology in the field of distribution network voltage Control : A review", Renewable and Sustainable Energy Reviews, 2024.
[14]. D. Quillen, E. Jang, O. Nachum, C. Finn, J. Ibarz and S. Levine, "Deep Reinforcement Learning for Vision-Based Robotic Grasping: A Simulated Comparative Evaluation of Off-Policy Methods," 2018 IEEE International Conference on Robotics and Automation (ICRA)