2021 8th International Conference on Information, Cybernetics, and Computational Social Systems (ICCSS) | 978-1-6654-0245-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICCSS53909.2021.9722012

Moving Target Shooting Control Policy Based on Deep Reinforcement Learning

Boyu Li[1*], Tao Jin[2*], Yuanheng Zhu[1], Haoran Li[1✉], Yingnian Wu[2✉] and Dongbin Zhao[1]

***Abstract*—Robots are playing a more and more important role in people's production and life, recently. However, robot control in dynamic environment is still a difficulty. With the great breakthrough of deep reinforcement learning in the field of video games, this method is also extended to the field of robots. Due to the gap between the simulation environment and the real environment, the deep reinforcement learning algorithm trained in the simulation environment is difficult to be applied to the real environment. Aiming at the gimbal control with two degrees of freedom (DOF), a pipline combining system identification and deep reinforcement learning is proposed. On the one hand, the shooting accuracy of the gimbal to moving objects is improved through deep reinforcement learning algorithm. On the other hand, the gap between simulation and reality is reduced through system identification. The method is verified in the RoboMaster University AI Challenge (RMUA) shooting system. The results show that the shooting accuracy is better than the classical control method.**

***Index Terms*—moving target, shooting control, deep reinforcement learning, system identification**

I. Introduction

In recent years, deep learning has made great breakthroughs in image processing, text recognition and speech processing, and its influence has also spread to the field of robotics. It has a far-reaching impact on robot perception, such as vehicle detection [1], road segmentation [2] and so on. At the same time, reinforcement learning, as a sequential decision-making method, has made a milestone breakthrough in solving the high-dimensional state space decision-making problem such as video games [3]. Deep reinforcement learning based on deep learning and reinforcement learning has gradually affected the field of robot control.

At present, deep reinforcement learning has been widely used in the field of robotics. Clavera *et al*. uses model-based method to solve the problem of manipulator grasping [4]. Xie *et al.* realize the navigation of the robot based on image by deep reinforcement learning [5]. Li *et al.* combine deep learning and reinforcement learning to solve the problem of lane keeping in automatic driving [6]. At present, most of the deep reinforcement learning methods applied to robots need to be trained in the simulation environment. However, due to the difference of dynamic response between the simulation environment and the real environment, the algorithm trained in the simulation environment is difficult to be directly applied to the physical robot.

Shooting control of moving objects is a dynamic tracking problem of the two DOF gimbal. Due to the tracking error of the gimbal, the response delay, and the uncertainty of the bullet in the flight, the stable gimbal control is hard. Moreover, the nonlinear motion of moving objects makes it difficult for the classical control and tracking methods to have a satisfying tracking result even under the ideal model. Therefore, by introducing the deep reinforcement learning method, this paper captures the nonlinear motion trajectory of the moving object, as well as the uncertainty of the gimbal and the flight process of bullet through the deep neural network.

Aiming at the shooting control problem of moving object based on two DOF gimbal, we introduce deep reinforcement learning method. Considering the gap between the simulation and the real environment faced by deep reinforcement learning in dealing with robot control problems, we combine deep reinforcement learning and system identification, and design a shooting control network for dynamic objects to improve the transfer performance. The experimental results in the simulation environment and the real environment show that the accuracy of this method is better than that of the classical control method.

II. Related Work

With the great success of the deep reinforcement learning method combining with deep learning and reinforcement learning in the fields of video games, Go and so on, this method is also gradually applied to the field of robots to deal with the problems of motion control in high-dimensional state space. For example, Li *et al.* uses the deep reinforcement learning method to realize lane keeping with image as input [6]. Li *et al.* combines reinforcement learning with autonomous exploration, proposes a fully convolutional Q network, and applies it to physical robot exploration [7]. In addition, deep reinforcement learning is also widely used in other robot fields, including aircraft control [8], manipulator control [9], [10], and quadruped robot control [11], [12].

[1]Boyu Li, Yuanheng Zhu, Dongbin Zhao and Haoran Li are with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, 100190, China, and also with the College of Artificial Intelligence, the University of Chinese Academy of Sciences, Beijing, China (email : liboyu2021@ia.ac.cn, yuanheng.zhu@ia.ac.cn, dongbin.zhao@ia.ac.cn, lihaoran2015@ia.ac.cn)

[2]Tao Jin and Yingnian Wu is with the School of Automation, Beijing Information Science and Technology University, Beijing, 100192, China (email : 1550288502@qq.com, wuyingnian@126.com)

* Boyu Li and Tao Jin contributed equally to this work.

✉Haoran Li and Yingnian Wu are the corresponding authors.

This work was supported in part by the National Key Research and Development Program of China under Grant 2018AAA0101005, in part by the Strategic Priority Research Program of Chinese Academy of Sciences under Grant No. XDA27030400, and also in part by Youth innovation Promotion Association CAS.

Although there are many deep reinforcement learning methods applied to robots, most of them are only trained and tested in the simulation environment, which is difficult to be directly applied to physical robots. The main reason is the difference between the dynamic model of the robot in the simulation environment and the real environment. At present, there have been a lot of work on this kind of problem. For example, Punjani et al. use nonlinear model to compensate the dynamic system of helicopter, so as to obtain a more accurate system model [13]. The difference between the parameter value in the linear system model and the real system can be solved by collecting the real data and using the least square method. For nonlinear systems or systems with complex models, heuristic optimization methods can usually be used to solve them, such as CMA-ES [14], Bayesian optimization [15], REPS [16], etc.

The shooting control problem of moving object is a dynamic target tracking problem under two-DOF gimbal. This problem comes from the RMUA. Its goal is to attack the armor on the enemy robot by controlling the two-DOF gimbal. Because the nonlinear movement of enemy robot armor and the strike delay caused by bullet flight time, the tracking control method based on PID or Kalman filter [17] has poor performance in dynamic situation. In addition, the response delay of the gimbal and the uncertainty of air friction during bullet flight make it very difficult to accurately strike long-distance moving targets. At present, some schemes [18] use external sensors to estimate the motion and assume the linear or circular motion of the enemy armor plate, but the effect of this method is not ideal in practice.

III. METHOD

In this section, we first introduce the problem of the two-DOF gimbal shooting control, then model each part of the system, determine the parameters in the system model through the system identification method. The simulation environment of two-DOF gimbal shooting system is established. Finally, the deep reinforcement learning method is used to train the gimbal control policy in the simulation environment.

A. The shooting problem

The shooting control of moving objects of two-DOF gimbal comes from the infantry countermeasure system in RMUA. RMUA is an adversarial robot competition. The competition system includes red and blue teams. Each team has two robots. Each robot is composed of gimbal system and chassis system. Each robot can launch bullets by controlling the shooting system on the gimbal, and the bullet launch angle can be controlled by controlling the angle of the two-DOF gimbal. Armors are installed around each robot, which can sense bullet attack, so as to realize confrontation. During the competition, the robots of both sides avoid the bullets fired by the other robot and hit the other robot through continuous movement, and finally beat the other robot as the winner in a limited time. Gimbal system control is the research object of this paper, and its structure is shown in the Fig. 1.

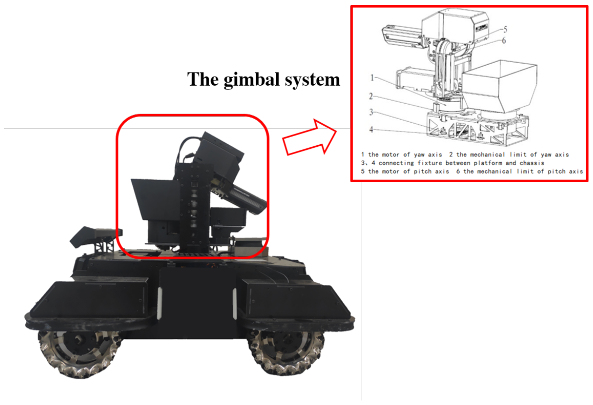
The gimbal system
1
2
3
4
5
6
1 the motor of yaw axis 2 the mechanical limit of yaw axis
3、4 connecting fixture between platform and chassis
5 the motor of pitch axis 6 the mechanical limit of pitch axis

Fig. 1. The infantry robot and the gimbal system.

The pitch angle and yaw angle are the angles to be controlled. The control range of pitch angle is $[-25, 10]$, and the control range of yaw angle is $[-75, 75]$.

B. System model

In the shooting control of moving objects, the system model includes three parts: the dynamic model of the gimbal, the bullet flight model and the motion model of enemy armor.

1) Dynamic model of pitch angle: With the angular momentum theorem, we have

$$m_p l_p^2 \ddot{\alpha} + b_p \dot{\alpha} + m_p g l_p \sin\alpha = T_p \tag{1}$$

where m_p is the weight of pitch arm, and l_p is the length of pitch swing arm. b_p is the damping coefficient of pitch axis. T_p is the motor output torque of pitch axis, and α is the pitch axis angle. The form of state model is

$$\begin{cases} \dot{x}_p^1 = x_p^2 \\ \dot{x}_p^2 = -\dfrac{g}{l_p}\sin x_p^1 - \dfrac{b_p}{m_p l_p^2} + T_p \end{cases} \tag{2}$$

where x_p^1 and x_p^2 are α and $\dot{\alpha}$, respectively.

2) Dynamic model of yaw angle: The establishment of yaw angle dynamic model is similar to that of pitch angle, except that the yaw angle does not consider the influence of gravity. The equation of state can be obtained as follows

$$\begin{cases} \dot{x}_y^1 = x_y^2 \\ \dot{x}_y^2 = -\dfrac{b_y}{m_y l_y^2} + T_y \end{cases} \tag{3}$$

where x_y^1 and x_y^2 are β and $\dot{\beta}$, respectively. β is yaw angle, and m_y is the weight of yaw arm. l_y is the length of yaw swing arm. b_y is the damping coefficient of yaw axis. T_y is the motor output torque of yaw axis.

3) Bullet flight model: The process of the bullet flying from the gimbal muzzle to the enemy armor is not linear. The flight trajectory of the bullet needs to be analyzed according to the initial velocity, mass and air resistance of the bullet. Through the force analysis of the bullet during flight, since the bullet is only affected by gravity and air resistance during flight, the following model can be obtained

$$\begin{cases} \dot{x}_b = v_b^x \\ \dot{y}_b = v_b^y \\ \dot{z}_b = v_b^z \\ \dot{v}_b^x = -\dfrac{k_b (v_b^x)^3}{|v_b^x|} \\ \dot{v}_b^y = -\dfrac{k_b (v_b^y)^3}{|v_b^y|} \\ \dot{v}_b^z = -\dfrac{k_b (v_b^z)^3}{m_b |v_b^z|} - g \end{cases} \tag{4}$$

where (x_b, y_b, z_b) is the bullet position. v_b^x, v_b^y and v_b^z are the flight speeds of the bullet in three directions respectively. k_b is the coefficient related to bullet material, air density, atmospheric pressure and bullet air contact area, and m_b is the mass of the bullet. The above model is a first-order differential equation, and its analytical solution is

$$\begin{cases} x_b = \dfrac{1}{k_b} \ln(k_b v_b^{x0} t + 1) \\ y_b = \dfrac{1}{k_b} \ln(k_b v_b^{y0} t + 1) \\ z_b = v_b^{z0} t - \dfrac{1}{2} g t^2 \end{cases} \tag{5}$$

where v_b^{x0}, v_b^{y0} and v_b^{z0} are the initial speeds of the bullet in three directions respectively.

In the real environment, we find that the initial velocity from the muzzle cannot strictly follow the muzzle direction. There is some disturbance. Therefore, we define the actual muzzle initial velocity distribution as follows.

$$\begin{cases} v_b^{x0} = v_0 \cos(\delta_0) \cos(\delta_1) \\ v_b^{y0} = v_0 \cos(\delta_0) \sin(\delta_1) \\ v_b^{z0} = v_0 \sin(\delta_0) \end{cases} \tag{6}$$

where $v_0 \sim N(\mu_v, \sigma_v)$ is the initial speed which is a random variable, and the disturbance angles $\delta_0 \sim N(\mu_0, \sigma_0)$ and $\delta_1 \sim N(\mu_1, \sigma_1)$ are shown in the Fig. 2.

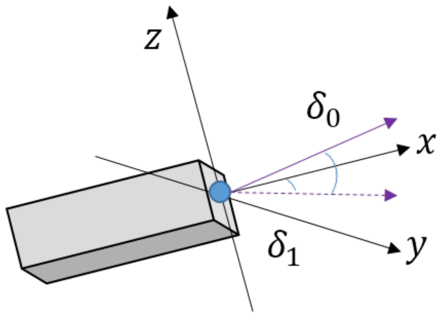

Fig. 2. The initial speed of the bullet.

4) Motion model of enemy armor: The movement of enemy armor is beyond our control and cannot be predicted in advance. In order not to lose generality, we use two motion models here. One is static model and the other is nonlinear motion model. Since the z-axis coordinate of the armor is unchanged, its motion is two-dimensional. Thus, curved motion reflects the normal motion of the armor. We choose parabolic motion for simulation. Assuming that the vertex of the parabola is (X_0, Y_0), when the X-axis coordinate exceeds $(X_0 - 0.04, X_0 + 0.04)$m, the target will return along the original trajectory and carry out cyclic motion. The coordinate system used is also the left-handed coordinate system. The equation of the parabola is:

$$x = 3 * (y - Y_0)^2 + X_0 (X_0 - 0.04 \le X \le X_0 + 0.04) \tag{7}$$

C. System identification

There are three commonly used system identification methods, namely black box system identification method, gray box system identification method and white box system identification method [19]. White box system identification method is relatively intuitive and easy to understand. For systems whcih we obtain the state transfer function through physical analysis, such as spring motion system and circuit system whose structure is known, we can directly build a mathematical model of the system to conduct systematic research. Black box identification system indicates that the internal structure of the system is too complicated, or we are lack of the internal information about the system.

The purpose of system identification is to determine the parameters in the system model, so as to construct the simulation environment and train the reinforcement learning algorithm. In the process of system identification of the gimbal dynamic model, we find that it is difficult to solve the pitch by the least square method because of its nonlinear part. In addition, there is a certain delay linearity in the process of system execution, which is not modeled by the system. Therefore, we adopt the method of black box system identification. Considering that the model is a second-order nonlinear system, we set the black box model as a fourth-order linear system to approximate it. On the one hand, it can ensure that the black box system has sufficient fitting ability to the actual system, on the other hand, it can prevent the system from over fitting due to too little data.

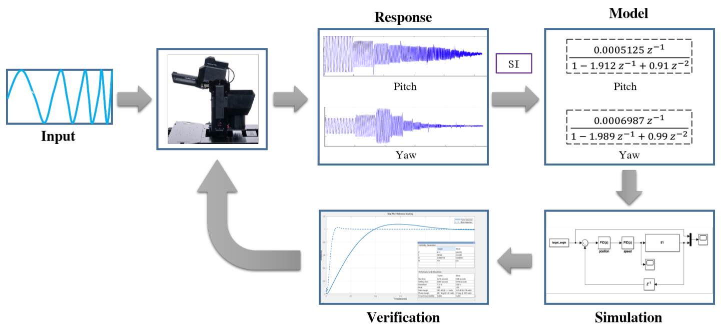
Input
Response
Pitch
Yaw
SI
Model
$\frac{0.0005125\,z^{-1}}{1-1.912\,z^{-1}+0.91\,z^{-2}}$
Pitch
$\frac{0.0006987\,z^{-1}}{1-1.989\,z^{-1}+0.99\,z^{-2}}$
Yaw
Simulation
Verification

Fig. 3. The system identification process of the gimbal dynamic model.

At this time we need to investigate the relationship between input and output of the system to predict the system function. In general, we will traverse the input and observe the response under various excitation, so as to ensure the rationality of the model. Finally, we generally use the step signal to verify the system model. In this paper, we use the black box method to build the shooting control model. The application conditions of the grey box identification method are between them, that is, we can know the structure but without parameters.

For the parameter identification of the bullet trajectory model, we record the firing velocity of the actual bullet and the position of the landing point at different distances, and obtain the initial velocity and angle disturbance distribution through maximum likelihood estimation.

D. Deep reinforcement learning

There are four basic components in reinforcement learning, which are state s (including environment state and self state), action a, state transition probability $P^a_{s\to s'}$(the probability that state changes from s to s' when state s and the action taken is a), and reward r. For the agent we study, in the current state s, it decides the action a by the policy π, and executes the action. Then, it will transfer to the new state s' with the $P^a_{s\to s'}$ probability and get the reward r. The policy is updated according to the reward situation, and the cycle repeats until our policy converges to maximum rewards.

Specific to shooting control of RMUA, the objective of the policy is to achieve a precise shot at armor of the enemy with the highest reward. In the case of a certain shooting speed, the shooting action of the agent is only determined by the shooting angle, so the action space is the rotation angle control variables of two axes of the platform. Since both the state of the infantry and the state of the target will affect the shooting action, the state space we set is also composed of two parts: the shooting angle of the past two moments and the target position at the current moment. We take the opposite number of Euclidean distance between the final bullet landing point and the center point of the armor as the reward function, so that when the agent finds the optimal policy, the reward function takes the maximum value at the same time. Considering the armor and the horizontal plane is almost vertical, we caculate the negative Euclidean distance of bullet and armor when they have the same x-coordinate. The reward r is as following.

$$r = -\sqrt{(Y_{target} - Y_{bullet})^2 + (Z_{target} - Z_{bullet})^2} \tag{8}$$

where (Y_{target}, Z_{target}) is the shooting target position, and (Y_{bullet}, Z_{bullet}) is the bullet landing point.

In this paper, two neural networks are used to realize shooting control policy. The action network inputs the current state of the agent to get the action at this moment, and the critic network estimates the possible reward under this state and action. The action network and the critic network we are using here are all three layers: an input layer, a hidden layer, and an output layer. The hidden layer has 256 neurons. ReLU is used as the activation function. The attenuation factor of the action network is 0.001, the attenuation factor of the critic network is 0.003. The input dimension of action network is 6, and the output dimension is 2.

We use PPO2 algorithm to speed up the convergence of the network. In deep reinforcement learning the objective function is shown as equation (9).

$$\begin{aligned} J(\theta) &= \int_I p_\theta(\tau) A^\tau(s_t, a_t) d\tau \\ &= \int_I p_\theta(\tau)(Q^\tau(s_t, a_t) - V^\tau(s_t, a_t)) d\tau \end{aligned} \tag{9}$$

$A^\tau(s_t, a_t)$ is the advantage function, $Q^\tau(s_t, a_t)$ is an estimate state-action value given by critic network while $V^\tau(s_t, a_t)$ is the state value which is a constant after sampling. PPO2 algorithm uses importance sampling and truncation function to improve sample data utilization and increase iteration step length. [20] The simplified objective function is shown as equation (10).

$$\begin{aligned} &J^{CLIP}_{\theta'}(\theta) = E_{(s_t, a_t \sim \pi_{\theta'})}[\min(\frac{P_\theta(a_t|s_t)}{P_{\theta'}(a_t|s_t)} A^{\theta'}(s_t, a_t), \\ &clip(\frac{P_\theta(a_t|s_t)}{P_{\theta'}(a_t|s_t)}, 1-\varepsilon, 1+\varepsilon) A^{\theta'}(s_t, a_t))] \end{aligned} \tag{10}$$

θ' represents the policy parameters that we sampled, while θ represents the parameters that we want to update. ε represents the restriction factor of truncation function, which is 0.2 in our experiment.

IV. Experiments and Results

In order to verify the accuracy of the model obtained by system identification and the effectiveness of deep reinforcement learning algorithm, experiments are carried out in simulation environment and real environment. During the experiment, the data required for model system identification are collected from the RMUA infantry platform. We first verify the results of model identification, and then test the gimbal control shooting accuracy based on deep reinforcement learning in simulation environment and real environment respectively.

A. Results of system identification

1) Dynamic model: The experiment conducts the identification of the platform according to the black box method. Firstly, 2-50Hz sweep signals are fed into the platform module by using the vehicle-mounted Xavier timer according to the actual situation. The parameters of the state model are fitted with least square algorithm. We get the state equation as follow

$$\begin{aligned} X_p(t+1) &= A_p X_p(t) + B_p U_p(t) \\ \alpha(t+1) &= C_p X_p(t) \end{aligned} \tag{11}$$

where X_p and U_p are the states and controls for pitch axis, and

$$A_p = \begin{bmatrix} 0.985 & -0.039 & 0.015 & -0.004 \\ 0.071 & 0.905 & 0.084 & 0.331 \\ 0.019 & -0.139 & -0.813 & 0.542 \\ 0.025 & 0.213 & -0.531 & -0.641 \end{bmatrix}, B_p = \begin{bmatrix} -7.782e-06 \\ -3.569e-05 \\ -0.00011 \\ -0.0003 \end{bmatrix} \tag{12}$$

$$C_p = \begin{bmatrix} 1183 & 12.61 & 0.1683 & -29.35 \end{bmatrix} \tag{13}$$

The state model for yaw axis is as the following.

$$\begin{aligned} X_y(t+1) &= A_y X_y(t) + B_y U_y(t) \\ \beta(t+1) &= C_y X_y(t) \end{aligned} \tag{14}$$

where X_y and U_y are the states and controls for yaw axis, and

$$A_y = \begin{bmatrix} 0.995 & -0.007 & 0.0002 & -0.008 \\ 0.052 & -0.267 & -0.551 & 0.696 \\ -0.006 & 0.422 & -0.799 & -0.419 \\ 0.049 & 0.639 & 0.228 & 0.156 \end{bmatrix}, B_y = \begin{bmatrix} 1.229e-06 \\ 2.354e-05 \\ -1.292e-07 \\ 8.957e-08 \end{bmatrix} \tag{15}$$

$$C_p = \begin{bmatrix} -8.353e+04 & -1378 & -372.7 & 207.4 \end{bmatrix} \tag{16}$$

Finally, we carry out the simulation of the system in Simulink based on the obtained system state parameters, and compare the step responses of the simulated system to the physical gimbal system. The results of pitch angle are shown in Fig. 4.

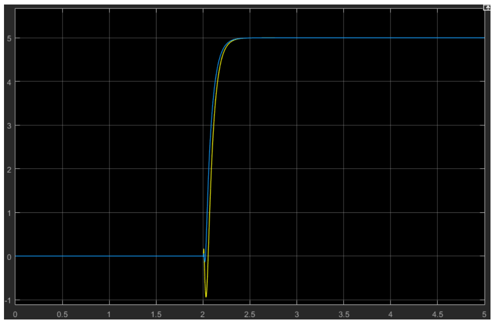
Fig. 4. The step responses from the physical system and simulated system.

The vertical axis represents angle, the horizontal axis represents time. The blue curve represents physical results, and the green curve represents simulation results. At this point, the simulation results are basically consistent with the actual situation, which means that the model parameters we set for the pitch axis are basically reasonable and can be used to build the simulation environment. After the same verification of yaw axis, the conclusion is consistent.

2) Bullet flight model: We carry out the experiment of the real environment and simulation environment, and verify the model with the distance of the falling point. The results are shown in Table I.

α represents the pitch angle, and β represents the yaw angle. d_1 represents the distance of simulation falling point, and d_2 represents the distance of real falling point. The error between the real point and the simulated point is less than 5%, which proves that our model reflects the motion of the bullet accurately.

TABLE I
THE RESULT OF THE BULLET FLIGHT MODEL

α	β	d_1(m)	d_2(m)	error
5°	5°	6.315	6.545	3.51%
5°	0°	6.308	6.510	3.10%
5°	-5°	6.311	6.417	1.66%
0°	5°	2.879	2.761	4.27%
0°	0°	2.876	2.846	1.05%
0°	-5°	2.883	2.893	0.35%
-5°	5°	1.957	1.955	0.12%
-5°	0°	1.946	2.017	3.52%
-5°	-5°	1.945	2.031	4.23%

B. Results of deep reinforcement learning based gimbal control

In the actual situation, the length and width of the armor are about 0.12m. At this time, we consider the maximum inner circle of the armor. We stipulate that the bullet within this range is considered to hit, otherwise it does not hit. That is, as long as the final reward value we get is greater than -0.06, the projectile can be considered to hit the target. After we substitute the obtained parameters into the simulation environment, the training process is obtained as shown in Fig. 5 and Fig. 6.

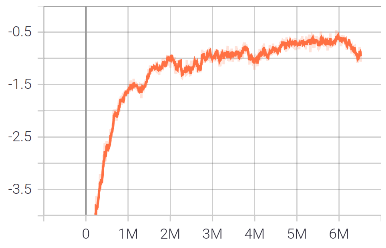

Fig. 5. Return during PPO training under static target shooting.

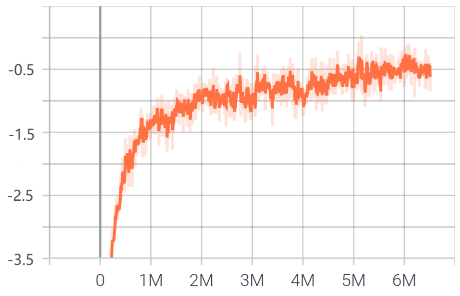

Fig. 6. Return during PPO training under dynamic target shooting.

It can be seen that after the final policy convergence, the result of dynamic shooting model is better than static model. In order to further verify the superiority of the algorithm, we compare our method to the traditional shooting control model which uses Kalman filter to tracking targets in the simulation environment and real environment. The results are shown in the Table II, Table III, Table IV and Table V.

In the simulation environment, the shooting accuracy of two methods is obtained after 2000 repeated experiments in the same situation. And in the real environment, the shooting accuracy of two methods is obtained after 50 repeated experiments

TABLE II
STATIC TARGET SHOOTING ACCURACY IN SIMULATION ENVIRONMENT

d(m)	PPO (%)	Kalman filter(%)
$1 \sim 2$	**83.80**	80.22
$2 \sim 3$	**84.75**	81.32
$3 \sim 4$	**84.35**	78.22
$4 \sim 5$	**84.55**	78.46

TABLE III
DYNAMIC TARGET SHOOTING ACCURACY IN SIMULATION ENVIRONMENT

d(m)	PPO(%)	Kalman filter(%)
$1 \sim 2$	**88.35**	76.84
$2 \sim 3$	**87.20**	72.68
$3 \sim 4$	**87.00**	68.96
$4 \sim 5$	**87.65**	68.84

TABLE IV
STATIC TARGET SHOOTING ACCURACY IN REAL ENVIRONMENT

d(m)	PPO(%)	Kalman filter(%)
$1 \sim 2$	78.00	**78.86**
$2 \sim 3$	76.00	**76.54**
$3 \sim 4$	72.00	**74.30**
$4 \sim 5$	66.00	**72.66**

TABLE V
DYNAMIC TARGET SHOOTING ACCURACY IN REAL ENVIRONMENT

d(m)	PPO (%)	Kalman filter(%)
$1 \sim 2$	**76.00**	70.44
$2 \sim 3$	**70.00**	68.36
$3 \sim 4$	**66.00**	62.68
$4 \sim 5$	**64.00**	60.56

in the same situation. d represents the distance between armor (the vertex of parabolic in dynamic situation) and the gimbal.

From the experimental results, the shooting control based on deep reinforcement learning in the simulation environment gets better result. Under the condition that distance is greater than 1m, it can achieve higher shooting accuracy and ensure good tracking of dynamic target compared with the traditional method based on Kalman filter. After transfering to the real robot, the accuracy suffers a sharp decline, but it still maintains better accuracy than traditional method under the dynamic target.

V. CONCLUSION

In this paper, aiming at the shooting control problem of moving objects with two DOF gimbal of RMUA infantry, we propose a pipeline combining system identification and deep reinforcement learning, which effectively reduces the difference between the simulation system and the real physical system. The experiments in the simulation environment and the real environment show that the proposed method has higher shooting accuracy than the Kalman filter method under static and dynamic targets.

REFERENCES

[1] Y. Chen, H. Li, R. Gao, and D. Zhao, "Boost 3-D object detection via point clouds segmentation and fused 3-D giou-l1 loss," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–12, 2020.
[2] H. Li, Y. Chen, Q. Zhang, and D. Zhao, "BiFNet: Bidirectional fusion network for road segmentation," *IEEE Transactions on Cybernetics*, pp. 1–12, 2021.
[3] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
[4] I. Clavera, J. Rothfuss, J. Schulman, Y. Fujita, T. Asfour, and P. Abbeel, "Model-based reinforcement learning via meta-policy optimization," *Conference on Robot Learning*, pp. 617–629, 2018.
[5] L. Xie, S. Wang, A. Markham, and N. Trigoni, "Towards monocular vision based obstacle avoidance through deep reinforcement learning," in *Robotics: Science and Systems Workshop 2017: New Frontiers for Deep Learning in Robotics*, 2017.
[6] D. Li, D. Zhao, Q. Zhang, and Y. Chen, "Reinforcement learning and deep learning based lateral control for autonomous driving [application notes]," *IEEE Computational Intelligence Magazine*, vol. 14, no. 2, pp. 83–98, 2019.
[7] H. Li, Q. Zhang, and D. Zhao, "Deep reinforcement learning-based automatic exploration for navigation in unknown environment," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 6, pp. 2064–2076, 2020.
[8] C. Tang and Y.-C. Lai, "Deep reinforcement learning automatic landing control of fixed-wing aircraft using deep deterministic policy gradient," in *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2020, pp. 1–9.
[9] C. Finn and S. Levine, "Deep visual foresight for planning robot motion," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 2786–2793.
[10] B. Burger, P. M. Maffettone, V. V. Gusev, C. M. Aitchison, Y. Bai, X. Wang, X. Li, B. M. Alston, B. Li, R. Clowes, N. Rankin, B. Harris, R. S. Sprick, and A. I. Cooper, "A mobile robotic chemist," *Nature*, vol. 583, no. 7815, pp. 237–241, 2020.
[11] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, 2019.
[12] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science Robotics*, vol. 5, no. 47, p. 5986, 2020.
[13] A. Punjani and P. Abbeel, "Deep learning helicopter dynamics models," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 3223–3230.
[14] A. Farchy, S. Barrett, P. MacAlpine, and P. Stone, "Humanoid robots learning to walk faster: from the real world to simulation and back," in *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, 2013, pp. 39–46.
[15] S. Zhu, A. Kimmel, K. E. Bekris, and A. Boularias, "Fast model identification via physics engines for data-efficient policy search." in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, 2018, pp. 3249–3256.
[16] Y. Chebotar, A. Handa, V. Makoviychuk, M. Macklin, J. Issac, N. Ratliff, and D. Fox, "Closing the sim-to-real loop: Adapting simulation randomization with real world experience," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 8973–8979.
[17] W. Li, "Research on target tracking control system based on vision," Ph.D. dissertation, Xi 'an University of Technology, 2019.
[18] X. Tang, C. Leng, Y. Guan, L. Hao, and S. Wu, "Development of tracking and control system based on computer vision for robomaster competition robot," in *2020 5th International Conference on Advanced Robotics and Mechatronics (ICARM)*, 2020, pp. 442–447.
[19] Y. P. Kondratenko, V. M. Kuntsevich, A. A. Chikrii, and V. F. Gubarev, *Advanced Control Systems: Theory and Applications*, 2021, pp. 54–59.
[20] D. Li, Z. Qiao, T. Song, and Q. Jin, "Adaptive natural policy gradient in reinforcement learning," in *2018 IEEE 7th Data Driven Control and Learning Systems Conference (DDCLS)*, 2018, pp. 605–610.