Journal on
Internet of Things

Tech Science Press

DOI: 10.32604/jiot.2022.031043
Article

Generation and Simulation of Basic Maneuver Action Library for 6-DOF Aircraft by Reinforcement Learning

Jinlin Wang[1], Jitao Teng[3], Yang He[1], Hongyu Yang[1,*], Yulong Ji[2,*], Zhikun Tang[4] and Ningwei Bai[5]

[1]Sichuan University National Key Laboratory of Fundamental Science on Synthetic Vision, Chengdu, 610000, China
[2]Sichuan University School of Aeronautics and Astronautics, Chengdu, 610000, China
[3]Chinese People's Liberation Army Unit 93216, Beijing, 10000, China
[4]National Airspace Management Center, Linxi, 054900, China
[5]Chengdu No. 7 High School, Chengdu, 610000, China
*Corresponding Authors: Hongyu Yang. Email: yanghongyu@scu.edu.cn; Yulong Ji. Email: jyl@scu.edu.cn
Received: 15 August 2022; Accepted: 15 October 2022

Abstract: The development of modern air combat requires aircraft to have certain intelligent decision-making ability. In some of the existing solutions, the automatic control of aircraft is mostly composed of the upper mission decision and the lower control system. Although the underlying PID (Proportional Integral Derivative) based controller has a good performance in stable conditions, it lacks stability in complex environments. So, we need to design a new system for the problem of aircraft decision making. Studies have shown that the behavior of an aircraft can be viewed as a combination of several basic maneuvers. The establishment of aircraft basic motion library will effectively reduce the difficulty of upper aircraft control. Given the good performance of reinforcement learning to solve the problem with continuous action space, in this paper, reinforcement learning is used to control the aircraft's rod and rudder to generate a basic maneuver action library, and the flight of the aircraft under the 6 degrees of freedom (6-DOF) simulation engine is effectively controlled. The simulation results verify the feasibility of the method on a visual simulation platform.

Keywords: Reinforcement learning; maneuver action library; air combat

1 Introduction

Control theory and controller based on PID (Proportional Integral Derivative) design have been widely used in industrial control process [1], including aircraft control, etc. Although the controller designed by PID theory has achieved many good effects in some fields, it still can't cover up the weakness of PID theory itself. Take the control of the aircraft for example, the control system of the aircraft has the characteristics of non-linearity, variable parameters, variable structure and so on [2]. It is difficult to complete the work to complete a slightly more complex action with a simple linear PID controller. In addition, the PID-based controller lacks the necessary flexibility, and is difficult to cope with the interference of other factors such as external weather. On the other hand, almost all the

components related to the control of the aircraft has a situation of dead-zone non-linearity [3], which also increases the necessary of the controller, which can be used in more complex cases.

From the perspective of military demand, as the pace of modern warfare increases, the complexity of war is rising, and the military demand for intelligent combat is also getting higher and higher. Intelligent air combat has become an important content of the future war. This means, the aircraft should have the ability to effectively integrate external information and make fast decisions to complete the given related tasks just with simple instructions from the commander. For the next generation of flight control systems to become intelligent, they need to design an approach that is capable of adapting to variable dynamics and environments.

The core content of air combat decision making is air combat maneuver decision. In the optimization methods of making air combat action decision, it is generally required a designed maneuver action library. Maneuver action library is the base of maneuvering decision in air combat [4]. The establishment of the action library can reduce the difficulty of designing the end-to-end intelligent air combat body.

Reinforcement learning is a machine learning algorithm designed for sequential decision problems that can be learned in interaction with the environment without any prior knowledge of dynamic models. Reinforcement learning algorithm can gradually improve controller performance from sampling data in the process of interaction with the environment [5]. Reinforcement learning algorithm can greatly reduce the difficulty of controller design. The use of reinforcement learning method in the controller design can change the situation that traditional controller design relies too much on the precise mathematical model of the controlled object and can also solve the problem that traditional controller can be interfered and fails in the control process. Due to its unique characteristics, reinforcement learning has been applied in many fields, including reducing the difficulty of control in control [6], playing a role in task scheduling [7], and showing strong adaptability in image processing [8].

In this paper, based on the 6-DOF flight dynamics model in JSBSim, a flight maneuver action library consisting of five basic maneuver actions including stabilized flight, left turn, right turn, climb and subduction is established.

2 Background

In this part, we will introduce the basic reinforcement learning theory and the aircraft dynamics model library we used in this paper. After that, we will introduce the related work in aircraft control.

2.1 Reinforcement Learning

Reinforcement learning algorithm is not a newly emerging machine learning algorithm, but the great achievements in the field of reinforcement learning in recent years increase people's attention to it.

In a typical reinforcement learning process, consider a finite episode problem, we assume that reinforcement learning agents exist in a complex environment E. At each discrete decision time point t, the agent will observe the environment and obtain the observed value of the environment at this time S_t. Then, the agent will select an action a_t based on its existing knowledge and strategy π and execute it. After an agent performs, the environment situation will change. At the next decision point, the environment will score the performance of the agent in the last step, and the score will be reflected

in the amount of the reward r given by the environment. During the training process, the interaction between the agent and the environment will cycle this process, just as the Fig. 1 shows.

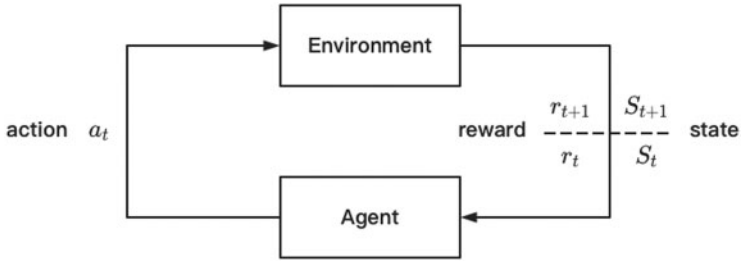


Figure 1: Reinforcement learning flowchart

In the reinforcement learning, the aim is to find a target policy π, which can lead to gain the maximum cumulative return reward. In the finite episode situation, the cumulative reward can be written like:

$$E_{s,a\sim\pi}\left[\sum_{t=1}^{T}\gamma^{t-1}r\left(s_t,a_t\right)\right] \tag{1}$$

The discount γ factor is used to adjust the agent's consideration of future rewards and T is the number of interactions in an episode.

In the reinforcement learning algorithm based on value update represented by Q learning, we use the parameter θ to estimate the action value function $Q\left(s,a;\theta\right)$ under the situation (s,a). With the help of Bellman equation, we can get the objective function like:

$$L_Q = E\left[\left(r\left(t\right)+\gamma\ max_{a'}Q\left(s',a';\theta^{-}\right)-Q\left(s,a;\theta\right)\right)^2\right] \tag{2}$$

Iterate with the least mean square deviation, we can get the best parameter θ, and the optimal strategy is to obtain the optimal action in different states by maximizing the action value function.

In the reinforcement learning algorithm based on policy update represented by Policy Gradient. we use the parameter θ to estimate the policy, in the learning progress, the gradient of the cumulative reward can be written as:

$$\frac{\partial E_{a\sim\pi}\left[R_{1:T}\right]}{\partial\theta} = E\left[\frac{\partial}{\partial\theta}log\,\pi\left(a|s\right)\left(Q^{\pi}\left(s,a\right)-V^{\pi}\left(s\right)\right)\right] \tag{3}$$

In the calculation process, the value function and action value function are replaced by the estimate.

In the process of interacting with the environment, the actions taken by an agent affect not only the immediate reward, but also the next state, and thus all subsequent rewards. To get the maximum reward, an agent must effectively explore the environment and choose the most effective action based on previous experience. This feature makes reinforcement learning more suitable for training agents to make continuous decisions compared with other machine learning methods.

2.2 Flight Dynamics Simulation

At present, the flight simulation of aircraft is mainly divided into three degrees of freedom simulation and six degrees of freedom simulation. The 3-DOF simulation model mainly contains the model of velocity, angle, and its change, without the equation of the relationship between mechanics

and torque. In the 6-DOF simulation model, based on the 3-DOF model, the relationship between mechanics and torque is also included. Compared with the 3-DOF flight simulation model, the 6-DOF simulation model can better describe the real motion of the aircraft, but at the same time, the control of the 6-DOF simulation model is also more difficult. The 6-DOF simulation model takes continuous state as input and continuous actions as output which are correspond to the rudder deviation and engine power. In the subsequent development, it will be more conducive to use in the real aircraft.

The reinforcement learning environment in this paper will be built based on the 6-DOF simulation model. The 6-DOF simulation model library used in this work is JSBSim, which is an open source, multi-platform, data-driven flight dynamics modeling framework [9]. This simulation platform is essentially a physical or mathematical model that uses classic coefficient construction to effectively model aerodynamics and torque and can support any type of aircraft model simulation. In addition, JSBSim defines the motion of the aircraft under the action of its internal forces and the forces from the nature, which has a high degree of simulation of the flight state, strong scalability, and has got high recognition in the field of flight simulation.

2.3 Related Work

NASA has been developing intelligent air combat systems based on expert predictions between the 1960s and 1990s, making several attempts to use artificial intelligence systems to assist and even replace pilots in air combat decisions [10]. Heuristic methods such as genetic algorithm and fuzzy tree are also explored and used [11,12]. In recent years, with the develop of machine learning theory and the improvement of computing power, intelligent algorithms represented by deep learning and reinforcement learning have shown great advantages in air combat. Huang Changqiang et al. assumed the unilateral decision in close air combat as Markov decision process, used Bayesian reasoning to calculate the air situation, and used fuzzy logic method to predict the movement of enemy aircraft. Changqiang et al. [13], using DDPG algorithm, have successfully obtained a reinforcement learning model capable of trajectory retention and velocity retention. Zhang's team [14] proposed an approximate dynamic programming method for solving maneuvers in air combat, which also performed well in environmental tests. Weiren McGrew et al. [15] also conducted relevant research on the intelligent decision-making of unmanned combat air vehicles in the case of air combat within the visual range and proposed the decision algorithm of reinforcement learning for the decision-making strategy of 1V1 air combat scenario. Kong et al. [16] used DDPG to control a 3-DOF aircraft to generate a decision model for air combat. For the complex problem of reward setting, they developed a battlefield situation assessment model to aid in the generation of rewards for agent decisions [17].

Although many papers published in aircraft control that has achieved good results, even some work has been conducted on the simulation platform to run, but it is important to note that the current flight simulation model used by most of the work is the 3-DOF model, and the setting is not unified. Zhuang Sheng et al. 's work is based on six degrees of freedom, but it also simplifies the real situation. There is still a long way to go to realize the effective controller based on the 6-DOF flight dynamics model, and then develop intelligent aircraft capable of real intelligent air combat based on it.

For the movement of aircraft, some scholars have made a summary and summary of it. Scholars of NASA in the United States have put forward a library of seven movements composed of common maneuvering methods in air combat, which mainly includes: maximum acceleration, minimum deceleration, maximum overload climb, maximum overload subduction, maximum overload right turn, maximum overload left turn, stabilized flight and so on. Kaneshige et al. divided the control of the aircraft into three layers: the decision layer, the planning layer and the control layer. The

control layer was mainly responsible for the execution of the maneuvering instructions of the aircraft. Kaneshige et al. [18] also verified the effectiveness of establishing basic action library in simple simulation scenarios [19]. Based on this, this paper proposes to build a basic motion action library for the 6-DOF aircraft simulation model to reduce the training difficulty of the agent and provide effective choices for the upper level of the confrontation decision.

3 Experiment

3.1 Soft Actor Critic Algorithm

Soft Actor Critic (SAC) is a reinforcement learning algorithm for continuous state input and continuous state output, which is based on actor-Critic structure design. In the structure, the actor receives the observation values, makes decisions according to policy, and interacts with the environment, the critic is responsible for evaluating the value of actions made by actor. The actor next adjusts the frequency of action selection according to the score of the value function.

More different from other policy iterative algorithms, SAC adds the entropy value of the action to the basic cumulative reward. The target policy can be written as:

$$\pi^* = \mathrm{argmax}_\pi \mathrm{E}_{(\mathrm{s_t},\mathrm{a_t})}\left[\sum\nolimits_t \mathrm{R}\left(\mathrm{s_t},\mathrm{a_t}\right) + \alpha\, \mathrm{H}\left(\pi\left(\cdot|\mathrm{s_t}\right)\right)\right] \tag{4}$$

with $\mathrm{H}(\mathrm{P}) = \mathrm{E}_{\mathrm{x}\sim\mathrm{P}}[-\log \mathrm{P}(\mathrm{x})]$.

The temperature parameter α determines the relative importance of the entropy term against the reward, and thus controls the stochasticity of the optimal policy. The existence of entropy can effectively increase the exploration of the environment without missing any feasible strategy. Models designed on this will be more adaptable for subsequent upper flight decisions.

In the depth model, we use multi-layer networks to estimate policies and state functions, using $\mathrm{V}_\psi(\mathrm{s_t})$ to evaluate the value function, using $\mathrm{Q}_\theta(\mathrm{s_t},\mathrm{a_t})$ to estimate the soft-Q function, using $\pi_\varphi(\mathrm{a_t}|\mathrm{s_t})$ to evaluate the strategy. ψ, θ, and φ are three parameters of deep network. Soft-V function is updated by minimizing residual square as:

$$J_V(\psi) = E_{s_t\sim D}\left[\frac{1}{2}\left(V_\psi(s_t) - E_{a_t\sim\pi_\phi}\left[Q_\theta(s_t,a_t) - log\,\pi_\phi(a_t|s_t)\right]\right)^2\right] \tag{5}$$

D is the replay buffer of previously sampled states and actions. The gradient of Eq. (5) can be written as:

$$\nabla_\psi J_V(\psi) = \nabla_\psi V_\psi(s_t)\left(V_\psi(s_t) - Q_\theta(s_t,a_t) + log\,\pi_\phi(a_t|s_t)\right) \tag{6}$$

As for the soft-Q function, the target function can be written as Eq. (8):

$$J_Q(\theta) = E_{(s_t,a_t)\sim D}\left[\frac{1}{2}\left(Q_\theta(s_t,a_t) - \hat{Q}(s_t,a_t)\right)^2\right] \tag{7}$$

with $\hat{Q}(s_t,a_t) = r(s_t,a_t) + \gamma E_{s_{t+1}\sim p}\left[V_{\bar{\psi}}(s_{t+1})\right]$

The gradient of formula (6) can be written as Eq. (9):

$$\nabla_\theta J_Q(\theta) = \nabla_\theta Q_\theta(a_t,s_t)\left(Q_\theta(a_t,s_t) - r(s_t,a_t) - \gamma V_{\bar{\psi}}(s_{t+1})\right) \tag{8}$$

In the update process, another estimate of the state function is used as the target network, which has the same structure as $\mathrm{V}_\psi(\mathrm{s_t})$ and different updating speed of parameters.

Finally, for the policy network, since the policy generates a non-deterministic policy, the value directly output from the network is the mean and covariance of the action distribution. Therefore, the parameter θ of the policy network is non-differentiable during the whole process, which results in that the parameters of the policy network cannot be directly updated through the way of back propagation. Here, a reparametrize method is used in SAC, let $a_t = \mathfrak{f}_\phi(\epsilon_t; s_t)$, then the target function of the policy network can be written as:

$$J_\pi(\phi) = E_{s_t \sim D, \epsilon_t \sim \mathcal{N}} \left(log\, \pi_\phi \left(\mathfrak{f}_\phi(\epsilon_t; s_t) | s_t \right) - Q_\theta \left(s_t, \mathfrak{f}_\phi(\epsilon_t; s_t) \right) \right) \tag{9}$$

Then, the gradient of formula (6) can be written as:

$$\nabla_\phi J_\pi(\phi) = \nabla_\phi log\, \pi_\phi(a_t|s_t) + \left(\nabla_{a_t} log\, \pi_\phi(a_t|s_t) - \nabla_{a_t} Q(s_t, a_t) \right) \nabla_\phi \mathfrak{f}_\phi(\epsilon_t; s_t) \tag{10}$$

There are many flight dynamics models of aircraft in JSBSim, and the training process will be introduced based on the flight simulation model of F15 fighter jet. The training process taking F15 as an example will also be theoretically applicable to the training of flight dynamics simulation models of other 6-DOF aircraft in JSBSim.

In the training process, the agent was provided the observation value of the aircraft's own state in one-dimensional vector format. By exploring the action space, the agent will generate strategies for different targets, to achieve the optimal controller of a series of end-to-end flight action libraries.

In the design of the aircraft basic action library, we use neural network to estimate the strategy, and design relevant reward functions for different action commands to guide the agent to learn the desired action mode and maintain the stable state of the aircraft at the same time.

3.2 Stabilized Flight

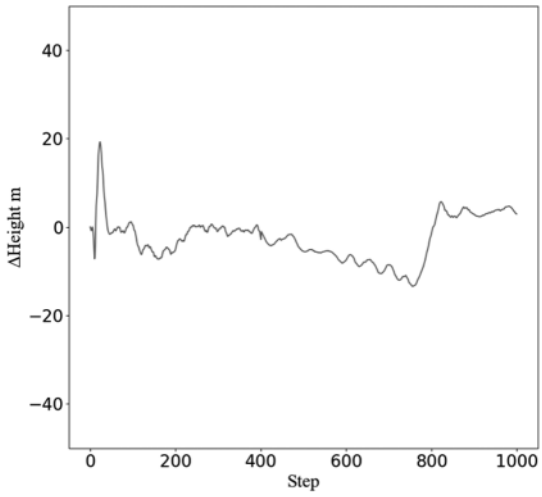
For the basic maneuver action of stabilized flight, we use a four-layer fully connected neural network to estimate the strategy. In this mission, we expect the trained agent has the ability to maintain the altitude before action started and maintain the heading. Assume that when the command is given, the altitude of the aircraft at this time is *alt*, and the aircraft's heading is *hg*. At this point, we set the observation as $[alt_t, v_t, hg_t, p_t]^T$ based on the observable properties of the actual aircraft, among them, alt_t is the altitude of the aircraft at time t, v_t is the speed shown on the plane's dashboard. hg_t is the heading angle of the aircraft of time t, p_t *i*s the pitch angle of the aircraft. In SAC algorithm, Actor network receives six inputs, namely, the current altitude att_t, the current speed v_t, the current heading angle and pitch angle of the aircraft, as well as the altitude *att* and the heading angle *hg* at the beginning of receiving the instruction. The three outputs of actor network are the position of the direction bar, the position of the lifting bar and the thrust ratio of the engine respectively. JSBSim will get these and help us convert the information acting on the direction sense and the lifting bar into the swing changes of the wing. The Critic network receives nine inputs, including six inputs to Actor network and three outputs from Actor network, while the output of Critic Network is just one. The only output represents the scoring of the current decision taken in this case. According to the current direct mission objectives, we get three main elements into consideration to set the reward function. In addition to the ability of the agent to maintain the height of aircraft, the ability to maintain the heading angle, we take the real aircraft control into consideration. In most cases, the actions of aircrafts are smooth, thus we ae added an estimate of the control of the smoothness of the decisions made by agents. The final reward function can be written as follows:

$$reward = \omega_\alpha |alt_t - \ alt| + \omega_\beta |h_t - hg| + \omega_\gamma ||a_t - \ a_{t-1}|| + r_s \tag{11}$$

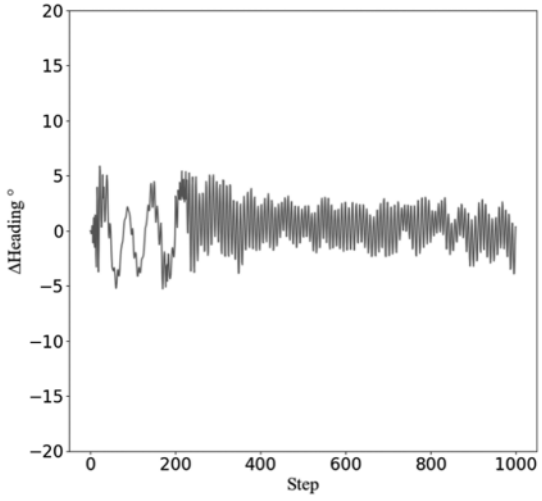
where $\omega_\alpha, \omega_\beta, \omega_\gamma$ are the proportions of the three parts in the overall reward. $a_t - a_{t-1}$ is used to estimate the difference between the two actions. And r_s is the reward once the agent step successfully.

In the training, the policy network and evaluation function network were randomly initialized, the length of single episode was set as 1000, and the ΔT per step was set as 0.1 s. The judgment condition for the end of episode is added. When the aircraft deviates from the initial height of 500 m or the initial heading of 15 degrees, the current round is terminated, and the next episode of interaction is started.

In the test, we recorded the altitude and heading of the aircraft, and obtained the results as shown in the Fig. 2. Under the control of the stabilized-flight agent, the state of the aircraft is not in the best state in the first few steps, after the adjustment of the agent in several steps, the altitude and heading angle of the aircraft change less and only fluctuate in a small range.



(a) Altitude changes in stabilized flight



(b) Heading changes in stabilized flight

Figure 2: The altitude and heading of the aircraft

As shown in Fig. 2a, under the influence of the initial state, the height of the aircraft controlled by the agent changes dramatically at the beginning, and gradually recovers to stability in the later period.

The agent can effectively adjust the height deviation in the process. As Fig. 2b shows, during the test, the heading of the aircraft swings in a small range. Before and after the test, the heading of the aircraft does not change much, indicating the effectiveness of the control.

3.3 Left- and Right-Turn

The training of left- and right-turn maneuvers action have certain relevance in design and training, so they are presented together. In the course changing mission, we use a five-layer fully connected neural network to estimate the strategy and value function. It is expected that the trained agent can reduce the radius required for course changing as much as possible and maintain the altitude at the same time. In these two missions, the information that the agent can observe is still controlled as $[alt_t, v_t, hg_t, p_t]^T$. The input of Actor network is still 6 in this mission, but hg in the stabilized flight is changed to the h_{t-1}, and the output does not change. The input and output of the Critic network are also the same, respectively as the network in a stabilized-flight mission. According to the current mission, we adjusted the reward. The reward in this mission still consists of four parts.

$$reward = \omega_\alpha \left| alt_t - \ alt \right| + \omega_\beta \left| h_t - h_{t-1} \right| + \omega_\gamma \left| \left| a_t - \ a_{t-1} \right| \right| + r_s \tag{12}$$

In this mission, the angular error term is modified to encourage the agent to change the aircraft's course as much as possible while maintaining stability and altitude.

In the training, the policy network and evaluation function network were randomly initialized, the length of single episode was set as 1000, and the ΔT per step was set as 0.1 s. When the aircraft deviates from the initial height of 500 m, the current round is terminated, and the next episode of interaction is started. In the test, we recorded the altitude and heading of the aircraft, and obtained the results as shown in the Fig. 3.

(a) and (c) are the change of height with the number of steps, (b) and (d) are the change of heading with the number of steps in the process of turning left and right respectively. In the turning process, the heading orientation has a stable periodic change, and the change of altitude gradually tends to ease. During the whole process, the change of altitude was not obvious.

3.4 Climb

In the climbing mission, we also use a five-layer network to estimate the strategy and value function respectively. We expect the aircraft to be able to raise the altitude as quickly as possible while maintaining its heading angle. The information that the agent can observe is still controlled as $[alt_t, v_t, hg_t, p_t]^T$. The input of Actor network is still 6 in this mission, but att in the stabilized flight is changed to the alt_{t-1}, and the output does not change. The input and output of the Critic network are also the same, respectively as the network in a stabilized-flight mission. Similarly, we have adjusted the return function. The adjusted return function is shown as follows.

$$reward = \omega_\alpha \left| alt_t - \ alt_{t-1} \right| + \omega_\beta \left| h_t - hp \right| + \omega_\gamma \left| \left| a_t - \ a_{t-1} \right| \right| + r_s \tag{13}$$

This time, the altitude error term is modified to encourage the agent to change the aircraft's altitude as much as possible while maintaining stability and heading angle.

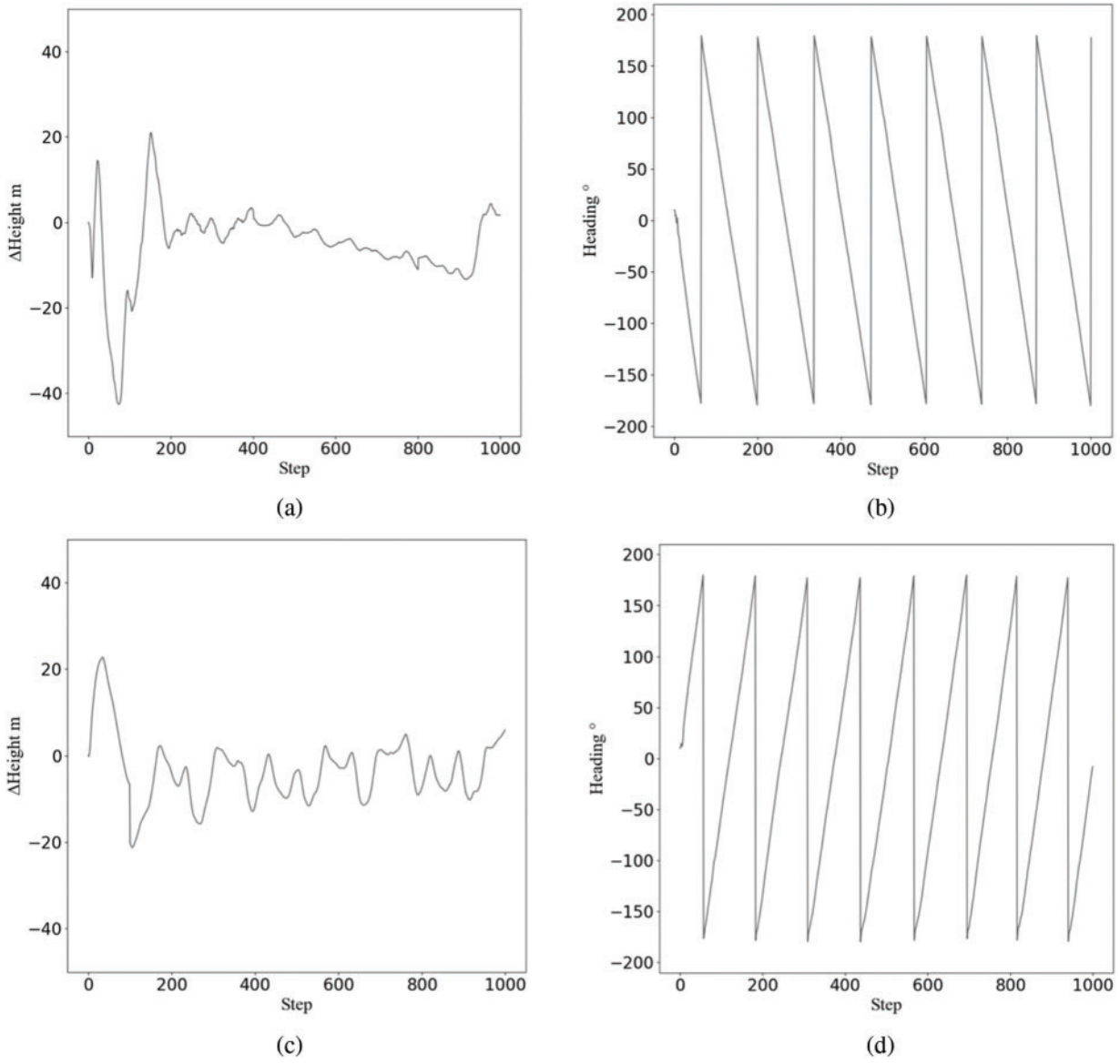


Figure 3: Altitude and heading changes in left- and right-turn

In the training, the policy network and evaluation function network were randomly initialized, the length of single episode was set as 500, and the ΔT per step was set as 0.1 s. When the aircraft deviates from the initial heading of 15 degrees, the current round is terminated, and the next episode of interaction is started.

The altitude and heading recorded during testing are shown below.

As shown in the Fig. 4, with the increase of the number of test steps, the height of the aircraft steadily rises under the control of the agent, and at the same time, the heading of the aircraft gradually becomes stable.

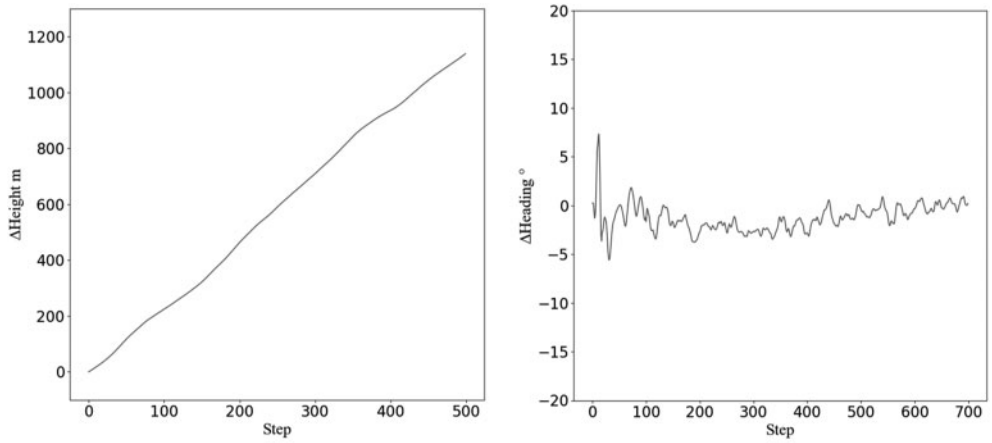


Figure 4: Altitude and heading changes in climb

3.5 Subduction

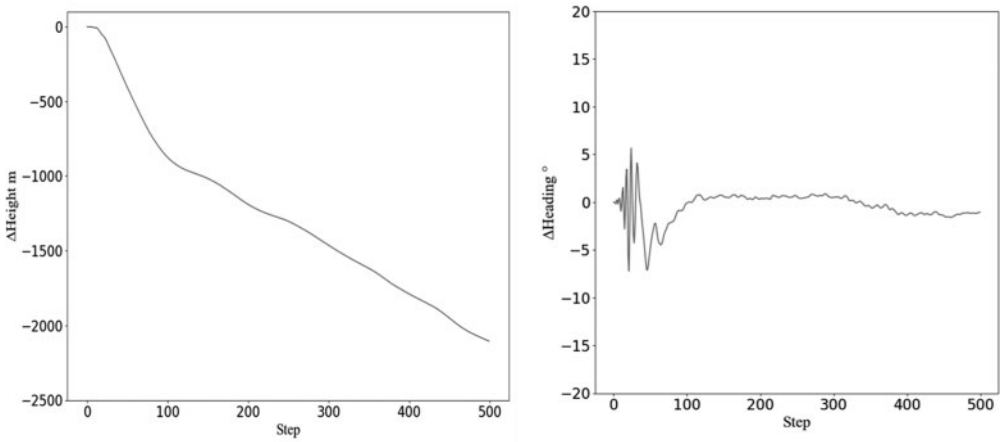
In the subduction mission, we also use a five-layer network to estimate the strategy and value function respectively. We expect the aircraft to be able to lower the altitude as quickly as possible while maintaining its heading angle. But different from the climbing mission, if effective control is not done, the intelligent body is very likely to adopt a nose-down flight state, resulting in aircraft crash, which is contrary to the purpose of flight control. The information that the agent can observe is still controlled as $[att_t, v_t, hg_t, p_t]^T$. The input and output of the Actor network and Critic network are the same, respectively as the network in a climb mission. Again, we've adjusted the return function. The adjusted return function is shown as follows.

$$reward = \omega_\alpha |att_t - \ att_{t-1}| + \omega_\beta |h_t - hp| + \omega_\gamma ||a_t - \ a_{t-1}|| + r_s \tag{14}$$

In the training, the policy network and evaluation function network were randomly initialized, the length of single episode was set as 500, and the ΔT per step was set as 0.1 s. When the aircraft deviates from the initial heading of 15 degrees, the current round is terminated, and the next episode of interaction is started. In the subduction mission, we added the reward for the agent's successful single-step execution in amount to encourage the agent to keep the vehicle as controllable as possible while lowering altitude, so that it could fly steadily after lowering altitude.

The altitude and heading recorded during testing are shown below.

As shown in the Fig. 5, with the increase of the number of test steps, the height of the aircraft steadily decreases under the control of the intelligent agent. Meanwhile, the heading of the aircraft gradually becomes stable and can effectively maintain the initial heading.



(a) (b)

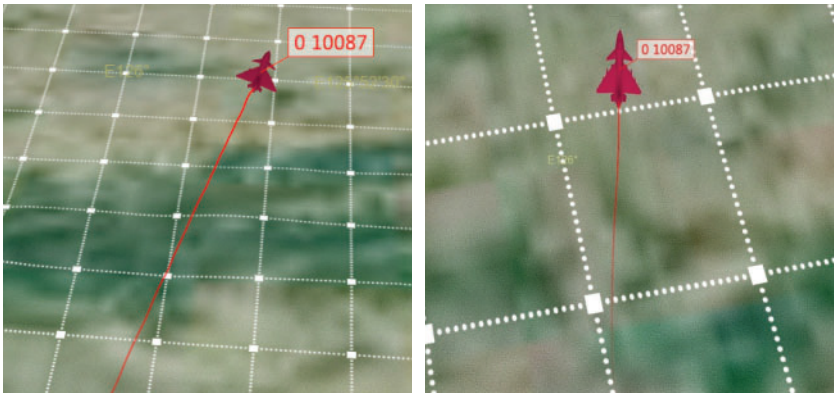
Figure 5: Altitude and heading changes in subduction

3.6 Simulation Results

In order to have a more intuitive understanding of the flight state of the aircraft in the execution of the action, we carried out a visual display of the state and track of the aircraft. With the Gisplay software designed and developed within the group, we visualized the training results. In the display, to see the attitude of the aircraft more intuitively, we enlarged the model of the aircraft.

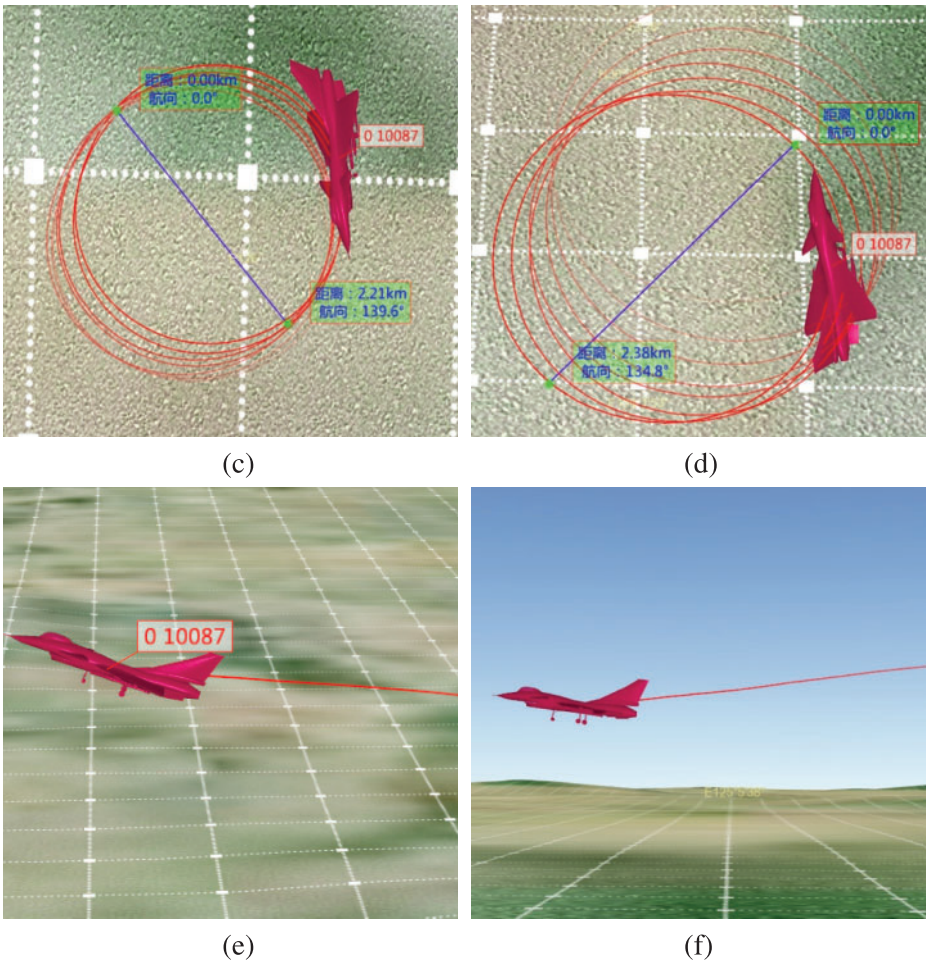
The software Gisplay receives the data packet sent by the program, and according to the description information of the position and attitude of the aircraft, it can draw the state of the aircraft in real time in the scene and retain the flight track of the aircraft in a previous period of time, which is of great help for the intuitive analysis of the flight state of the aircraft.

Fig. 6 shows the flight tracks of the aircraft when performing different actions, where (a) and (b) are observations from two angles of stabilized flight. From (c) to (f) are visualizations performed during left turn, right turn, pull up, and dive, respectively. On the posture software, we can see that the trajectory of the vehicle is smooth, with the help of the Gisplay measurement tools, we can know that the turning radius of the aircraft in the simulation is only about 2.2 km, this makes sense for high speed aircraft.



(a) (b)

Figure 6: (Continued)



(c) (d)

(e) (f)

Figure 6: Flight trajectory of aircraft in different actions

4 Conclusion

In this paper, we use reinforcement learning to design a basic maneuver action library based on the F15 fighter jet, a six-degree-of-freedom flight model in JSBSim. In the experiment, we use the network to estimate the agent's strategy and action value function and design the reward function with similar structure for five kinds of maneuver actions, namely stabilized flight, left turn, right turn, climb and subduction, which successfully guide the agent to learn five kinds of basic action library. In the experiment, the design of the return function is simple, and the structure of the reward function between different actions is similar, and there is the possibility of migration between various models.

In addition, the motion library of this work is developed based on the JSBSim flight simulation model with six degrees of freedom, and the simulation degree is greatly improved compared with three degrees of freedom. The establishment of the maneuver action library not only proves the feasibility of reinforcement learning under the control of 6-DOF aircraft, but also provides a feasibility for the control of 6-DOF aircraft for complex actions, which reduces the complexity of upper decision-making. In the future, we will continue to expand the basic maneuver action library, and on this basis, design and develop the countermeasures decision strategy model based on the 6-DOF aircraft.

Acknowledgement: I would like to express my thanks to those who have helped with the work covered in this article. I would like to express my sincere and ardent thanks to my instructor Ji. And thanks to the students and senior brothers who provided guidance and help in the control theory of the aircraft.

Funding Statement: This work was supported by grant of No. U20A20161 from the State Key Program of National Natural Science Foundation of China.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] H. Wu and W. H. Su, "PID controllers: Design and tuning methods," in *Proc. IEEE Conf. on Industrial Electronics and Applications*, Hangzhou, China, pp. 808–813, 2014.
[2] K. H. Ang, C. Gregory and Y. Li, "PID control system analysis, design, and technology," *IEEE Transactions on Control System Technology*, vol. 13, no. 4, pp. 559–576, 2005.
[3] Q. Hu, "Variable structure output feedback control of a spacecraft under input dead-zone non-linearity," *Aerospace Engineering*, vol. 221, no. 2, pp. 289–303, 2007.
[4] X. J. Dong, M. J. Yu and S. Song, "Research on air combat maneuver library and control algorithm design," in *Proc. China Command and Control Conf.*, Beijing, China, pp. 188–193, 2020.
[5] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, Cambridge, MA, US: MIT Press, 2018. [Online]. Available: https://go.gale.com/ps/i.do?id=GALE%7CA61573878&sid=googleScholar&v=2.1&it=r&linkaccess=abs&issn=07384602&p=AONE&sw=w.
[6] J. Arshad, A. Khan, M. Aftab, M. Hussain, A. U. Rehman *et al.*, "Deep deterministic policy gradient to regulate feedback control systems using reinforcement learning," *Computers, Materials & Continua*, vol. 71, no. 1, pp. 1153–1169, 2022.
[7] F. Rasheed, K. A. Yau, R. M. Noor and Y. Chong, "Deep reinforcement learning for addressing disruptions in traffic light control," *Computers, Materials & Continua*, vol. 71, no. 2, pp. 2225–2247, 2022.
[8] W. Fang, L. Pang and W. N. Yi, "Survey on the application of deep reinforcement learning in image processing," *Journal on Artificial Intelligence*, vol. 2, no. 1, pp. 39–58, 2020.
[9] Z. Feng, "Research on the techniques of desktop real time aviation emulation and its implementation," M.S. Dissertation, University of Jilin, China, 2008.
[10] Q. Yang, J. Zhang, G. Shi, J. Hu and Y. Wu, "Maneuver decision of UAV in short-range air combat based on deep reinforcement learning," *IEEE Access*, vol. 8, pp. 363–378, 2019.
[11] N. Ernest, D. Carroll, C. Schumacher, M. Clark, K. Cohen *et al.*, "Genetic fuzzy based artificial intelligence for unmanned combat aerial vehicle control in simulated air combat missions," *Journal of Defense Management*, vol. 6, no. 1, pp. 2167–0374, 2016.
[12] R. E. Smith, B. A. Dike, R. K. Mehra, B. Ravichandran and A. EI-Fallah, "Classifier systems in combat: Two-sided learning of maneuvers for advanced fighter aircraft," *Computer Methods in Applied Mechanics and Engineering*, vol. 186, no. 2, pp. 421–437, 2000.
[13] H. Changqiang, D. Kangsheng, H. Hanqiao, T. Shangqin and Z. Zhuoran, "Autonomous air combat maneuver decision using Bayesian inference and moving horizon optimization," *Journal of Systems Engineering and Electronics*, vol. 29, no. 1, pp. 86–97, 2018.
[14] S. Zhang, X. Du, J. Xiao, J. Huang and K. He, "Reinforcement learning control for 6 DOF flight of fixed-wing aircraft," in *Proc. Chinese Control and Decision Conf.*, Kunming, China, pp. 134–141, 2021.
[15] J. S. McGrew, *Real-Time Maneuvering Decisions for Autonomous Air Combat*, Cambridge, MA, USA: Massachusetts Institute of Technology, pp. 127–129, 2008.
[16] W. Kong, D. Zhou, Z. Yang, K. Zhang and L. Zeng, "Maneuver strategy generation of UCAV for within visual range air combat based on multi-agent reinforcement learning and target position prediction," *Applied Sciences*, vol. 10, no. 15, pp. 5198–5207, 2020.

[17] Q. Yang, Y. Zhu, J. Zhang, S. Qiao and J. Liu, "UAV air combat autonomous maneuver decision based on DDPG algorithm," in *Proc. Int. Conf. on Control and Automation*, Edinburgh, UK, pp. 37–42, 2019.
[18] J. Kaneshige, K. Krishnakumar and F. Shung, "Tactical maneuvering using immunized sequence selection," in *2nd AIAA "Unmanned Unlimited" Conf. and Workshop & Exhibit*, San Diego, California, pp. 6640, 2006.
[19] K. Q. Zhu and Y. F. Dong, "Research on design mode of air combat maneuvering action library," *Aeronautical Computing Technology*, vol. 31, no. 4, pp. 50–52, 2001.