Hindawi
Scientific Programming
Volume 2022, Article ID 6385972, 14 pages
https://doi.org/10.1155/2022/6385972

Research Article

Research on the Agricultural Machinery Path Tracking Method Based on Deep Reinforcement Learning

Hongchang Li,[1] Fang Gao,[1] and GuoCai Zuo [1,2]

[1]*Changzhou Vocational Institute of Mechatronic Technology, College of Vehicle Engineering, Changzhou, Jiangsu 213164, China*
[2]*Hunan Software Vocational and Technical University, Hunan Xiangtan 411100, China*

Correspondence should be addressed to GuoCai Zuo; zuoguocai@hnsoftedu.com

Received 19 December 2021; Revised 4 January 2022; Accepted 13 January 2022; Published 22 March 2022

Academic Editor: Baiyuan Ding

With the rapid development of information technology, industry and service industries have achieved rapid development in recent years. Then, looking at the development of agriculture, the popularity of informatization lags far behind industry and service industries, directly hindering the digital development of agriculture. Starting from the current agricultural machinery driving operation scene, this paper carried out a simplified research on the traditional agricultural machinery driving operation method through the agricultural machinery kinematics model, and based on the related theory of deep reinforcement learning to study the agricultural machinery path tracking in the agricultural operation scene, it carried out the controller design, built the agricultural machinery autonomous path tracking framework operating mechanism under deep reinforcement learning, and further researched through experimental design and found that the agricultural machinery autonomous path tracking control can achieve better automatic control after empirical learning. I-DQN algorithm enables agricultural robots to adapt to the environment faster when performing path tracking, which improves the performance of path tracking. It has important guiding significance for further promoting the automatic navigation and control of agricultural machinery to realize the efficient operation of agricultural mechanization.

1. Introduction

Automatic navigation control of agricultural machinery is a key technology to support precision agriculture. This technology can improve the working accuracy and efficiency of agricultural machinery, so that the driver can get rid of long-time tired and repetitive driving work and have enough time to monitor and operate agricultural machinery. Therefore, the automatic navigation control of agricultural machinery has broad development prospects.

The path tracking methods that are at the core of the automatic navigation control of agricultural machinery mainly include model-based control methods and model-independent control methods. In terms of model-based control methods, related scholars have separately studied the path following control methods based on the kinematics model and dynamics model of agricultural machinery [1–9]. However, among these methods, the method based on the kinematics model is mainly to approximate the model with a small angle linearization and design the controller under the assumption of constant speed. This introduces not only linearization error, but also the controller's performance when the speed changes. Robustness also deteriorates; while the control method based on the dynamic model can fully consider the dynamic characteristics of agricultural machinery, the dynamic model parameters are difficult to obtain online and in real time. In terms of model-independent control methods [10–15], the online adaptive determination of the forward-looking distance in the pure tracking method has not been well solved although the intelligent method has some human-like intelligence and incomparable traditional control methods. It has linear mapping ability, but its design requires certain experience knowledge and complex learning and training process. Aiming at the outstanding advantages of intelligent methods in agricultural machinery path control, this paper proposes

an agricultural machinery path tracking method based on deep reinforcement learning. The research of this method has certain practical significance for the development of efficient agricultural operation methods.

2. Related Theories

The research in this paper will use the deep combination of reinforcement learning and deep learning and make full use of the decision-making advantages of reinforcement learning and the perceptual advantages of deep learning [13, 16, 17] to carry out research. In deep reinforcement learning, reinforcement learning is used to define problems and optimization goals, deep learning is used to solve strategy functions or value functions, and backpropagation algorithms are used to optimize the objective function. To a certain extent, deep reinforcement learning has general intelligence to solve complex problems.

2.1. Deep Learning. Deep learning is derived from the idea of artificial neural network, which combines low-level features to form higher-level features and attribute categories. The most basic unit of artificial neural network is neuron, also known as perceptron. A deep neural network is called a multilayer perceptron. The difference from a single-layer perceptron is that it adds multiple hidden layers and can have multiple outputs. In the hidden layer, more complex feature information can be learned and multiple values can be output. It also enables the neural network model to solve more types of problems, such as classification, regression, dimensionality reduction, and clustering. At the same time, combining deep neural networks with different activation functions can further enhance the expressive ability of the model [13, 16, 17].

The deep neural network model is shown in Figure 1. The structure can be divided into input layer, hidden layer, and output layer. The input layer refers to information obtained through sensors or from the environment, such as radar data of agricultural intelligent harvesting vehicles. Each hidden layer is a feature level, in which each neuron represents a feature attribute. The output of the output layer is the required variables, such as the angular velocity and linear velocity of agricultural intelligent harvesting vehicles.

In DNN, each layer of neural network is fully connected; that is, the neurons in each i+1 layer are connected by the second layer of neurons. Assume that there are m neurons in the $l-1$ layer network, and W_{ij} represents the weight between the jth neuron in layer 1 and the kth neuron in $l-1$ layer, b_j^l is the bias of the kth neuron in the lth layer, and $\sigma(z)$ is the activation function. Then, for the output a_j^l of the jth neuron of the lth layer, there are

$$\begin{aligned} a_j &= \sigma\left(z_j^l\right) \\ &= \sigma\left(\sum_{k=1}^{m} w_{jk}^l a_k^{l-1} + b_j^l\right). \end{aligned} \tag{1}$$

The above process is the forward propagation of the neural network, but to optimize the parameters of the neural network, backpropagation is required. In order to calculate the error between the model output and the real training sample output, the neural network needs to first define the loss function for training, as defined in

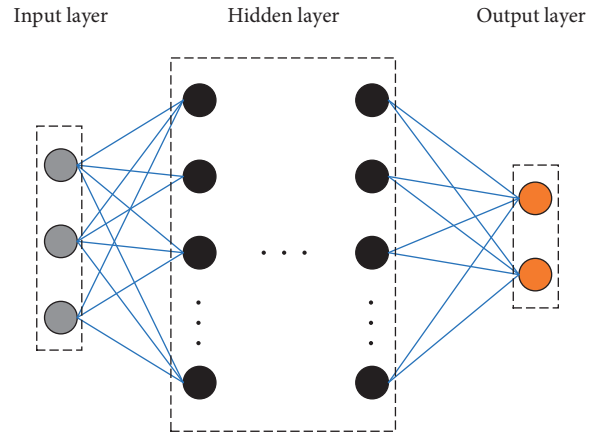


Figure 1: Deep neural network model.

$$J(W, b, x, y) = \frac{1}{2} a^L - y_2^2. \tag{2}$$

Finally, the error is used to update the weight of each neuron, and finally a better model is obtained. This is the process of backpropagation of the deep neural network.

2.2. Reinforcement Learning. Different from deep learning that focuses on perception and expression, reinforcement learning focuses on finding problem-solving strategies [18, 19]. Reinforcement learning is mainly composed of agent and environment. Since the interaction between the agent and the environment is similar to the interaction between the organism and the environment, it can be considered that reinforcement learning is a general learning framework, which represents the future development trend of general artificial intelligence algorithms [20, 21].

The basic framework of reinforcement learning is shown in Figure 2. Agents interact with the environment through states, actions, and rewards. Suppose that the state of the environment at time t in Figure 2 is denoted as s_t, and the agent performs a certain action a_t in the environment. At this time, the action a_t changes the original state of the environment and makes the agent reach a new state s_{t+1} at time t+1. In the new state, the environment generates a feedback reward r_t to the agent. The agent performs a new action a_{t+1} based on the new state s_{t+1} and the feedback reward r_{t+1} and iteratively interacts with the environment through feedback signals [22].

The ultimate goal of the above process is to maximize the cumulative reward for the agent. Equation (3) is the calculation process of the cumulative reward G.

$$G = r_1 + r_2 + \cdots + r_n. \tag{3}$$

In the above process, the rule of selecting actions according to the state s and the reward r is called the strategy

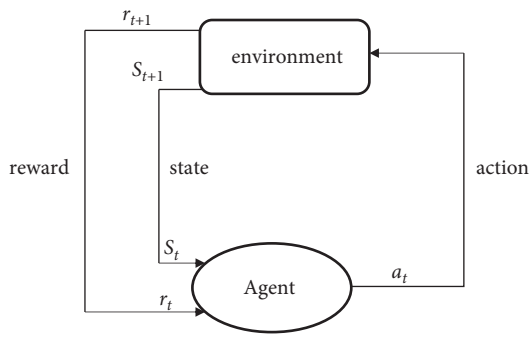


Figure 2: The basic framework of reinforcement learning.

π, where the value function v is the expectation of the cumulative reward.

Reinforcement learning is to continuously perform trial-and-error learning according to the feedback information of the environment and then adjust and optimize its own state information. The purpose is to find the optimal strategy or the maximum reward.

There are two types of environments in which an agent is located [23]: one is that the environment is known, which is called model-based; the other is that the environment is unknown, which is called model-free.

The relationship between model-based tasks and model-free tasks is shown in Figure 3. The line following agricultural robot shown in Figure 3(a) controls its walking by sensing the black course on the ground through sensors. Since the black route on the ground is planned in advance and the surrounding environment is also controllable and known, it can be regarded as a model-based task. Figure 3(b) is the autopilot system of a car. In the real traffic environment, many things cannot be estimated in advance, such as the behavior of passers-by, the trajectory of passing vehicles, and other emergencies, so it can be regarded as a model-free task.

2.3. Deep Q-Learning (DQN) Algorithm. The DQN algorithm is a famous work of the Google DeepMind team. They used reinforcement learning to propose a deep learning network model for solving control strategy problems, opening a new era of deep reinforcement learning [24–27].

The Q-learning algorithm stores Q values in the form of Q tables, as shown in Figure 4. This method of storing the Q value can handle maze problems when the state space and action space are very small, but when the problem has a large action or state space, the method of applying the Q table will cause a very large amount of data. The DQN algorithm combines Q-learning and deep learning algorithms, using deep convolutional nerves as shown in Figure 5.

The DQN algorithm has made the following improvements on the basis of the reinforcement learning algorithm:

(1) DQN uses a deep neural network to simulate the Q value function. The value function here corresponds to the weight θ of each layer in the convolutional neural network, that is, $Q(s,a;\theta) \approx Q^{\pi}(s,a)$. In this way, the update process of the Q value function is essentially an update of the weight θ of the neural network. When the parameter θ of the neural network is determined, the value function Q is also determined.

(2) Use experience playback technology to train neural networks. The deep neural network used by DQN is a supervised neural network model. The input data needs to be independent of each other and meet the same distribution. Since the data collected by the agent in the environment is continuous, there is a correlation between adjacent data. When the algorithm uses a set of continuous data for training, the direction of gradient descent will become the same. Calculating the gradient under the same training step size may cause the result to not converge. The experience playback mechanism puts the data collected by the agent into a memory bank, then uniformly randomly samples from the memory bank, and extracts the data from it for neural network training. By using experience replay, the behavior distribution can be averaged in its many previous states, thus smoothing the learning process and avoiding fluctuations or divergence of parameters. At the same time, assign priority to each conversion in the experience replay memory, which can greatly improve the learning efficiency compared with the uniform sampling from the experience replay memory.

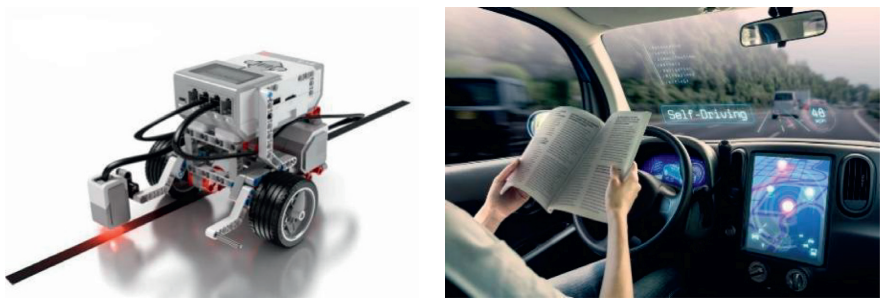
(3) The Q target network is set up to calculate the TD error. When using the convolutional neural network to approximate the Q value network, the parameter θ is processed by the gradient descent method, and the update process is

$$\theta_{i+1} = \theta_i + \alpha r + \gamma \max_{a'} Q(s', a'; \theta) - Q(s, a; \theta) \nabla Q(s, a; \theta). \tag{4}$$

In (4), $r + \gamma \max_{a'} Q(s', a'; \theta)$ is called the TD target, and the network used in calculating the TD target is called the target network. The neural network used to approximate the Q value function is called the estimation network. From the above formula, it can be seen that the parameters used by the target network are the same as the parameters of the estimated Q network, so that the results obtained by the calculation will have relevance. The training results of reinforcement learning are unstable. To solve this problem, the DQN algorithm expresses the parameters of the target network as θ^{-}. In the update of the neural network, the parameter θ of the estimated network is updated in real time, and the parameter θ^{-} of the target network is obtained by assigning the parameters of the estimated network to the target network after N rounds of iterations, so (4) changes to

$$\theta_{i+1} = \theta_i + \alpha r + \gamma \max_{a'} Q(s', a'; \theta^{-}) - Q(s, a; \theta) \nabla Q(s, a; \theta). \tag{5}$$

In the update of the neural network, the loss function is defined by the mean square error:



(a) (b)

Figure 3: Specific examples of model-based and model-free. (a) Line patrol agricultural machinery robot. (b) Autopilot system.

	a1	a2	a3	a4
s1	Q (1, 1)	Q (1, 2)	Q (1, 3)	Q (1, 4)
s2	Q (2, 1)	Q (2, 2)	Q (2, 3)	Q (2, 4)
s3	Q (3, 1)	Q (3, 2)	Q (3, 3)	Q (3, 4)
s4	Q (4, 1)	Q (4, 2)	Q (4, 3)	Q (4, 4)

Figure 4: Q table of Q-learning algorithm.

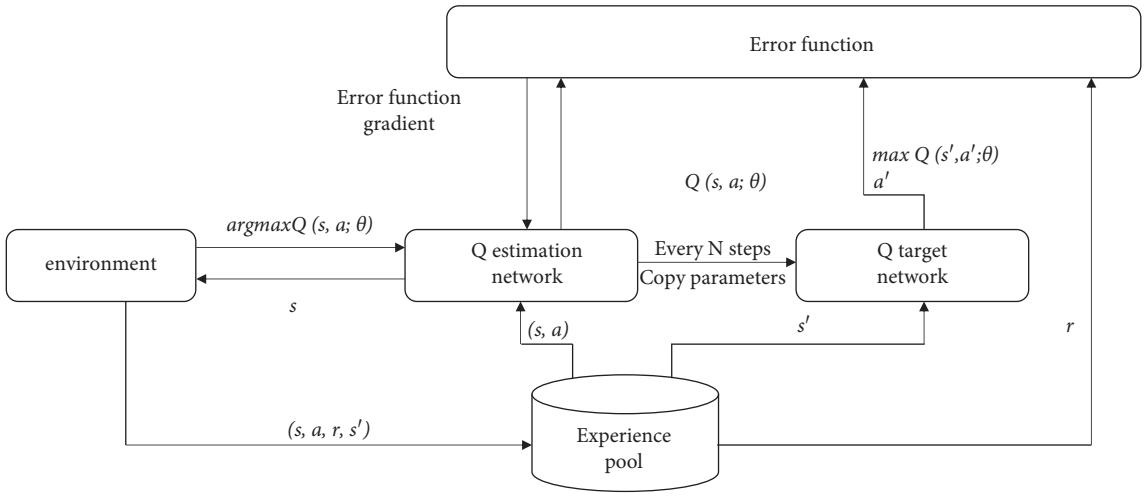


Figure 5: DQN algorithm principle diagram.

$$L(\theta) = E\left[\left(r + \max_{a'} Q(s', a'; \theta^{-}) - Q(s, a; \theta)\right)^2\right]. \quad (6)$$

Error function gradient:

$$\nabla L(\theta) = E\left[\left(r + \max_{a'} Q(s', a'; \theta^{-}) - Q(s, a; \theta)\right)\nabla Q(s, a; \theta)\right]. \quad (7)$$

After updating the network of (7) and obtaining the value of $Q(s, a; \theta)$, you can use $\nabla Q(s, a; \theta)$ to obtain the optimal Q value for the nerve of (5).

3. Agricultural Machinery Kinematics Model

3.1. Behavioral Learning Theory. Considering the application of agricultural machinery in actual agricultural land, agricultural machinery should have high flexibility and stability in complex environments. Therefore, this paper adopts a four-wheel agricultural machinery movement model, which provides power for the agricultural machinery movement through two rear wheels. The two front wheels adopt different steering angles to ensure the smooth steering of the mobile agricultural machinery. The movement model is shown in Figure 6.

When the agricultural machinery system is turning, its turning process can be simplified into a bicycle model as shown in Figure 7.

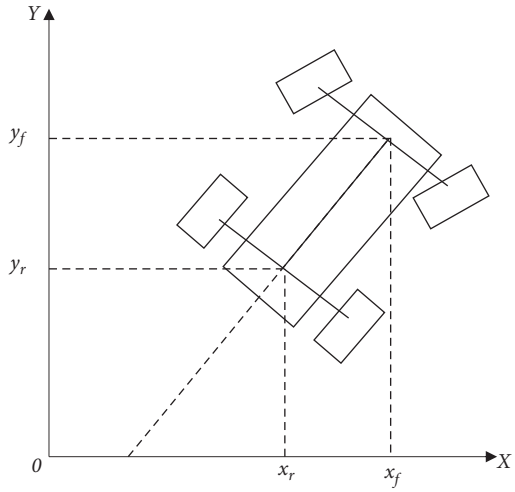


Figure 6: Schematic diagram of agricultural machinery steering.

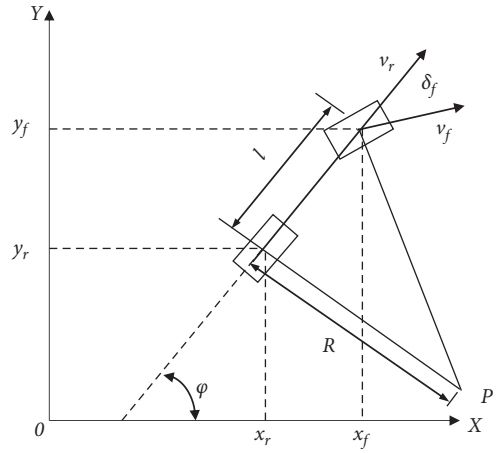


Figure 7: Model of the agricultural machinery steering bicycle.

In the map coordinate system, (x_r, y_r) and (x_f, y_f), respectively, represent the coordinates of the center position of the two rear wheels of the agricultural machine and the coordinates of the center position of the two front wheels, and v_r and v_f, respectively, represent the center position of the front wheel and the center of the rear wheel of the agricultural machine. The speed of the position, φ, is the heading angle of the agricultural machine in the map coordinate system, δ_f is the deflection angle of the front wheel of the agricultural machine, and l is the distance between the center position of the front wheel and the center position of the rear wheel. P is the instantaneous turning center of the rear wheel center position of the agricultural machinery during the turning process; R is the turning radius of the center point of the rear wheel of the agricultural machinery, assuming that the deflection angle of the center of mass of the moving agricultural machinery does not change during the turning process; that is, the instantaneous turning radius and the radius of curvature of the path are the same. Then the speed of the rear wheel center (x_r, y_r) of the agricultural machinery is v_r:

$$v_r = \dot{x}_r \cos\varphi + \dot{y}_r \sin\varphi. \tag{8}$$

I also know the kinematic constraints of the center of the front and rear wheels of agricultural machinery:

$$\begin{cases} \dot{x}_r \cos\varphi - \dot{y}_r \sin\varphi = 0, \\ \dot{x}_f \sin\left(\varphi + \delta_f\right) - \dot{y}_f \cos\left(\varphi + \delta_f\right) = 0. \end{cases} \tag{9}$$

Combining (8) and (9) can get

$$\begin{cases} \dot{x}_r = v_r \cos\varphi, \\ \dot{y}_r = v_r \sin\varphi. \end{cases} \tag{10}$$

According to the relationship between the center coordinates of the rear and front wheels (x_r, y_r) and (x_f, y_f):

$$\begin{cases} x_f = x_r + l\cos\varphi, \\ y_f = y_f + l\sin\varphi. \end{cases} \tag{11}$$

Incorporating (10) into (11) can reach the angular velocity ω when the agricultural machinery turns:

$$\omega = \frac{v_r}{l}\tan\delta_f. \tag{12}$$

ω is the angular velocity at which the agricultural machinery rotates around the instantaneous rotation center P. And the moving speed of the agricultural machinery v_r can get the turning radius R and the front wheel deflection angle δ_f:

$$\begin{cases} R = \dfrac{v_r}{\omega}, \\ \delta_f = \tan^{-1}\left(\dfrac{l}{R}\right). \end{cases} \tag{13}$$

Finally, the kinematics model of mobile agricultural machinery can be obtained as

$$\begin{bmatrix} \dot{x}_r \\ \dot{y}_r \\ \dot{\varphi} \end{bmatrix} = \begin{bmatrix} \cos\varphi \\ \sin\varphi \\ \tan\dfrac{\delta_f}{l} \end{bmatrix} v_r. \tag{14}$$

4. Design of the Control Strategy for the Agricultural Machinery Path following Deep Reinforcement Learning

4.1. Design of the Autonomous Path Tracking Framework for Agricultural Machinery. When agricultural machinery uses reinforcement learning to achieve autonomous path tracking and obstacle avoidance tasks in unknown environments, it must first meet the MDP model. When using MDP, it is necessary to define the state space, action space, and reward and punishment functions. When the agricultural machinery first interacts with the environment, it cannot distinguish between obstacles and targets. It can only

adjust its own strategy according to the reward and penalty values feedback from the environment in the process of exploring the environment and finally realize the task of path tracking. The framework of the path tracking algorithm is shown in Figure 8. The Autolabor four-wheeled vehicle is used to simulate the operation of agricultural machinery in the design of the path tracking framework.

In the above framework, the agricultural machinery obtains external information through the Lidar sensor and executes action a, tries different states S_t, and at the same time obtains the corresponding reward value r according to the set reward and punishment function. When exploring the environment, OU noise is added to increase the exploration degree of the action space, and the experience explored in the environment is stored in the form of tuples and placed in the experience playback pool. When training the network, the priority playback mechanism is used to sample and learn the important experience samples first, reducing the training time of the mobile agricultural machine, and finally the mobile agricultural machine learns to track autonomously in the environment. The following will design the state space, action space, and reward and punishment functions in the algorithm framework.

4.1.1. Agent State and Space Design. In order to simplify the path tracking model, it is assumed that the agricultural machine is moving at a fixed speed; that is, the agricultural machine has a fixed moving distance in each time step, so the steering angle φ of the machine is taken as the action space, and the dimension is 1.

In deep reinforcement learning training, the purpose of agricultural machinery is to move to the target path while avoiding obstacles. Therefore, the state space of agricultural machinery needs to include its own positional relationship with obstacles and target paths. This article defines the state space of agricultural machinery as follows:

$$S = \begin{cases} \dfrac{(x, y)}{k}, \\ \dfrac{\theta}{2\pi}, \\ \dfrac{d_{\text{obj}}}{k}, \\ \dfrac{\left(\left(x - x_{\text{obj}}\right), \left(y - y_{\text{obj}}\right)\right)}{k}, \\ \dfrac{d_{\text{aim}}}{k}, \\ \dfrac{\left(\left(x - x_{\text{aim}}\right), \left(y - y_{\text{aim}}\right)\right)}{k}. \end{cases} \tag{15}$$

Among them, (x, y) and θ represent the position and orientation of the agricultural machine in the current map, and k is the standardized coefficient; d_{obj} and d_{aim} represent the distance between the agricultural machine and the nearest obstacle and the target path; $(x - x_{\text{obj}})$, $(y - y_{\text{obj}})$ and $(x - x_{\text{aim}})$, $(y - y_{\text{aim}})$, respectively, represent the distance information of the agricultural machinery from the nearest obstacle and the target path.

In actual movement, the real-time pose of the agricultural machine in the environment can be obtained through SLAM technology, and the distance between the agricultural machine robot and the obstacle is obtained through sensors.

4.1.2. Reward Function Design. The reward and punishment value is the feedback signal given to the agent by the environment, which reflects the pros and cons of the actions performed by the agent during the task learning process. When the agricultural machinery obtains a higher reward value from the environment, it indicates that the current behavior of the agricultural machinery is more conducive to the path tracking task; on the contrary, if the mobile agricultural machinery receives a large penalty value in the environment, it means that the behavior performed by the mobile agricultural machinery is not good for the path tracking task and should be avoided as much as possible. Finally, the mobile farming opportunity adjusts its strategy according to the rewards and punishments in the environment. During the training of mobile agricultural machinery, when the agricultural robot reaches the target point or touches obstacles and walls, the agricultural robot is given a fixed reward. When the agricultural robot has not reached the target or touched an obstacle, the reward value contains two parts: one is the negative reward value of the distance information between the agricultural machine and the nearest obstacle; the second is the positive reward value of the distance information between the agricultural robot and the target path. The sum of the two parts of the reward value is used as the final reward value obtained by the agricultural robot after each action, set as follows:

$$\text{reward} = \text{reward}_{\text{att}} + \text{reward}_{\text{rep}} = \frac{1}{2^{d_{\text{aim}}/k}} - \frac{1}{2^{d_{\text{obj}}/k}}. \tag{16}$$

Therefore, the reward function of agricultural machinery action is

$$\begin{cases} 200 & d_{\text{aim}} < d_{\text{obj}}, \\ \dfrac{1}{2^{d_{aim}/k}} - \dfrac{1}{2^{d_{obj}/k}} & d_{\text{aim}} < d < d_{\text{obj}}, \\ -200 & d_{\text{aim}} > d_{\text{obj}}. \end{cases} \tag{17}$$

The rewards in the above reward and punishment function are divided into continuous rewards and instant rewards. Continuous rewards are rewards that are generated every time the agricultural robot takes an action; that is, rewards are rewards that are given immediately under certain circumstances.

4.1.3. Design of Autonomous Path Tracking Control for Agricultural Machinery. The path tracking process design of mobile agricultural machinery under the deep

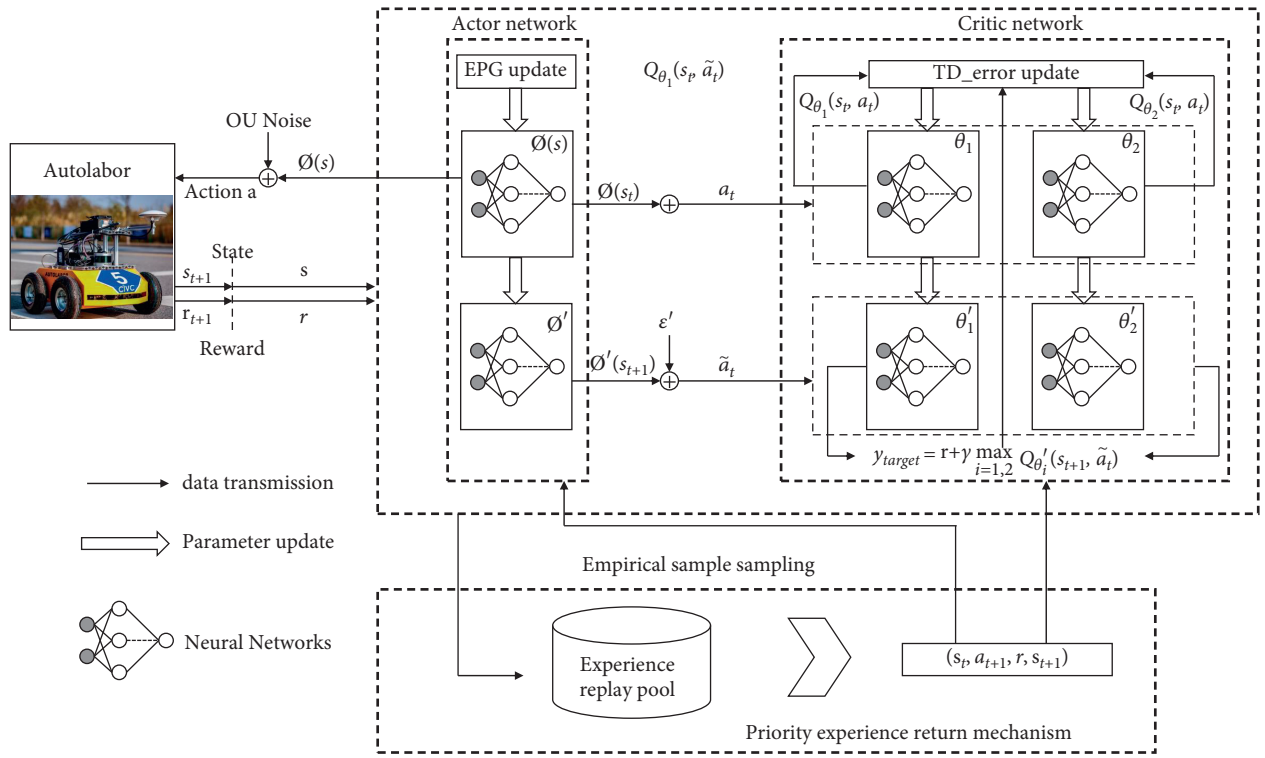


Figure 8: Agricultural machinery path tracking framework based on Autolabor.

reinforcement learning algorithm is shown in Figure 9. The agricultural machinery first obtains environmental information through sensors and calculates the orientation and distance of obstacles and targets and selects the corresponding action value according to the exploration noise and exploration attenuation rate. At the same time, it is judged whether it is the end state or the target state. If it is the end state, reset the environment and restart; otherwise continue to learn in the environment; if it is the target state, continue to judge whether the algorithm has converged; if it converged, the program ends; otherwise continue to generate target endpoints and interact with the environment until the end.

4.2. Deep Neural Network Structure Design. The deep neural network of agricultural machinery is based on the Actor-Critic framework. In the current state, the mobile agricultural machinery obtains and executes actions through the Actor network and interacts with the environment to reach the next state and obtain reward values. At this time, the Critic network takes the actions and state values output in the Actor network as input and outputs the evaluation of the current action value. This evaluation indicates the pros and cons of the action value of the mobile agricultural machine in the current state. The structure design of the network is shown in Figure 10.

In the Actor network, the input is the state S of the agricultural machinery robot. The number of neurons in the hidden layer is 400 and 300, the activation function is Relu, and the output layer is the linear velocity v and angular velocity v of the mobile agricultural machinery. Since the retreat of agricultural machinery is not considered, the linear velocity w has only positive values, and the angular velocity$w$is a vector, and the positive and negative values indicate the direction, so the Sigmoid and Tanh activation functions are used to output the action values in the continuous action space. In the Critic network, the hidden layer uses the same number of neurons and activation function as the Actor network. The Q value of the output layer does not require an activation function to perform a nonlinear transformation and directly performs a linear transformation. Finally, the smallest Q value is selected from two Critic networks of the same structure to avoid overestimation of the deviation. According to the set reward and punishment mechanism, network parameters will be continuously optimized, so that the Actor network can get a higher reward value after performing actions. In the Critic network, the value calculated in the Actor network is scored, and the score result is sent back to the Actor network. The Actor network will update according to the score result. The combination of the two networks can improve the efficiency of algorithm update.

5. Experimental Design and Results

5.1. Simulation Environment Settings. This chapter will adopt the mobile agricultural machinery model. Autolabor is a ROS-based mobile four-wheeled vehicle instead of agricultural machinery. It has programmable, SLAM mapping navigation, and motion control functions. At the same time, Autolabor software is also provided in open source form. In RVIZ, the models of agricultural machinery robots are

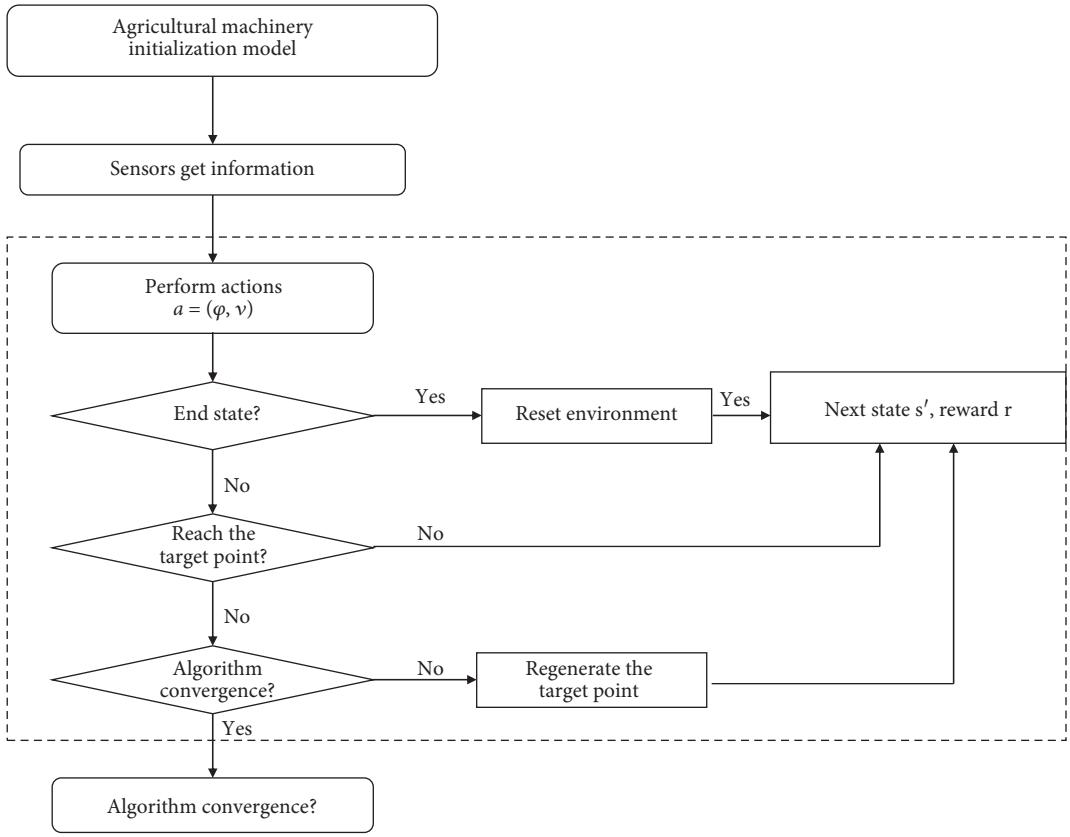


Figure 9: Design of autonomous path tracking control for agricultural machinery.

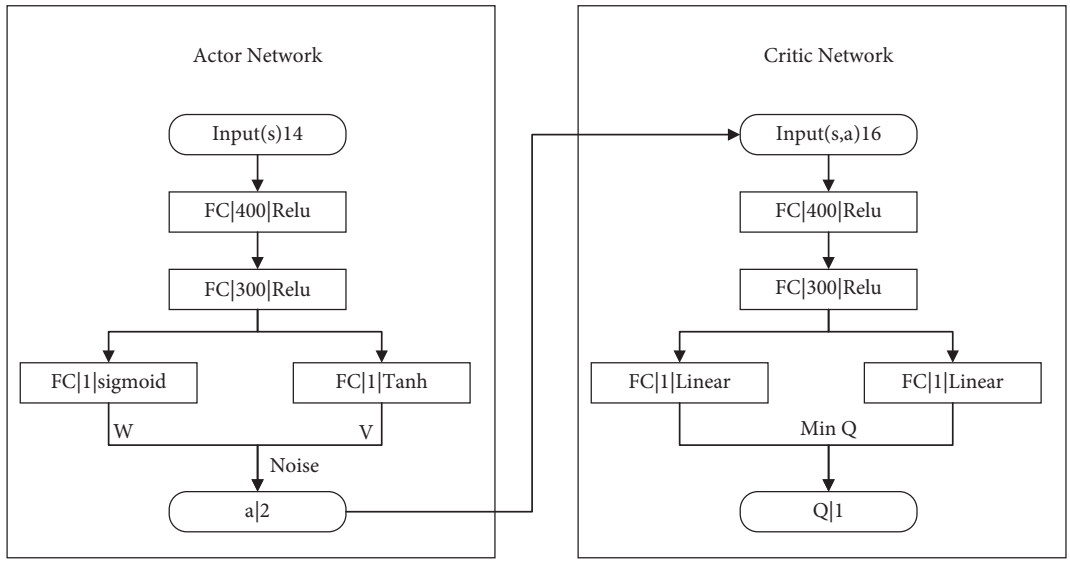


Figure 10: Actor-Critic network structure design.

commonly described in URDF and XARCO files, and their essence is in XML format. Autolabor's model files are shown in Figure 11.

After the model is built, start the model for testing. Create the file display.launch in the launch folder Figure 12. The first input parameter model is the path to the urdf file to be launched. The two input parameters gui specify whether to enable the joint rotation control panel window. Two parameters indicate describing the model description file to be started (urdf) and the joint to the control window (gui, corresponding to each joint), respectively. Three nodes are used to send joint information, robot control information, and rviz start.

Among them, Link and Joint can be compared to human skeletons and joints, which are the basis for describing the

```
<robot name="autolabor_description">
  <link name="base_link">
  <inertial>
    <origin
      xyz="0. 0. 0."
      rpy="0. 0. 0." />
    <mass
      value="0.251988675650349" />
    <inertia
      ixx="0.000595579869264794"
      ixy="5.99238175321912E-08"
      ixz="-1.98242615307314E-08"
      iyy="0.00102462329604677"
      iyz="-1.73115625503396E-05"
      izz="0.00060561972360446" />
  </inertial>
  <visual>
    <origin
      xyz="0. 0. 0.05"
      rpy="1.57 0. 1.57" />
    <geometry>
      <mesh
        filename="package://autolabor_description/meshes/base_link.stl" />
    </geometry>
    <material
      name="">
      <color
        rgba="0.792156862745098 0.819607843137255 0.933333333333333 1" />
    </material>
  </visual>
</link>
</robot>
```

Figure 11: Autolabor model file.

```
<launch>
  <arg name="model" />
  <arg name="gui" default="false" />

  <param name="robot_description" textfile="$(find autolabor_description)/urdf/autolabor_description.urdf" />
  <param name="use_gui" value="$(arg gui)" />

  <node name="joint_state_publisher" pkg="joint_state_publisher" type="joint_state_publisher" />
  <node name="robot_state_publisher" pkg="robot_state_publisher" type="state_publisher" />
  <node name="rviz" pkg="rviz" type="rviz" args="-d $(find autolabor_description)/urdf.rviz" />
</launch>
```

Figure 12: Model startup file.

structure of agricultural machinery and agricultural machinery robots and are constructed in a tree structure. The main body, wheels, and joints of the agricultural machinery and agricultural machinery robot are defined in the link, and some attributes are given: <visual> defines the appearance attributes of the link; <geometry> defines the shape of the structure; <inertial> and <collision> specify, respectively, inertial properties and collision properties. The final Autolabor model in RVIZ is shown in Figure 13.

Next, create a topographic map of agricultural land based on the topographic characteristics of agricultural land, as shown in Figure 14.

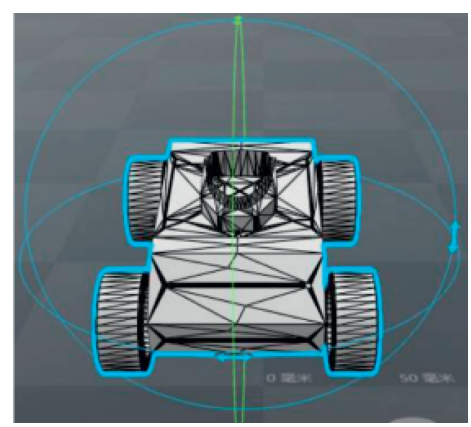


Figure 13: Autolabor model.

5.2. Training Process and Experimental Parameter Settings

5.2.1. Pretest Results and Analysis of Physical Fitness. When the mobile agricultural machinery is undergoing training experiments, it is essentially a process in which the agricultural robot explores the environment and adjusts its action strategy according to the feedback of the environment and finally realizes the path tracking and obstacle avoidance of the agricultural robot. During the training of agricultural robots, the starting point is the starting point, and the target end point is randomly generated in the set simulation environment. The same coordinate range as the obstacle collision area cannot be set as the target end point. When the agricultural machinery robot reaches the target, it means that it has successfully completed a path tracking task and uses this point as the starting point to continue to the next randomly generated target end position. When an agricultural robot fails to track the path, it is regarded as a terminal state. The terminal state includes that the

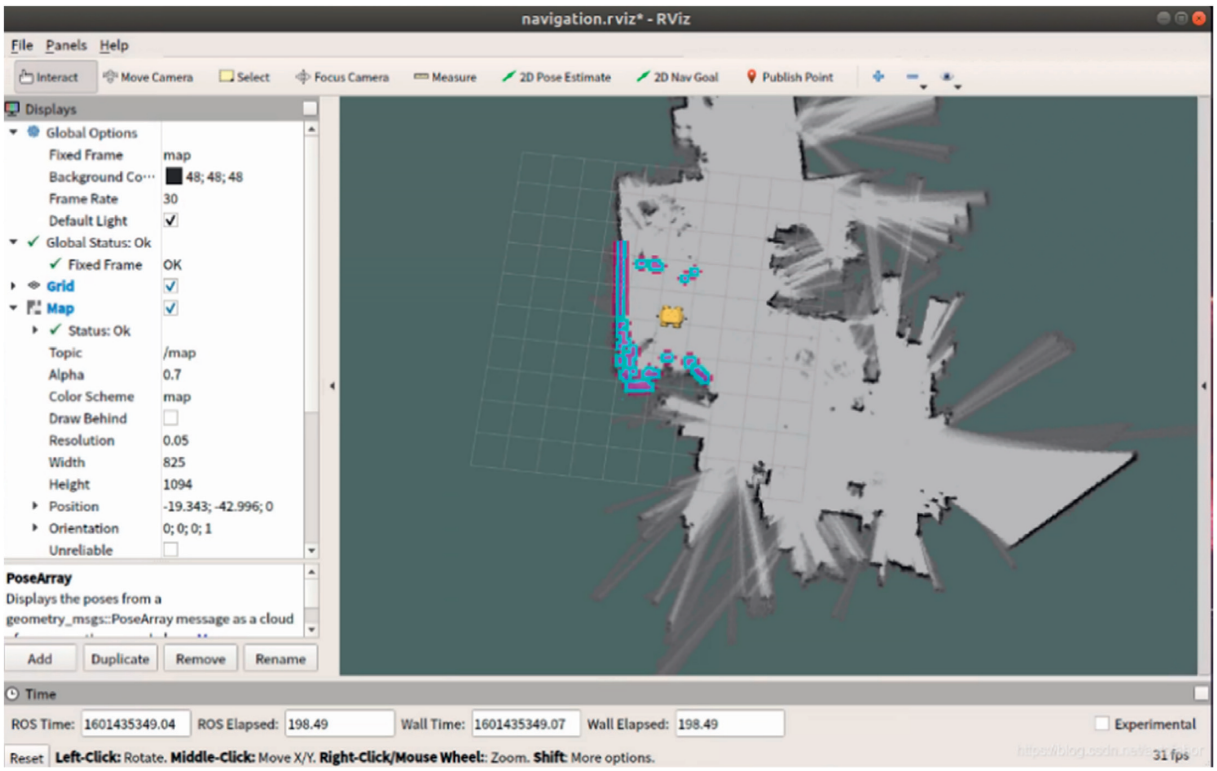


Figure 14: Topographic map of simulated agricultural land.

agricultural robot encounters an obstacle, a wall, or reaches the upper limit of the planned number of steps. At this time, the agricultural robot will start the next training from the planned starting point. Finally, the training is completed after reaching the set maximum number of training rounds. The training process is shown in Figure 15.

5.2.2. Experimental Parameter Settings. In order to improve the reliability of the experimental data, the experiments in this chapter are all completed under the environment ubuntu 6.04+cuda9.0+pytorch0.4.1, and the experimental hardware conditions are i7-8750H + GeForce GTX1060 + 16G. The specific settings of the experimental parameters are shown in Table 1.

5.3. Experiment and Result Analysis. In this section, experiments will be conducted on static obstacle scenes and dynamic obstacle scenes, respectively. In each scene, the path tracking results of the DQN algorithm and the agricultural robot proposed in this paper will be tested, and the results will be analyzed.

5.3.1. Static Obstacle Experiment. The reward value of the first 1000 training rounds is plotted as a reward curve, as shown in Figure 16. In the initial stage of training, since the agricultural robot has just started to interact with the environment, it will often drive away from the target and finally collide with obstacles or walls, so the penalty value is high, and the initial reward is basically around -500 to -400. In the 200 rounds before training, because the DQN algorithm cannot distinguish the importance of experience, it can only continue to explore and try to learn, and the curve fluctuates greatly. The agricultural machinery algorithm uses a priority playback mechanism, which will give priority to learning some important experiences. Compared with the DQN algorithm, it reduces the volatility of the curve, and the I-DQN algorithm starts to accelerate the convergence in about 100 rounds; however, the DQN algorithm does not start to increase the rewards until 250 rounds. As the training time increases, the I-DQN algorithm basically converges after 300 rounds, and the DQN algorithm gradually converges around 450 rounds. Therefore, in scenario 1 under the same training conditions, the I-DQN algorithm has better convergence and stability.

Figure 17 shows the path tracking success rate in the 1000 rounds before the training of the agricultural robot. The trend line of the success rate and the reward value curve are roughly the same. The I-DQN algorithm starts around 100 rounds, and the success rate is greatly improved, reaching 70% in 200 rounds. In about 300 rounds, the success rate of the I-DQN algorithm basically reached 90%; in contrast, the DQN algorithm had fewer successes in the early stage and lacked stability. In 200 rounds of training, there was only a 50% success rate until after 450 rounds. The success rate has gradually reached 90%. Therefore, the importance area of the experience samples in the experience pool can make the agricultural machinery robot better learn path tracking planning tasks and finally learn to use experience to avoid obstacles and reach the end.

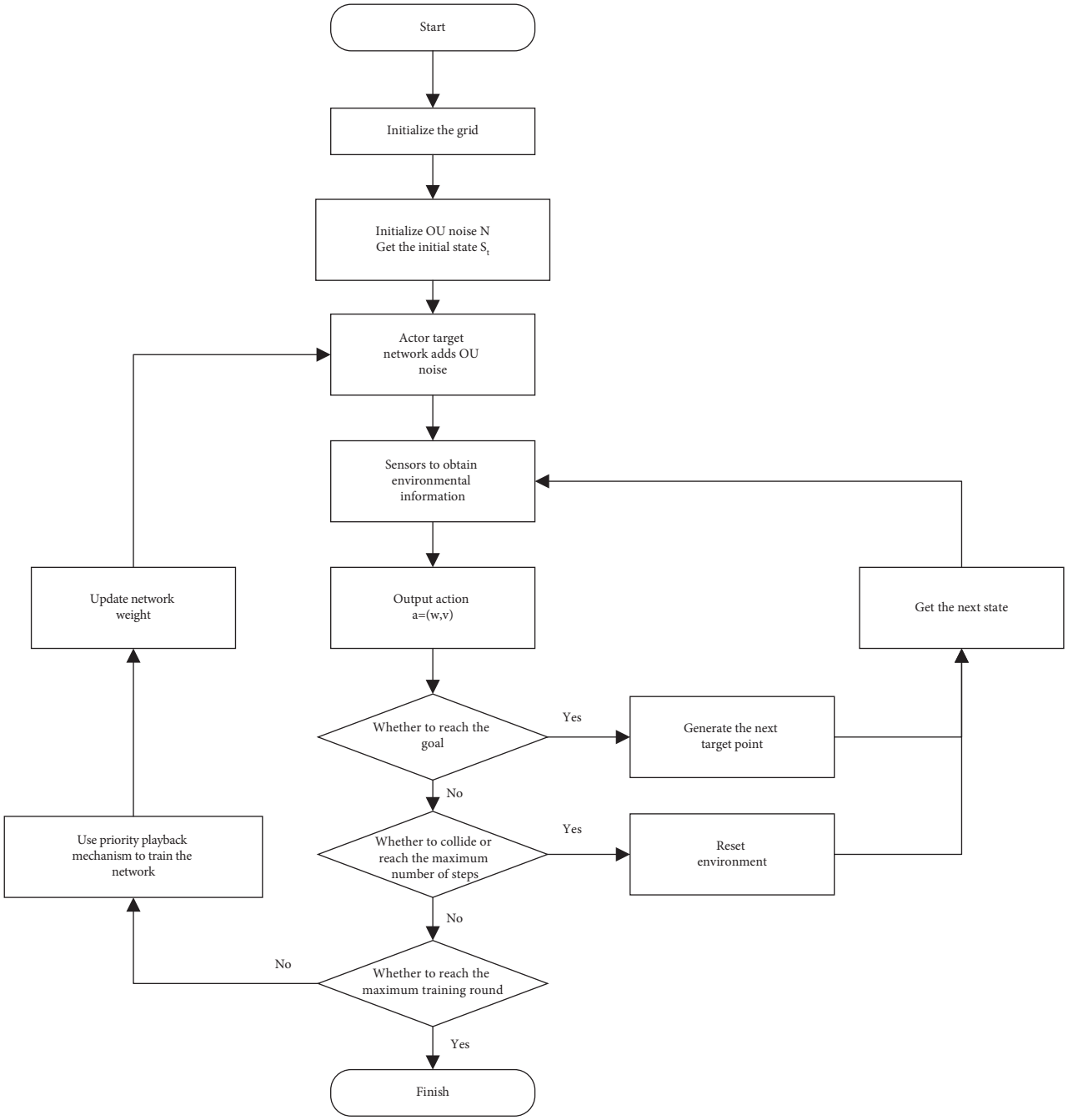


Figure 15: Experimental training process.

5.3.2. Dynamic Obstacle Experiment. In order to test the path tracking ability of the agricultural robot in the DQN and I-DQN algorithms under different types of obstacles, a dynamic obstacle path tracking test was performed in scenario 2. After the agricultural robot enters the termination state, the dynamic obstacle also returns to the original point. The agricultural machinery robot restarts path tracking. Analyze the reward value and success rate during training under the dynamic obstacle scene, and test the path length and planning time.

Figure 18 shows the reward value curve of the two-algorithm training under scenario 2. Similar to scenario 1, the agricultural robot is trying to learn how to avoid obstacles under the two algorithms in the early stage, because the dynamic obstacle avoidance process is more complicated, and the agricultural robot is more likely to collide with obstacles at first, and it takes longer to learn. After 150 rounds, the volatility of the I-DQN algorithm began to decrease, and the reward value increased rapidly in the subsequent 200 rounds, and finally the algorithm gradually converged around 400 rounds. The DQN algorithm fluctuated greatly in the first 200 rounds. The 250 rounds began to rise gradually and did not begin to converge until 550 rounds.

Table 1: Experimental parameter setting table of the mobile agricultural machinery robot.

Parameter name	Parameter assignment
Reward discount rate γ	0.9
Actor network learning rate	0.001
Critic network learning rate	0.001
Priority parameter α	0.6
Correction error parameter β	0.4
Target network delay update TAU	0.001
OU explores noise	$\sigma = 0.2, c = 0.15$
Experience playback pool capacity	200000
BATCH_SIZE	256
Optimizer	Adam
Maximum travel distance per round	3m
Total rounds	15000
Experience pool capacity	50000
Batch capacity	32

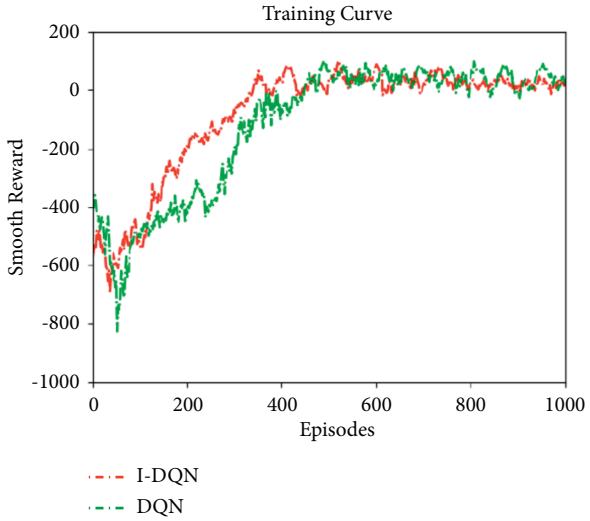


Figure 16: Reward curve of the static obstacle experiment.

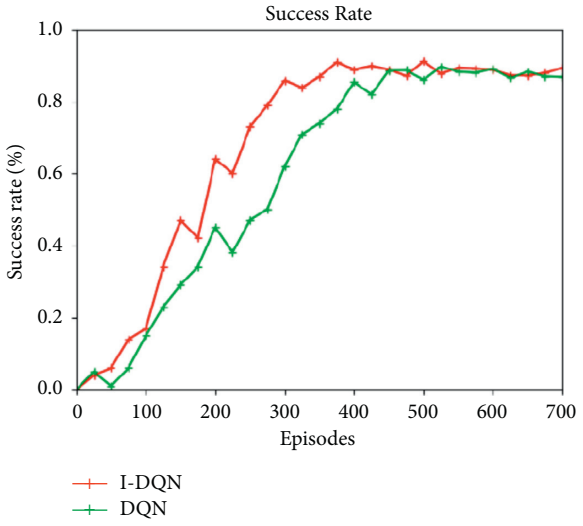


Figure 17: The success rate of static obstacle experiments.

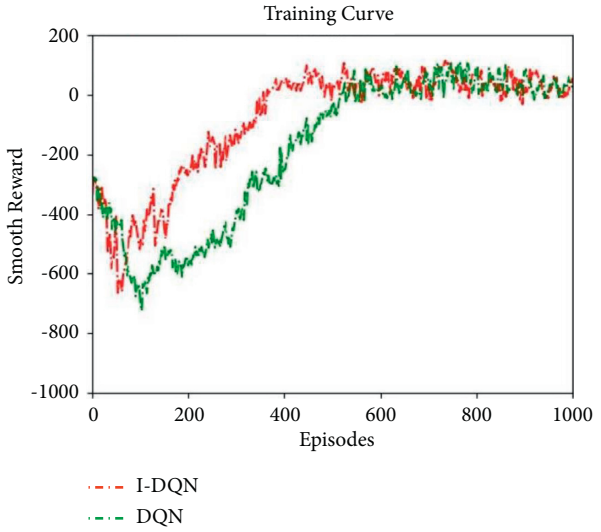


Figure 18: Dynamic obstacle experiment reward curve.

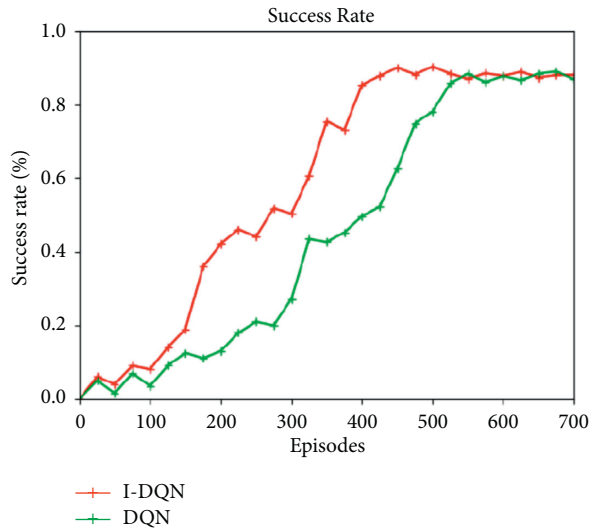


Figure 19: Success rate of the dynamic obstacle experiment.

The path tracking success rate results under scenario 2 are shown in Figure 19. It can be seen that the two algorithms have low success rates in the first 150 rounds, but the success rate of the I-DQN algorithm is greater than the DQN algorithm in the subsequent 200 rounds. The success rate of 350 rounds reaches 75%, which is about 30% higher than the success rate of the DQN algorithm. In 400 rounds, the success rate of I-DQN algorithm basically reached 90%, while DQN had the same success rate in 550 rounds, which proves that I-DQN is better than DQN in path tracking under dynamic obstacle scenarios.

After the training, the dynamic obstacle avoidance process of the mobile agricultural machine in the Gazebo environment is shown in Figure 20. The agricultural robot has been able to continuously reach different target paths while avoiding dynamic obstacles.

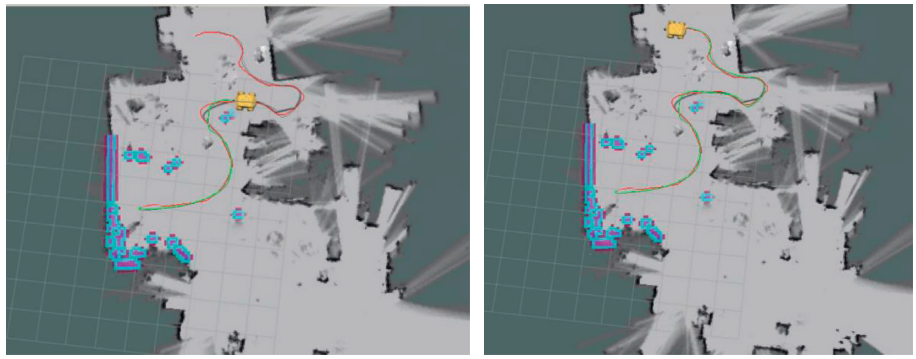


Figure 20: Continuous dynamic obstacle avoidance process of mobile agricultural machinery.

Table 2: I-DQN algorithm path tracking results.

	Target end	Path length (m)	Moving time (s)
1	(−1.13, 0.67)	1.41	7.73
2	(0.84, 1.04)	1.44	7.96
3	(−0.27, 0.34)	1.46	8.27
4	(−1.17, −0.96)	1.62	8.58
5	(0.86, 1.36)	1.69	9.10
6	(0.05, 1.66)	1.73	9.57
7	(−0.91, 1.41)	1.75	9.97
8	(−1.18, 1.33)	1.86	10.11
9	(−1.29, 1.50)	2.11	11.22
10	(1.44, 1.98)	—	—

Table 3: DQN algorithm path tracking results.

	Target end	Path length (m)	Moving time (s)
1	(−1.13, 0.67)	1.49	8.19
2	(0.84, 1.04)	1.49	8.21
3	(−0.27, 0.34)	1.61	8.78
4	(−1.17, −0.96)	1.73	9.16
5	(0.86, 1.36)	1.85	9.43
6	(0.05, 1.66)	1.89	9.90
7	(−0.91, 1.41)	1.95	10.60
8	(−1.18, 1.33)	1.86	10.83
9	(−1.29, 1.50)	2.17	11.56
10	(1.44, 1.98)	—	—

Because obstacle avoidance is more complicated in dynamic obstacle scenarios, the requirements for obstacle avoidance and path tracking of mobile agricultural machinery are higher. Therefore, in order to better test the algorithm, the path length and movement time of the path tracking in scene 2 are tested. Before the test starts, ten coordinate points are randomly generated as the target end point. In order to increase the reliability of the experiment, the coordinate range is set outside the obstacle bypass area, so that the agricultural robot must pass through the obstacle area and will not appear when tracking the path. The target path is very close to the agricultural robot. After the end point is set, perform ten experiments on the I-DQN and DQN algorithms in scenario 2, starting from the origin each time, and use ten randomly generated end points as the target path to perform path tracking, respectively. According to the target path, the straight-line distance length of the starting point is sorted, and the final moving results are shown in Table 2.

Comparing Tables 2 and 3, under the same target end point, the average path length and planning time of the I-DQN algorithm are shorter than those of the DQN algorithm, and the gap gradually increases as the target path distance increases, which proves that the agricultural machinery algorithm tends to make the agricultural machinery robot learn to take a shorter path in a dynamic obstacle scene, and the time is shorter, which improves the performance of path tracking. The average results of the ten times of I-DQN and DQN path tracking are shown in Table 4.

Table 4: Comparison table of path tracking movement results.

	Path length (m)	Moving time (s)	Success rate
I-DQN algorithm	1.67	9.17	90
DQN algorithm	1.76	9.63	90

Based on the analysis of the above experimental results, the DQN algorithm realizes the autonomous path tracking of mobile agricultural machinery in an unknown environment. At the same time, whether in static or dynamic obstacle scenarios, the I-DQN algorithm has a faster convergence speed, allowing agricultural robots to learn to avoid obstacles and reach the target destination faster, and the stability and path tracking performance are improved.

6. Conclusion

With the rapid development of information technology and the realization of smart agriculture, digital agriculture has become an inevitable trend in agricultural development now and in the future. Based on this background, this paper studies the automatic navigation control of agricultural machinery, adopts deep reinforcement learning theory, designs an autonomous path tracking control strategy for agricultural machinery, and conducts experimental simulations through two operating scenarios. The DQN and I-DQN algorithms are applied. In the path tracking task, a number of experiments were designed to verify and analyze

the results. The analysis of experimental results shows that the DQN algorithm realizes the autonomous path tracking of mobile agricultural machinery in unknown environments. At the same time, the I-DQN algorithm has a fast convergence speed. Whether in static or dynamic obstacle scenarios, it can make the agricultural machinery robot learn to avoid obstacles and reach the destination faster, so as to improve the stability and path tracking performance. This research simplifies the motion model and, to a certain extent, does not achieve the true restoration of the actual scene. It has certain limitations for practical applications, but the ideas provided have laid a theoretical foundation for subsequent practical application research.

Data Availability

The dataset can be accessed from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

Natural Science Foundation of Hunan Province (No. 2020JJ7007).

References

[1] D. Chen, Z. Shi, P. Yuan et al., "Trajectory tracking control method and experiment of AGV," in *Proceedings of the AdvancedMotion Control (AMC), 2016 IEEE 14th International Workshop on*, pp. 24–29, IEEE, Auckland, April 2016.

[2] J. E. Normey-Rico, I. Alcalá, J. Gómez-Ortega, and E. F. Camacho, "Mobile robot path tracking using a robust PID controller," *Control Engineering Practice*, vol. 9, no. 11, pp. 1209–1214, 2001.

[3] X. Li, C. Luo, Y. Xu, and P. Li, "A Fuzzy PID controller applied in AGV control system," in *Proceedings of the Advanced Robotics andMechatronics (ICA RM), International Conference on*, pp. 555–560, IEEE, Macau, China, August 2016.

[4] C.-F. Juang and Y.-C. Chang, "Evolutionary-group-based particle-swarm-optimized fuzzy controller with application to mobile-robot navigation in unknown environments," *IEEE Transactions on Fuzzy Systems*, vol. 19, no. 2, pp. 379–392, 2011.

[5] Y.-H. Lee, G.-G. Jin, and M.-O. So, "Level control of single water tank systems using Fuzzy-PID technique," *Journal of the Korean Society of Marine Engineering*, vol. 38, no. 5, pp. 550–556, 2014.

[6] Y. Kanayama, Y. Kimura, F. Miyazaki, and T. Noguchi, "A stable tracking control method for an autonomous mobile robot," in *Proceedings of the Robotics and Automation IEEE International Conference on*, pp. 384–389, IEEE, Cincinnati, OH, USA, May 1990.

[7] N. Hung, J. S. Im, S. Jeong, H. Kim, and B. Sang, "Design of a sliding mode controller for an automatic guided vehicle and its implementation," *International Journal of Control, Automation and Systems*, vol. 8, no. 1, pp. 81–90, 2010.

[8] W. Wu, H. Chen, and Y. Wang, "Adaptive exponential stabilization of mobile robots with uncertainties," in *Proceedings of the Decision and Control 38th IEEE Conference on*, vol. 4, pp. 3484–3489, IEEE, 1999.

[9] T.-L. Bui, P.-T. Doan, D.-T. Van, H.-K. Kim, and S.-B. Kim, "Hybrid control of a tricycle wheeled AGV for path following using advanced fuzzy-PID," *Journal of the Korean Society of Marine Engineering*, vol. 38, no. 10, pp. 1287–1296, 2014.

[10] G. Campion, G. Bastin, and B. Dandrea-Novel, "Structural properties and classification of kinematic and dynamic models of wheeled mobile robots," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 1, pp. 47–62, 1996.

[11] P. Wawzynski, "Control policy with autocorrelated noise in reinforcement learning for robotics," *InternationalJournal of Machine Learning and Computing*, vol. 5, no. 2, pp. 91–95, 2015.

[12] L. Samson, B. O. K. Intelligentie, and E. Gavves, *Deep Reinforcement Learning Applied to the Game Bubbleshooter*, University of Amsterdam, Amsterdam, Netherlands, 2016.

[13] D. Silver, G. Lever, and N. Heess, "Deterministic policy gradient algorithms," in *Proceedings of the International Conference on Machine Learning*, pp. 387–395, 2014.

[14] T. P. Lillicrap, J. J. Hunt, A. Pritzel et al., "Continuous control with deep reinforcement learning," in *Proceedings of the International Conference on Learning Representations*, pp. 1–14, 2016.

[15] A. Barreto, W. Dabney, R. Munos et al., "Successor features for transfer in reinforcement learning," in *Proceedings of the Advances in neural information processing systems*, pp. 4055–4065, 2017.

[16] F. Shoeleh and M. Asadpour, "Graph based skill acquisition and transfer Learning for continuous reinforcement learning domains," *Pattern Recognition Letters*, vol. 87, pp. 104–116, 2017.

[17] M. Fortunato, M. G. Azar, B. Piot et al., "Noisy networks for exploration," 2017, https://arxiv.org/abs/1706.10295.

[18] T. P. Lillicrap, J. J. Hunt, A. Pritzel et al., "Continuous control with deep reinforcement learning," *Computer Science*, vol. 8, no. 6, 2015.

[19] M. Volodymyr, P. B. Adrià, M. Mirza et al., "Asynchronous methods for deep reinforcement learning," in *Proceedings of the International Conference on Machine Learning*, 2016.

[20] W. Meng, Q. Zheng, Y. Shi, and G. Pan, "An off-policy trust region policy optimization method with monotonic improvement guarantee for deep reinforcement learning," *in IEEE Transactions on Neural Networks and Learning Systems*.

[21] A. Kendall, J. Hawke, D. Janz et al., "Learning to drive in a day," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 8248–8254, IEEE, Montreal, QC, Canada, May 2019.

[22] R. S. Sutton and A. G. Barto, *Introduction to Reinforcement Learning*, MIT press, Cambridge, England, 1998.

[23] Q. Cai and B. Zhang, "A reinforcement learning model and application research based on agent team," *Computer Research and Development*, vol. 37, no. 9, pp. 1087–1093, 2000.

[24] V. Mnih, K. Kavukcuoglu, D. Silver et al., "Playing atari with deep reinforcement learning," 2013, https://arxiv.org/abs/1312.5602.

[25] J. Li, Y. Chen, X. N. Zhao, and J. Huang, "An improved DQN path planning algorithm," *The Journal of Supercomputing*, vol. 78, pp. 1–24, 2021.

[26] Y. Wang, L. Chen, H. Zhou et al., "Flexible transmission network expansion planning based on DQN algorithm," *Energies*, pp. 488–491, 2021.

[27] Y. Liu and Y. Xu, "Free gait planning of hexapod robot based on improved DQN algorithm," in *Proceedings of the IEEE 2nd International Conference on Civil Aviation Safety and Information Technology (ICCASIT)*, October 2020.