DOI: 10.13009/EUCASS2019-33

A New Spacecraft Attitude Stabilization Mechanism Using Deep Reinforcement Learning Method

Yuejiao Wang, Zhong Ma, Yidai Yang, Zhuping Wang and Lei Tang***
**Xi'an Microelectronics Technology Institute, China Aerospace Science and Technology Corporation (CASC)*
No.198, Taibai Road, Xi'an 710065, China, wang_yuejiao@126.com
*** Xi'an Microelectronics Technology Institute, China Aerospace Science and Technology Corporation (CASC)*
No.198, Taibai Road, Xi'an 710065, China, mazhong@mail.com; y88y01d07@sina.com; zxjwl@126.com; tanglei@163.com

Abstract

Aiming at the problem of sudden changes in the attitudes and the mass parameters encountered by spacecrafts when performing complex tasks such as discarding a payload or capturing a target, this paper proposes a new attitude stabilization mechanism based on the deep reinforcement learning method to restore the spacecraft to a stable attitude. Specifically, a three-dimensional space simulation environment which simulates the spacecraft's attitude is firstly established according to the real-time control torque. A neural network model based on the segmented weighted reward function is then built, which takes the attitude of the spacecraft as input, and outputs the discretized control torque to control the spacecraft. Moreover, the Deep Q Network algorithm performs the attitude stabilization training for the spacecraft in the simulation environment, the parameters of the neural network model are continuously updated in the training phase. Simulation experiments illustrate that, by continuous self-learning and self-evolution, deep reinforcement learning method gradually learns to re-stabilize spacecraft's attitude after unknown disturbance. As a contrast, we compare the proposed method with PD controller and backstepping controller. The PD controller is unable to re-stabilize the attitude due to its dependence on the mass parameters. The backstepping controller has robustness against mass parameter uncertainty, but it can only handle the constant control cycle. Compared with two controllers, our attitude stabilization mechanism based on deep reinforcement learning has competitive performance for uncertainties in the mass parameters, while allows the control cycle changes during the training phase. The proposed mechanism is an innovative application of artificial intelligence technology in attitude stabilization field, which provides support for achieving the intelligent control and makes the technical foundation for completing the service and maintenance of spacecrafts on-orbit. For future work, we plan to train and validate the mechanism on a semi-physical simulation platform consisting of a three-dimensional turntable and an actuator under the complex space simulation environment.

1. Introduction

Good attitude control methods are critical to the stable orbital operation of spacecraft. In-orbit operation of spacecraft, due to long-term consumption of fuel, changes in spacecraft configuration (such as the deployment of solar panels or some large antennas), orbital capture and release of loads (such as the release of satellites from spacecraft; capture) Targets, removal of orbital debris, etc., and docking with other spacecraft, etc., will cause changes in the motion state and quality characteristics of the system, and many changes are dramatic (such as capturing and releasing satellites; docking with targets, etc.) Know (such as the operation of non-cooperative targets, the clean-up of orbital garbage, etc.). Most of the existing attitude control algorithms rely on the quality parameters of the controlled object [1] (including mass, moment of inertia, etc.), and the quality parameters need to be identified by various means. In this case, it is difficult to give accurate parameter identification, and Such system dynamics models are complex and have strong nonlinearities, which easily lead to the failure of existing attitude control systems. Therefore, there is an urgent need for a highly autonomous attitude control technology with a high degree of intelligence to solve the problem of high-performance spacecraft control in the case of in-orbit changes in spacecraft quality characteristics that are difficult to deal with.

DOI: 10.13009/EUCASS2019-33

In view of the attitude control problem in such complex situations, Yoon H et al. proposed a new control law for nonlinear Hamiltonian MIMO systems for the uncertainty of inertia in spacecraft attitude control [2]. Queiroz M S D et al. designed a nonlinear adaptive control using the dynamic model of the complete system and proved the global asymptotic convergence of the tracking error of the closed-loop system when the interference is unknown [3]. Miao Shuangquan solved the vibration problem in the maneuvering process of large flexible spacecraft by using an adaptive sliding mode control strategy [4]. In general, the algorithms for attitude control of spacecraft are currently less intelligent and are usually designed for specific applications and do not have universality. Therefore, with the increasing complexity of space exploration missions, it is necessary to design a posture control technology with a high degree of intelligence.

Deep reinforcement learning is a technique for learning control strategies directly from high-dimensional raw data [5], and DQN (Deep Q Network) is one of the typical algorithms for deep reinforcement learning. It combines deep learning with reinforcement learning to achieve an entirely new algorithm for end-to-end learning from perception to action. DQN's input is the original image data as state, and the output is the value evaluation (Q value) corresponding to each action [6]. In 2013, Google's DeepMind team presented the DQN algorithm at the NIPS Deep Learning Seminar [7]. By directly inputting the original video of the game for intensive learning, the 6 models of the 7 games in the Atari game platform exceeded the previous algorithms, and 3 of them exceeded the human level, demonstrates the enormous potential of such algorithms for intelligent decision making.
This paper intends to use the deep reinforcement learning algorithm to solve the problem of spacecraft intelligent attitude stabilization, and propose a mechanism of spacecraft intelligent attitude stabilization through autonomous learning. Breaking through the existing methods relies on the limitations of the controlled object with complex dynamic models and strict quality parameters, solves the problem of spacecraft attitude instability caused by sudden random disturbances, and improves the attitude stability and control accuracy of the attitude control algorithm.

2. Spacecraft Attitude Stabilization Mechanism

2.1 Construction of Overall design process

The spacecraft attitude stabilization problem addressed in this paper is defined formally as follows:

When a spacecraft performs complex tasks such as discarding a payload or capturing a target, it will encounter a sudden disturbance of the attitude and the mass parameters, causing unstable flying and rolling of the spacecraft. In such circumstances, the change of the movement and mass characteristics are unpredictable. How to operate the control torque to stabilize the attitude of the spacecraft as an initial state?

In order to reproduce this scene and solve the problem, the overall design process is divided into two stages. The first one is to establish a three-dimensional space simulation environment which simulates the spacecraft's attitude using the visualization software. The inputs of the simulation environment are the control torque of the spacecraft, and the outputs are the attitude of the spacecraft. The second one is to establish a fully connected neural network. Using the classical deep reinforcement learning algorithm DQN, to perform the intelligent autonomous attitude stabilization training for the spacecraft in the simulation environment [11]. Taking the attitude of the spacecraft as input, the weight parameters of the neural network model are obtained by using the DQN algorithm. The control torque required for the spacecraft is intelligently output, and the attitude of the spacecraft is sent into the simulation environment again. It continues to be input into the neural network for continuous deep reinforcement training. By continuous self-learning and self-evolution, the weight value of the neural network is constantly updated. The interactive process of the simulation environment and the fully connected neural network is shown in Figure 1.

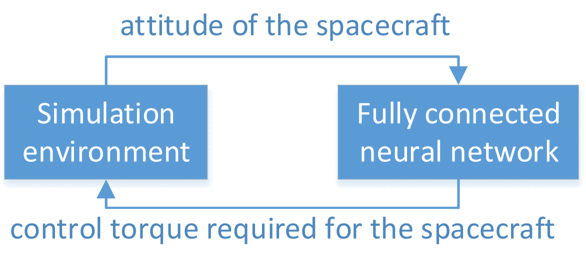

attitude of the spacecraft
Simulation environment
Fully connected neural network
control torque required for the spacecraft

Figure 1: The interactive process of the simulation environment and the neural network

The two important steps are described separately as the simulation environment and the training algorithm. The design process is shown in Figure 2.

DOI: 10.13009/EUCASS2019-33

The visualization software was used to create a three-dimensional space visualization environment simulation platform for the spacecraft and its cameras. In the three-dimensional space simulation environment, the intelligent controller gives the desired control torque output of the spacecraft attitude stabilization system based on the measurement information; the actuator model uses this as an input to give the actual torque output of the actuator under the actual situation; the sum of the actual torque and the external disturbance is the external torque of the spacecraft, is calculated by the attitude dynamics. The attitude angular velocity and attitude angle of the spacecraft derived from the attitude dynamics model provides the required input information for the intelligent controller; and the three-dimensional visualization module simulators the attitude of the spacecraft according to the real-time attitude input maneuver.

In the DQN training algorithm, we built a fully connected neural network as the intelligent agent, which takes the attitude of the spacecraft as input, and outputs the control torque required for the spacecraft. The input layer of the fully connected neural network has 14 nodes, corresponding to the 14-dimensional representation of the spacecraft's attitude. The hidden layer has two layers, the first layer has 512 nodes, and the second layer has 1024 nodes. The output layer has 7 nodes, corresponding to the seven types of value vectors after the control torque is discretized. Weight parameters of the network are obtained by using the DQN. At each time step, the control torque is sent back into the simulation environment, and the environment continues to output the attitude of the spacecraft to feed the network for continuous deep reinforcement training.

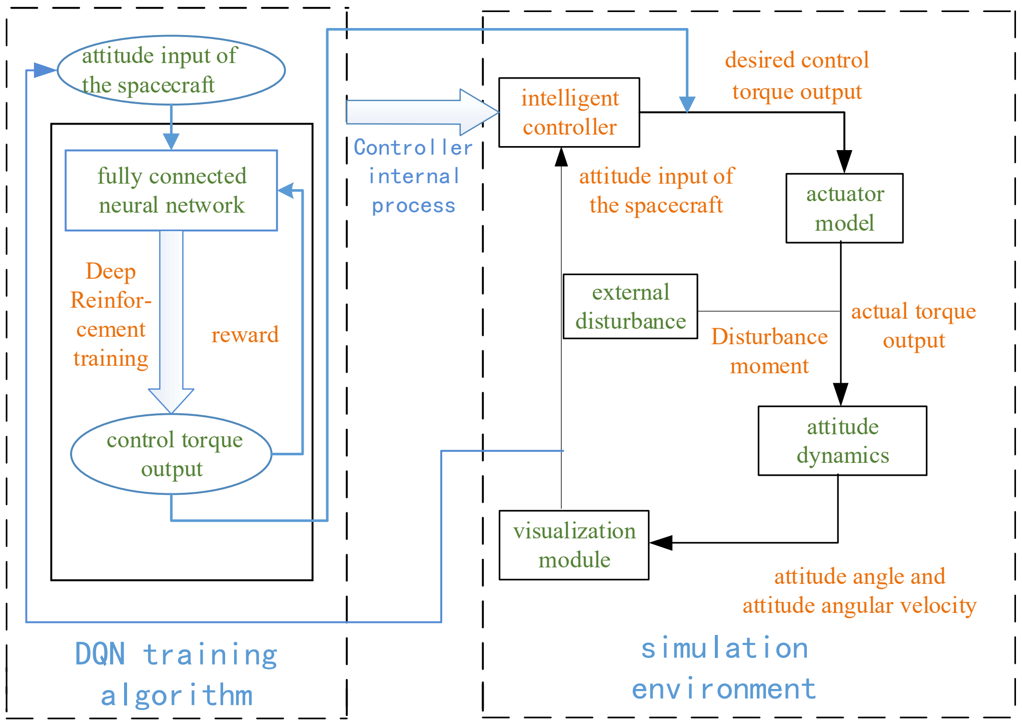

attitude input of the spacecraft
fully connected neural network
Deep Reinfor-cement training
reward
control torque output
Controller internal process
intelligent controller
desired control torque output
attitude input of the spacecraft
actuator model
external disturbance
Disturbance moment
actual torque output
attitude dynamics
visualization module
attitude angle and attitude angular velocity
DQN training algorithm
simulation environment

Figure 2: Sketch map of space debris capturing method based on deep reinforcement learning

2.2 Establishment of Dynamic Environment

It is assumed that the spacecraft is established on the orbital coordinate system. In order to study the dynamic model of the spacecraft, in order to describe the attitude of the spacecraft, correct attitude dynamics and kinematics model must be established.

（1）Constructing Attitude dynamics model

The dynamic model of the spacecraft can be described by the Euler dynamic equation of a single rigid body as follows:

$$\dot{\omega}=I^{-1}\left(T-\omega\times\left(I\cdot\omega\right)\right) \quad \textbf{(1)}$$

Where T is the control torque acting on the centroid of the rigid body, I is the moment of inertia matrix of the rigid body, $\omega=[\omega_x,\omega_y,\omega_z]^T$ is the attitude angular velocity of the rigid body. If the initial value of the attitude angular

DOI: 10.13009/EUCASS2019-33

velocity is known as ω_0, and set control torque T to a fixed value, it is possible to obtain the attitude angular velocity at any time by solving the upper formula.

（2）Constructing Attitude kinematics model

Because the attitude angle of the spacecraft can be described by the attitude quaternion, the quaternion is used to characterize the change of the attitude of the spacecraft that the human eyes can observe. The following equation is the quaternion-based attitude kinematics equation of the spacecraft, $\omega=[\omega_x,\omega_y,\omega_z]^T$ is the attitude angular velocity. If the spacecraft is known at its initial attitude quaternion Q_0, the attitude of the spacecraft can be represented at any time by integrating.

$$\begin{bmatrix} \dot{q}_0 \\ \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & -q_3 & q_2 \\ q_2 & q_3 & q_0 & -q_1 \\ q_3 & -q_2 & q_1 & q_0 \end{bmatrix} \begin{bmatrix} 0 \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{2}$$

（3）Constructing dynamic environment

According to the dynamics model, the simulation environment is constructed as follows:

Step1: Randomly initialize the attitude angular velocity ω_0 and attitude quaternion Q_0 of the spacecraft;

Step2: Network model gives the control torque T of the spacecraft;

Step3: Equations (1)-(2) are integrated sequentially to solve the attitude angular velocity ω_i and attitude quaternion Q_i, circulatory output the final value (ω, Q) of the spacecrafts;

Step4: The initial attitude of the spacecraft is input into the neural network for subsequent deep reinforcement learning training.

2.3 Discretization of Control Torque

DQN is a discrete control-oriented algorithm, that is, the output of the action is discrete, corresponding to the Atari game, only a few discrete keyboard or handle buttons to control. DQN can't face continuous action, because the update of Q value needs to be achieved by seeking the largest Action. However, in the problem of spacecraft attitude stabilization that needs to be resolved, the output of the control torque is continuous and high-dimensional and cannot be solved using the traditional DQN method. Therefore, the output control torque is discretized here.

The control torque of the spacecraft is assumed to be a three-dimensional vector $T \in R^3$, with a possible value range $1.0e-02*\{-1,0,1\}$ of each direction component in $T=[T_x,T_y,T_z]^T$, that is only one direction component of control torque has only one certain value in the set of $1.0e-02*\{-1,0,1\}$ for each maneuver, and the other components have a value of zero. There are 7 kinds of torque distribution methods according to the above range of values, with the flag vector for each of which set as $[1,0,0,0,0,0,0]$、$[0,1,0,0,0,0,0]$、$[0,0,1,0,0,0,0]$、$[0,0,0,1,0,0,0]$、$[0,0,0,0,1,0,0]$、$[0,0,0,0,0,1,0]$、$[0,0,0,0,0,0,1]$, respectively. These vectors can represent the control torque for an iterative update of the Q value. Among them, $[1,0,0,0,0,0,0]$ represents that the control torque of the agent' output is zero, that is, the spacecraft has no external torque applied, and only relies on its original angular velocity to continue to rotate. The correspondence between the control torque and flag vector is shown in the following table.

Table 1: Discretization of control torque

Direction component of control torque T	**Flag vector**
$T_x = T_y = T_z = 0$	$[1,0,0,0,0,0,0]$
$T_x = -1.0e-02,\ T_y = T_z = 0$	$[0,1,0,0,0,0,0]$
$T_x = 1.0e-02,\ T_y = T_z = 0$	$[0,0,1,0,0,0,0]$
$T_x = 0,\ T_y = -1.0e-02,\ T_z = 0$	$[0,0,0,1,0,0,0]$

DOI: 10.13009/EUCASS2019-33

$T_x = 0,\ T_y = 1.0e-02,\ T_z = 0$	$[0,0,0,0,1,0,0]$
$T_x = T_y = 0,\ T_z = -1.0e-02$	$[0,0,0,0,0,1,0]$
$T_x = T_y = 0,\ T_z = 1.0e-02$	$[0,0,0,0,0,0,1]$

2.4 Design of Reward Function and Terminating Condition

The goal of deep reinforcement training is to output the optimal control torque of the spacecraft, so that the deviation between the initial attitude angular velocity ω_0 and the spacecraft's attitude angular velocity ω_i obtained through the dynamic model is almost zero. However, the goal of the training is to get as much reward as possible. Therefore, the reward function needs to have the nature of a decreasing function, where the smaller the angular velocity difference (defined as ω_i - ω_0) is, the greater the reward. In this paper, we use the Gaussian function to construct the reward function:

$$g\left(\omega_i - \omega_0\right) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}\left(\omega_i - \omega_0\right)^2} \tag{3}$$

We have also set the terminating condition of the training process of the DQN algorithm. The criteria of whether to complete the training is dependent on whether the torque can control the spacecraft to restore to a stable attitude. Here, we calculate the deviation between the actual attitude angular velocity and the desired attitude angular velocity of the spacecraft at each iteration, when the deviation falls into a predefined range, the training process is terminated. The predefined range is the deviation of the spacecraft's attitude angular velocity in the three-directional components being less than 10^(-6).

2.5 Establishment of Deep Reinforcement Training

Intelligent attitude stabilization of the spacecraft is a complex, high-dimensional issue. The neural network is regarded as the control agent which decides how much control torque is to be given based on the current attitude of the spacecraft. Using a fully connected neural network as the calculation component, taking the spacecraft 's current attitude angular velocity and attitude quaternion as input, outputting a value indicating the probability of the decision size (e.g. a certain control torque), each iteration will sample the actual movement to be performed from this distribution. In simple terms, using a probabilistic strategy for sampling action decisions, action decisions that happen to lead to good output will been couraged in the future, and action decisions that lead to bad results are suppressed. In summary, the deep reinforcement training process based on DQN is as follows:

Table 2: Control torque training process based on DQN algorithm

DQN algorithm flow:
1. Initialize the capacity of the experience pool D for N, which is used to store training samples; 2. Use a deep neural network as the Q-value network to initialize weight parameters θ; 3. Set the total number of control task training as M, loop start: Initialize the network input state x_1, and calculate network output a_1. 1） Randomly select the action a_t with probability epsilon (decreasing with the number of iterations) or the maximum Q value $\arg\max_a Q\left(x_t, a; \theta\right)$ output through the network; 2）After performing a_t in the environment, get reward r_t and input x_{t+1} for the next network; 3）Save the parameter vector $\left(x_t, a_t, r_t, x_{t+1}\right)$ as D at the moment (D holds the state of N moments); 4）When D accumulates to a certain degree, the minibatch states are randomly taken out of D after each execution of 1-3 steps; 5）Calculate the target value for each state $\left(x_j, a_j, r_j, x_{j+1}\right)$:

DOI: 10.13009/EUCASS2019-33

$$y_j = \begin{cases} r_j, x_{j+1} \textit{ terminates the task} \\ r_j + \gamma \max_{a'} Q\left(x_{j+1}, a'; \theta\right), x_{j+1} \textit{ does not terminate the task} \end{cases};$$

6) The network weight parameter is updated through SGD and the loss function is defined using the mean squared error $\left(y_j - Q\left(x_j, a_j; \theta\right)\right)^2$.

Loop execution of the above 1-6 steps, continuous training model.

4. Multiple training to get the model.

3. Numerical Simulation and Results Analysis

To verify the effectiveness of the proposed autonomous attitude stabilization method, simulation tests were performed in this section. Firstly, we gave a detailed description of the simulation environment. Then, we establish a three-dimensional visualization environment of the spacecrafts using the visualization software Unity. Specifically, to simulate the tasks such as target capture or payload release performed by spacecrafts, an instantaneous burst random disturbance torque is applied to the attitude of spacecraft based on the above dynamic model, and the moment of inertia has randomly been changed to simulate changes in spacecraft's quality parameters at the same time. The proposed method should continuously output control torque in this state to control the spacecraft to restore a stable flight attitude.

3.1 Description of the Simulation Environment

In the above-mentioned intelligent attitude stabilization mechanism, the visualization software simulates the dynamic attitude of the spacecraft under external control. The deep reinforcement training based on DQN provides the external control for the spacecraft.

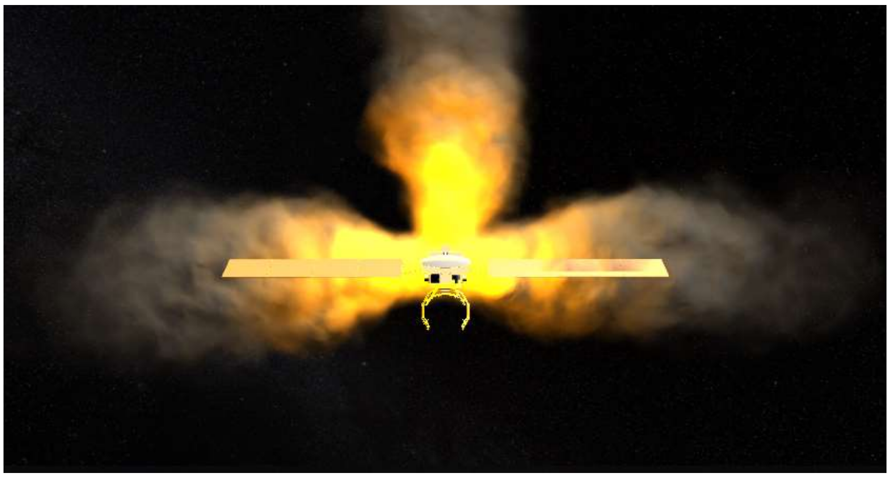
Before the start of each simulation, the spacecraft initial attitude quaternion and angular velocities are initialized randomly, and the entire spacecraft attitude maneuvering process is observed based on sensor acquisition information. When the attitude quaternion and angular velocity deviation no longer decrease with time, the attitude maneuvering process is considered to be finished, and the attitude accuracy and stability of the attitude stabilization system are evaluated based on the final angular velocity deviation.

First, we randomly initialize the attitude angular velocity ω_0, attitude quaternion Q_0 and the moment of inertia matrix I of the spacecraft. Then, an instantaneous burst random disturbance torque which simulates the tasks such as target capture performed by spacecraft is defined as TR. The expected experiment results should be, after this disturbance, the attitude stabilization mechanism continuously outputs the control torque, so that the spacecraft's attitude angular velocity can converge to a certain value, and the deviation between this value and the desired attitude angular velocity tends to zero, indicating that the attitude stabilization mechanism is effective. For comparison, this paper also tests the traditional attitude stabilization method based on PD controller and backstepping controller. The mechanism is implemented using the Anaconda3 software package and TensorFlow deep learning software framework.

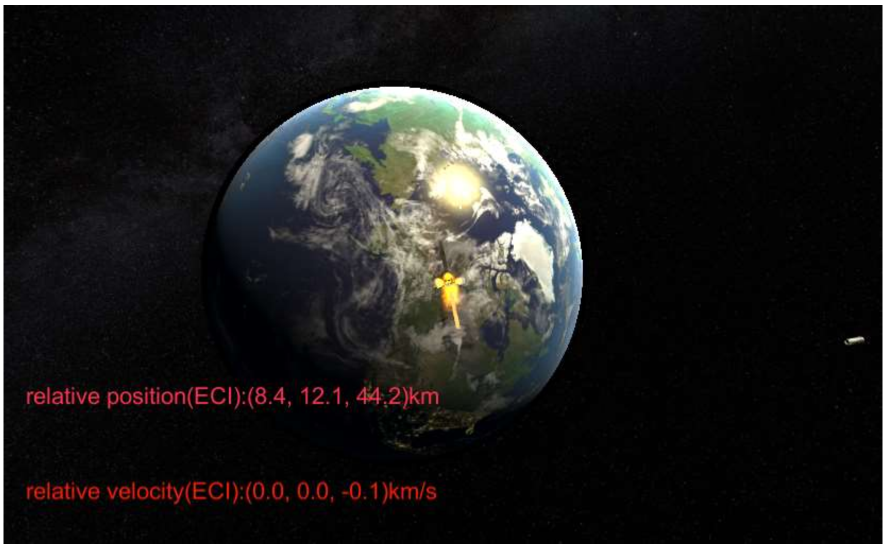
3.2 Three-Dimensional Visualization Environment

The attitude stabilization mechanism utilizes the visualization software Unity to simulate the mechanical characteristics of the spacecrafts, and uses Python programming to perform deep enhancement training, and establishes a three-dimensional visualization environment of the spacecrafts. The input of the environment is continuous control torque along the three axes of the spacecraft body coordinate system, the output is the attitude information of the spacecrafts at the next moment, and the output is visualized. The 3D visualization environment is shown in Figure 3. The environment simulates the dynamic state of the spacecrafts under additional control and disturbance actions. (a) provides an analysis engine for calculating data and displaying various forms of 2D maps. The ability is to generate position and attitude data, such as the instantaneous relative position (8.4, 12.1, 44.2) km and speed (0.0, 0.0, -0.1) km/s of an spacecraft and another spacecraft generated in (b), generated in (c) the instantaneous attitude angular velocities of the two spacecrafts are (0.0, 0.0, 0.0) rad/s and (0.0, 0.0, 7.0) rad/s, respectively, which can visualize the input and output of the attitude stabilization system, and can be directly artificial at any time. Interfering with the attitude of the spacecraft provides a good training environment for the intelligent attitude stabilization mechanism.
training environment for the intelligent attitude stabilization mechanism.

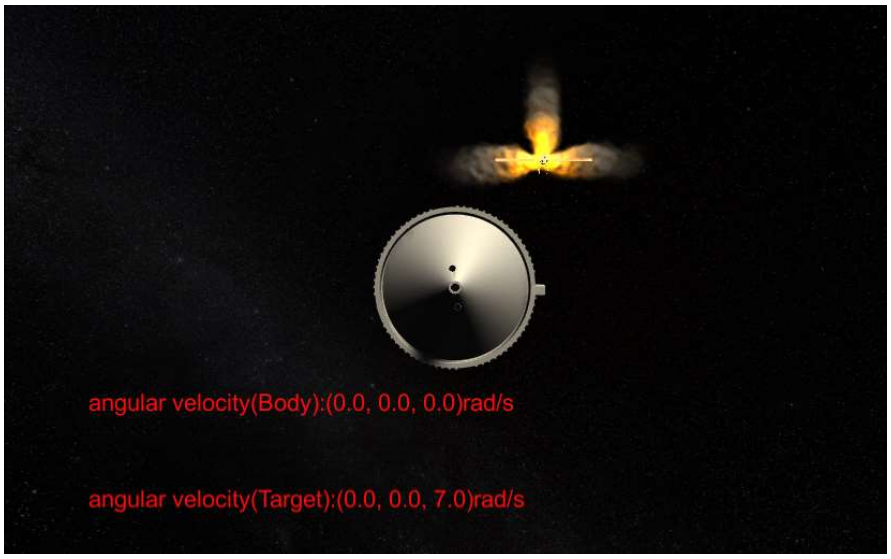
DOI: 10.13009/EUCASS2019-33



(a). the spacecraft with control


relative position(ECI):(8.4, 12.1, 44.2)km
relative velocity(ECI):(0.0, 0.0, -0.1)km/s

(b). Relative position and velocity of the two spacecrafts


angular velocity(Body):(0.0, 0.0, 0.0)rad/s
angular velocity(Target):(0.0, 0.0, 7.0)rad/s

(c) Attitude angular velocity of the two spacecrafts

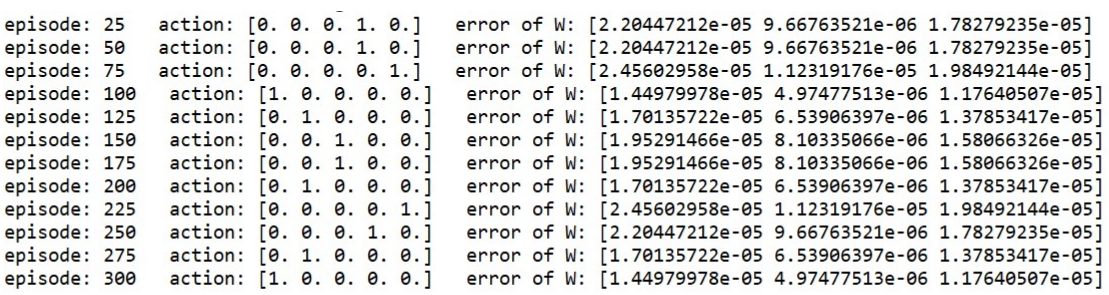
Figure 3: Three-dimensional visualization environment of the spacecrafts

DOI: 10.13009/EUCASS2019-33

3.3 Attitude Stabilization Experiment Based on DQN Training

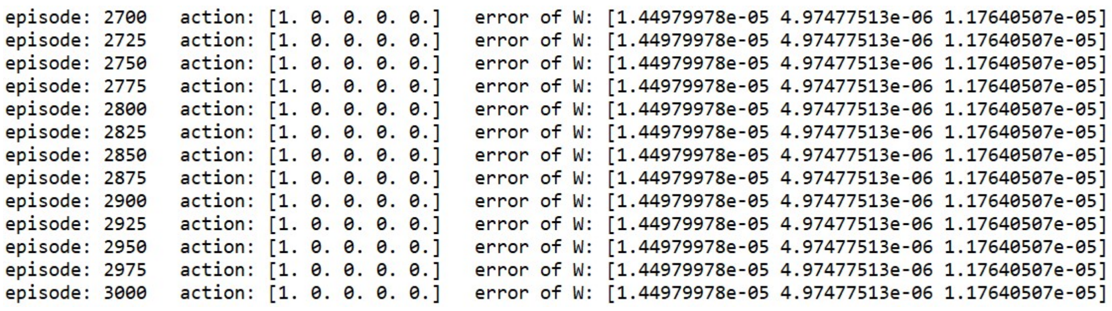
Specifically, define instantaneous burst random disturbance torque $TR = 1.0\text{e-}02 * [r1, r2, r3]^T$, where $r1, r2, r3$ are randomly generated numbers. Initialize the attitude angular velocity $\omega_0 = [0.001, 0.001, 0.001]^T rad/s$ and attitude quaternion $Q_0 = [1,0,0,0]^T$ of the spacecraft. I is randomly been changed by adding a lower order 3 × 3 random symmetric matrix based on a basic matrix $\begin{bmatrix} 2.0257 & 0.6498 & 1.1226 \\ 0.6498 & 0.7998 & 0.1833 \\ 1.1226 & 0.1833 & 1.2753 \end{bmatrix}$, each element in the random symmetric matrix is drawing from an uniform distribution with range of [0,1].

The number of iterations is defined as episode=3000, the capacity of the experience pool is 500, and the discount factor is 0.99. The initial value of the neural network weight parameter is 0.01. The moment of inertia matrix takes the random value satisfying the symmetry and the largest diagonal element. The deep reinforcement training based on the DQN algorithm is performed 25 times per iteration, that is, the error (deviation) of the control torque (action) and the error of the attitude angular velocity (error of W) are displayed every 25 episodes. The training result output is shown in Figure 4. It can be seen from the figure that in the initial, i.e. completely random 300 episodes, the action and the error of W are randomly changed, thereby storing the training samples and eliminating the correlation of the data. As of training to 3000 episodes, the action and error of W tend to be stable. Figure 5 is the output of the last 300 episodes of training.



```
episode: 25   action: [0. 0. 0. 1. 0.]   error of W: [2.20447212e-05 9.66763521e-06 1.78279235e-05]
episode: 50   action: [0. 0. 0. 1. 0.]   error of W: [2.20447212e-05 9.66763521e-06 1.78279235e-05]
episode: 75   action: [0. 0. 0. 0. 1.]   error of W: [2.45602958e-05 1.12319176e-05 1.98492144e-05]
episode: 100   action: [1. 0. 0. 0. 0.]   error of W: [1.44979978e-05 4.97477513e-06 1.17640507e-05]
episode: 125   action: [0. 1. 0. 0. 0.]   error of W: [1.70135722e-05 6.53906397e-06 1.37853417e-05]
episode: 150   action: [0. 0. 1. 0. 0.]   error of W: [1.95291466e-05 8.10335066e-06 1.58066326e-05]
episode: 175   action: [0. 0. 1. 0. 0.]   error of W: [1.95291466e-05 8.10335066e-06 1.58066326e-05]
episode: 200   action: [0. 1. 0. 0. 0.]   error of W: [1.70135722e-05 6.53906397e-06 1.37853417e-05]
episode: 225   action: [0. 0. 0. 0. 1.]   error of W: [2.45602958e-05 1.12319176e-05 1.98492144e-05]
episode: 250   action: [0. 0. 0. 1. 0.]   error of W: [2.20447212e-05 9.66763521e-06 1.78279235e-05]
episode: 275   action: [0. 1. 0. 0. 0.]   error of W: [1.70135722e-05 6.53906397e-06 1.37853417e-05]
episode: 300   action: [1. 0. 0. 0. 0.]   error of W: [1.44979978e-05 4.97477513e-06 1.17640507e-05]
```

Figure 4: Output data in the first 300 episodes



```
episode: 2700   action: [1. 0. 0. 0. 0.]   error of W: [1.44979978e-05 4.97477513e-06 1.17640507e-05]
episode: 2725   action: [1. 0. 0. 0. 0.]   error of W: [1.44979978e-05 4.97477513e-06 1.17640507e-05]
episode: 2750   action: [1. 0. 0. 0. 0.]   error of W: [1.44979978e-05 4.97477513e-06 1.17640507e-05]
episode: 2775   action: [1. 0. 0. 0. 0.]   error of W: [1.44979978e-05 4.97477513e-06 1.17640507e-05]
episode: 2800   action: [1. 0. 0. 0. 0.]   error of W: [1.44979978e-05 4.97477513e-06 1.17640507e-05]
episode: 2825   action: [1. 0. 0. 0. 0.]   error of W: [1.44979978e-05 4.97477513e-06 1.17640507e-05]
episode: 2850   action: [1. 0. 0. 0. 0.]   error of W: [1.44979978e-05 4.97477513e-06 1.17640507e-05]
episode: 2875   action: [1. 0. 0. 0. 0.]   error of W: [1.44979978e-05 4.97477513e-06 1.17640507e-05]
episode: 2900   action: [1. 0. 0. 0. 0.]   error of W: [1.44979978e-05 4.97477513e-06 1.17640507e-05]
episode: 2925   action: [1. 0. 0. 0. 0.]   error of W: [1.44979978e-05 4.97477513e-06 1.17640507e-05]
episode: 2950   action: [1. 0. 0. 0. 0.]   error of W: [1.44979978e-05 4.97477513e-06 1.17640507e-05]
episode: 2975   action: [1. 0. 0. 0. 0.]   error of W: [1.44979978e-05 4.97477513e-06 1.17640507e-05]
episode: 3000   action: [1. 0. 0. 0. 0.]   error of W: [1.44979978e-05 4.97477513e-06 1.17640507e-05]
```

Figure 5: Output data in the last 300 episodes

In one test, the average value of the attitude angular velocity and its deviation was calculated and recorded per 100 iteration, and the trend of 3000 iterations is shown in Figure 6-7. The variation of the average value of the three angular directions of the attitude angular velocity indicates that as the number of iterations increases, the average angular velocity of the spacecraft converges to [0.00097544 0.00101123 0.00101985] rad/s, and the average value curve of the angular velocity deviation with the desired attitude converges to [2.45602958e- 05 1.12319176e-05 1.98492144e-05], that is, the values in all three directions are reduced and converged, indicating that the attitude of the service aircraft has reached a steady state.

Calculate the average results of reward value per 100 iterations in different experiments and show the changing trend of reward. As shown in the Figure 8, we can see that from the 500th iteration, the average reward is increasing rapidly. When the 2000th iteration is reached, the reward value has stabilized and has only small fluctuations. The fluctuations are mainly caused by constant disturbance. After entering the stationary period, there is not much improvement, which shows that the training is converged after about 2000 iterations.

DOI: 10.13009/EUCASS2019-33

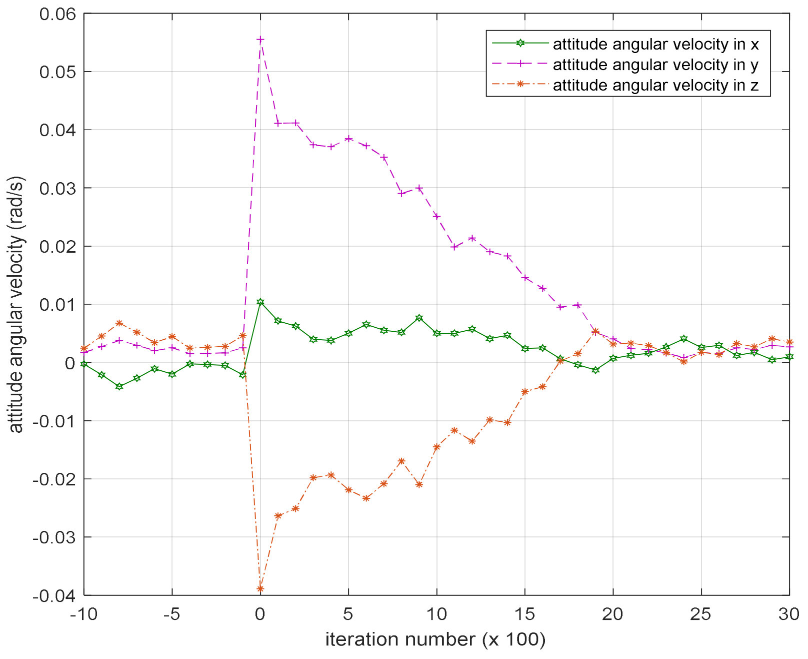


Figure 6: Iterative process of average value of the attitude angular velocity

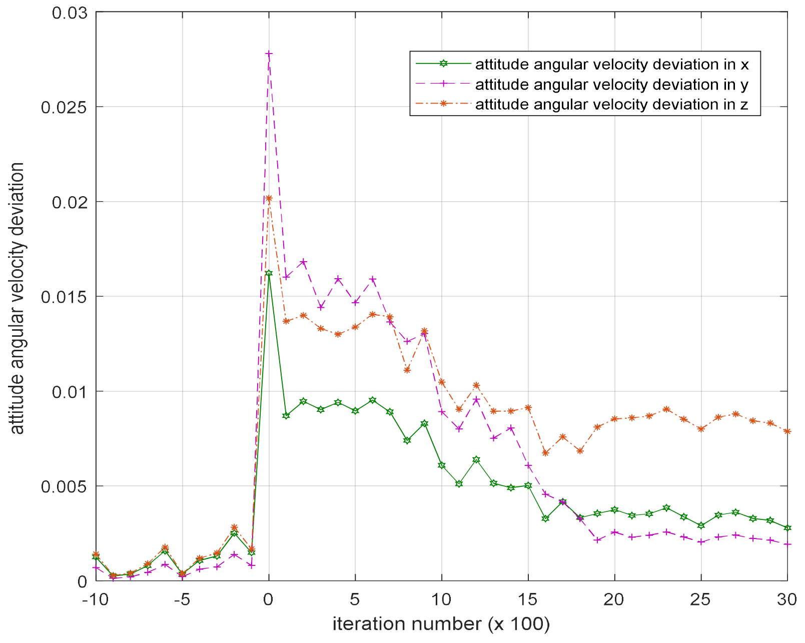


Figure 7: Iterative process of average value of the attitude angular velocity deviation

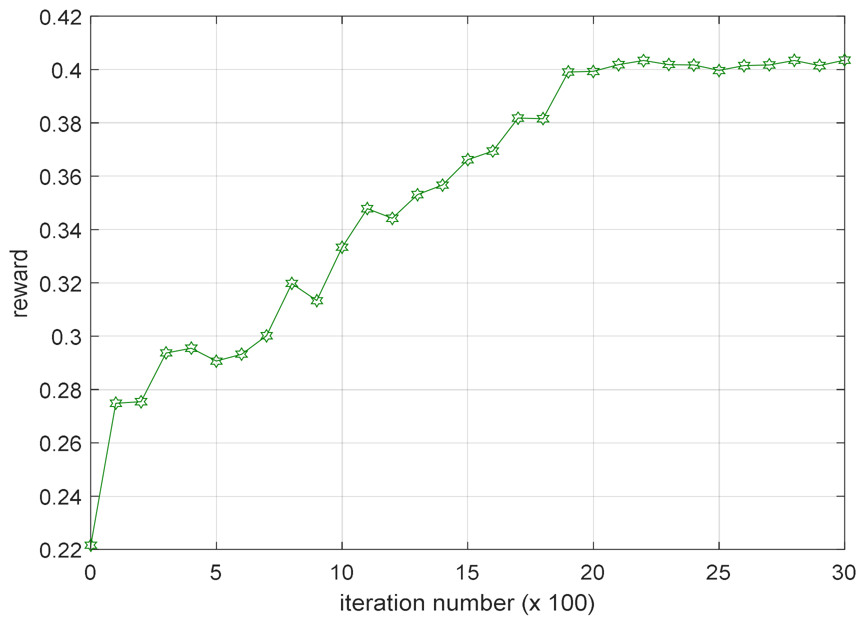


Figure 8: Iteration process of the average reward

3.4 Attitude Control Experiment Based on PD Controller

The PD controller strictly depends on the mass parameter, i.e. the moment of inertia of the controlled object. In the figures below, the attitude angular velocity and its deviation of the spacecraft over 3000 iterations is shown when takes a random disturbance.

Figure 9 demonstrates that the attitude angular velocity gradually increases as the number of iterations increases, failing to converge. This divergent deviation between the attitude angular velocity and the desired attitude angular velocity shown in Figure 10 indicates that the spacecraft cannot maintain the attitude stabilization.

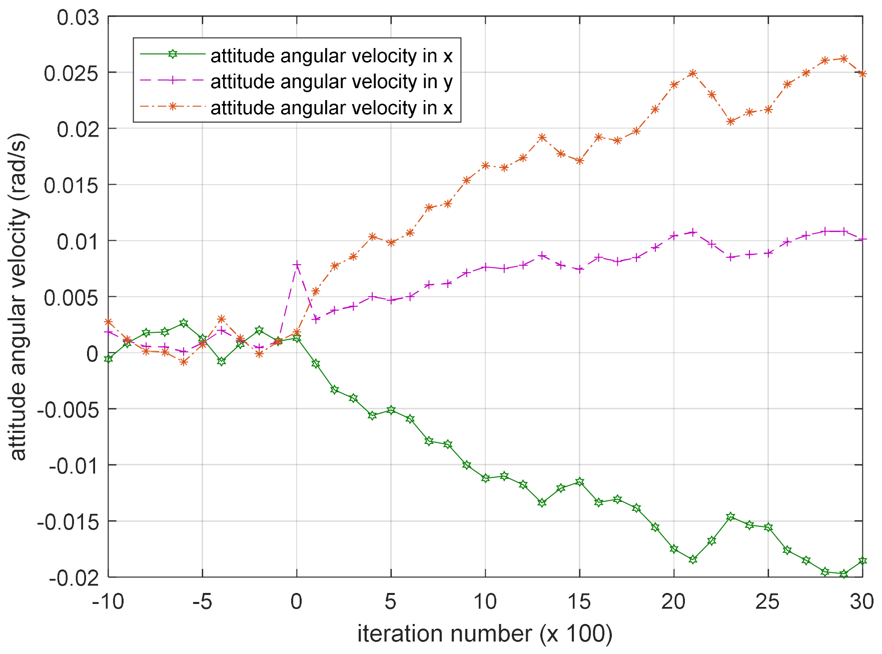


Figure 9: Iteration process of the attitude angular velocity in PD controller

DOI: 10.13009/EUCASS2019-33

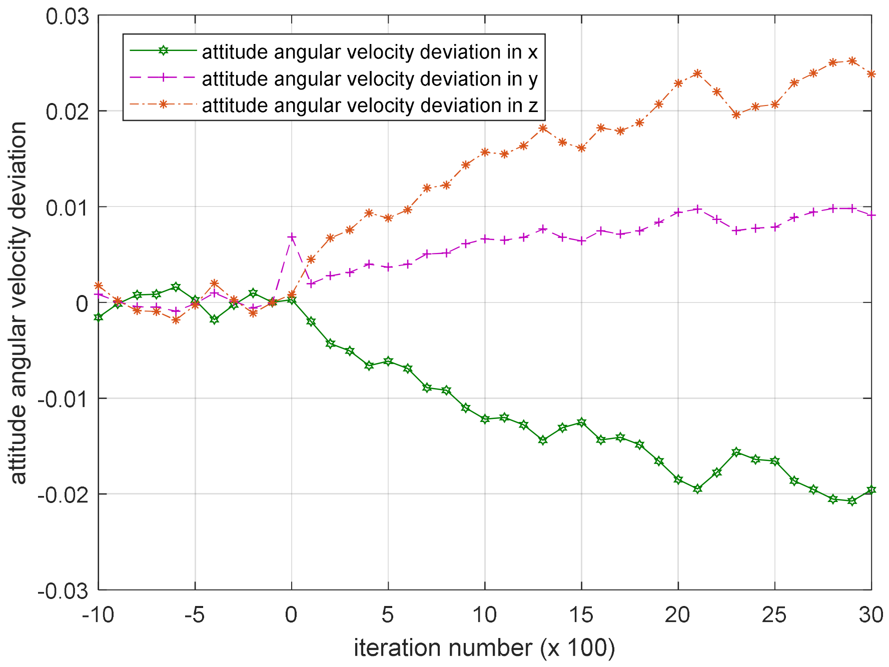


Figure 10: Iteration process of the attitude angular velocity deviation in PD controller

Considering the uncertainties of the mass parameters such as the moment of inertia matrix, the comparison with Section 3.3 also verifies that deep reinforcement learning technology still has the advantage of stabilizing spacecraft attitude under random parameter changes.

It can be seen that the optimal intelligent output of the control torque is obtained as the attitude angular velocity tends to be stable as a reward. The space-enhanced spacecraft attitude control-based deep reinforcement learning algorithm has been able to autonomously stabilize spacecraft attitude after sudden random disturbance and is superior to conventional PD controllers in maintaining spacecraft attitude stabilization.

3.5 Attitude Control Experiment Based on Backstepping Controller

To further validate the effectiveness of proposed method, we compared the proposed method with a kind of robust attitude control method based on backstepping. This method uses the adaptive control theory to stabilize the spacecraft, and the effectiveness of this method can be proved analytically with the Lyapunov method. The experiment results of this method for the same problem are shown in Figures 11–12.

The results show that the backstepping controller has robustness against mass parameter uncertainty. However, the backstepping controller can only handle the constant control cycle; when the control cycle is constant, iteration process of the attitude angular velocity can converge. When the control cycle changes during the iteration process, it becomes divergent. Compared with the backstepping controller, our method based on deep reinforcement learning has competitive performance, while it allows the control cycle changes during iteration. That is, regardless of whether the control cycle changes or not, our method can control the spacecraft to restore a stable flight attitude.

DOI: 10.13009/EUCASS2019-33

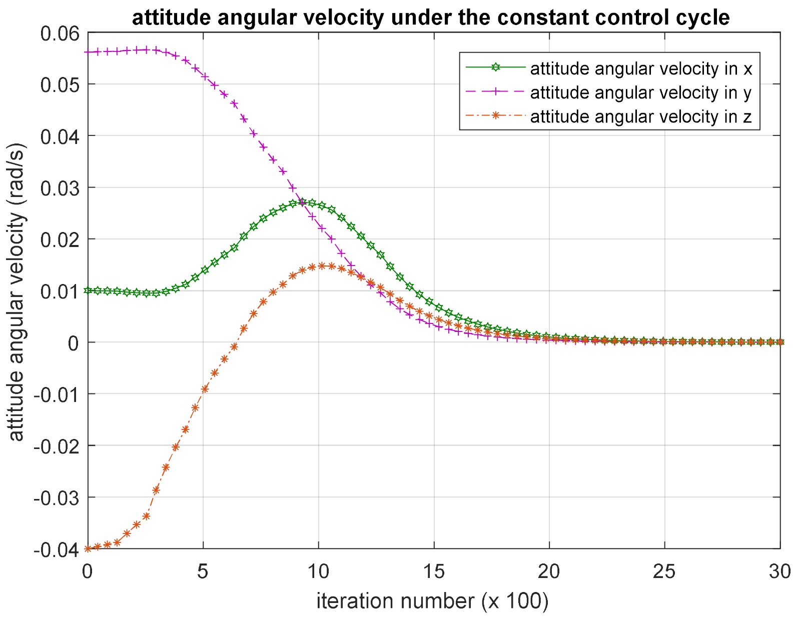


Figure 11: Iteration process of the attitude angular velocity under the constant control cycle in backstepping controller

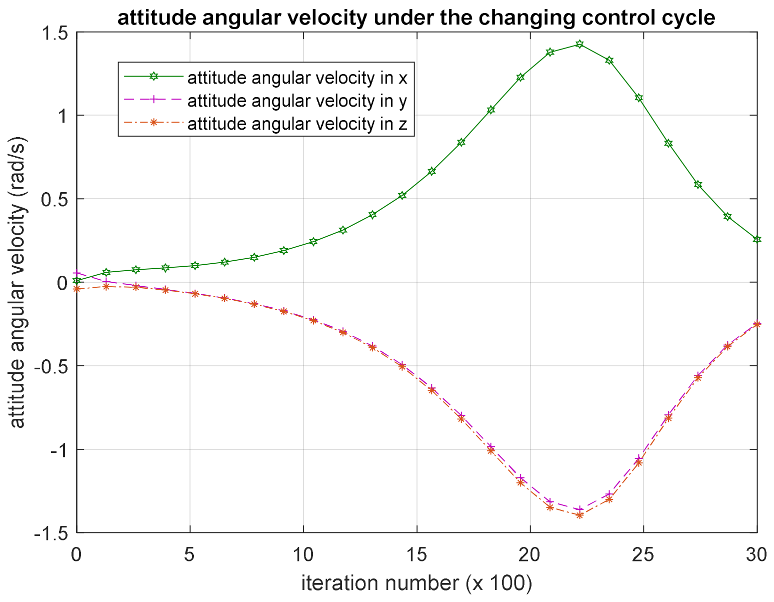


Figure 12: Iteration process of the attitude angular velocity under the changing control cycle in backstepping controller

4. Conclusion

In this paper, the deep reinforcement learning technique is used to stabilize the attitude and change the quality parameters of the spacecraft in order to restore the spacecraft to a stable attitude when performing complex tasks. Building the simulation environment using dynamic model, discretize the control torque, and the attitude angular velocity tends to be stable as the reward to obtain the optimal output. The simulation shows that the deep reinforcement learning algorithm can stabilize the spacecraft's attitude when the spacecraft is disturbed by the quality parameters, which breaks through the difficulties of the traditional PD controller depending on the quality parameters and backstepping controller handling the constant control cycle of the controlled object.

DOI: 10.13009/EUCASS2019-33

5. Acknowledgement

This work was supported by the National Natural Science Foundation of China (Grants No. 61702413).

References

[1] Wei W. S. 2013. Research on the parameters identification and attitude tracking coupling control for the mass body attached spacecraft. PhD Thesis. Harbin Institute of Technology.

[2] Yoon H., and Agrawal B. N. 2009. Adaptive control of uncertain hamiltonian multi-input multi-output systems: with application to spacecraft control. *IEEE Transactions on Control Systems Technology*. 17(4):900–906.

[3] Queiroz M. S. D., Kapila V., Yan Q. 2015. Adaptive nonlinear control of multiple spacecraft formation flying. *Journal of Guidance Control & Dynamics*. 23(3):385–390.

[4] Miao S. Q., Cong B. L., Liu X. D. 2013. Adaptive sliding mode control of flexible spacecraft on input shaping. *Acta Aeronauticaet Astronautica Sinica*. 34(8):1906–1914.

[5] Busoniu L., Babuska R., De Schutter B. 2008. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems Man & Cybernetics Part C Applications & Reviews*. 38(2):156–172.

[6] Jin B. B., Yang J. N., Huang X .S., et al. 2017. Deep deformable Q-Network: an extension of deep Q-Network. *International Conference on Web Intelligence*. 963–966.

[7] Mnih V., Kavukcuoglu K., Silver D., et al. 2013. Playing atari with deep reinforcement learning. *Computer Science*.