actuators

MDPI

Article

A Deep Reinforcement Learning Algorithm Based on Tetanic Stimulation and Amnesic Mechanisms for Continuous Control of Multi-DOF Manipulator

Yangyang Hou, Huajie Hong *, Dasheng Xu, Zhe Zeng, Yaping Chen and Zhaoyang Liu

College of Intelligence Science and Technology, National University of Defense Technology, Changsha 410073, China; houyangyang14@nudt.edu.cn (Y.H.); ixudasheng@163.com (D.X.); zengzhe@nudt.edu.cn (Z.Z.); yaping_chen2021@163.com (Y.C.); ZhaoyangLiuNUDT@163.com (Z.L.)
* Correspondence: honghuajie@nudt.edu.cn; Tel.: +86-138-7313-0046

Abstract: Deep Reinforcement Learning (DRL) has been an active research area in view of its capability in solving large-scale control problems. Until presently, many algorithms have been developed, such as Deep Deterministic Policy Gradient (DDPG), Twin-Delayed Deep Deterministic Policy Gradient (TD3), and so on. However, the converging achievement of DRL often requires extensive collected data sets and training episodes, which is data inefficient and computing resource consuming. Motivated by the above problem, in this paper, we propose a Twin-Delayed Deep Deterministic Policy Gradient algorithm with a Rebirth Mechanism, Tetanic Stimulation and Amnesic Mechanisms (ATRTD3), for continuous control of a multi-DOF manipulator. In the training process of the proposed algorithm, the weighting parameters of the neural network are learned using Tetanic stimulation and Amnesia mechanism. The main contribution of this paper is that we show a biomimetic view to speed up the converging process by biochemical reactions generated by neurons in the biological brain during memory and forgetting. The effectiveness of the proposed algorithm is validated by a simulation example including the comparisons with previously developed DRL algorithms. The results indicate that our approach shows performance improvement in terms of convergence speed and precision.

Keywords: multi-DOF manipulator; tetanic stimulation; amnesia mechanism; deep reinforcement learning

Citation: Hou, Y.; Hong, H.; Xu, D.; Zeng, Z.; Chen, Y.; Liu, Z. A Deep Reinforcement Learning Algorithm Based on Tetanic Stimulation and Amnesic Mechanisms for Continuous Control of Multi-DOF Manipulator. *Actuators* **2021**, *10*, 254. https://doi.org/10.3390/act10100254

Academic Editors: Ioan Doroftei and Karsten Berns

Received: 11 August 2021
Accepted: 20 September 2021
Published: 29 September 2021

1. Introduction

Deep Reinforcement Learning (DRL) is an advanced intelligent control method. It uses a neural network to parameterize the Markov decision processes (MDP). DRL has been successfully applied to the field of robots [1–4], Machine Translation [5], Auto-Driving [6], Target Positioning [7], and shows strong adaptability. There are two kinds of the DRL algorithms, one is based on value function such as Deep Q Network (DQN) [8] and Nature DQN [9]. The output of DRL algorithm based on value function is discrete state-action value. The other is policy-based, such as Deep Deterministic Policy Gradient (DDPG) [10], Trust Region Policy Optimization (TRPO) [11], Asynchronous Advantage Actor-Critic (A3C) [12], Distributed Proximal Policy Optimization (DPPO) [13,14], and Twin-Delayed Deep Deterministic Policy Gradient (TD3) [15]. For continuous action space, advanced search policy can improve the sampling efficiency of the underlying algorithms [10]. Many research results focus on improving exploration policy. Among them, Fortunato et al. [16] and Plppert et al. [17] put forward a noise-based exploration policy in adding noise to the action space and observation space. Bellemare et al. propose an exploratory algorithm based on pseudo-counting for efficient exploration. The algorithm evaluates frequency by designing a density model that satisfies certain properties and calculates pseudo-counts that are generalized in continuous space to encourage exploration [18]. In [19], Fox et al.

innovatively propose a framework, DORA, that uses two parallel MDP processes to inject exploration signals into random tasks. Reward-based exploration results in a slower approximation of functions and failure to provide an intrinsic reward signal in time. So, Badia et al. [20] propose a "never give up" exploration strategy (NGU), which is designed to quickly prevent repeated access to the same state in the same episode.

Due to the coupling and complexity, a multi-degree-of-freedom (multi-DOF) manipulator is a hot application background of DRL. The direct applications of DRL algorithms have, so far, been restricted to simulated settings and relatively simple tasks, due to their apparent high sample complexity [21]. Therefore, the application of DRL in multi-DOF robots requires a more effective DRL algorithm. Neuroscience provides a rich source of inspiration for new types of algorithms and architectures, independent of and complementary to the mathematical and logic-based methods and ideas that have largely dominated traditional approaches to AI [22].

In this paper, we design a new DRL algorithm named ATRTD3 based on the research results of neuroscience and the analysis of the human brain memory learning process. Our team shows a biomimetic view to speed up the converging process by biochemical reactions generated by neurons in the biological brain during memory and forgetting. We apply the neural network parameter updating mechanism with Tetanic stimulation and Amnesia mechanism to the DRL algorithm to further improve the efficiency in the application of manipulator.

2. Related Work

The advancement of DRL has led to the development of intelligent control of a multi-DOF manipulator in recent years. Kim et al. [23] propose a motion planning algorithm for robot manipulators using TD3, which designed paths are smoother and shorter after 140,000 episodes than those designed by Probabilistic Roadmap. Based on the classic DDPG algorithm, Zhang et al. smoothly add Gaussian parameters to improve the exploratory nature of the algorithm, dynamically sets the robot grasping space parameters to adapt to the workspace of multiple scales, and realizes the accurate grasping of the robot [24]. Robert Kwiatkowski et al. [25] used DL methods to make the manipulator build a self-model after 35 h of training. By comparing the application of DDPG and Proximal Policy Optimization (PPO) in the manipulator, Lriondo et al. [26] concluded that the current DRL algorithm could not obtain robust motion ability and acceptable training efficiency. The difficulty of applying DRL to the motion control of the multi-DOF manipulator lies in how to improve the exploration efficiency and improve the robustness of the output action of the manipulator. Therefore, it is necessary to get inspiration from the research results of neuroscience. For flexible manipulators, some research [27,28] are very interesting, which brings some help to kinematics modeling and control design in this paper.

Long-term potentiation (LTP) is a form of activity-dependent plasticity which results in a persistent enhancement of synaptic transmission. LTP has been a source of great fascination to neuroscientists since its discovery in the early 1970s [29] because it satisfies the criteria proposed by Donald Hebb for a synaptic memory mechanism in his influential book 'The Organization of Behavior' [30].

LTP is a persistent enhancement of excitatory synaptic transmission induced by some kinds of preceding operations of high-frequency stimulation (HFS) [31]. In LTP, stimulation changes the synaptic proteins, that is, changes the sensitivity of postsynaptic neurons to presynaptic neurons, thus changing the strength and efficiency of synaptic signal transmission. Memory formation is considered to be the result of long-term synaptic plasticities, such as long-term depression (LTD) and LTP [32].

LTP and LTD have another potentially important role in modern neuroscience, and that is the possibility that they may be exploited to treat disorder and disease in the human central nervous system (CNS). A variety of neurological conditions arise from lost or excessive synaptic drive due to sensory deprivation during childhood, brain damage, or disease [33]. Memory and forgetting are the stages that the human brain must go through in

the process of accepting knowledge and accumulating experience. This paper will modify the Actor-network module in DRL, and change the Actor-network module optimized by gradient descent into a network module with biological characteristics.

3. Methods

ATRTD3 is a DRL algorithm proposed in this paper to improve the motion ability of a multi-DOF manipulator. The innovation of the algorithm is to transform the research results of neuroscience into DRL. This algorithm is based on the Twin-Delayed Deep Deterministic Policy Gradient algorithm with Rebirth Mechanism (RTD3) [34] and improves the update of network weight parameters of Actor-network module. The highlight of the algorithm is that it uses tetanic stimulation and amnesia mechanism to randomly enhance and weaken the weighting parameters of the neural network, thus realizing the bionic update of the neural network. The Actor-network module obtained through the deterministic policy gradient needs to be further updated through the above two mechanisms. Compared with other DRL algorithms, ATRTD3 increases the network update part of biological characteristics and further expands the scope of exploration. Figure 1 shows the framework of the overall algorithm.

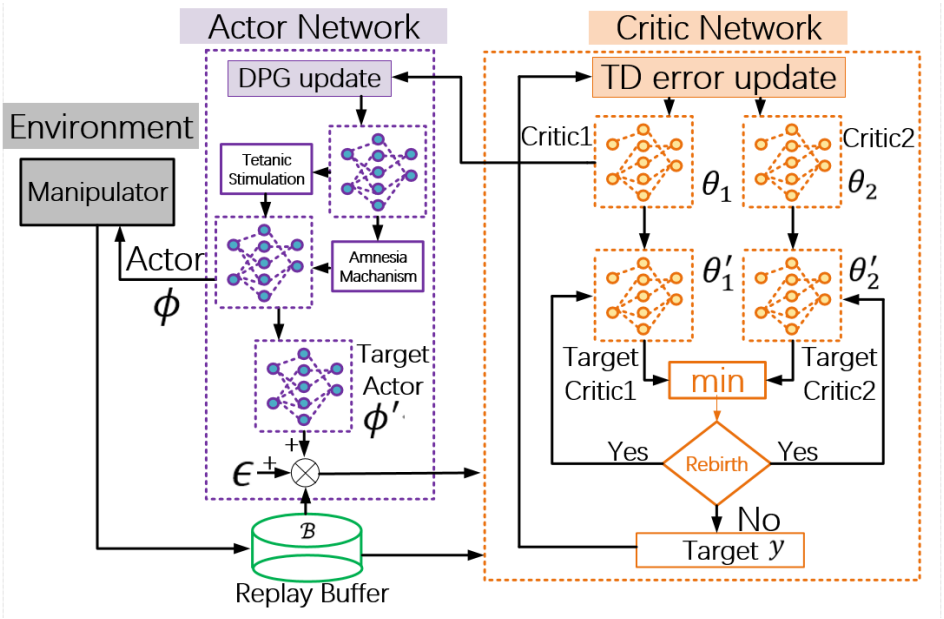


Figure 1. ATRTD3 algorithm framework.

The pseudo-code of ATRTD3 is shown in Appendix A at the end of this paper. The following is a detailed description of the Tetanic stimulation and Amnesia mechanism.

3.1. Tetanic Stimulation

Tetanic stimulation is the memory part of the Actor-network module. In the process of back propagation, the neural network obtains the updating quantity of parameters by gradient descent, to realize the iterative updating of the network. By evaluating the updating of network parameters, we can get which neural nodes' weights are enhanced and which ones are weakened. For the parameters of the strengthened neurons, there are also differences in the strengthened degree. Therefore, it is necessary to evaluate and sort the parameters of the strengthened neuron nodes, select the ranking of the strengthened degree, and then obtain the neuron nodes qualified for Tetanic stimulation, and conduct Tetanic stimulation on the above parameters of the neuron nodes to achieve LTP, as shown in Figure 2, and the specific pseudo code is shown in Algorithm 1. Directly modifying the parameters of neural nodes will directly affect the nonlinear expression results of neural networks. This is a kind of immediate influence, which will immediately show the change

effect of neuron weight in the continuous MDP, so we need to control this kind of influence in a reasonable range, and constantly update the parameters of the neural network in the iterative process. The designed Tetanic stimulation mechanism is nested in the fixed delay update step, which can exert the effect of Tetanic stimulation mechanism to a certain extent, and will not affect the overall effect of network update iteration, to ensure that the network converges towards the direction of performance improvement in the training process, and has the function of not weakening the attempt to explore the process.

Algorithm 1 Tetanic stimulation

1: Tetanic stimulation coefficient κ
2: Load Actor network ϕ, $W \leftarrow Actor.fc.weight$
3: Update Actor network $\phi' \leftarrow \phi$
4: Load New Actor network, $W^{new} \leftarrow Actor.fc.weight$
5: $\Delta W = W^{new} - W$
6: Select the serial number (row_{list}, col_{list}) of the T largest data in ΔW
7: **For** t = 1 **to** T **do:**
8: **If** $random(0,1) < \kappa$:
9: **If** $A.w(row_t, col_t) > 0$
10: $A.w(row_t, col_t) \leftarrow (1 + random(0, 0.01)) \times A.w(row_t, col_t)$
11: **else:**
12: $A.w(row_t, col_t) \leftarrow (1 - random(0, 0.01)) \times A.w(row_t, col_t)$
13: **End if**
14: **End if**
15: **End for**

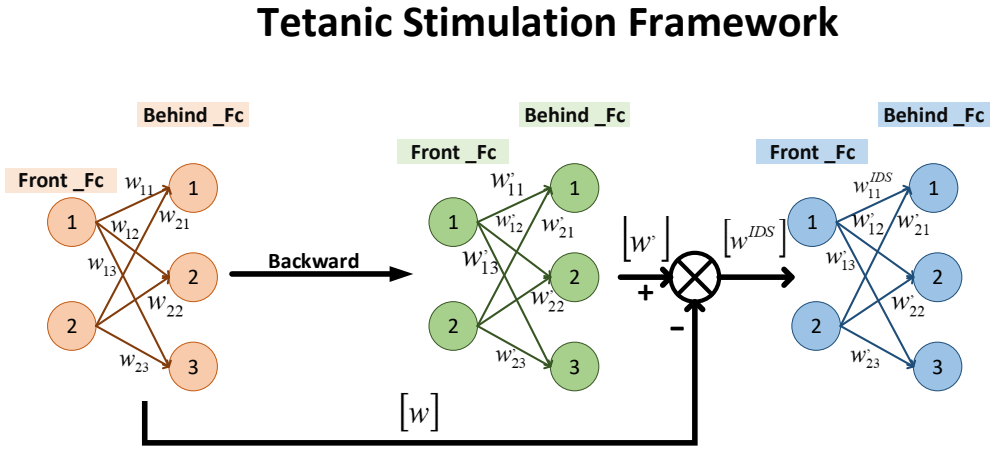


Figure 2. Schematic diagram of working mechanism of Tetanic stimulation.

3.2. Amnesia Mechanism

The Amnesia mechanism is the forgetting part of the Actor-network module. When there are problems in information transmission between neurons, synapses cannot normally perform the function of neurotransmitter transmission, the brain begins to have problems in information transmission, forgetting begins to occur, at this time, the huge redundant brain neural network begins to have problems, and some memory and logic units begin to have problems. Neuron function is not always in a stable and good working state, there will be all kinds of accidents, just as the world's best snooker players cannot guarantee that every shot will be accurate. So, there is forgetting in the whole period and process of neurons. For this reason, the Amnesia mechanism is added to the neural network by randomly selecting the neurons in the network with small probability, and then weakening the parameters of the neuron nodes. When using the Amnesia mechanism to weaken the representation ability of the neural network, the influence of this weakening must be controllable, that is to say, it will not affect the convergence trend of the neural network.

Therefore, to ensure that the influence of the Amnesia mechanism can be controlled, the weights of neurons are weakened by random force (a random percentage number) in this paper, as shown in Figure 3, and the specific pseudo code is shown in Algorithm 2.

Algorithm 2 Amnesia Framework

Load Actor network, $W \leftarrow A.w(Actor.fc.weight)$
N is the number of the W's node
For i = 1 **to** N:
    Random(0, 1) number ξ, Amnesia threshold value τ, Mutation coefficient k
    If $\xi < \tau$:
        $Actor.fc.weight[i] \leftarrow Actor.fc.weight[i] \times (1 - k \times random(0,1))$
    End if
End for

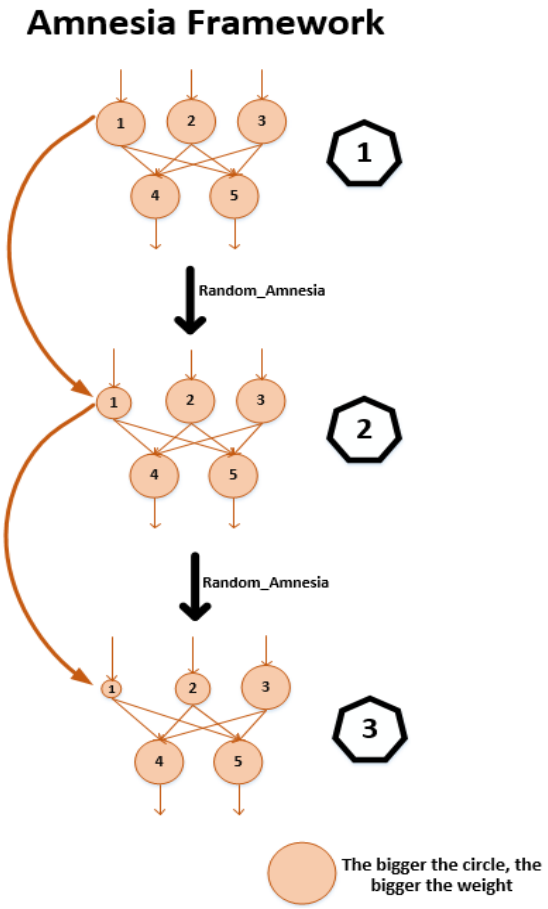


Figure 3. Schematic diagram of Amnesia framework.

4. Experiment

4.1. Experiment Setup

For the control problem of the multi-DOF manipulator, if we only consider the kinematics model of the manipulator, and the motion of the multi-DOF manipulator is regarded as a discrete process from one position of the end effector to another position, the DRL method of deterministic policy gradient, such as RTD3, can achieve good results. However, if the motion of the manipulator is regarded as a continuous motion process, a group of new inputs and outputs of DRL must be found. The idea adopted in this paper is to discretize the motion process of the manipulator in time, take the position deviation of the end-effector from the target, the angular velocity of the joints, and the angular of the joints as the input information of the DRL in this paper, and then take the angular acceleration

command for the next interval of the control joint as the output information, as shown in Figure 4. In this way, by controlling the angular acceleration of the joint, the problem of discrete processes in the previous control position process can be solved. However, this change will inevitably lead to the increase in model dimensions, which not only puts forward new requirements for the ability of the DRL algorithm but also puts forward new requirements for the redesign of the reward function.

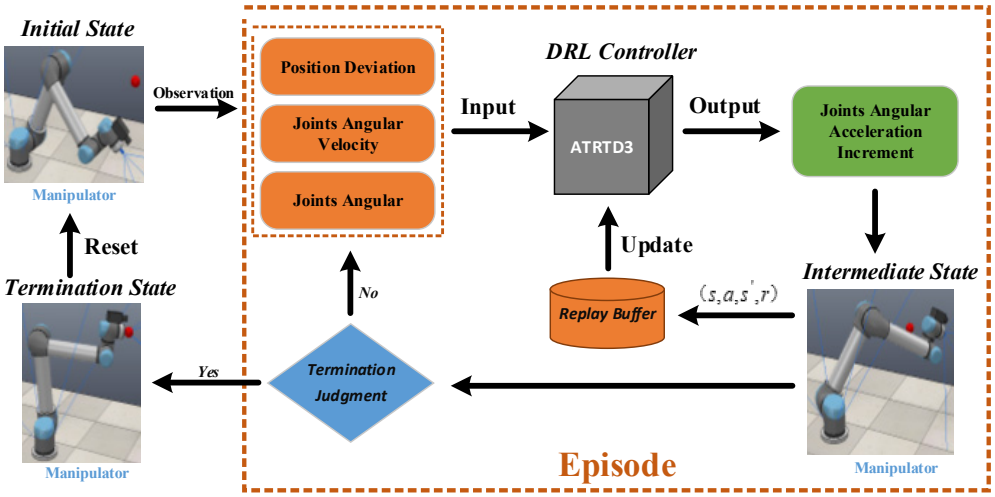


Figure 4. Schematic diagram of control flow.

UR manipulator is a representative manipulator in industrial production and scientific research. Therefore, this paper will use the structural size and joint layout of manipulator to establish the simulation manipulator in this paper.

4.2. Task Introduction

In this paper, the DRL algorithm is used to train a model controller of the multi-DOF manipulator. The model can control the angular acceleration of the joints of the manipulator so that the manipulator can start to move from the initial position of the workspace in the static state, and then move to the target position and stop. In the whole training process, the target position of the task is a fixed position in the workspace of the manipulator. The core of the task is that the manipulator can reach the target position and when it reaches the target position at the same time, each joint of the manipulator is at static state. Finally, the manipulator can reach the target position smoothly by controlling the angular acceleration of joints. In order to limit the boundaries of the entire task, the entire training process must be restricted. Each episode is divided into twenty steps. This setting mainly takes into account that the training convergence process takes a long time and the time of a single episode must be shortened. This task is a simulation experiment to test the convergence ability and learning ability of the improved algorithm ATRTD3.

4.3. Simulation Environment Construction

The DRL algorithm establishes the manipulator model through the standard DH [35] method and uses the positive kinematics solution to solve the spatial pose of the end effector through the joint angle. DH modeling method is a general modeling method of multi-link mechanism. Standard DH models are used for serial structure robots. In Table 1, we show the DH parameters of this manipulator, where a is the length of the link, d is the offset of the link, α is the twist angle of the link, θ is the joint angle. The units of a and d are meters, and the units of α and θ are radians.

Table 1. The D-H parameters of manipulator.

Joint	α	d	α	θ
Base	0	0.0892	$\pi/2$	θ_1
Shoulder	-0.4250	0	0	θ_2
Elbow	-0.3923	0	0	θ_3
Wrist1	0	0.1092	$\pi/2$	θ_4
Wrist2	0	0.0946	$-\pi/2$	θ_5
Wrist3	0	0.0823	0	θ_6

During the experiment, only base, shoulder, and elbow are controlled, wrist1, wrist2, and wrist3 were locked. Because the problem studied in this paper focuses on the position reaching ability of the manipulator, it can be realized only by using three joints, so the three joints of the wrist joints are locked. The homogeneous transformation matrix is established through D–H parameters, as shown in Equation (1).

$$ {}^{i-1}_{i}T = \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i)\cos(\alpha_i) & \sin(\theta_i)\sin(\alpha_i) & a_i\cos(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i)\cos(\alpha_i) & -\cos(\theta_i)\sin(\alpha_i) & a_i\sin(\theta_i) \\ 0 & \sin(\alpha_i) & \cos(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1} $$

The solution of the forward kinematics of the manipulator can be obtained by multiplying the homogeneous transformation matrix, as shown in Equation (2). In the base coordinate system {B}, the position of the end of the manipulator can be obtained.

$$ {}^{0}_{6}T = {}^{0}_{1}T{}^{1}_{2}T{}^{2}_{3}T{}^{3}_{4}T{}^{4}_{5}T{}^{5}_{6}T \tag{2} $$

In each episode, the target position is randomly generated in the workspace of the manipulator. In the experiment, the distance difference between the center position of the end effector and the target position in three directions (dx, dy, and dz), the angular velocity (ω_{Joint_Base}, $\omega_{Joint\_Shoulder}$, and ω_{Joint_Eblow}) and absolute angle of the first three joints (θ_{Joint_Base}, $\theta_{Joint\_Shoulder}$, and θ_{Joint_Eblow}) are used as the input of DRL, and the angular acceleration control commands of the base, shoulder and elbow ($\dot{\omega}_{Joint\_Base}$, $\dot{\omega}_{Joint\_Shoulder}$, and $\dot{\omega}_{Joint\_Elbow}$) are output by DRL. In order to ensure the safe operation of the virtual manipulator, the angular acceleration (θ/s^2) is limited, as shown in Equation (3).

$$ \dot{\omega}_i \in (-0.5, 0.5), i \in (Base, Shoulder, Eblow) \tag{3} $$

When the DRL outputs the angular acceleration control command $\dot{\omega}_i$, the joint angle increment $\Delta\theta_i$ obtained in this step is calculated by Equation (4) according to the interval time t = 0.1 s and the angular velocity of the previous step ω_{i}. The current joint angle $\theta_{i\_}$ is updated through the joint angle increment $\Delta\theta_i$ and the joint angle of the previous step θ_{i} in Equation (5). The position of the manipulator end effector in {B} coordinate is obtained by homogeneous transformation matrix ${}^{0}_{6}T$ calculation. The joint angular velocity is updated as shown in Equation (6).

$$ \Delta\theta_i = \omega\_{i}t + \frac{1}{2}\dot{\omega}_i t^2, i \in (Base, Shoulder, Eblow) \tag{4} $$

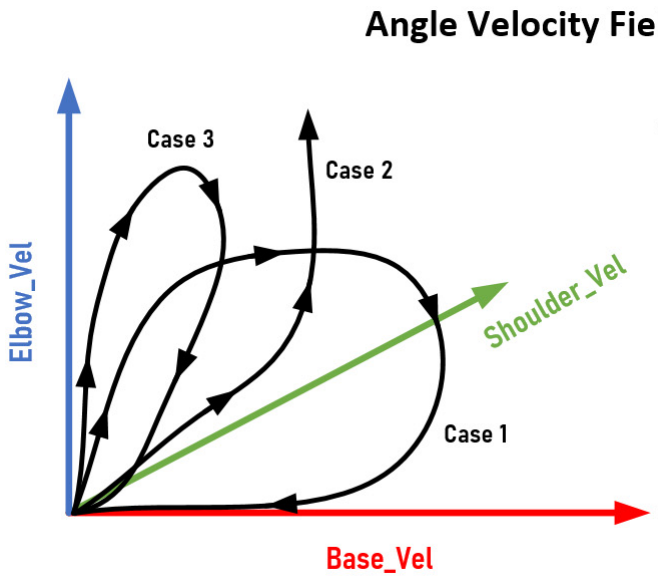
$$ \theta_{i\_} = \theta\_{i} + \Delta\theta_i \tag{5} $$

$$ \omega_{i\_} = \omega\_{i} + \dot{\omega}_i t \tag{6} $$

In this motion process, the DRL model sends the angular acceleration command (the output of the DRL algorithm) of the joints to manipulator according to the perceived environment and its state (the input of the DRL algorithm) and gives the termination command when judging the motion.

4.4. Rewriting Experience Playback Mechanism and Reward Function Design

The motion process of a multi-DOF manipulator is no longer multiple discrete spatial position points. As can be seen from Figure 5 below, this experiment completes the specified task by controlling the angular acceleration of the three joints. In the joint angular velocity field drawn by three joint angular velocities, in Case 1, the manipulator can reach the target position and stop, which means that the task is successfully completed; in Case 2, the angular velocity of the joint does not stop at all, or it passes through the target position quickly and fails to complete the task; for Case 3, although the manipulator stops in the scheduled time, the end of the manipulator does not reach the target position, and the task fails to complete. Figure 5 also shows that the task of this experiment is more difficult than the task of only achieving the goal through discrete spatial position.



Case	Target Point	Angle Velocity
1	Yes	$\sum_i \lvert\omega_i\rvert < \xi$
2	No	$\sum_i \lvert\omega_i\rvert > \xi$
3	No	$\sum_i \lvert\omega_i\rvert < \xi$

$i \in (Base, Shoulder, Elbow)$

Figure 5. Schematic diagram of the angular velocity field of joints. ξ is the threshold for determining joint stop.

Therefore, the design of the reward function must be reconsidered, that is to say, the angular velocity information of the joint must be introduced. In the whole process of the manipulator movement, each instance of acceleration and deceleration has an impact on the angular velocity of each joint when the termination condition is reached and the iteration of the round stops. Therefore, it is necessary to further improve the experience playback mechanism and change the experience pool storage. In other words, the final angular velocity of each joint should be shared by all the previous continuous movements. As shown in Equation (7), here the absolute value of the angular velocity of the joint will be taken, multiplied by the constant λ_i, divided by the number of steps T_{Stop}, and the corresponding reward value will be obtained.

$$R_{Joint\_Vel} = -\frac{\lambda_1|\omega_{Base}| + \lambda_2|\omega_{Shoulder}| + \lambda_3|\omega_{Elbow}|}{T_{Stop}}, \lambda_i > 0, i = \{1,2,3\} \quad (7)$$

This part of the reward as a negative reward is added to the corresponding reward in the experience pool, so as to realize the feedback of the joint angular velocity state in the neural network update parameters, as shown in Figure 6.

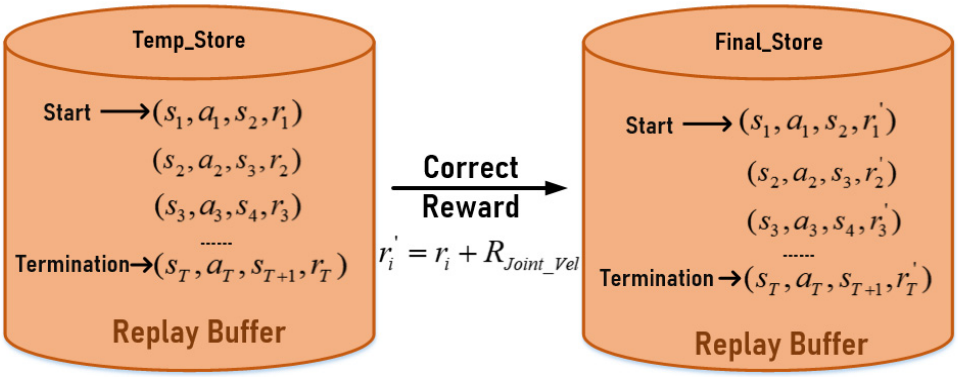


Figure 6. Schematic diagram of rewriting experience playback mechanism.

The design of the reward function in this paper adds the angular velocity reward part to the Step-by-Step reward function ($r_{Step-by-Step}$) [35]. The Step-by-Step reward function mainly includes two parts: the first part is the negative value of the Euclidean distance between the end of the manipulator and the target. The second part is the reward obtained by comparing the distance closed to the target position between the current position and the last position of the manipulator end during the movement. Therefore, the reward function in this paper is shown in Equation (8):

$$r' = \lambda_4 r_{Step-by-Step} + \lambda_5 R_{Joint\_Vel} \tag{8}$$

where λ_4 and λ_5 are two constants.

4.5. Simulation Experimental Components

In the application of the DRL algorithm, a problem that cannot be avoided is the random generation of a large number of neural network parameters. It is because of the randomness of the parameters that we cannot train and learn effectively in the face of specific tasks, so we need to explore a more efficient, faster convergence, and more stable algorithm framework to make up for this disadvantage. Since ATRTD3, RTD3, and TD3 are all improved and innovated on the basis of DDPG, the contrast group of this experiment chooses the above four algorithms. In the contrast experiment, we train with the same task and acquire the ability to solve the target task through learning and training. In the experiment, we specify two kinds of evaluation indexes. The first index is to calculate all the reward scores in an episode and divide them by the total number of steps in the episode to get the average score. The second index is to record the absolute value of the angular velocity of base, shoulder, and elbow at the end of an episode. Additionally, in the same group of experiments, in order to ensure a fair comparison of the practical application ability and model convergence ability of the four algorithms, we use the same initialization model as the test algorithm model.

5. Discussion

Through the comparison of DDPG, TD3, RTD3, and ATRTD3 in Figure 7a, we can clearly see the improvement effect of ATRTD3 learning ability. Therefore, we need to further analyze the evaluation index of the average score when the final distance error is roughly the same. From Figure 7a, we can see that the average score of ATRTD3 is higher than other algorithms when the final distance error is the same in purple areas A, B, and C, which indicates that the learning effect of ATRTD3 is better. The reason is that part of the reward function is the negative reward introduced by the absolute value of the angular velocity of each joint after each termination of an episode. Through purple areas A, B, and C of Figure 7b, ATRTD3 can better guide the multi-DOF manipulator to move to the target position and stop better, so ATRTD3 is more efficient and stable than other algorithms.

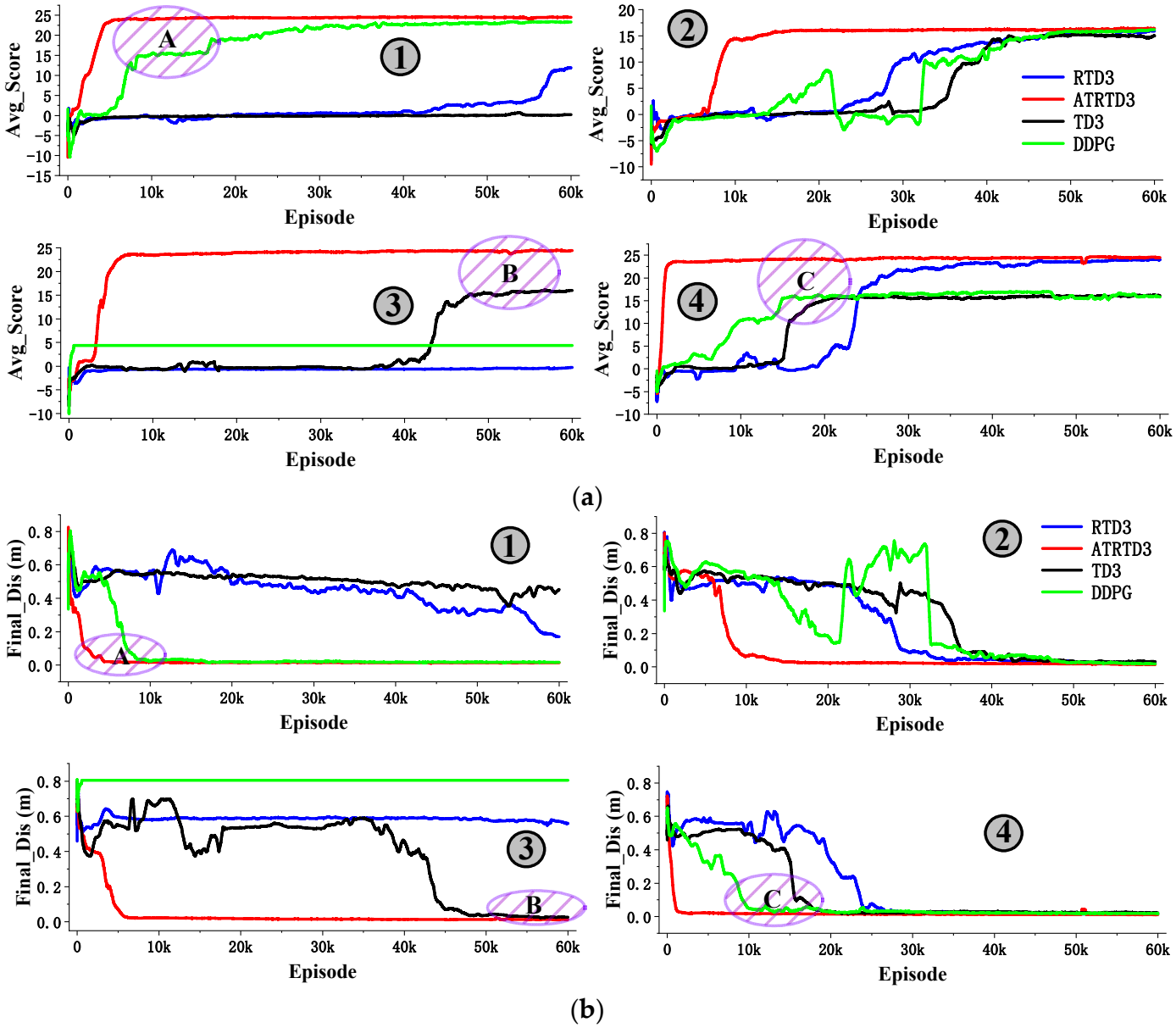
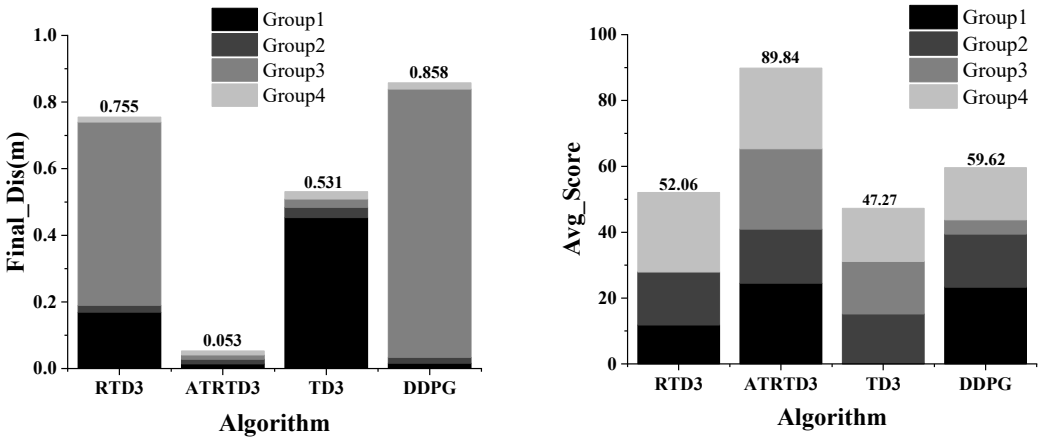


Figure 7. Four randomized experiments (Group1, Group2, Group3, and Group4) were conducted to evaluate the performance of four algorithms (RTD3, ATRTD3, TD3 and DDPG). There are Group1, Group2, Group3, and Group4 in the average score (Avg_score) in (**a**) and the final error distance (Final_Dis) in (**b**). Purple areas A, B, and C need special attention.

Secondly, through Figure 7, we can draw the conclusion that ATRTD3 shows stronger stability than other algorithms in the late training period. As can be seen in Figure 8, although DDPG has reached the score level close to that of ATRT3 through the later training, we can clearly see from the average score curve and the final error distance curve that ATRTD3 has better stability in the later training stage compared with DDPG, and the two curves of ATRTD3 are straighter, there are many spikes in the two curves of DDPG.

From Figure 9, we can see that ATRTD3 improves the average score by at least 49.89% compared with the other three algorithms. In terms of stability, ATRTD3 performs better, with an improvement of at least 89.27% compared with the other three algorithms.



(**a**) (**b**)

Figure 8. The results of four experiments (Group1, Group2, Group3, and Group4) of four algorithms (RTD3, ATRTD3, TD3, and DDPG) are compared in a stacked way. From left to right, (**a**) represents the final error distance; (**b**) represents the average score.

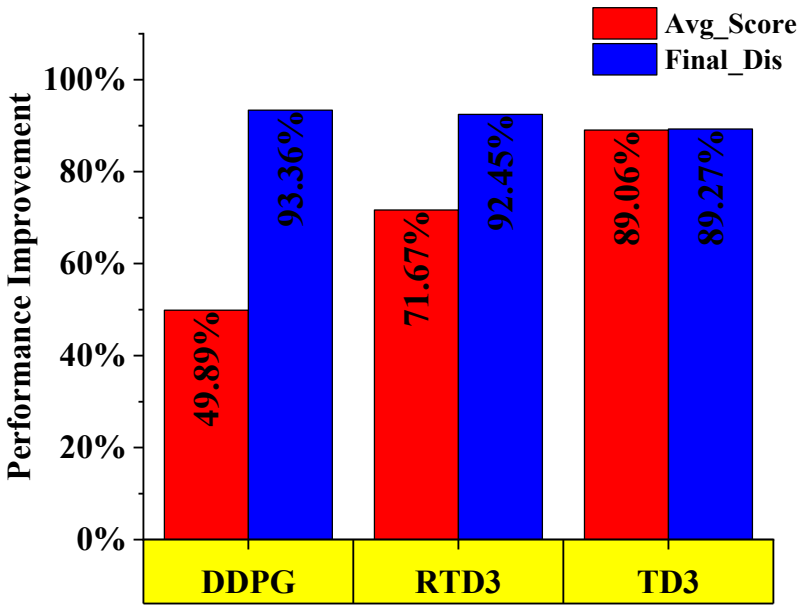


Figure 9. Combined with four groups of experiments (Group1, Group2, Group3, and Group4), the ATRTD3 algorithm improves the average score and final error distance performance of the model.

To further demonstrate the advantages of the ATRTD3 from the underlying control variables, we will collect the end angular speeds of the three joints during model training as shown in Figure 10a,b. In Figure 10a, we show the final angular velocity of the three joints of the high score model (Group1). In Figure 10b, we show the final angular velocity of the three joints of the arm controlled by a low score model (Group2). By placing a local magnification in each image, we can clearly see the final angular velocity of the three joints at the end of the training process, where the curve represented by the red color is the speed of each joint under the guidance of the ATRTD3 throughout the training process. ATRTD3 has obvious advantages over the other three algorithms.

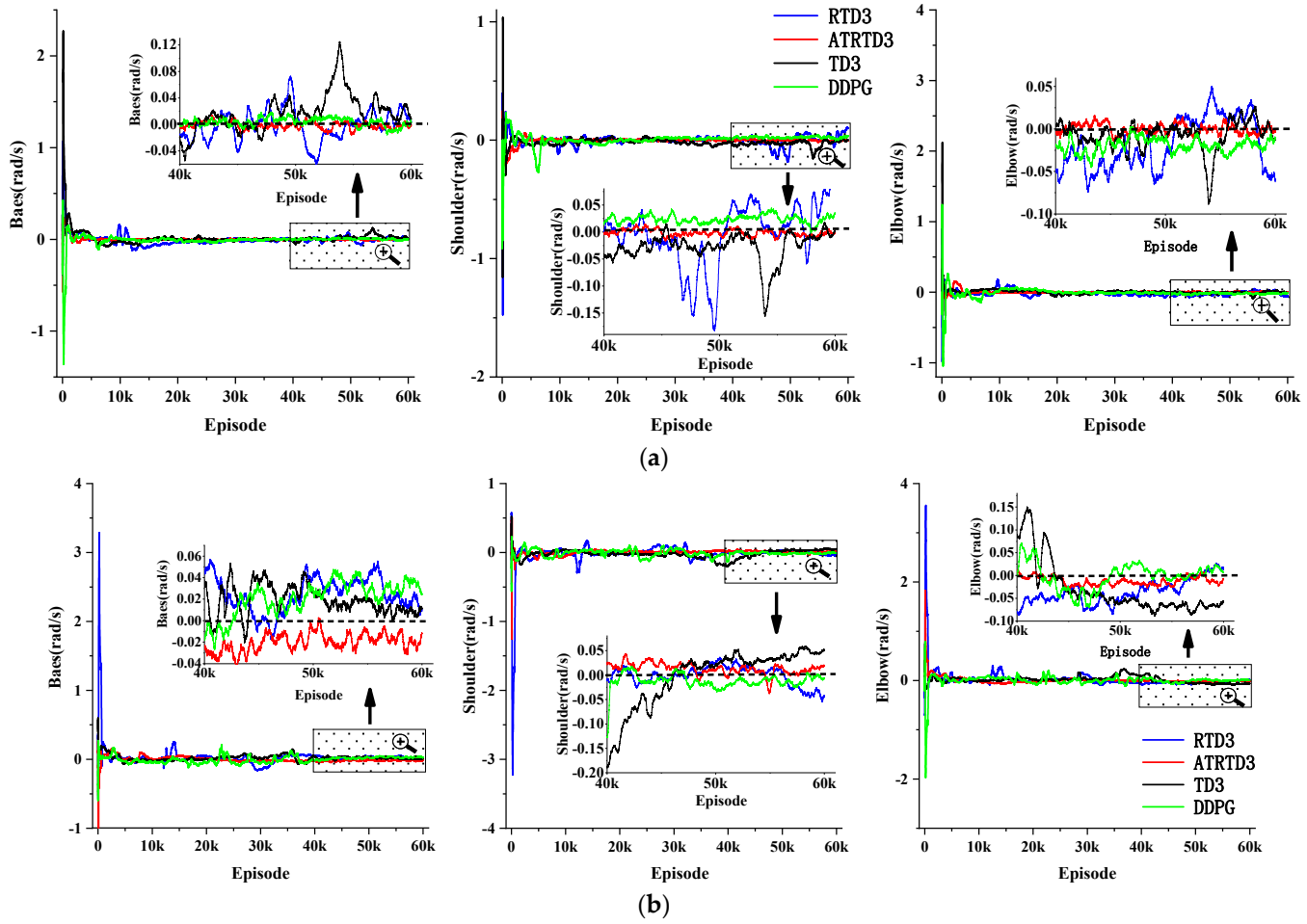


Figure 10. (**a**) The final angular velocity of the joints in the high score model Group1; (**b**) The final angular velocity of the joints in the low score model Group2.

The final joint angular velocity of the manipulator based on ATRTD3 is always around 0 (rad/s), with the minimum fluctuation and the best stability. In the high score model (Group1), only DDPG can achieve a score similar to ATRTD3 in the training score, so only the joint angular velocity of DDPG and ATRTD3 is compared. It can be seen from Table 2 that DDPG has an order of magnitude disadvantage in the comparison of the angular velocity of the other two joints except that the final angular velocity of the base joint is roughly the same as that of ATRTD3. By comparing the variances in Table 2, it is not difficult to find that ATRTD3 has absolute advantages in stability, which is generally one order of magnitude higher. Because the smaller the variance is, the more stable the training is. In the low model (Group2), the advantages of ATRTD3 cannot be seen in Table 3. After accumulating and comparing the angular velocity and angular velocity variance of the three joints, it can be seen from Figure 11 that ATRTD3 is still better than the other three algorithms.

Table 2. The mean and variance of joints angular velocity in local enlarged images of high score model (Group1).

Joint	Base		Shoulder		Elbow	
Angular Velocity	Average (rad/s)	Var	Average (rad/s)	Var	Average (rad/s)	Var
ATRTD3	-2.50×10^{-3}	2.80×10^{-5}	2.51×10^{-3}	3.09×10^{-5}	-3.90×10^{-4}	2.83×10^{-5}
RTD3	-4.65×10^{-3}	4.58×10^{-4}	-3.13×10^{-2}	1.07×10^{-3}	3.35×10^{-3}	6.81×10^{-4}
TD3	3.13×10^{-2}	7.26×10^{-4}	-2.79×10^{-2}	1.32×10^{-3}	-6.69×10^{-3}	4.73×10^{-4}
DDPG	2.66×10^{-3}	4.60×10^{-5}	2.52×10^{-2}	6.41×10^{-5}	-2.01×10^{-2}	3.95×10^{-5}

Table 3. The mean and variance of joints angular velocity in local enlarged images of low score model (Group2).

Joint	Base		Shoulder		Elbow	
Angular Velocity	Average (rad/s)	Var	Average (rad/s)	Var	Average (rad/s)	Var
ATRTD3	-1.71×10^{-2}	4.17×10^{-5}	8.04×10^{-3}	1.08×10^{-4}	-1.23×10^{-2}	6.18×10^{-5}
RTD3	3.03×10^{-2}	1.14×10^{-4}	-2.50×10^{-3}	5.87×10^{-4}	-1.73×10^{-2}	4.48×10^{-4}
TD3	1.39×10^{-2}	3.40×10^{-5}	3.67×10^{-2}	9.69×10^{-5}	-6.58×10^{-2}	6.95×10^{-5}
DDPG	3.20×10^{-2}	5.38×10^{-5}	-1.40×10^{-2}	7.07×10^{-5}	8.21×10^{-3}	1.27×10^{-4}

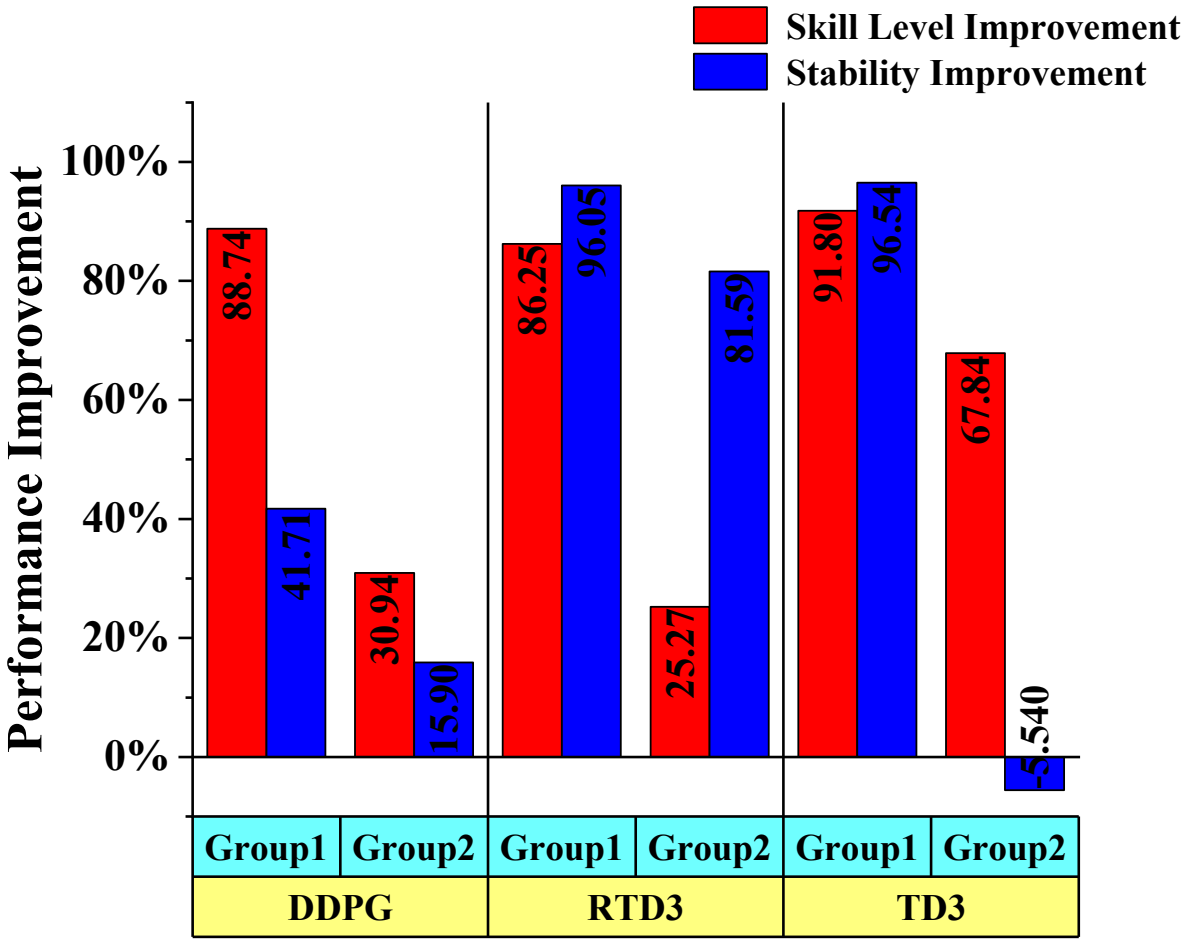


Figure 11. In the high score model (Group1) and low score model (Group2), the ATRTD3 algorithm compared with the other three algorithms in the control of joint angular velocity skill level and late training stability are two aspects of improvement effect.

Through Figure 11, we can see that ATRTD3 can significantly improve the skill level and stability compared with the other three algorithms. ATRTD3 generally improves the skill level by more than 25.27% and improves stability by more than 15.90%. Compared with TD3, the stability improvement of ATRTD3 in Group 2 is −5.54%. The main reason is that TD3 falls into a more stable local optimization, so the performance of ATRTD3 cannot be questioned under this index.

In the Actor neural network, the Tetanic stimulation mechanism changes the parameters of some neurons compulsively and improves the exploration space of action in a certain range. Through Tetanic stimulation, the weight of selected neurons can be increased again, which helps to improve the convergence speed of the neural network and speed up the training speed. The Amnesia mechanism makes the parameters of neurons changed compulsively, which is in line with the situation of biological cell decay.

6. Conclusions

In this paper, we propose an algorithm named ATRTD3, for continuous control of a multi-DOF manipulator. In the training process of the proposed algorithm, the weighting parameters of the neural network are learned using Tetanic stimulation and Amnesia mechanism. The main contribution of this paper is that we show a biomimetic view to speed up the converging process by biochemical reactions generated by neurons in the biological brain during memory and forgetting. The integration of the two mechanisms is of great significance to expand the scope of exploration, jump out of the local optimal solution, speed up the learning process and enhance the stability of the algorithm. The effectiveness of the proposed algorithm is validated by a simulation example including the comparisons with previously developed DRL algorithms. The results indicate that our approach shows performance improvement in terms of convergence speed and precision in the multi-DOF manipulator.

The ATRTD3 we proposed successfully applies the research results of neuroscience to the DRL algorithm to enhance its performance, and uses computer technology to carry out biological heuristic design through code to approximately realize some functions of the biological brain. In the future, we will further learn from neuroscience achievements, improve the learning efficiency of DRL, and use DRL to achieve reliable and accurate manipulator control. Furthermore, we will continue to challenge the scheme of controlling six joints to realize the cooperative control of position and orientation.

Author Contributions: Conceptualization, H.H. and Y.H.; methodology, Y.H.; software, Y.H.; validation, Y.H., D.X. and Z.Z.; formal analysis, Y.H.; investigation, Y.H.; resources, H.H.; data curation, Y.C.; writing—original draft preparation, Y.H.; writing—review and editing, Y.C. and Z.L.; visualization, Y.H.; supervision, H.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The authors exclude this statement.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Pseudo code of Algorithm A1.

Algorithm A1. ATRTD3

Initialize critic networks $Q_{\theta_1}, Q_{\theta_2}$,and actor network π_ϕ with random parameters θ_1, θ_2, ϕ
Initialize target networks $Q_{\theta_1'}, Q_{\theta_2'}, \pi_{\phi'}$
Target network node assignment $\theta_1 \to \theta_1'; \theta_2 \to \theta_2'; \phi \to \phi'$
Initialize replay buffer $\mathcal{B}$
for $t = 1$ **to** T **do**
    Amnesia framework
    Select action with exploration noise $a \sim \pi_\phi + \varepsilon, \varepsilon \sim N(0, \sigma)$
    Temporary store transition tuple (s, a, s', r) in $\mathcal{B}$
    Fix transition tuple $r' \leftarrow r + R_{joint\_Vel}$, angular velocity correction R_{Joint_Vel}
    Store transition tuple (s, a, s', r') in $\mathcal{B}$
    If $Sum_{\mathcal{B}} < mini\_batch$ **then**
        return
        Sample mini-batch of N transition (s, a, r, s') from $\mathcal{B}$
        $\widetilde{a} \leftarrow \pi_{\phi'}(s') + \varepsilon, \varepsilon \sim clip(N(0, \widetilde{\sigma}), -c, c)$
        $y = r + \min_{i=1,2} Q_{\theta_i'}(s', \widetilde{a})$
        Statistical calculation of Q value utilization ratio $w_{\theta_1'}, w_{\theta_2'}$
        If $\overline{w_{\theta_1'}} < Mini\_utilization$ **then**
            Rebirth target networks θ_1'
        End if
        If $\overline{w_{\theta_2'}} < Mini\_utilization$ **then**
            Rebirth target networks θ_2'
        End if
        Update critics $\theta_i \leftarrow \text{argmin}_{\theta_i} N^{-1} \sum (y - Q_{\theta_i}(s, a))^2$
        If t mod d **then**
            Update ϕ by the deterministic policy gradient:
            $\nabla_\phi J(\phi) = N^{-1} \sum \nabla_a Q_{\theta_1}(s, a)\big|_{a=\pi_\phi(s)} \nabla_\phi \pi_\phi(s)$
            Tetanic stimulation Framework
            Update target networks:
            $\theta_i' \leftarrow \tau\theta_i + (1 - \tau)\theta_i'$
            $\phi' \leftarrow \tau\phi + (1 - \tau)\phi'$
        End if
    End if
End for

References

1. Levine, S.; Pastor, P.; Krizhevsky, A.; Ibarz, J.; Quillen, D. Learning Hand-Eye Coordination for Robotic Grasping with Large-Scale Data Collection. In *International Symposium on Experimental Robotics*; Springer: Cham, Switzerland, 2016; pp. 173–184.
2. Zhang, M.; Mccarthy, Z.; Finn, C.; Levine, S.; Abbeel, P. Learning deep neural network policies with continuous memory states. In Proceedings of the International Conference on Robotics and Auto-mation, Stockholm, Sweden, 16 May 2016; pp. 520–527.
3. Levine, S.; Finn, C.; Darrell, T.; Abbeel, P. End-to-end training of deep visuomotor policies. *J. Mach. Learn. Res.* **2016**, *17*, 1–40.
4. Lenz, I.; Knepper, R.; Saxena, A. DeepMPC:learning deep latent features for model predictive control. In Proceedings of the Robotics Scienceand Systems, Rome, Italy, 13–17 July 2015; pp. 201–209.
5. Satija, H.; Pineau, J. Simultaneous machine translation using deep reinforcement learning. In Proceedings of the Workshops of International Conference on Machine Learning, New York, NY, USA, 24 June 2016; pp. 110–119.
6. Sallab, A.; Abdou, M.; Perot, E.; Yogamani, S. Deep reinforcement learning framework for autonomous driving. *Electron. Imaging* **2017**, *19*, 70–76. [CrossRef]
7. Caicedo, J.; Lazebnik, S. Active Object Localization with Deep Reinforcement Learning. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 11–18 December 2015; pp. 2488–2496.
8. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; Riedmiller, M. Playing Atari with Deep Reinforcement Learning. *arXiv* **2013**, arXiv:1312.5602.
9. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Figjeland, A.K.; Ostrovski, G.; et al. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533. [CrossRef]

10. Lillicrap, T.; Hunt, J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; Wierstra, D. Continuous control with deep reinforcement learning. *arXiv* **2015**, arXiv:1509.02971.
11. Schulman, J.; Levine, S.; Moritz, P.; Jordan, M.I.; Abbeel, P. Trust Region Policy Optimization. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 1889–1897.
12. Mnih, V.; Badia, A.; Mirza, M.; Graves, A.; Lillicrap, T.; Harley, T.; Silver, D.; Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In Proceedings of the International Conference on Machine Learning, New York, NY, USA, 19–24 June 2016; pp. 1928–1937.
13. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal Policy Optimization Algorithms. *arXiv* **2017**, arXiv:1707.06347.
14. Heess, N.; Dhruva, T.B.; Sriram, S.; Lemmon, J.; Silver, D. Emergence of Locomotion Behaviours in Rich Environments. *arXiv* **2017**, arXiv:1707.02286.
15. Fujimoto, S.; Hoof, H.V.; Meger, D. Addressing function approximation error in actor-critic methods. In Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; pp. 1587–1596.
16. Fortunato, M.; Azar, M.G.; Piot, B.; Menick, J.; Osband, I.; Graves, A.; Mnih, V.; Munos, R.; Hassabis, D.; Pietquin, O.; et al. Noisy Networks for Exploration. *arXiv* **2017**, arXiv:1706.10295.
17. Plappert, M.; Houthooft, R.; Dhariwal, P.; Sidor, S.; Chen, R.Y.; Chen, X.; Asfour, T.; Abbeel, P.; Andrychowicz, M. Parameter Space Noise for Exploration. *arXiv* **2017**, arXiv:1706.01905.
18. Bellemare, M.; Srinivasan, S.; Ostrovski, G.; Schaul, T.; Saxton, D.; Munos, R. Unifying count-based exploration and intrinsic motivation. *Adv. Neural Inf. Process. Syst.* **2016**, *29*, 1471–1479.
19. Choshen, L.; Fox, L.; Loewenstein, Y. DORA The Explorer: Directed Outreaching Reinforcement Action-Selection. *arXiv* **2018**, arXiv:1804.04012.
20. Badia, A.; Sprechmann, P.; Vitvitskyi, A.; Guo, D.; Piot, B.; Kapturowski, S.; Tieleman, O.; Arjovsky, M.; Pritzel, A.; Bolt, A.; et al. Never Give Up: Learning Directed Exploration Strategies. *arXiv* **2020**, arXiv:2002.06038.
21. Gu, S.; Holly, E.; Lillicrap, T.; Levine, S. Deep Reinforcement Learning for Robotic Manipulation with Asynchronous Off-Policy Updates. *arXiv* **2016**, arXiv:1610.00633.
22. Hassabis, D.; Kumaran, D.; Summerfield, C.; Botvinick, M. Neuroscience-Inspired Artificial Intelligence. *Neuron* **2017**, *95*, 245–258. [CrossRef]
23. MyeongSeop, K.; DongKi, H.; JaeHan, P.; JuneSu, K. Motion Planning of Robot Manipulators for a Smoother Path Using a Twin Delayed Deep Deterministic Policy Gradient with Hindsight Experience Replay. *Appl. Sci.* **2020**, *10*, 575.
24. Zhang, H.; Wang, F.; Wang, J.; Cui, B. Robot Grasping Method Optimization Using Improved Deep Deterministic Policy Gradient Algorithm of Deep Reinforcement Learning. *Rev. Sci. Instrum.* **2021**, *92*, 1–11.
25. Kwiatkowski, R.; Lipson, H. Task-agnostic self-modeling machines. *Sci. Robot.* **2019**, *4*, eaau9354. [CrossRef]
26. Iriondo, A.; Lazkano, E.; Susperregi, L.; Urain, J.; Fernandez, A.; Molina, J. Pick and Place Operations in Logistics Using a Mobile Manipulator Controlled with Deep Reinforcement Learning. *Appl. Sci.* **2019**, *9*, 348. [CrossRef]
27. Giorgio, I.; Del Vescovo, D. Energy-based trajectory tracking and vibration control for multilink highly flexible manipulators. *Math. Mech. Complex Syst.* **2019**, *7*, 159–174. [CrossRef]
28. Rubinstein, D. Dynamics of a flexible beam and a system of rigid rods, with fully inverse (one-sided) boundary conditions. *Comput. Methods Appl. Mech. Eng.* **1999**, *175*, 87–97. [CrossRef]
29. Bliss, T.V.P.; Lømo, T. Long-lasting potentiation of synaptic transmission in the dentate area of the anaesthetized rabbit following stimulation of the perforant path. *J. Physiol.* **1973**, *232*, 331–356. [CrossRef]
30. Hebb, D.O. *The Organization of Behavior*; Wiley: New York, NY, USA, 1949.
31. Thomas, M.J.; Watabe, A.M.; Moody, T.D.; Makhinson, M.; O'Dell, T.J. Postsynaptic Complex Spike Bursting Enables the Induction of LTP by Theta Frequency Synaptic Stimulation. *J. Neurosci.* **1998**, *18*, 7118–7126. [CrossRef] [PubMed]
32. Dayan, P.; Abbott, L.F. *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*; The MIT Press: New York, NY, USA, 2001.
33. Bliss, T.V.P.; Cooke, S.F. Long-Term Potentiation and Long-Term Depression: A Clinical Perspective. *Clinics* **2011**, *66* (Suppl. 1), 3–17. [CrossRef]
34. Hou, Y.Y.; Hong, H.J.; Sun, Z.M.; Xu, D.S.; Zeng, Z. The Control Method of Twin Delayed Deep Deterministic Policy Gradient with Rebirth Mechanism to Multi-DOF Manipulator. *Electronics* **2021**, *10*, 870. [CrossRef]
35. Denavit, J.; Hartenberg, R.S. A Kinematic Notation for Lower-Pair Mechanisms. *J. Appl. Mech.* **1955**, *77*, 215–221. [CrossRef]