agronomy

MDPI

Article

Research on Path Planning of Agricultural UAV Based on Improved Deep Reinforcement Learning

Haitao Fu [1], Zheng Li [1], Weijian Zhang [1], Yuxuan Feng [1], Li Zhu [1], Xu Fang [2] and Jian Li [1,*]

[1] College of Information Technology, Jilin Agricultural University, Changchun 130118, China; fht@jlau.edu.cn (H.F.); 20231308@mails.jlau.edu.cn (Z.L.); 20231257@mails.jlau.edu.cn (W.Z.); fengyuxuan@jlau.edu.cn (Y.F.); zhuli@jlau.edu.cn (L.Z.)

[2] School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798, Singapore; fa0001xu@e.ntu.edu.sg

* Correspondence: lijian@jlau.edu.cn; Tel.: +86-139-4419-5488

Abstract: Traditional manual or semi-mechanized pesticide spraying methods often suffer from issues such as redundant coverage and cumbersome operational steps, which fail to meet current pest and disease control requirements. Therefore, there is an urgent need to develop an efficient pest control technology system. This paper builds upon the Deep Q-Network algorithm by integrating the Bi-directional Long Short-Term Memory structure to propose the BL-DQN algorithm. Based on this, a path planning framework for pest and disease control using agricultural drones is designed. This framework comprises four modules: remote sensing image acquisition via the Google Earth platform, task area segmentation using a deep learning U-Net model, rasterized environmental map creation, and coverage path planning. The goal is to enhance the efficiency and safety of pesticide application by drones in complex agricultural environments. Through simulation experiments, the BL-DQN algorithm achieved a 41.68% improvement in coverage compared with the traditional DQN algorithm. The repeat coverage rate for BL-DQN was 5.56%, which is lower than the 9.78% achieved by the DQN algorithm and the 31.29% of the Depth-First Search (DFS) algorithm. Additionally, the number of steps required by BL-DQN was only 80.1% of that of the DFS algorithm. In terms of target point guidance, the BL-DQN algorithm also outperformed both DQN and DFS, demonstrating superior performance.

Keywords: precision agriculture; deep Q-learning; Bi-directional Long Short-Term Memory; pest control; remote sensing

Citation: Fu, H.; Li, Z.; Zhang, W.; Feng, Y.; Zhu, L.; Fang, X.; Li, J. Research on Path Planning of Agricultural UAV Based on Improved Deep Reinforcement Learning. *Agronomy* **2024**, *14*, 2669. https://doi.org/10.3390/agronomy14112669

Academic Editor: Gniewko Niedbała

Received: 14 October 2024
Revised: 31 October 2024
Accepted: 10 November 2024
Published: 13 November 2024

1. Introduction

Monitoring data indicate that in 2024, major crops in China, such as grains, oilseeds, and vegetables, will confront severe pest and disease threat, with affected areas expected to reach 15.541 million hectares, representing a significant increase compared with previous years. The potential food loss is projected to exceed 150 million tons [1]. Currently, traditional manual or semi-mechanical pesticide spraying methods suffer from incomplete coverage and low efficiency, failing to meet the current pest and disease control requirements [2,3]. Thus, accelerating the development of an efficient modern pest control technology system is urgent. The application of drones in pest and disease control is of great significance. By optimizing the flight paths of individual or swarm drones, more precise spraying can be achieved, thereby reducing redundant coverage, improving control efficiency, conserving resources, and lowering costs [4–7].

Recently, drone technology has gained increasing global applications due to its low cost, ease of use, and operational capabilities in high-risk or hard-to-reach areas. This trend has brought significant benefits and opportunities across various fields, including agriculture, healthcare, and military applications. Agricultural drones, as part of precision agriculture technology, differ from drones used in other sectors by typically requiring

sensors such as hyperspectral and thermal imaging systems. These sensors are employed to monitor plant health, soil moisture, and other agricultural concerns. Additionally, agricultural drones may be equipped with liquid spraying systems for precise pesticide or fertilizer application, a feature that is relatively uncommon in other types of drones [8–11].

The coverage path planning (CPP) problem, as a critical research area in drone path planning, aims to design a path within a given area such that the drone can traverse every point or cover each sub-region of the map with the shortest number of steps [12–14]. Solutions to the CPP problem can be roughly categorized into four types. First, Depth-First Search (DFS) is a graph traversal algorithm that explores as far as possible along each branch before backtracking. It is often used for solving problems that can be modeled as a graph, where the goal is to visit all nodes or find a specific path. DFS uses a stack to manage the nodes to be explored and systematically searches each branch to its end before retracing its steps [15]. Second, heuristic algorithms, such as the A* algorithm, model the search space as a tree structure and use heuristic search techniques to solve the problem [16]. Third, the Artificial Potential Field (APF) method, as a local obstacle avoidance path planning algorithm, completes the planning task by simulating the potential fields between the target and obstacles [17]. Fourth, the Deep Q-Network (DQN) algorithm, based on deep reinforcement learning (DRL) [18,19], approximates the Q-value function through deep neural networks, allowing the agent to learn and optimize path planning strategies [20].

Cai et al. proposed a coverage path planning algorithm based on an improved A* algorithm, which efficiently accomplishes coverage tasks for cleaning robots by incorporating a U-turn search algorithm. However, despite its effectiveness in high-precision maps, the A* algorithm is associated with significant computational complexity and node redundancy issues [21].

Wang et al. proposed a multi-agent coverage path planning method based on the Artificial Potential Field (APF) theory, which guides the movement of agents by simulating the interactions of forces in a physical field. However, the APF method is susceptible to becoming trapped in local optima, faces challenges with complex parameter adjustments, and may encounter potential singularity issues during planning [22].

Tang et al. proposed a coverage path planning method based on region-optimal decomposition, which combines an improved Depth-First Search (DFS) algorithm with a genetic algorithm to achieve efficient coverage inspection tasks for drones in port environments [23]. Although the improved DFS algorithm successfully completes the coverage path planning tasks, it may encounter local optima in certain complex environments and faces challenges in ensuring safety during actual drone flight operations.

Unlike classical algorithms that rely on predetermined rules, the DQN algorithm, based on deep reinforcement learning, introduces two optimization techniques: “target network” and “experience replay”. The target network updates its parameters at regular intervals to maintain relative stability in the target values, thereby reducing fluctuations during the training process. Experience replay allows for the reuse of past experiences, sampling from diverse previous interactions to mitigate the instability issues caused by data correlation. Through this continuous improvement of experience, the drone is capable of making decisions under complex environmental conditions, making it particularly effective in dynamic and challenging environments [24–26].

In recent years, Mirco Theile and colleagues have addressed the CPP problem for drones under fluctuating power limitations by leveraging the DDQN algorithm to balance battery budgets, enabling the achievement of full coverage maps [27]. S.Y. Luis and colleagues approached the problem of patrolling water resources by modeling it as a Markov Decision Process (MDP) and using DQN and DDQN algorithms for training. However, the large number of parameters involved made it challenging to ensure algorithm stability [28].

In the field of agricultural applications, Li and their team introduced an algorithm for multi-region task path planning utilizing a DDQN to tackle the difficulties associated with precise fertilization using agricultural drones [29]. (1) This research partially extends and supplements the application of DQN within the scope of agricultural drone CPP; however, it primarily focuses on the formation control of drones in multi-task areas and does not adequately account for the variations in actual farmland terrain, limiting the generalizability of the algorithm. (2) Furthermore, this study did not consider the drone recovery issue when designing the reward function, resulting in uncertainty regarding the landing position after task completion, which poses significant recovery challenges. (3) Although the improved DDQN algorithm has demonstrated some success in path planning, it still exhibits shortcomings in obstacle avoidance, leading to issues such as the presence of overlapping areas within the task region and difficulties in evading obstacles. To address these concerns, this paper attempts to integrate a Bi-directional Long Short-Term Memory (Bi-LSTM) structure with the DQN algorithm, resulting in an improved BL-DQN algorithm. Through the use of the Google Earth platform, multiple farmland areas were randomly selected to construct planning maps, and the reward function was adjusted to better fit real agricultural application scenarios, thereby enhancing the algorithm's generalizability and optimizing issues related to drone recovery, repeated regions, and obstacle avoidance.

The primary contributions of this paper are outlined as follows:

1. A framework for pest and disease control path planning for agricultural drones has been developed using the BL-DQN algorithm. This framework includes four modules: remote sensing image acquisition via the Google Earth platform, task area segmentation using the deep learning U-Net model, grid-based environmental map creation, and coverage path planning.
2. A new BL-DQN algorithm is proposed, which effectively integrates the BL-LSTM structure with the target network in the DQN algorithm to achieve high-performance information processing and learning.
3. To address the drone task retrieval issue, a target-oriented reward function is designed, taking into account the priority of target areas, path efficiency, and task requirements.

The organization of this paper is structured as follows: Section 2 details the creation of environmental maps, design of the reward function, and improvements to the DQN algorithm; Section 3 describes the experimental design, presents experimental validation and results analysis, and outlines future prospects; and Section 4 concludes with a summary of research findings.

2. Materials and Methods

The path planning framework for pest and disease control in agricultural drones proposed in this paper utilizes the GE platform in conjunction with the deep learning U-Net algorithm to construct the task environment maps. The drone then employs the BL-DQN algorithm to complete the coverage task and locate target arrival points, thus facilitating the path planning task for pest and disease control. The comprehensive structure of the proposed approach is depicted in Figure 1.

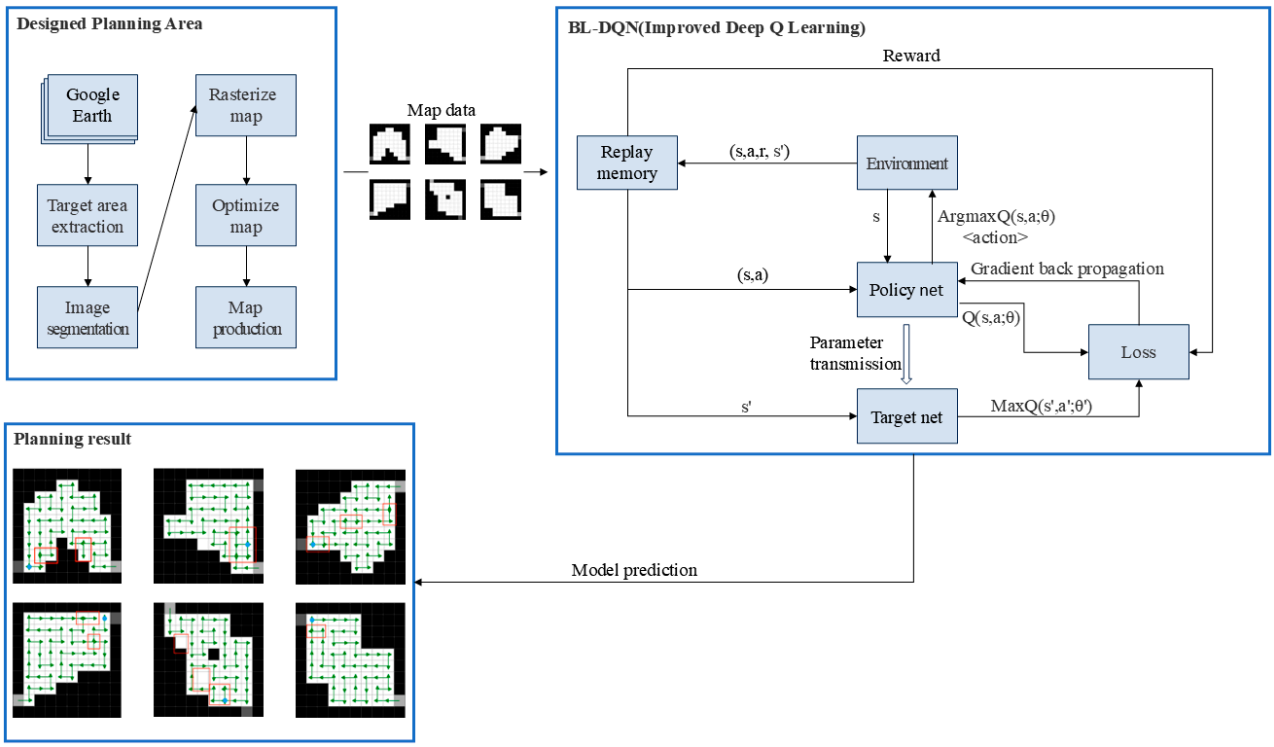

Figure 1. The comprehensive process of the framework.

2.1. Designed Planning Area Description

This research explores notable topographical differences in various agricultural production settings, focusing on Jilin Province to address the diversity found in farmlands. Situated in the heart of the Northeast Plain, Jilin Province is part of one of the globe's three main black soil areas and serves as a vital agricultural region and significant grain production hub in China. The province features diverse terrain with higher elevations in the southeast and lower elevations in the northwest. Its landscape predominantly comprises plains and mountainous regions, covering an area of approximately 187,400 square km, with elevations ranging from 5 m to 2691 m.

High-resolution remote sensing images of farmlands in Jilin Province from April 2021 to October 2022 were acquired using the Google Earth platform. The geographic coordinates of the study area range from 123°01′ to 128°08′ east longitude and 43°14′ to 45°36′ north latitude [30]. Six regions were randomly selected for analysis, as depicted in Figure 2. Furthermore, the U-Net model was utilized for detailed segmentation of the farmland areas, classifying them into two categories: (1) task areas and (2) non-task areas, as illustrated in Figure 3.

Additionally, this study employs a gridded map approach, dividing the environment into a 10 × 10 grid and using a two-dimensional integer array for storage and operations. Based on this, the drone's environment map is defined as a state matrix with 100 elements, where each element represents a grid cell on the map. The side length of the map is denoted by L, and $M_{(x,y)}$ represents the environmental state at position (x, y) on the map. Each position is assigned one of five distinct values to characterize its specific environmental features, as illustrated in Table 1.

In this paper, each grid cell is considered as a unit of the map for each movement of the drone. When an action A_i is executed, the corresponding state matrix changes, transitioning from the current state S_i to the next state S_{i+1}, as illustrated in Figure 4.

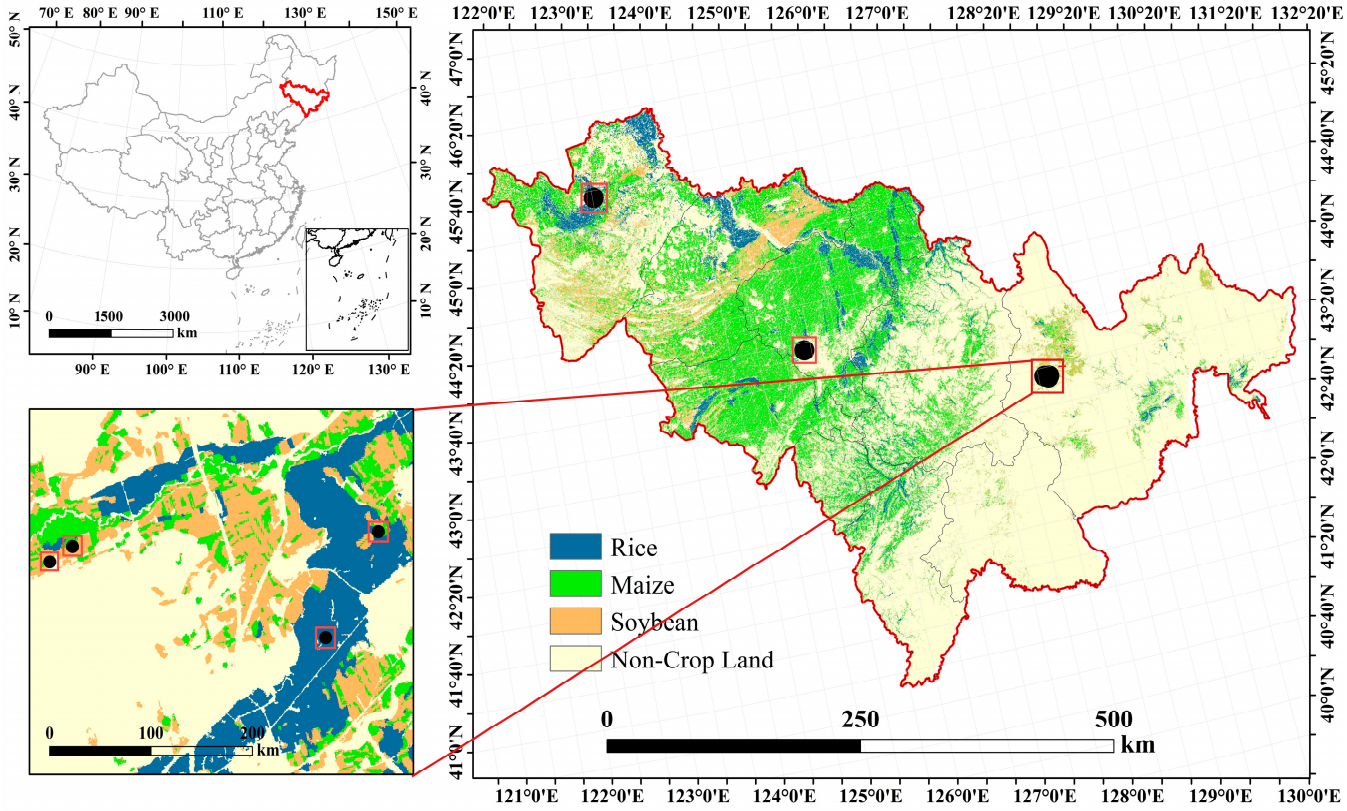

Figure 2. Remote sensing map extraction (task areas are highlighted with red boxes).

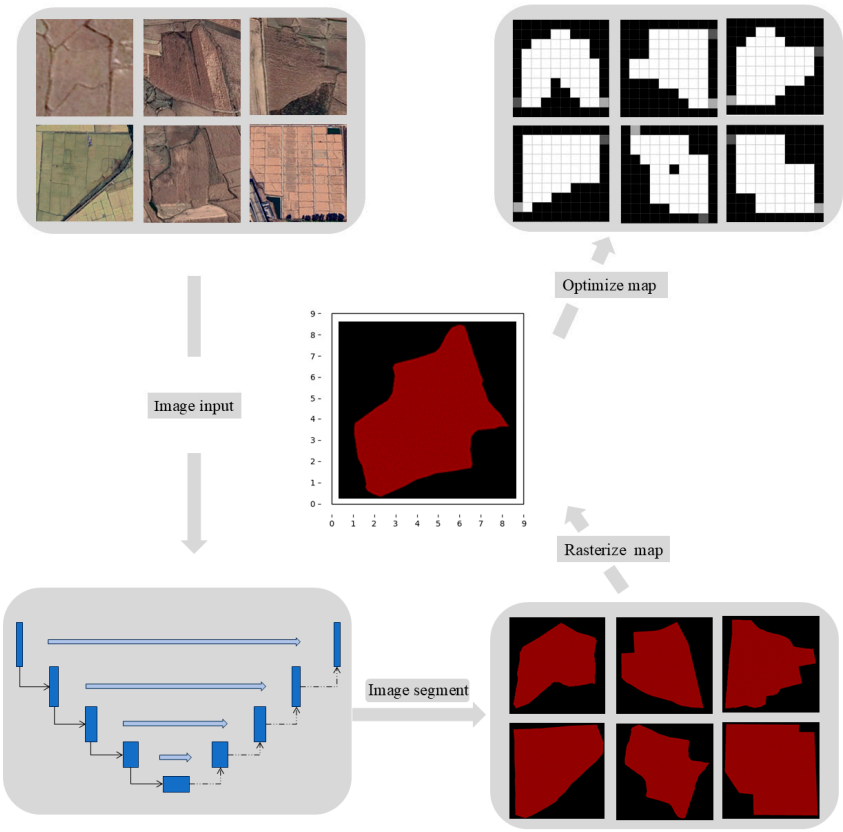

Figure 3. Designed planning area process.

Table 1. Map state corresponding to different values.

Value ($M_{(x,y)}$)	Description
0	Non-task area
1	Task area
2	Current position of UAV
3	Target location
4	Obstacle area

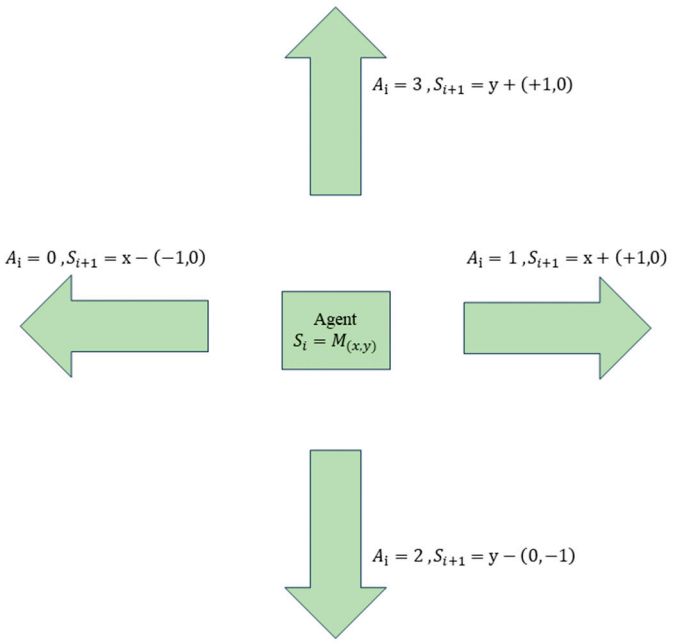

Figure 4. UAV action selection space.

Here, A_i = {1, 2, 3, 4} represents the four allowed movement directions for the drone at its current position: left, right, down, and up, respectively.

2.2. Basic Theory

2.2.1. Deep Q-Learning

In 2016, Volodymyr Mnih proposed the DQN algorithm [31], which integrates the characteristics of DL with the principles of reinforcement learning (RL). This method utilizes DL models to directly acquire control strategies from complex sensory information and assesses the value through neural networks. The DQN algorithm implements an experience replay system, in which experience tuples created while the agent interacts with the environment are saved in a replay buffer, as shown in Equation (1). These experiences are then randomly sampled from the buffer for training.

$$< s_t, a_t, r_t, s_{t+1} > \quad (1)$$

Furthermore, DQN utilizes two networks that share the same structure and parameters: the policy network and the target network. Throughout training, only the parameters of the policy network are consistently updated. At specific intervals, these parameters are transferred to the target network to reduce instability caused by the frequent updates to the target Q-values. The DQN algorithm has shown impressive performance in various classic Atari 2600 games, reaching near-human-level proficiency through learning, and has consequently become a significant subject in artificial intelligence research in recent years.

2.2.2. Reward

Since the model relies solely on feedback rewards obtained through interactions with the environment to guide learning, the design of an effective reward mechanism typically determines the efficiency and accuracy of the model's learning process. Therefore, a well-designed reward function should be simple and reflect the specific requirements of the task. The reward function used in traditional DQN path planning algorithms is represented by Equation (2).

$$r_t = \begin{cases} r_{overlay}, & \text{Task area coverage} \\ r_{crash}, & \text{Collision} \\ 0, & \text{Other cases} \end{cases} \tag{2}$$

Based on the different outcomes at the next timestep, the rewards are divided into three parts. The action $r_{overlay}$ for reaching the target is given a positive value to encourage the model to find the target. Conversely, the action r_{crash} for collisions is assigned a negative value to penalize collision behavior. However, sparse rewards, which only occur when reaching the target or experiencing collisions, result in a lack of valuable feedback during each action. This not only reduces learning efficiency and increases exploration difficulty but also complicates policy optimization [32].

2.3. A Reward Function Based on Goal Orientation

To tackle the issues of high complexity and slow convergence in traditional reward function strategies, this paper focuses on practical tasks for agricultural drones. Specifically, its goal is to devise the best route from the starting point to the target location, minimizing steps, covering the entire task area, and evading obstacles. The reward function has been optimized to improve these aspects, and a new reward function design method is proposed, as illustrated in Equation (3).

$$r_t = \begin{cases} r_{reach}, & \text{Reaching the target point} \\ r_{carsh}, & \text{Collision} \\ r_{overlay}, & \text{Task area coverage} \\ r_{step}, & \text{Maximum remaining score/Maximum number of steps} \\ 0, & \text{Other cases} \end{cases} \tag{3}$$

1. Coverage Reward: This approach allocates rewards according to the percentage of the task area on the map that has been explored.

2. Target Guidance Reward: This method sets a reward to guide the drone towards the target points.

These optimizations enable the drone to cover the map more quickly and reach target points, thereby accelerating the model's convergence speed.

The map coverage reward is given by Equation (4), where M_{new_x,new_y} denotes the agent's existing location. Areas marked with the number 0 represent non-task regions, while those marked with the number 1 indicate task areas. $C_{initial}$ is the initial number of task areas on the map, $C_{current}$ is the count of task areas at the present moment, and φ is the adjustable coverage reward scaling factor. The coverage rate is evaluated by comparing the number of current task areas and the distance to the target point, which allows for dynamic adjustments of both the coverage reward and the target guidance reward. This process effectively guides the drone in selecting a new position.

$$r_{overlay} = \begin{cases} 0.05 + \varphi e^{\frac{C_{initial} - C_{current}}{C_{initial}}} + R_{goal} & \text{if } M_{new_x,new_y} = 1 \\ -0.1 & \text{if } M_{new_x,new_y} = 0 \end{cases} \tag{4}$$

The target guidance reward for the drone is designed as shown in Equation (4). In this Equation, $current\_distance$ refers to the distance between the initial position and the destination, while $new\_distance$ is the distance to the destination from the new position

after the movement. M_x and M_y represent the current position coordinates of the drone on the map, M_{new_x} and M_{new_y} are the coordinates of the agent's position after executing the current action, selected based on a greedy strategy, and G_x and G_y are the coordinates of the target location on the map. β is the adjustable scaling factor for the target point reward. Equations (5)–(7) illustrate these calculations.

To calculate the distance measured from the current location to the destination, use the Euclidean distance equation:

$$current\_distance = \sqrt{(M_x - G_x)^2 + (M_y - G_y)^2} \tag{5}$$

To calculate the distance between the new position and the destination, use the Euclidean distance equation:

$$new\_distance = \sqrt{(M_{new_x} - G_x)^2 + \left(M_{new_y} - G_y\right)^2} \tag{6}$$

To calculate the reward for reaching the target R_{goal} based on the difference between the two distances, use the following Equation:

$$R_{goal} = \begin{cases} \beta(current\_distance - new\_distance) & \text{if } new_d < current_d \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

To avoid the program getting stuck in local optima and to reduce the training burden, this paper presents three solutions, where "True" indicates that the task has been completed or failed, signaling that the current episode has ended, as illustrated in Equation (8).

1. The current episode is terminated when the drone collides with an obstacle.
2. The current episode is terminated if the drone exceeds the maximum step limit or if the drone has not scored for a number of consecutive steps beyond the specified threshold.
3. The current episode is terminated when the drone reaches the target point and there are no uncovered areas within the task zone.

$$R, Done = \begin{cases} r_{carsh}(-1 \; and \; \text{True}), \text{ if } M_{new_x,new_y} = 4 \\ r_{step}(-0.1 \; and \; \text{True}), \; S \geq S_{max} \; or \; N \geq N_{max} \\ r_{reach}(1 \; and \; \text{True}), \text{ if } M_{new_x,new_y} = 3 \; and \; ones\_ratio = 0 \end{cases} \tag{8}$$

2.4. BL-DQN

Bi-LSTM is an extended LSTM network that simultaneously considers data from both past and future contexts by employing two LSTM layers operating in opposite directions to capture contextual relationships. This structure enables the model to obtain more comprehensive information when processing data, thereby enhancing performance. This paper extends the DQN algorithm by integrating the BL-LSTM structure, which improves the focus on multi-temporal action values through deep neural networks. The network architecture consists of two LSTM layers in different directions to increase model depth and capture more complex sequence dependencies, and a fully connected layer that maps the high-dimensional feature vectors from the Bi-LSTM layers to a low-dimensional space matching the number of output classes for the task, as illustrated in Figure 5c.

The entire model inference process is illustrated in Figure 5b. Initially, the input data were converted into an array and subjected to normalization. Subsequently, the data were transformed into tensors and dimensions were added to meet the input requirements of the subsequent model, as shown in Figure 5a. The preprocessed data were then input into the model after being flattened and reshaped, allowing the model to generate output results that provided guidance for path planning.

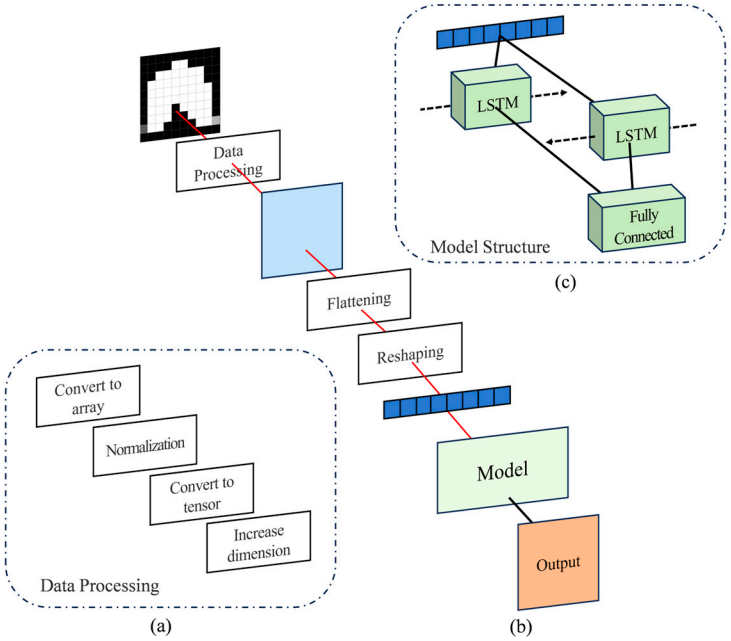

Figure 5. (**a**) Data processing. (**b**) Model inference process. (**c**) Model structure.

At each time step t, the Bi-LSTM's output consisted of the merging of hidden states from both the forward and backward LSTM layers, as shown in Equation (9).

$$h_t^{BiLSTM} = [h_t, h_t'] \tag{9}$$

Here, h_t and $h_t\prime$ denote the hidden states of the forward and backward LSTM layers at time step t. Subsequently, the output of the Bi-LSTM layer will be processed through a fully connected layer to estimate the Q-values.

$$Q(s,a) \approx Q(s,a;\theta) = W \cdot h_t^{BiLSTM} + b \tag{10}$$

The Q-value update equation is given by Equation (11).

$$Q'^{(s,a)} \leftarrow Q(s,a) + \alpha[r + \gamma max_{a'} Q(s',a') - Q(s,a)] \tag{11}$$

Here, $Q(s,a)$ denotes the Q-value for taking action a in state s, where α represents the learning rate, r is the reward, and γ is the discount factor. A larger γ emphasizes long-term rewards more heavily, whereas a smaller γ prioritizes short-term gains.

This paper employs a greedy strategy to balance the concepts of exploitation and exploration when selecting actions at each time step, as shown in Equation (12). In this equation, RN represents a random number generated in the range of 0 to 1 for each time step. ϵ is a hyperparameter used to balance exploitation and exploration, dynamically adjusted throughout the training phase, as shown in Equation (13). When $RN > \epsilon$, the action with the highest Q-value for the current state is chosen for exploitation. Otherwise, a random action is selected for exploration.

$$Action = \begin{cases} argmax(Q(s,a;\theta)) & \text{if } RN > \epsilon \\ random\ action & \text{otherwise} \end{cases} \tag{12}$$

$$\epsilon = \epsilon_{min} + (\epsilon_{initial} - \epsilon_{min}) \times e^{-Decay\ rate \times currunt\ episode} \tag{13}$$

This paper uses the Smooth L1 Loss as the loss function, which smooths the input values close to zero to reduce the occurrence of extreme gradient values during gradient descent. This loss function applies squared error for small errors and linear error for larger errors. By computing the loss between Q_{value} and Q_{target}, and optimizing parameters through backpropagation, the agent learns the actions that maximize expected rewards in a given state after extensive training and optimization, as shown in Equation (14).

$$L(y, \hat{y}) = \begin{cases} 0.5(Q(s, a; \theta) - y)^2 & \text{if } |x - y| < 1 \\ |Q(s, a; \theta) - y| - 0.5 & \text{otherwise} \end{cases} \tag{14}$$

Here, y represents the Q_{target} value given by the target network, as shown in Equation (15).

$$y = r + \gamma max_{a'} Q(s', a') \tag{15}$$

3. Results and Discussion

This section validates the robustness and efficiency of the agricultural drone pesticide application path planning algorithm through simulation experiments. This includes establishing the simulated training environment tasks, setting algorithm parameters, and optimizing the model path.

3.1. Experimental Setup

All simulation experiments in this study were performed on a desktop computer equipped with NVIDIA GeForce RTX 3090 GPUs (24 GB × 2) and running the Ubuntu operating system, using Python 3.11.5 for programming. The parameter settings for the proposed algorithm are shown in Table 2.

Table 2. The parameters of the BL-DQN algorithm.

Parameter	Value	Description
E_P	50,000	maximum episode
S_{max}	100	maximum step size
N_{max}	5	maximum consecutive unrewarded steps
γ	0.95	discount rate
B	128	batch size
M	100,000	experience replay buffer capacity
LR	1×10^{-3}	learning rate
$\epsilon\_initial$	1.0	initial exploration rate
ϵ_min	0.1	minimum exploration rate
Decay rate	3000	controlling the speed at which ϵ epsilonϵ decreases
$Layers_{LSTM1}$	128	the number of neurons in LSTM1
$Layers_{LSTM2}$	128	the number of neurons in LSTM2
n	10	network update frequency
$N\_actions$	4	action space size
Optimizer	Adam	optimizer

Table 2 presents the hyperparameter configurations for the BL-DQN algorithm. The maximum training episodes (E_P) was set to 50,000, with a maximum step count per episode (S_{max}) of 100 to prevent excessive training time. The maximum number of consecutive steps without reward (N_{max}) was set to 5, encouraging exploration in the absence of rewards. The discount factor (γ) was 0.95, highlighting the importance of long-term returns. The batch size (B) was set to 128, and the capacity of the experience replay buffer (M) was 100,000. The

learning rate (LR) was 1×10^{-3}, affecting the step size for weight adjustments. The initial exploration rate ($\epsilon\_initial$) was set to 1.0 to encourage the agent to explore by selecting random actions during the early training phase. The minimum exploration rate (ϵ_min) was 0.1, ensuring that the agent retains some randomness for exploration in the later training stages. The decay rate was set to 3000, controlling the speed at which ϵ epsilonϵ decreases. Each LSTM layer contained 128 neurons ($Layers_{LSTM1}$ and $Layers_{LSTM2}$), enhancing the model's expressive capability. The network update frequency (n) was set to 10, meaning that the parameters of the policy network would be transferred to the target network every 10 training episodes. The action space size ($N\_actions$) was 4, corresponding to movements in four directions. Finally, the Adam optimizer was selected to accelerate convergence and improve learning efficiency.

3.2. Results and Analysis

Due to its high applicability and flexibility, the DFS algorithm was easy to deploy quickly and ensured that all potential paths were explored during the search process. This makes it suitable for tasks requiring complete coverage of specific areas. Therefore, this study conducted a detailed comparison of the DFS algorithm with the BL-DQN and DQN algorithms on six randomly selected grid maps. In these experiments, black grids represent obstacles, gray grids indicate the starting points for the drones, white grids denote task areas, and brown grids mark the target points on the map.

The BL-DQN algorithm was tested on six randomly selected grid maps, comparing the DQN and DFS algorithm. In these experiments, black grids represented obstacles, gray grids indicated the starting points for drones, white grids denoted task areas, and brown squares marked the target points on the map.

The pseudocode for the agricultural drone path planning algorithm and its training process is illustrated in Algorithm 1.

The hyperparameters were initialized, and in each episode, the agent initialized the map and reset the reward and loss values. During the episode, the agent determined its actions based on the current ϵ value: if a randomly generated number was less than ϵ, a random action was selected for exploration; otherwise, the action with the highest Q-value for the current state was chosen for exploitation. After executing the action, the agent observed the reward received and updated the environmental map. Whenever the episode count reached a multiple of n, the parameters of the policy network were copied to the target network, and the loss was calculated to update the network weights. This process continued until the maximum number of episodes (E_P) was reached.

The results of the path planning are depicted in Figures 6–11, where the areas enclosed by red boxes represent repeated paths. Under the predefined task conditions, the BL-DQN algorithm, as a type of statistical learning method, demonstrated superior performance with respect to coverage, number of steps, and repeat coverage rates in comparison to both the DQN and DFS algorithms, while also effectively considering target point planning. The DFS algorithm achieved complete coverage of the map but failed to plan specifically for target points, resulting in numerous non-predefined action trajectories that significantly reduced the operational safety of the drones. Additionally, the traditional DQN algorithm did not meet the task requirements for complete regional coverage or target-oriented planning. In contrast, the proposed BL-DQN algorithm exhibited exceptional performance across all metrics, including complete coverage of task areas, repeat coverage rates, number of steps, and overall task completion.

Algorithm 1 BL-DQN algorithm for agricultural UAV path planning

```
Input: E_P ← 50,000 // Maximum episodes
       n ← 10 // Network update frequency
       ϵ_initial ← 1.0 // Initial exploration rate
       ϵ_min ← 0.1 // Minimum exploration rate
Initialize hyperparameters(learning rate, gamma,)
Initialize Police and Target networks with parameters
for episode in range(E_P) do
  Initialize Map
  Set episode_reward to 0
  Set episode_loss to 0
  done ← false
  while (not Done) do
    if random() < ϵ then
      action ← random_action()
    else
      action ← argmax(Q(state, action))
    end if
    Execute action in Map and observe Reward R and done
    Update Map environment
    if episode mod n = 0 then
      Copy parameters from PolicyNet to TargetNet
    end if
      Passing the parameters of policy net to Target net
    Calculate Loss
    Update networks using backpropagation
  end while
end for
```

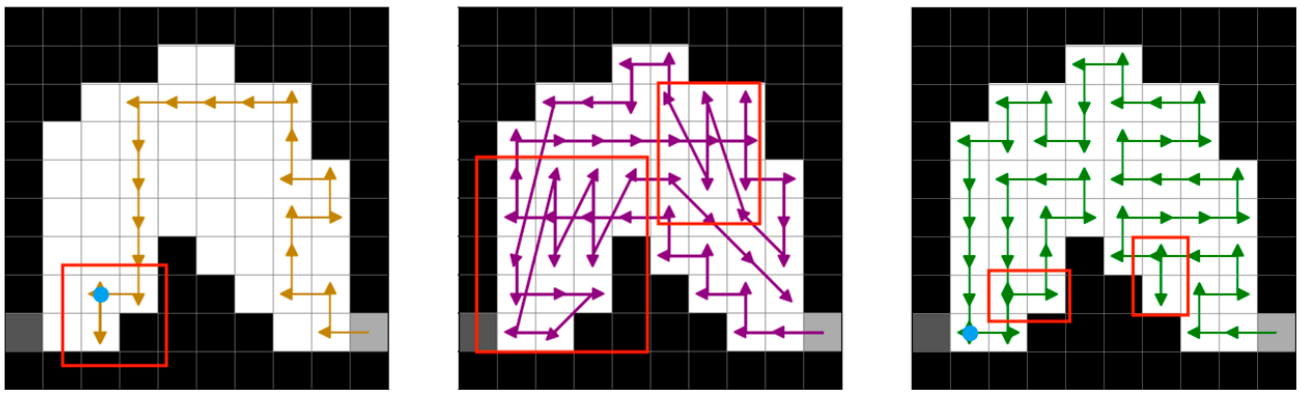

Figure 6. Path planning results of the DQN (**left**), DFS (**middle**), and BL-DQN (**right**) on Map 1.

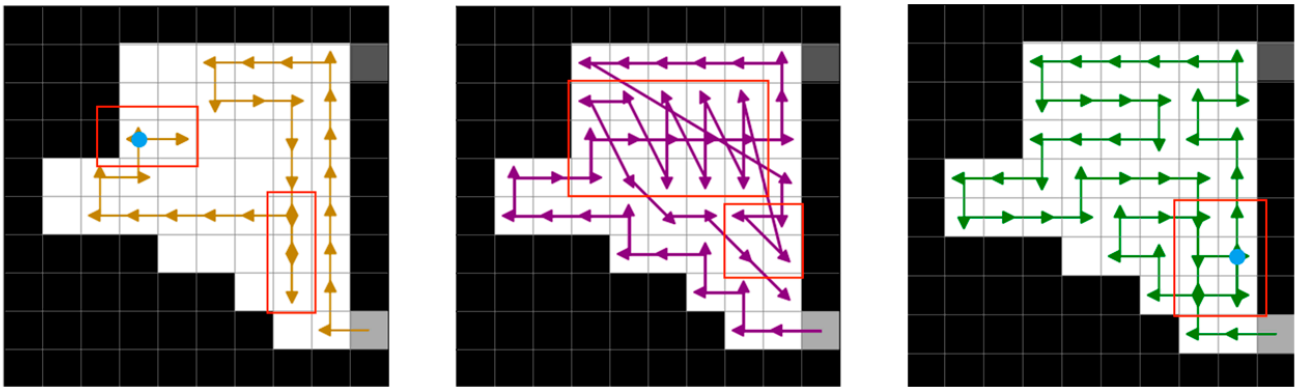

Figure 7. Path planning results of the DQN (**left**), DFS (**middle**), and BL-DQN (**right**) on Map 2.

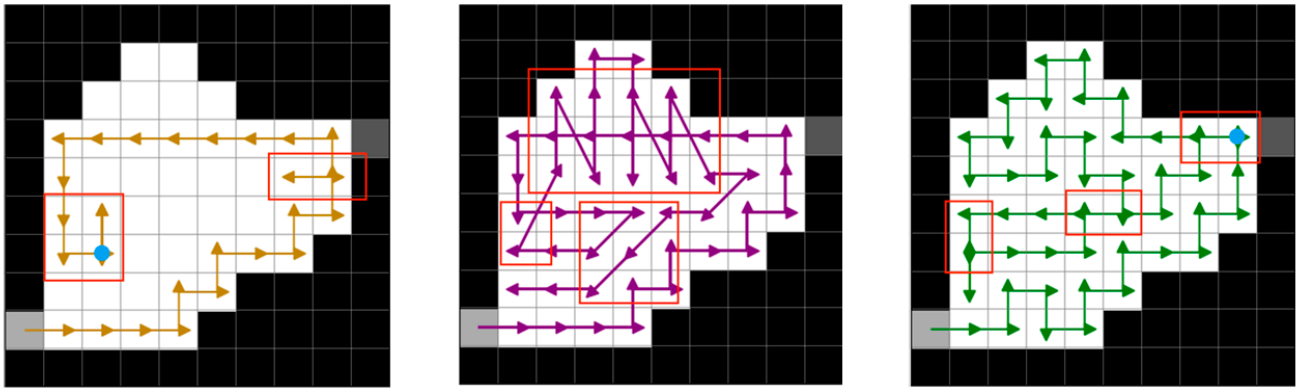

Figure 8. Path planning results of the DQN (**left**), DFS (**middle**), and BL-DQN (**right**) on Map 3.

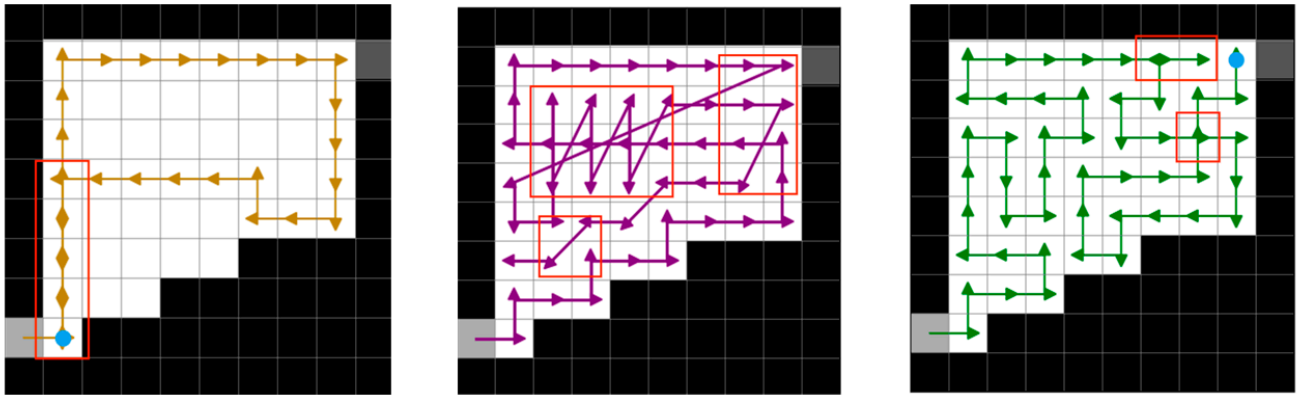

Figure 9. Path planning results of the DQN (**left**), DFS (**middle**), and BL-DQN (**right**) on Map 4.

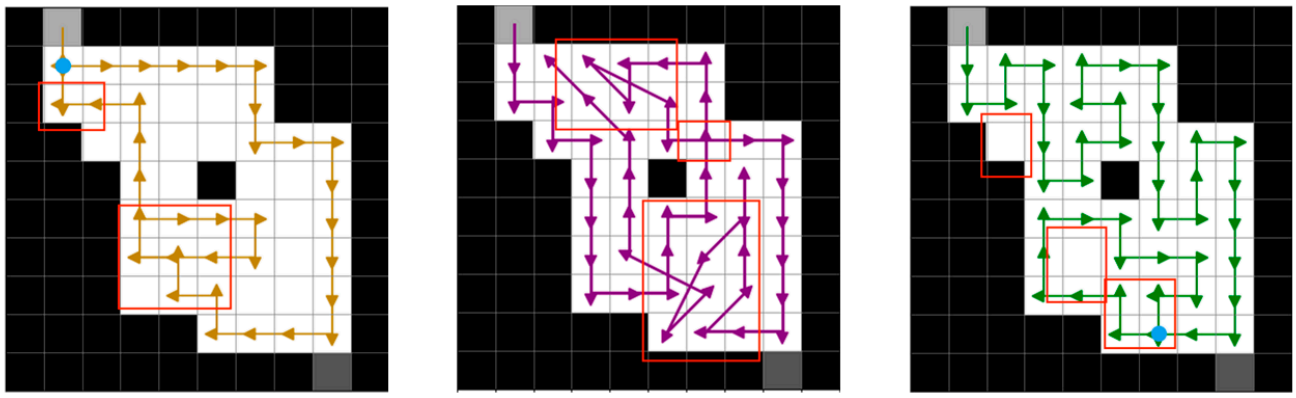

Figure 10. Path planning results of the DQN (**left**), DFS (**middle**), and BL-DQN (**right**) on Map 5.

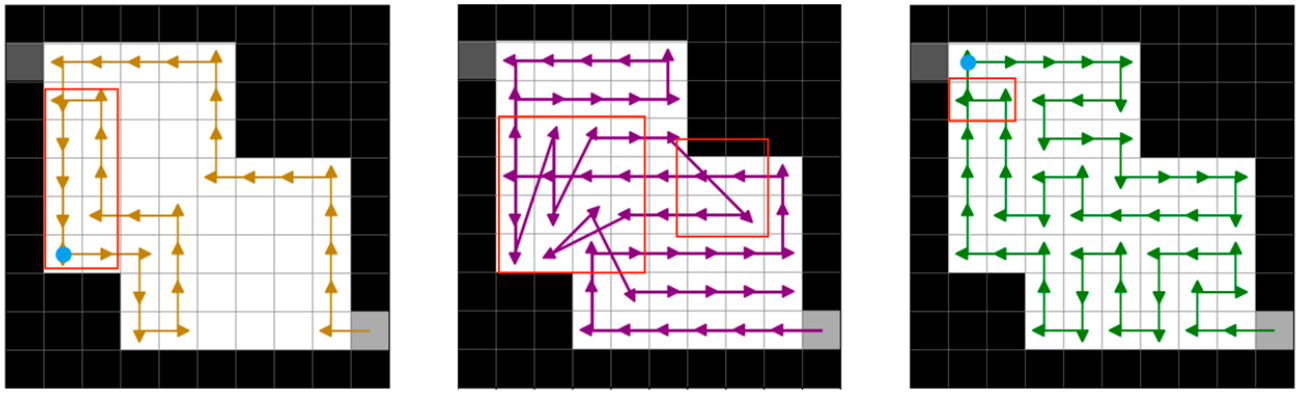

Figure 11. Path planning results of the DQN (**left**), DFS (**middle**), and BL-DQN (**right**) on Map 6.

As shown in Figure 12, the loss values of the BL-DQN algorithm were generally lower than those of the DQN algorithm. During training, the BL-DQN demonstrated greater stability and more efficient problem-solving strategy learning capabilities, with better generalization performance. Although there may be significant fluctuations in the loss values of the BL-DQN under certain specific conditions, leading to occasional performance degradation, its overall performance remained superior to that of the DQN algorithm.

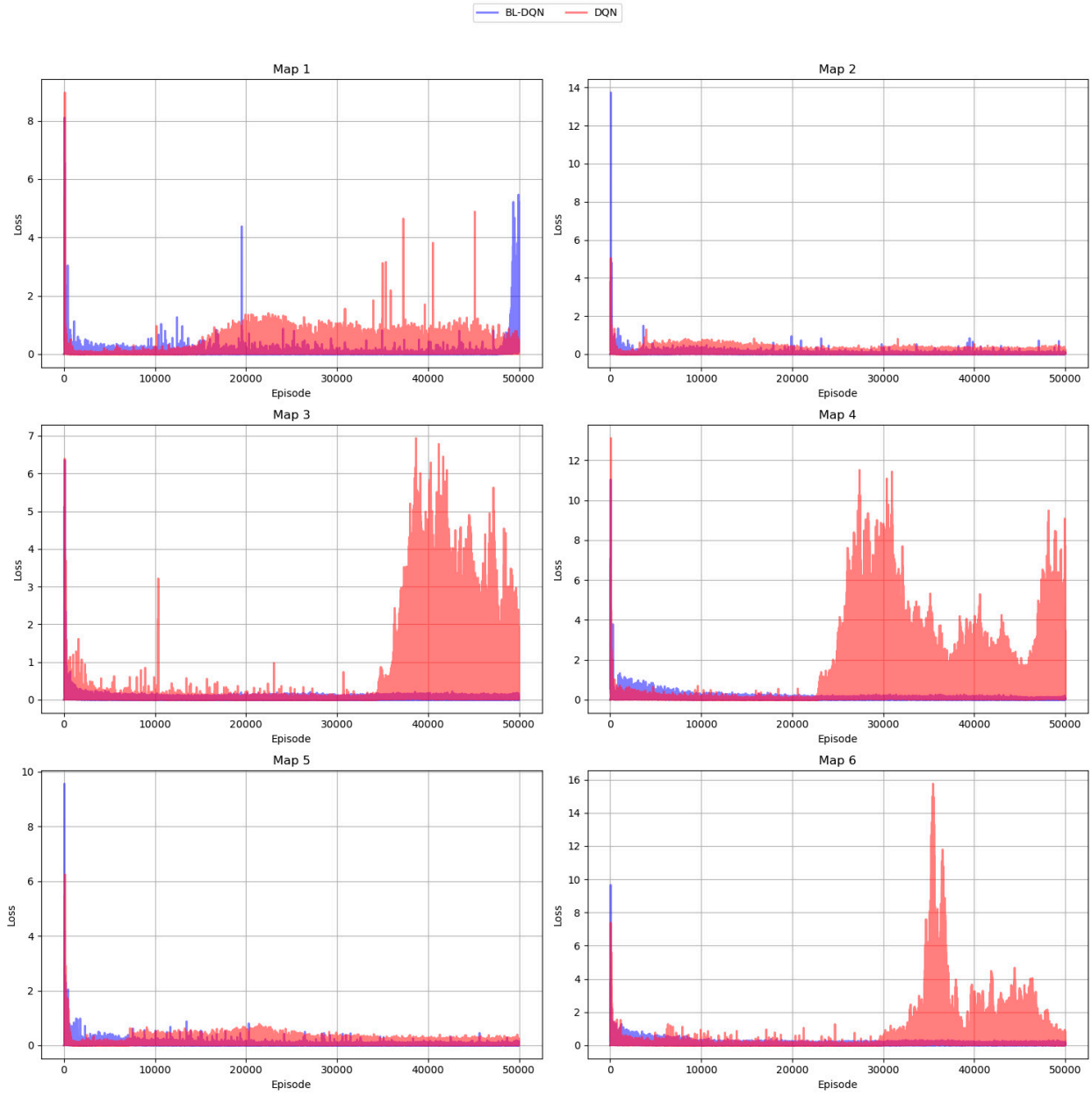

Figure 12. Comparison of loss between BL-DQN algorithm and DQN algorithm.

As illustrated in Figure 13, the reward value of the proposed BL-DQN algorithm significantly outperformed that of the DQN algorithm during training. However, as the complexity of the map increased, the model's performance also exhibited greater fluctuations.

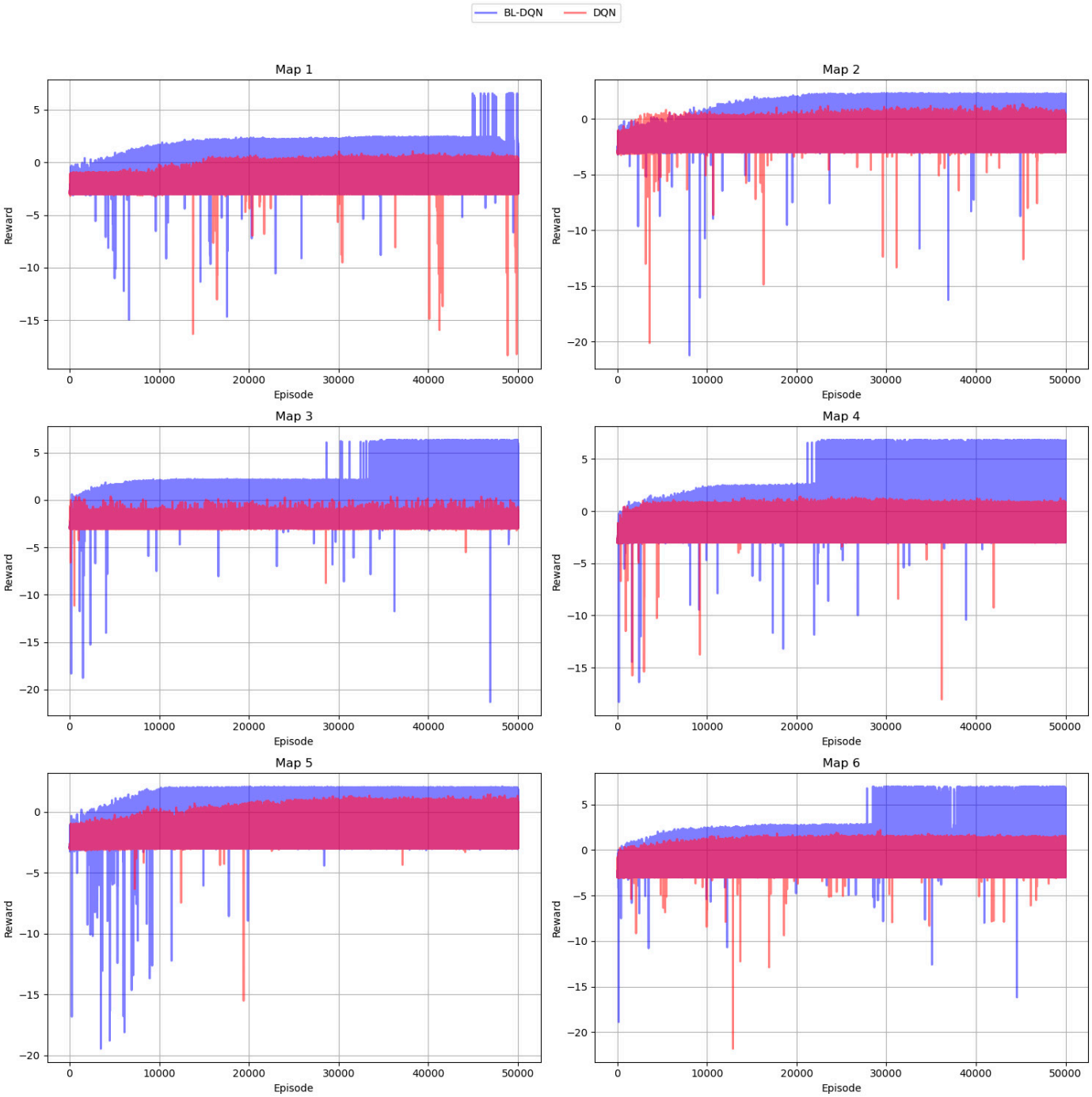

Figure 13. Comparison of reward between BL-DQN algorithm and DQN algorithm.

To perform a quantitative assessment of the three algorithms, the research measured the repetition rate, the steps involved in path planning, coverage rate, number of collisions, target point arrival status, and adherence to rules (including collisions with obstacles and deviations from the specified direction) for each algorithm in completing the task area coverage. The analysis results are shown in Table 3. The analysis shows that the BL-DQN algorithm surpasses the other algorithms in terms of drone path planning, coverage rate, number of steps, target point guidance, and adherence to rules. After 50,000 training iterations, the DQN algorithm did not achieve full coverage and effective target point guidance. Although the DFS algorithm showed stable coverage, it did not match the BL-DQN in terms of target point accuracy and task completion, and it exhibited higher repeat rates and rule violations.

Table 3. Comparison of the experimental results.

Map	Algorithms	Step	Repeat (%)	Coverage (%)	Reach Target	Offense Against Rule	Complete the Task
Map 1	Ours	52	8.3%	100%	True	False	True
	DQN	24	12.5%	43.75%	False	False	False
	DFS	65	35.42%	100%	False	True	False
Map 2	Ours	50	11.11%	97.78%	False	False	False
	DQN	33	13.33%	62.22%	False	False	False
	DFS	62	37.78%	100%	False	True	False
Map 3	Ours	51	10.87%	100%	True	False	True
	DQN	29	6.52%	56.52%	False	False	False
	DFS	58	26.09%	100%	False	True	False
Map 4	Ours	51	4.08%	100%	True	False	True
	DQN	31	10.2%	53.06%	False	False	False
	DFS	69	40.82%	100%	False	True	False
Map 5	Ours	49	4.08%	95.92%	False	False	False
	DQN	40	18.37%	63.26%	False	False	False
	DFS	65	32.65%	100%	False	False	False
Map 6	Ours	53	3.92%	100%	True	False	True
	DQN	38	9.8%	64.71%	False	False	False
	DFS	63	23.53%	100%	False	False	False

3.3. Discussion

The BL-DQN algorithm outperformed traditional DQN and DFS algorithms in terms of the number of steps, coverage rate, and repeat coverage rate. It also achieved significant advancements in task-oriented guidance, indicating that the BL-DQN algorithm enhances efficiency in path planning while better optimizing drone recovery issues. However, despite the notable optimization effects demonstrated by the proposed BL-DQN algorithm in simulated environments, several limitations remain in the current research. In recent years, Pan and colleagues have made innovative improvements to the traditional APF method for formation control in three-dimensional constrained spaces by introducing the concept of rotational potential fields. They developed a novel formation controller utilizing potential function methods [33]. Additionally, Zhou and associates proposed a biologically inspired path planning algorithm for real-time obstacle avoidance in unmapped environments for unmanned aerial vehicles [34]. Fang et al. proposed a solution that integrates distributed network localization with formation maneuvering control. This approach utilizes relative measurement information among agents to achieve real-time positioning and coordinated formation management of multi-agent systems in multi-dimensional spaces [35]. Enrique Aldao and colleagues introduced a real-time obstacle avoidance algorithm based on optimal control theory, suitable for autonomous navigation of UAVs in dynamic indoor environments. By integrating pre-registered three-dimensional model information with onboard sensor data, this algorithm optimizes UAV flight paths and effectively avoids collisions with both fixed and moving obstacles in the environment [36]. He, Y et al. proposed a new stability analysis method for dealing with hybrid systems with double time delays, which has important implications for the design of control strategies in the field of UAVs [37]. Considering the research trends of the past three years, there remains room for exploration in the following areas within this study.

1. A limitation of the current model is its reliance on pre-defined map data. Future research should focus on integrating real-time environmental data, such as weather conditions, crop growth dynamics, pest distribution information, and other disturbances, to enable dynamic adjustments in path planning, ensuring the stability of the drones. The development of such adaptive algorithms will substantially enhance the robustness and effectiveness of the model in practical agricultural applications.
2. Extending the existing single-agent model to a multi-agent framework holds promise for further improving operational efficiency and coverage in large-scale farmland. Investigating how to coordinate multiple drones for joint path optimization, while considering communication constraints and task allocation strategies, represents a challenging yet promising direction for future research.
3. As depicted in Figures 7 and 10, the complexity of the maps resulted in target points being unmet in Map 2 and Map 5. This indicates that there is potential for enhancement. Future efforts will focus on refining the model and adjusting parameters to improve planning efficacy.

4. Conclusions and Future Work

This study improves the Deep Q-Network algorithm by incorporating a Bi-directional Long Short-Term Memory structure, resulting in the BL-DQN algorithm. A target point-oriented reward function suitable for complex farmland environments was designed based on this, and a path planning framework for agricultural drones was developed. This framework includes four modules: remote sensing image acquisition based on the Google Earth platform, task area segmentation using the deep learning U-Net model, grid-based environmental map creation, and coverage path planning. Through simulation experiments, the BL-DQN algorithm achieved a 41.68% improvement in coverage compared with the traditional DQN algorithm. The repeat coverage rate for the BL-DQN was 5.56%, which is lower than the 9.78% achieved by the DQN algorithm and the 31.29% of the DFS algorithm. Additionally, the number of steps required by the BL-DQN was only 80.1% of that of the DFS algorithm. In terms of target point guidance, the BL-DQN algorithm also outperformed both the DQN and DFS, demonstrating superior performance.

These improvements not only highlight the advantages of the BL-DQN algorithm, but also hold significant practical implications for enhancing precision and intelligence in modern agriculture. This indicates that drones equipped with the BL-DQN algorithm can more effectively cover target areas during pest and disease control operations, reducing the impact of multiple applications and missed spray areas. Consequently, this leads to significant savings in time and energy, lowers operational costs, and improves overall efficiency in crop management.

Although positive results were achieved under the assumption of a constant search environment, future research will focus on integrating real-time environmental data (such as weather conditions, crop growth dynamics, and pest distribution) into path planning to develop dynamic environment-adaptive algorithms. Additionally, coordinating multiple drone fleet path planning while considering communication constraints and task allocation strategies will be explored, with the aim of adapting the framework for agricultural drones to further enhance precision farming efficiency and intelligence.

Author Contributions: Conceptualization, H.F., Z.L., X.F. and J.L.; methodology, J.L. and W.Z.; software, H.F. and Y.F.; investigation, L.Z. and W.Z.; resources, Z.L. and Y.F.; writing—original draft, Z.L.; writing—review and editing, H.F.; visualization, Z.L.; supervision, H.F., J.L. and X.F.; funding acquisition, L.Z.; validation, L.Z. and X.F.; data curation, W.Z.; project administration, X.F. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Jilin Province Science and Technology Development Plan Project (20240302092GX).

Data Availability Statement: The original contributions presented in the study are included in the article; further inquiries can be directed to the corresponding author.

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

References

1. Liu, J. Trend forecast of major crop diseases and insect pests in China in 2024. *China Plant Prot. Guide* **2024**, *44*, 37–40.
2. Tudi, M.; Li, H.; Li, H.; Wang, L.; Lyu, J.; Yang, L.; Tong, S.; Yu, Q.J.; Ruan, H.D.; Atabila, A.; et al. Exposure Routes and Health Risks Associated with Pesticide Application. *Toxics* **2022**, *10*, 335. [CrossRef]
3. Benbrook, C.M. Trends in Glyphosate Herbicide Use in the United States and Globally. *Environ. Sci. Eur.* **2016**, *28*, 3. [CrossRef]
4. Fang, X.; Xie, L.; Li, X. Distributed Localization in Dynamic Networks via Complex Laplacian. *Automatica* **2023**, *151*, 110915. [CrossRef]
5. Kim, J.; Kim, S.; Ju, C.; Son, H.I. Unmanned Aerial Vehicles in Agriculture: A Review of Perspective of Platform, Control, and Applications. *IEEE Access* **2019**, *7*, 105100–105115. [CrossRef]
6. Fang, X.; Li, J.; Li, X.; Xie, L. 2-D Distributed Pose Estimation of Multi-Agent Systems Using Bearing Measurements. *J. Autom. Intell.* **2023**, *2*, 70–78. [CrossRef]
7. He, Y.; Zhu, D.; Chen, C.; Wang, Y. Data-Driven Control of Singularly Perturbed Hybrid Systems with Multi-Rate Sampling. *ISA Trans.* **2024**, *148*, 490–499. [CrossRef]
8. Nazarov, D.; Nazarov, A.; Kulikova, E. Drones in Agriculture: Analysis of Different Countries. *BIO Web Conf.* **2023**, *67*. [CrossRef]
9. Ayamga, M.; Akaba, S.; Nyaaba, A.A. Multifaceted Applicability of Drones: A Review. *Technol. Forecast. Soc. Change* **2021**, *167*, 120677. [CrossRef]
10. Tsouros, D.C.; Bibi, S.; Sarigiannidis, P.G. A Review on UAV-Based Applications for Precision Agriculture. *Information* **2019**, *10*, 349. [CrossRef]
11. An, D.; Chen, Y. Non-Intrusive Soil Carbon Content Quantification Methods Using Machine Learning Algorithms: A Comparison of Microwave and Millimeter Wave Radar Sensors. *J. Autom. Intell.* **2023**, *2*, 152–166. [CrossRef]
12. Cabreira, T.M.; Brisolara, L.B.; Paulo, R.F., Jr. Survey on Coverage Path Planning with Unmanned Aerial Vehicles. *Drones* **2019**, *3*, 4. [CrossRef]
13. Aggarwal, S.; Kumar, N. Path Planning Techniques for Unmanned Aerial Vehicles: A Review, Solutions, and Challenges. *Comput. Commun.* **2020**, *149*, 270–299. [CrossRef]
14. Fang, X.; Xie, L. Distributed Formation Maneuver Control Using Complex Laplacian. *IEEE Trans. Autom. Control* **2024**, *69*, 1850–1857. [CrossRef]
15. Tarjan, R. Depth-First Search and Linear Graph Algorithms. In Proceedings of the 12th Annual Symposium on Switching and Automata Theory (swat 1971), East Lansing, MI, USA, 13–15 October 1971; pp. 114–121.
16. Tang, G.; Tang, C.; Claramunt, C.; Hu, X.; Zhou, P. Geometric A-Star Algorithm: An Improved A-Star Algorithm for AGV Path Planning in a Port Environment. *IEEE Access* **2021**, *9*, 59196–59210. [CrossRef]
17. Sang, H.; You, Y.; Sun, X.; Zhou, Y.; Liu, F. The Hybrid Path Planning Algorithm Based on Improved A* and Artificial Potential Field for Unmanned Surface Vehicle Formations. *OCEAN Eng.* **2021**, *223*, 108709. [CrossRef]
18. Hu, L.; Hu, H.; Naeem, W.; Wang, Z. A Review on COLREGs-Compliant Navigation of Autonomous Surface Vehicles: From Traditional to Learning-Based Approaches. *J. Autom. Intell.* **2022**, *1*, 100003. [CrossRef]
19. Ning, Z.; Xie, L. A Survey on Multi-Agent Reinforcement Learning and Its Application. *J. Autom. Intell.* **2024**, *3*, 73–91. [CrossRef]
20. Li, L.; Wu, D.; Huang, Y.; Yuan, Z.-M. A Path Planning Strategy Unified with a COLREGS Collision Avoidance Function Based on Deep Reinforcement Learning and Artificial Potential Field. *Appl. Ocean Res.* **2021**, *113*, 102759. [CrossRef]
21. Cai, Z.; Li, S.; Gan, Y.; Zhang, R.; Zhang, Q. Research on Complete Coverage Path Planning Algorithms Based on A* Algorithms. *Open Cybern. Syst. J.* **2014**, *8*, 418–426.
22. Wang, Z.; Zhao, X.; Zhang, J.; Yang, N.; Wang, P.; Tang, J.; Zhang, J.; Shi, L. APF-CPP: An Artificial Potential Field Based Multi-Robot Online Coverage Path Planning Approach. *IEEE Robot. Autom. Lett.* **2024**, *9*, 9199–9206. [CrossRef]
23. Tang, G.; Tang, C.; Zhou, H.; Claramunt, C.; Men, S. R-DFS: A Coverage Path Planning Approach Based on Region Optimal Decomposition. *Remote Sens.* **2021**, *13*, 1525. [CrossRef]
24. Liu, L.; Wang, X.; Yang, X.; Liu, H.; Li, J.; Wang, P. Path Planning Techniques for Mobile Robots: Review and Prospect. *Expert Syst. Appl.* **2023**, *227*, 120254. [CrossRef]
25. Qin, H.; Shao, S.; Wang, T.; Yu, X.; Jiang, Y.; Cao, Z. Review of Autonomous Path Planning Algorithms for Mobile Robots. *Drones* **2023**, *7*, 211. [CrossRef]
26. Patle, B.K.; Babu, L.G.; Pandey, A.; Parhi, D.R.K.; Jagadeesh, A. A Review: On Path Planning Strategies for Navigation of Mobile Robot. *Def. Technol.* **2019**, *15*, 582–606. [CrossRef]
27. Theile, M.; Bayerlein, H.; Nai, R.; Gesbert, D.; Caccamo, M. UAV Coverage Path Planning under Varying Power Constraints Using Deep Reinforcement Learning. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 24 October 2020–24 January 2021; pp. 1444–1449.
28. Luis, S.Y.; Reina, D.G.; Marín, S.L.T. A Deep Reinforcement Learning Approach for the Patrolling Problem of Water Resources Through Autonomous Surface Vehicles: The Ypacarai Lake Case. *IEEE Access* **2020**, *8*, 204076–204093. [CrossRef]
29. Li, J.; Zhang, W.; Ren, J.; Yu, W.; Wang, G.; Ding, P.; Wang, J.; Zhang, X. A Multi-Area Task Path-Planning Algorithm for Agricultural Drones Based on Improved Double Deep Q-Learning Net. *Agriculture* **2024**, *14*, 1294. [CrossRef]

30. Ma, C.; Wang, L.; Chen, Y.; Wu, J.; Liang, A.; Li, X.; Jiang, C.; Omrani, H. Evolution and Drivers of Production Patterns of Major Crops in Jilin Province, China. *Land* **2024**, *13*, 992. [CrossRef]
31. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; et al. Human-Level Control through Deep Reinforcement Learning. *Nature* **2015**, *518*, 529–533. [CrossRef]
32. Guo, S.; Zhang, X.; Du, Y.; Zheng, Y.; Cao, Z. Path Planning of Coastal Ships Based on Optimized DQN Reward Function. *J. Mar. Sci. Eng.* **2021**, *9*, 210. [CrossRef]
33. Pan, Z.; Zhang, C.; Xia, Y.; Xiong, H.; Shao, X. An Improved Artificial Potential Field Method for Path Planning and Formation Control of the Multi-UAV Systems. *IEEE Trans. Circuits Syst. II-Express Briefs* **2022**, *69*, 1129–1133. [CrossRef]
34. Zhou, Y.; Su, Y.; Xie, A.; Kong, L. A Newly Bio-Inspired Path Planning Algorithm for Autonomous Obstacle Avoidance of UAV. *Chin. J. Aeronaut.* **2021**, *34*, 199–209. [CrossRef]
35. Fang, X.; Xie, L.; Li, X. Integrated Relative-Measurement-Based Network Localization and Formation Maneuver Control. *IEEE Trans. Autom. Control* **2024**, *69*, 1906–1913. [CrossRef]
36. Aldao, E.; Gonzalez-deSantos, L.M.; Michinel, H.; Gonzalez-Jorge, H. UAV Obstacle Avoidance Algorithm to Navigate in Dynamic Building Environments. *Drones* **2022**, *6*, 16. [CrossRef]
37. He, Y.; Zhu, G.; Gong, C.; Shi, P. Stability Analysis for Hybrid Time-Delay Systems with Double Degrees. *IEEE Trans. Syst. Man Cybern. -Syst.* **2022**, *52*, 7444–7456. [CrossRef]