Space Robot Target Intelligent Capture System Based on Deep Reinforcement Learning Model

Liang Binyan1*, Chen Zhihong1, Guo Meishan1, Wang Yao1, Wang Yanbo1
Beijing Research Institute of Precise Mechatronics and controls, Beijing, China
liang_by@foxmail.com

Abstract. There are many on-orbit capture tasks for space robots. At present, most space robots capture methods are based on the trajectory planning of robot kinematics. This kind of method has poor control precision in space environment. The intelligence degree of robot capture task is very low. We built a simulation environment for robot space target capture task based on physics engine. A real-time online simulation training platform is established in the simulation environment. We design a robot deep reinforcement learning motion control model based on Actor-Critic algorithm. We optimize the reward function of the DRL model. The reward function shortens the training time and improves the score performance of the model. The experiment data show that the DRL model converges in 800 steps. The average score and standard deviation of the model indicate that the model has successful completed the capture task of space robot.

1. Introduction
In recent years, there are more and more target capture tasks such as spacecraft maintenance, in-orbit assembly, satellite filling and recovery. These tasks put forward higher requirements for the control of space robots. In the aspects of intelligence, dynamic and environmental adaptability, the existing control algorithms are difficult to meet the requirements of space capture tasks.

The existing space robot control algorithm mostly relies on robot teaching control. The space robot can capture the spatial position of the target according to the measurement of a vision sensor. The robot generates the trajectory of the manipulator arm through the kinematics algorithm. Because of its fast speed and high control precision, this method is widely used in manipulator motion control. Xu et al. [1] designed an anti-interference automatic control method for the space manipulator. This method realized the intelligent motion control of space robot through their adaptive motion control algorithm. W Xu et al. [2] proposed a new robot model method for the capture task of space robotic. They optimized the objective function by using genetic algorithm, which effectively reduced the influence of space floating on the space robot capture task. Teveira et al. [3] established the operation control model of the space capture manipulator by using neural network. They took the uncertainty in the model into account and optimized the entire capture task.

The space environment is extremely complex. And the space environment has the following characteristics: 1) The complex space light environment is different with light on the ground. The recognition accuracy and stability of the vision sensor bring difficulties to the capture task. 2) The space temperature difference is large. In order to ensure that the robot can both work in high and low temperature, robot structure must be designed to large margin. The large structure margin leads to low accuracy in robot control. 3) The radiation of space particles is complex, which may cause the control failure of the robot. In the case of the robot failure, the control algorithm cannot continue to execute the capture task.
In view of the above problems, using artificial intelligence method to solve the robot capture task, has become a new trend. Artificial intelligence method has a certain degree of fault-tolerant ability. The AI method does not need very high joint control precision. [4] proposed a method based on reinforcement learning to construct the agent control model. The RL method can complete the autonomous operation task of the agent. [5] built a neural network models based on the DDPG algorithm. They proposed the SAC reinforcement learning training model to accelerate the time of training convergence. The method can complete the tasks such as robot assembly and handling. [7] built a reinforcement learning model based on TD3 algorithm. The obstacle avoidance robot control model is built to realize the safety capturing operation.

In this paper we design a robot deep reinforcement learning model that solves the space capturing problems. Our contributions can be summarized as follows:

• We design a simulation environment of robot space target capture task based on physics engine. Through online real-time simulation robot capture operation, we completed the training of the deep reinforcement learning model.

• We build the robot reinforcement learning model based on Actor-Critic algorithm, and realize the intelligent control of the space robot.

• We designed a special reward function for the robot capture space target. And we trained the DRL model to convergence through analysis and experiment.

2. The CNN model design

The environment of space robot task is difficult to simulate on the ground. We establish a real-time online simulation training platform. The training platform is based on the physics engine. So the simulation conditions are very close to the real space environment. At the same time, we build a real-time online training system, so that the reinforcement learning model of the robot can interact with the environment in real time. The real-time online training system ensures the rapidity of training and reduces the convergence time.

![Diagram](image)

Figure 1. The real-time online simulation training platform

The real-time online simulation training platform consists of three parts: real-time physical simulation module, reinforcement learning model training module and training data management module. The real-time physics simulation module accepts the action data of the reinforcement learning model. It generates the state data and training flag according to the physics engine calculation. On the
one hand, the state data is returned to the reinforcement learning training module to generate the next action output. On the other hand, the state data of the robot is sent to the training data management module. the training data management module store and distribute state data. Then the training data set is sent to the reinforcement learning module through data sampling functions. The real-time online simulation training platform is shown in the figure1 below.

3. Space Robot DRL Capture Model

The space robot reinforcement learning capture model is composed of three parts[8-10], namely Value Network $V(s)$, Policy Network $P(s|a)$ and Q Network $Q(s|a)$. The Network structure of reinforcement learning model is shown in the Figure 2 below. $s$ is the state space. The state space contains the angular of each joint of the robot, the Euclidean distance of robot tool and the target, the last action of the robot, and the task end flag. $a$ is the action space, which is the angle movement of each robot joint. The input layer is a layer constructed by $s$ and $a$. The structure of the robot’s reinforcement learning capture model is shown in the Figure 2 below.

The objective optimization Value Network is defined as follows:

$$J_V = E \left[ \frac{1}{2} (V(s) - E[Q(s|a)]^2) \right]$$

(1)

Then the Q Network is defined as follows:

$$J_Q = E \left[ \frac{1}{2} (Q(s|a) - \hat{Q}(s|a))^2 \right]$$

(2)

The Policy Network outputs the mean $\mu(s)$ and log variance $\log(\sigma(s))$ of the Gaussian distribution of actions[11]. The objective optimization function of Policy Network is defined as:

$$J_P = E \left[ \log(P(s) + \varepsilon(s)) - (Q(s) + \varepsilon(s)) \right]$$

(3)

Where, $\varepsilon(s)$ is a neural network noise output.

The pseudocode of the training process of reinforcement learning model of robot is as follows:

**Algorithm 1** Space robot capture model

| Initial parameters $V P Q$ |
|---------------------------|
| **For** each episode **do** |
| **For** each step **do** |
| $a^* \sim P(s|a)$ |
| Generate action |
Generate state

\[ s' \sim \mathcal{T}(s|a) \]

\[
D \leftarrow D \cup \{(s, a, r(s, a), s')\}
\]

Update dataset

\[ s \leftarrow s \]

\[ \text{End For} \]

For each gradient step do

\[
V \leftarrow V - \beta V J V
\]

\[
P \leftarrow P - \beta P J P
\]

\[
Q_i \leftarrow Q_i - \beta Q_i J Q_i \text{, for } i \in 1, 2
\]

\[
\dot{V} \leftarrow \tau V -(1-\tau)V
\]

End For

End For

4. Model Optimization

The reward function optimization is the key point of the Robot reinforcement learning model training. The reward function determines the direction of the model parameter adjustment. The better reward function is, model training time is shorter and task success rate is higher. Bad reward function may lead the no convergence of the model. Before we optimize the reward function, we define some parameters as follows:

\[ x, y, z \] is the position error vector of the target coordinate to the tool coordinate[12].

\[ \alpha, \beta, \gamma \] is the attitude error vector of the target coordinate to the tool coordinate.

\[ d = \sqrt{x^2 + y^2 + z^2} \] is the Euclidean distance of the target coordinate to the tool coordinate.

\[ \text{rot} = \sqrt{\alpha^2 + \beta^2 + \gamma^2} \] is the second-order norm of the Euler Angle of the target coordinate to the tool coordinate.

First, the Euclidean distance of the tool to the target and the second-order norm of the Euler Angle were used to measure the reward. The physical meaning is that the closer the end tool is to the target, the higher the goal will be. And the smaller the absolute value of the Euler Angle, the higher the goal will be.

\[ \text{reward}_1 = (2-d)^2 + (1-d/2)(\sqrt{3}\pi - \text{rot})^2 - 33.6 \] \hspace{1cm} (4)

However, we found that it was difficult for the model to converge in this way. Because when the tool coordinate system was far away from the target object, the distance had the greatest influence. The angle had very little influence on the reward. However, in reward function 1, the two effects on the reward were equivalent, resulting in that the reward function could not correctly reflect the movement trend of the robot.

Then we designed a segmentally reward function. That is, when the tool coordinate system is far from the target coordinate system, only \( d \) is used to construct the reward function [13-15]; when the tool coordinate system is close to the target, the Angle is assumed into the reward function. In this way, the model quickly converges to the pre-capture position.

\[ \text{reward}_2 = \begin{cases} 
-2d - 0.1r - 1.9, & d \geq 0.1 \\
-(e' - 1) - 10d^2, & d < 0.1 
\end{cases} \] \hspace{1cm} (5)

This kind of reward function brings a problem that the model can get close to the target very quickly. But the angle adjustment takes a long time, and the convergence time of the whole model becomes very long. Through analysis, it is found that there is a certain error in the rot using. The measurement for angles and positions are not consistent. Then we designed the reward function based on the trigonometric function of attitude Angle:
This function accelerates the convergence time. Reward function 3 unifies the evaluation criteria of distance and Angle through trigonometric function transformation. That makes the model converge and gets a higher score, which successfully solves the optimization problem.

5. Experiment

This paper tests the training process of three kinds of reward functions, and calculates the convergence steps of reward functions. The model score and the stability of the model is counted. The reward curves of the three reward functions are shown in the Figure 3 below.

In the figure above, the curve smooth factor is 0.8, and about 2k training steps were counted. According to the data, we counted the number of convergent steps of each reward function. The mean score of the reward function was $\mu$, and the stability of the reward function was $std$. The obtained statistical data are shown in the Table 1 below.
Table 1. Optimize Function Performance.

| Reward Function | Number of Convergence Steps | Mode Mean Score | Mode Stability |
|-----------------|-----------------------------|-----------------|---------------|
| Reward 1        | >2.5k                       | -627.1          | 133.51        |
| Reward 2        | 1.2k                        | -493.8          | 82.49         |
| Reward 3        | 0.8k                        | -162.4          | 67.17         |

It can be seen from the table that the convergence times of reward function 3 are the least, with 800 converges. While the reward function 1 requires more than 2.5k steps to converge. The average score of reward function 3 was the highest (-162.4). And the standard deviation of reward function 3 was the least 67.17. Therefore, it can be concluded that reward function 3 has the fastest convergence, the highest score of the model and the highest stability. In this paper, reward function 3 is finally adopted in the reinforcement learning model of the space robot.

The final space robot target capture system is shown in Figure 6.

Figure 6. Space robot target capture system

6. Conclusion
In this paper we design a robot deep reinforcement learning model that solves the space capturing problems. We build a simulation environment of robot space target capture task. Through online real-time simulation robot capture operation, we completed the training the robot reinforcement learning model based on Actor-Critic algorithm. We optimize a special reward function for DRL model. The experiment data show that the robot deep reinforcement learning model converges very quickly. The space robot deep reinforcement learning model has successfully performed the space capture task.

For future work, we hope to use more effective models to achieve the space robot capture tasks. We plan to improve our simulation environment by using more efficient algorithm [16-17]. And we will move forward to search for methods to speed up the DRL model training time.

Acknowledgments
Thanks for the support of the research work by Beijing Research Institute of Precise Mechatronics and controls and China Academy of Launch Vehicle Technology (CALT). The authors would like to acknowledge the support of the team members for their outstanding contributions to the paper.

References
[1] Xu, Y., Shum, H. Y., Kanade, T., & Lee, J. J. (1994). Parameterization and adaptive control of space robot systems. Aerospace and Electronic Systems, IEEE Transactions on.
[2] Xu, W., Liu, Y., Liang, B., & Qiang, X. W. (2008). Autonomous path planning and experiment study of free-floating space robot for target capturing. Journal of Intelligent & Robotic Systems.

[3] Taveira, T. D. F. P. A., Siqueira, A. A. G., & Terra, M. H. (2006). Adaptive nonlinear $H_\infty$ controllers applied to a free-floating space manipulator. Computer Aided Control System Design.

[4] Cai, J. (2020). WD3-MPER: A Method to Alleviate Approximation Bias in Actor-Critic. Neural Information Processing.

[5] Aslani, M., Mesgari, M. S., & Wiering, M. (2017). Adaptive traffic signal control with actor-critic methods in a real-world traffic network with different traffic disruption events. Transportation Research Part C: Emerging Technologies, 85(dec.), 732-752.

[6] Chen, T., Ma, X., You, S., & Zhang, X. (2019). Soft Actor-Critic-Based Continuous Control Optimization for Moving Target Tracking. Image and Graphics.

[7] El-Shamouty, M., Wu, X., Yang, S., Albus, M., & Huber, M. F. (2020). Towards Safe Human-Robot Collaboration Using Deep Reinforcement Learning. IEEE International Conference on Robotics and Automation (ICRA). IEEE.

[8] Geng, X., Zhang, M., Bruce, J., Caluwaerts, K., & Levine, S. (2016). Deep reinforcement learning for tensegrity robot locomotion.

[9] Long, P., Fan, T., Liao, X. (2018). Towards Optimally Decentralized Multi-Robot Collision Avoidance via Deep Reinforcement Learning. 2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE.

[10] Anschel, O., Baram, N., & Shimkin, N. (2016). Averaged-dqn: variance reduction and stabilization for deep reinforcement learning.

[11] Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. arXiv preprint arXiv:1801.01290, 2018.

[12] Kapturowski, S., Ostrovski, G., Quan, J., Munos, R., & Dabney, W. (2019). Recurrent Experience Replay in Distributed Reinforcement Learning. ICLR.

[13] Liang, B., Wang, Y., Wang, Y., Chen, Z., & Lin, J. (2020). Garbage sorting system based on composite layer cnn and multi-robots. Journal of Physics: Conference Series, 1634(1), 012083 (8pp).

[14] Binyan, L., Yanbo, W., Zhihong, C., Jiayu, L., & Junqin, L. (2017). Object detection and robotic sorting system in complex industrial environment. Chinese Automation Congress (pp.7277-7281).

[15] Liang, B., Li, T., Chen, Z., Wang, Y., & Liao, Y. (2018). Robot Arm Dynamics Control Based on Deep Learning and Physical Simulation. 2018 37th Chinese Control Conference (CCC).

[16] Ficuciello, F., & Siciliano, B. (2016). Learning in robotic manipulation: the role of dimensionality reduction in policy search methods: comment on "hand synergies: integration of robotics and neuroscience for understanding the control of biological and artificial hands" by marco santello et al. Physics of Life Reviews, 36-37.

[17] Qureshi, A. H., Nakamura, Y., Yoshikawa, Y., & Ishiguro, H. (2017). Robot gains social intelligence through multimodal deep reinforcement learning.