Article

A Deep Reinforcement Learning Strategy Combining Expert Experience Guidance for a Fruit-Picking Manipulator

Yuqi Liu, Po Gao, Change Zheng , Lijing Tian and Ye Tian

School of Technology, Beijing Forestry University, Beijing 100083, China; liuyuqi97@bjfu.edu.cn (Y.L.); pogao1009@163.com (P.G.); T7190221@bjfu.edu.cn (L.T.); tytoemail@bjfu.edu.cn (Y.T.)

* Correspondence: zhengchange@bjfu.edu.cn

Abstract: When using deep reinforcement learning algorithms for path planning of a multi-DOF fruit-picking manipulator in unstructured environments, it is much too difficult for the multi-DOF manipulator to obtain high-value samples at the beginning of training, resulting in low learning and convergence efficiency. Aiming to reduce the inefficient exploration in unstructured environments, a reinforcement learning strategy combining expert experience guidance was first proposed in this paper. The ratios of expert experience to newly generated samples and the frequency of return visits to expert experience were studied by the simulation experiments. Some conclusions were that the ratio of expert experience, which declined from 0.45 to 0.35, was more effective in improving learning efficiency of the model than the constant ratio. Compared to an expert experience ratio of 0.35, the success rate increased by 1.26%, and compared to an expert experience ratio of 0.45, the success rate increased by 20.37%. The highest success rate was achieved when the frequency of return visits was 15 in 50 episodes, an improvement of 31.77%. The results showed that the proposed method can effectively improve the model performance and enhance the learning efficiency at the beginning of training in unstructured environments. This training method has implications for the training process of reinforcement learning in other domains.

Keywords: fruit picking; deep reinforcement learning; path planning; manipulator; expert experience

Citation: Liu, Y.; Gao, P.; Zheng, C.; Tian, L.; Tian, Y. A Deep Reinforcement Learning Strategy Combining Expert Experience Guidance for a Fruit-Picking Manipulator. *Electronics* **2022**, *11*, 311. <https://doi.org/10.3390/electronics11030311>

Academic Editors: Josip Musić, George A. Papakostas and Amir Mosavi

Received: 20 November 2021

Accepted: 17 January 2022

Published: 19 January 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Automatic fruit-picking systems based on a multi-DOF (degree of freedom) manipulator have become a major direction in fruit harvesting in order to increase efficiency and reduce production costs [1]. However, it is more difficult for an automatic multi-DOF manipulator to pick fruits in complex natural environments. Cluttered branches will seriously hinder the fruit-picking process, and the different ripening sequences of fruits present difficulties in picking fruits. Therefore, picking-path planning in unstructured natural environments is one of the main research topics of automatic fruit-picking systems [2–4].

A variety of path-planning algorithms have been proposed, such as the A* algorithm [5,6], ant colony algorithm [7–9], raster algorithm [10], artificial potential field algorithm [11,12], etc. However, these algorithms rely on real-time modelling of the multi-DOF manipulator and the environments, and it is difficult to accurately model the natural picking environments because the variability and the computational complexity of modelling increases exponentially with the number of DOFs. Deep reinforcement learning (DRL) [13–15] is a self-learning approach that enables the agent to interact with the environment to obtain an optimal policy for solving a problem. In recent years, it has become a new solution for the problem of path planning of multi-DOF manipulators in unstructured environments, DeepMind, UC Berkeley, and many others have applied DRL to the trajectory planning problem of robotic arms [13–15]. Even proposed a path-planning method for a multi-arm manipulator based on the SAC (soft actor–critic) algorithm with hindsight replay (HER), which is suitable for multiarm manipulators with static and periodically mobile

obstacles [16]. Chun proposed a deep reinforcement learning algorithm framework that combined the advantages of convolutional neural network (CNN) and deep deterministic policy gradient (DDPG) algorithms to solve how to use delivery task information and automated guided vehicles (AGVs) travel time in the problem of dynamic scheduling of AGV [17]. Xu proposed a good convergence algorithm based on deep reinforcement learning, and designed a reward function including process rewards, such as a speed tracking reward, which solved the problem of sparse rewards [18]. Yu proposed a learning-based, end-to-end path-planning algorithm with security constraints, which included a security reward function, and used it as the reward feedback of the current state to improve the safety guarantee of autonomous exploration process [19]. Wang proposed an online learning method, DDPG, combined with a particle swarm optimization algorithm to improve the speed control performance [20].

However, the low sampling efficiency of DRL limits the application of the algorithm and leads to the slow convergence speed in high-dimensional complex environments. To address these problems, the experience replay was introduced into a deep Q-network (DQN), which improved the sampling efficiency and the learning rate of the algorithm by randomly sampling the samples in the experience replay buffer [21,22]. In addition, the quality of each sample in the experience replay buffer varied greatly, which had a significant impact on the learning of the network. Schaul proposed a prioritized experience replay [23], the core idea of which was that the agent evaluated the quality of samples by their temporal-difference error and more frequently replaying the samples with high expectations. Such an optimization method could result in a loss of sample diversity, which was alleviated with stochastic prioritization and diversity bias. Xie et al. [24] proposed a new dense reward function based on the idea of reward shaping, which improved the training efficiency of DRL-based path planning methods of a multi-DOF manipulator in unstructured environments. The function included an orientation reward function that improved the efficiency of local path planning, and a subtask-level reward function that reduced the ineffective exploration of a multi-DOF manipulator globally. Zheng et al. [25] proposed a deep deterministic policy gradient algorithm based on a stepwise migration strategy, which introduced spatial constraints for stepwise training in an obstacle-free environment, thus speeding up the network convergence, after which the obtained prior knowledge was used to guide the path planning task of a multi-DOF manipulator in a complex unstructured environment.

The above-mentioned methods improved the disadvantages of DRL in unstructured environments, enhancing the performance of the models and improving training efficiency. The DRL-based tasks were performed by calculating cumulative rewards to obtain an optimal policy model, which would have a better performance when a large amount of high-value training samples were available [26]. However, for the fruit-picking task, there were too few valid samples at the beginning of the training due to the randomness and uncertainty of the target fruit and obstacle locations. In addition, there is still a large search space for the cumulative return-based learning approach in unstructured environments; thus, much of the blind exploration behavior of multi-DOF manipulators results in low learning efficiency in the early stage of training [27]. In imitation learning, an agent imitates expert behavior by expert experience provided by human experts to learn the target policy. In this paper, a deep reinforcement learning strategy combined with expert experience was proposed to improve [28] the learning efficiency of the algorithm at the beginning of training period and reduce the blind exploration of the multi-DOF manipulator.

In the fruit-picking task guided by expert experience, the training samples consisted of expert experience that was used to implicitly provide environmental information and improve the initial policy, and self-generated samples that consistently improved the generalization of the policy during the training process. In addition, it was critical to the performance of the algorithm in the ratio between expert experience, frequency of return visits, and self-generated samples in the training samples in the training process.

Therefore, the two main factors of a deep reinforcement learning strategy combined with expert experience were analyzed in this paper.

For the convenience of the readers, the main acronyms used in the paper are described in Table A1/Appendix A.

2. Materials and Methods

2.1. DDPG

Fruit picking by a multi-DOF manipulator can be described as a continuous state-action model in a high-dimensional space, and the deep deterministic policy gradient (DDPG) algorithm can effectively solve the continuous action space task. The DDPG algorithm is based on the actor-critic network architecture, which consists of a policy network (actor) and a value network (critic) [29]. The network architecture of DDPG is shown in Figure 1; it had four neural networks, including the actor network, the actor target network, the critic network, and the critic target network. The actor network was used to select the action that corresponded to the DDPG, and the critic network assessed the strengths and weaknesses of the selected action by calculating the value function. The actor target network and the critic target network were used to update the value network system and the actor network system, but not to carry out online training and updating of network parameters.

Figure 1. The network architecture of DDPG algorithm. S_i denotes the current state, including the joint angle, angular velocity, and the target position of the end-effector; a_i denotes the angular velocity of the multi-degree-of-freedom robot arm; r_i denotes the timely reward of the current environment; and S_{i+1} denotes the state of the next moment [15].

The input of the actor network is the current state of multi-DOF manipulator, s_t , including the information such as the angle, angular velocity of each joint and the target position of the end-effector; and the output of the actor network is the angular velocity of the multi-DOF manipulator, a_t . Then, according to the distance between the current position and the target position of the end-effector, the environment feeds back an immediate reward, r_t . By constantly interacting with the environment and performing the appropriate actions, the multi-DOF manipulator can solve the task of path planning. There are two states for termination: (1) the end-effector of the multi-DOF manipulator reaches the target point or encounters an obstacle; and (2) the number of steps interacting with the environment reaches the upper limit, and Path-planning algorithm for multi-DOF manipulator were in Algorithm 1.

Algorithm 1. Path-planning algorithm for multi-DOF manipulator

1. Initialize the critic network with θ^Q and the actor network with θ^μ .
2. Initialize target networks $\theta'^Q \leftarrow \theta^Q$, $\theta'^\mu \leftarrow \theta^\mu$.
3. Initialize replay buffer R.
4. for episode = 1, M do.
5. Receive initial state s_1 .
6. for $t = 1, T$ do.
 - a. Select the action of the multi-DOF manipulator a_t , $a_t = \mu(s_t|\theta^\mu)$.
 - b. Multi-DOF manipulator executes action a_t and observes reward r_t and new state s_{t+1} .
 - c. Store transition (s_t, a_t, r_t, s_{t+1}) in R.
 - d. Sample a random minibatch of N transitions (s_t, a_t, r_t, s_{t+1}) from R, and update the actor and the critic networks, θ^μ and θ^Q .
 - e. Update the target networks:

$$\theta'^\mu \leftarrow \tau\theta^\mu + (1 - \tau)\theta'^\mu$$

$$\theta'^Q \leftarrow \tau\theta^Q + (1 - \tau)\theta'^Q$$

τ —Update parameters

- f. If s_{t+1} is terminated, the current episode ends, otherwise skip to step a Training terminated.

2.2. Deep Reinforcement Learning Strategies with Expert Experience

In the early stages of fruit picking, the complexity and disorder of the target locations and the fact that the network parameters are randomly generated at the initial stage make the model inefficient and difficult for the network to converge during the training process. Therefore, in this paper, expert experience was used as some initial training samples to train the initial policy of the DRL algorithm, which could reduce the exploration time and the learning difficulty at the beginning of the training process. In this paper, the rapid-exploration random tree (RRT) algorithm was adopted to obtain sufficient expert experience.

The RRT algorithm is a probability-complete global path-planning algorithm that obtains path points by random sampling in the search space, and then achieves a feasible path from the start point to the goal point. The RRT algorithm generates new points by random sampling in the workspace. The random tree expansion diagram for the RRT algorithm is shown in Figure 2. Due to the large randomness of the RRT algorithm, there is a large gap between the generated paths, which increases the diversity of the expert sample and helps to enhance the generalization ability of the model.

Figure 2. Random tree expansion diagram for the RRT algorithm. The red circle indicates the starting point, the green circle indicates the target point, The black circle indicates the path point, the black solid line indicates the path, and the black dashed line with an arrow indicates the current expansion direction of the random tree. P_{rand} denotes the randomly sampled point, P_{parent} denotes the parent point, and P_{new} denotes the new point [1].

The workspace set in the simulation environment was a $0.5\text{ m} \times 0.8\text{ m} \times 0.5\text{ m}$ three-dimensional space centered on the point $(0.25, 0, 1.002)$, and the initial coordinate of the end of the picking robot arm was set to $(-0.094, -0.025, 1.345)$.

$P(x, y, z)$ and the initial position of the end-effector of the multi-DOF manipulator, $P_0(x_0, y_0, z_0)$, were randomly set, and the steps of obtaining expert experience were in Algorithm 2:

Algorithm 2. The steps of obtaining expert experience

- a. Initialize the position of target point P , and the position of the end-effector of the multi-DOF manipulator, P_0 .
 - b. Plan a path from P_0 to P with the RRT algorithm.
 - c. Execute the path to observe the state and obtain a series of states and actions.
 - d. Build a new set of state-action pairs $D = \{(s_1, a_1), (s_2, a_2), (s_3, a_3), \dots\}$
 - e. Calculate the reward, r_i , by the state sequence and reassemble D into (s_i, a_i, r_i, s_{i+1}) .
 - f. Repeat the above steps until obtaining sufficient expert experience.
-

The above steps were used to obtain a large amount of expert experience, then a combination of expert experience and newly generated sample experience in the sample pool was randomly used for training.

3. Experiments and Results

3.1. The Simulation Platform

In this paper, C4D and CoppeliaSim were used to build an apple-picking simulation environment for a multi-DOF manipulator mounted on a mobile platform, as shown in Figure 3. When apple picking, the mobile platform would be braked hard to avoid any movement. The computer configuration in the experiments is shown in Table 1.

Figure 3. Picking scene. On the left side is an apple tree model, in which red spheres indicate ripe apples and green spheres indicate unripe apples; on the right side is a multi-DOF picking manipulator fixed on top of a mobile platform.

Table 1. Computer configuration used in the experiments.

Category	Details
Operating system	Ubuntu 16.04
CPU	64Intel(R) Core(TM) i7-8750H
GPU	NVIDIA GTX 1060
Graphics memory	16 GB
7-DOF manipulator	Franka
Simulation environment	CoppeliaSim
Programming language	Python/MATLAB

3.2. The Multi-DOF Manipulator

To verify the validity of the method, the 7-degree-of-freedom manipulator, Franka, was adopted as the picking robot in this paper. Figure 4 shows the structure of the multi-DOF manipulator.

Figure 4. The multi-DOF manipulator with 7 revolute joints, obtained from software CoppeliaSim.

All joints of a Franka are rotating joints, and joint 7 is the end actuating claw. Figure 5 shows the simplified model of the multi-DOF manipulator, and Table 2 shows the D-H parameters of the multi-DOF manipulator.

Figure 5. Simplified model of the multi-DOF manipulator. The figure shows the kinematic model of the robotic arm built by the D-H method.

Table 2. D-H parameters of the multi-DOF manipulator.

Joint i	$\alpha_i(^{\circ})$	$a_i/(mm)$	$d_i/(mm)$	$\theta_i(^{\circ})$	
				Initial Value	Range
1	0	0	33.3	0	−166~166
2	0	0	0	0	−101~101
3	−90	0	0	90	−166~166
4	0	0	37.8	−90	−176~−4
5	−90	0	0	90	−166~166
6	90	8.7	0	0	−1~215
7	90	8	8	0	−166~166

3.3. Model of Untargeted Fruit

During apple picking, the target objects are the mature fruits in the outer canopy of the tree, where most obstacles are relatively few untargeted fruits, with minimal impact from branch obstacles. Therefore, obstacles in this paper were mainly untargeted fruits, and untargeted fruit obstacles were modelled by simplification of shapes [30–32]. As shown in Figure 6, a sphere model was adopted for an apple obstacle.

Figure 6. Untargeted fruit obstacle model. A sphere envelope with the radius of 5 cm was used to simplify the model of an apple obstacle.

3.4. The Impact of Different Amounts of Expert Experience

Compared to the traditional training process of DDPG, expert experience can implicitly give priority information on the unstructured environment to the multi-DOF manipulator, enabling it to adapt to the environment faster in the early stage of training. However, if only expert experience is used to train, the model tends to be less effective when encountering unknown states. To improve the sample diversity, two experience replay buffers were set up, one for expert experience and the other for newly generated experience. When the network model was trained, m_1 and m_2 samples were respectively taken from the two experience replay buffers and were sent to the network for training as the training samples. As shown in Formula (1), k is the ratio of the number of expert samples to the total number of training samples:

$$k = \frac{m_1}{m_1 + m_2} \quad (1)$$

where m_1 is the number of the expert samples, m_2 is the number of the newly generated samples, and $(m_1 + m_2)$ is the total number of training samples.

As the feedback after performing an action, the reward value is the basis for optimizing the reinforcement learning strategy [33]. Figure 7 shows the reward values changed with

the number of the episode. The curve with a value for k of 0 represented the training curve without expert experience, and was set as the baseline. The other colored curves corresponded to the reward value curves with different k values. As seen in Figure 6, the reward gradually rose, and eventually reached a state of convergence.

Figure 7. The curves of reward value with different constant k values. The rapid growth phase of the reward values is presented in the red circle.

As can be seen in the red circle in Figure 7, the reward values increased faster at the beginning of training due to the involvement of expert experience and reached a high level within 500 episodes, while the reward value of the baseline increased at a relatively slow rate.

The total reward values in the first 500 episodes were calculated at different k values, as shown in Figure 8. In the figure, the lowest total reward value was -151.6 when k was 0, and the highest total reward value was -44 when k was 0.45, an improvement of 70.98% compared to the baseline, indicating that the models combined with expert experience gave the multi-DOF manipulator a larger reward and improved the learning efficiency of the algorithm at the beginning of the training.

Figure 8. The total reward for the first 500 episodes for different constant k values.

Figure 9 shows the average success times and success rate for different k values during the training process, and the point with k value of 0, named the baseline, was not involved with expert experience. Both the average success times and success rate of the baseline were the lowest. Compared with the baseline, the average success times and success rate of the experiments with other k values were both improved. This indicated that under the unstructured environment, expert experience could play an important role in avoiding too much unnecessary exploration of the multi-DOF manipulator in the training process. As the k value gradually increased, both curves tended to rise and then fall, which demonstrated that too much expert experience made the generalization performance of the model worse.

In Figure 8, when the value k was 0.35, compared to the baseline, the average success times increased from 1450 to 3018, an increase of 51.95%, and the average success rate increased from 0.447 to 0.712, an increase of 37.22%. This suggested that expert experience could contribute to the training of the multi-DOF manipulator in unstructured environments, improving the overall learning efficiency.

Figure 9. The curves of average success times and success rate for different constant k values. The curve in (a) indicates the average number of successes. The curve in (b) indicates the average success rate.

Figure 10 shows the path-planning results of the multi-DOF manipulator for a value k of 0.35. The red apple was the target fruit, and the green apple was the obstacle. The red line was the planning path of the multi-DOF manipulator.

Figure 10. Path planning result for constant k value. The red curve indicates the trajectory of the end-effector of the multi-DOF manipulator; the red spheres and green spheres indicate the ripe and unripe apples, respectively; and the background is the land model.

In the above-mentioned training process, the value of k was a constant value, and the learning performance of the multi-DOF manipulator was the best at the value k of 0.35 in terms of the average success times and success rate. However, as can be seen in Figure 9, the reward value with the value k of 0.35 was not the highest. This indicated that different values of k could have different effects on the training process. Therefore, instead of setting k to a constant value at the initial stage of training, the dynamic decline from an initial value to 0.35 was used to evaluate the effect of the dynamic k value on the training process.

Figure 11 shows the reward value curves for dynamic k values from different initial values to 0.35. The decline in the k value was made within the first 500 episodes, after which the k value of 0.35 was maintained. The red line indicates the training curve with the value k of 0, and no expert experience involved; namely, the baseline. The other colored

curves indicate the curves respectively corresponding to different initial k values. As can be seen in Figure 10, the reward value gradually increased, and eventually reached a state of convergence.

Figure 11. Reward value curves for dynamic k values.

The total reward values for the first 500 episodes corresponding to different initial k values are shown in Figure 12, in which the points with the value k of 0 are the reward values of the baseline experiment and the remaining points are the reward values with expert experience involved. When the k value was dynamic, the training process with expert experience had a higher reward value than the baseline experiment in the first 500 episodes, and the highest reward value was -40.15 when the k value was 0.45, an improvement of 73.52% compared to the baseline, indicating that the initial performance of the model was significantly improved by expert experience at the early stage of training, which accelerated the learning speed.

Figure 12. The total reward for the first 500 episodes for dynamic k values.

Figure 13 shows the average success times and success rate curves corresponding to different initial k values, in which the points with the value k of 0 are the average success times and success rate of the baseline experiments. As the initial k value gradually increased, all the curves in Figure 12 showed a trend of increasing first, and then decreased. Compared to the baseline experiment, the average success times and success rate of each experiment increased when the k value was dynamic, and when the initial k value was 0.45, the average success times increased from 1450 to 3088, an increase of 53.04%, and the average success rate increased from 0.447 to 0.721, an increase of 38%.

Figure 13. The curves of average success times and success rate for dynamic k values. The curve in (a) indicates the average number of successes. The curve in (b) indicates the average success rate.

Figure 14 shows the path-planning results of the multi-DOF manipulator with an initial k value of 0.45. The red apple was the target fruit and the green apple was the obstacle. The red line was the planning path of the multi-DOF manipulator.

Figure 14. Path-planning result for dynamic k values from 0.45 to 0.35. The red curve indicates the trajectory of the end-effector of the multi-DOF manipulator; the red spheres and green spheres indicate the ripe and unripe apples, respectively; and the background is the land model.

Table 3 compares the performance of the constant k value with dynamic k values and the baseline. It shows that the dynamic k value had the highest reward value in the early training period, 8.75% higher than the constant k value of 0.45 and 17.64% higher than the constant k value of 0.35; the dynamic k value also outperformed the constant k value in terms of the average success times and success rate. It was concluded that dynamic k values in the early training period were more effective in improving the learning efficiency of the model, and more expert experience involved in the early stage of training could enable the model to adapt to the environment faster; thus, the number of expert experience needed to be appropriately reduced, and newly generated samples were added in the following stage of training to overcome the disadvantage of insufficient diversity of expert experience and reduce its influence in policy updating.

To test the performance of the model, the average time and average path length of 100 successful path planings were chosen as evaluation metrics, and comparative experiments were made between the proposed method and the RRT algorithm, as shown in Table 4 [1]. The path-planning result of the RRT algorithm is shown in Figure 15.

Table 3. The total reward for the first 500 episodes with different k values.

	k = 0	Constant Value of k		Dynamic Value of k
		0.35	0.45	From 0.45 to 0.35
Total reward	−151.6	−48.75	−44	−40.15
Success times	1450	3018	2617	3088
Success rate	0.447	0.712	0.599	0.721

Table 4. Comparison of test results.

	RRT Algorithm	Baseline	Constant Value of k		Dynamic Value of k
			0.35	0.45	From 0.45 to 0.35
Time/s	8.21	7.40	5.51	6.12	5.14
Path length/m	0.83	0.76	0.69	0.71	0.67

Figure 15. Path-planning result for RRT algorithm. The red curve indicates the trajectory of the end-effector of the multi-DOF manipulator; the red spheres and green spheres indicate the ripe and unripe apples, respectively; and the background is the land model.

In Table 4, it can be seen that the path-planning time of the baseline was decreased by 9.86% and the path length declined by 7.52%, compared to those of the RRT algorithm. It was concluded that the deep reinforcement learning method had an advantage over the RRT algorithm. In addition, the performance of the dynamic k value was improved by 30.54% in the path-planning time and 11.84% in the path length compared to the baseline, which showed that the proposed method in this paper was verified. Compared with the paths in Figures 10 and 14, the path length of the RRT algorithm with more inflection points was longer than the proposed algorithm, and the time to complete a single picking was also longer than the proposed algorithm. Moreover, for a traditional algorithm such as RRT, the result of path planning is affected by many factors, such as step length, the number of iterations, and so on, which also made the traditional algorithm more complicated than the proposed algorithm in this paper.

3.5. Frequency of Return Visits to Expert Experience

In this section, we compare expert experience and policy as teacher and student, respectively. Analyzed from the perspective of reinforced learning, policy learning by students may not require the full guidance of the teacher, who usually provides regular guidance to students to correct errors they encounter in their learning. The model may be given some guidance by visiting expert experience to correct its blind exploration in

the training process. When the frequency of return visit is too low, the guidance given to the model will be ineffective, while when the frequency of return visit is too high, the generalization of the model will be reduced. Therefore, this section explores the effect of a regular return visit to expert experience on the learning process through experiments, which were set to introduce expert experience every 50 episodes.

Figure 13 shows the curve of reward values corresponding to the different numbers of return visits. The expert experience was set to be engaged within the first 500 episodes, after which the expert experience was accessed at a set frequency of visits. The red line was the training curve without visiting expert experience, named as the baseline, and the other colored curves represented the training curves according to the different frequency of visiting expert experience. As can be seen from the curves, the reward gradually increased as the training proceeded, and eventually reached a state of convergence.

As can be seen in Figure 16, the reward with expert experience at the early stages of training increased rapidly and reached a high level within 500 episodes, while the reward of the baseline experiment was relatively small and was still increasing within 500 episodes. As a result of regular visits to expert experience, the curves of reward increased faster than the baseline experiment, and the reward remained high, but the reward values decreased slightly in the following training period.

Figure 16. Reward value curves with different return visit times.

Figure 17 shows the curves of the average success times and success rates corresponding to the different numbers of visits. As the frequency of return visits gradually increased, all the curves tended to rise and then fall. Compared to the baseline experiment, when the number of return visits was 15, the average success times were 2549, an increase of 43.11%; the average success rate was 0.589, an increase of 24.11%. In addition, the times for 100 successful path planings of the baseline model and the model with the number of return visits of 15 were 740 s and 650 s, respectively.

The above experiments showed that the reward increased to a higher score within a short period time after the initialization of the policy using expert experience, denoting that the guidance based on expert experience could avoid too much unnecessary exploration, while regular visits to expert experience could enable the multi-DOF manipulator to receive some guidance at the following stage of training, which could correct the errors in policy updating and maintain the reward at a high level. However, it was found in the experiments that when the number of return visits was too large, the performance of the training was greatly reduced. This was because the higher the frequency of return visits was, the less stable the updated policy was, making the curve less stable and slightly decreasing the reward value in the following stages of training.

Figure 17. The curves of average success times and success rate. The curve in (a) indicates the average number of successes. The curve in (b) indicates the average success rate.

Figure 18 shows the path-planning results for the multi-DOF manipulator when the number of return visits was 15. The red apple was the target fruit, the green apple was the obstacle, and the red line was the planning path of the multi-DOF manipulator.

Figure 18. Path-planning result of the multi-DOF manipulator. The red curve indicates the trajectory of the end-effector of the multi-DOF manipulator; the red spheres and green spheres indicate the ripe and unripe apples, respectively; and the background is the land model.

4. Discussion

In this paper, a deep reinforcement learning strategy incorporating expert experience was proposed to address the problem of blind exploration of the robot arm in the early stage of training for models in unstructured environments. In the complex obstacle scene, too much blind exploration made the early learning efficiency very low. The effects of different k values and different numbers of return visits on the training results were mainly discussed and demonstrated in the simulation environment. In this paper, the k value was obtained by the enumeration method, with the advantage of being easy to implement. However, there may be more complex relationships between path-planning results and the proportion of expert experience. Enlightened by [33], we will discuss this by using the neural network to optimize the k value more deeply in further work.

In order to obtain effective expert experience, we needed to manually choose the samples to form an expert sample base. Meanwhile, the simulation environment of this paper was relatively simple compared with the actual picking scenario. In the actual picking scenario, too small of a distance between fruits and too small of a distance between fruits and branches will increase the complexity of the environment, which will put higher requirements on the path-planning accuracy of the robotic arm. Therefore, a future study

will include more complex experimental environments in the training process to improve the applicability of the model in more realistic scenes.

5. Conclusions

To solve the problem of blind exploration of DDPG at the initial stage of training, a reinforcement learning strategy combined with expert experience, generated by the RRT algorithm, was proposed in this paper. Moreover, the ratios of expert experience to newly-generated samples and the frequency of return visits to expert experience were analyzed by the simulation experiments. In terms of the average success times and success rate of the simulation experiments, some conclusions can be made in this paper. First, the proposed method was verified by comparing with and without expert experience. Second, the dynamic k value, which declined from 0.45 to 0.35, was more effective in improving learning efficiency of the model than the constant k value. Third, when the frequency of return visits was 15 in 50 episodes, the highest success rate was achieved. The training method proposed in this paper has implications for the training process of reinforcement learning in other domains. However, picking experiments in real orchards need to be further conducted and analyzed in subsequent studies due to the gap between the simulation environment and the natural picking environment.

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

Funding: This research was funded by the National Natural Science Foundation of China under grant number 31971668.

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

Appendix A

We summarized all the words containing acronyms in the paper and listed them in the order of occurrence in Table A1.

Table A1. List of acronyms.

Acronyms	Full Name
DOF	Degree of freedom
DRL	Deep reinforcement learning
SAC	Soft actor-critic
HER	Hindsight replay
CNN	Convolutional neural network
DDPG	Deep deterministic policy gradient
AGV	Automated guided vehicles
DQN	Deep Q-network
RRT	Rapid-exploration random tree

References

1. Cao, X.; Zou, X.; Jia, C.; Chen, M.; Zeng, Z. RRT-based path planning for an intelligent litchi-picking manipulator. *Comput. Electron. Agric.* **2019**, *156*, 105–118. [[CrossRef](#)]
2. Liu, X.; Zhao, D.; Jia, W.; Ruan, C.; Ji, W. Fruits segmentation method based on super pixel features for apple harvesting robot. *Trans. Chin. Soc. Agric. Mach.* **2019**, *50*, 15–23.
3. Liu, J.Z.; Zhu, X.X.; Yuan, Y. Depth-sphere transversal method for on-branch citrus fruit recognition. *Trans. Chin. Soc. Agric. Mach.* **2017**, *48*, 32–39.
4. Nguyen, T.T.; Kayacan, E.; De Baedemaeker, J.; Saeys, W. Task and motion planning for apple harvesting robot. *IFAC Proc. Vol.* **2013**, *46*, 247–252. [[CrossRef](#)]

5. Herich, D.; Vaščák, J.; Zolotová, I.; Brecko, A. Automatic Path Planning Offloading Mechanism in Edge-Enabled Environments. *Mathematics* **2021**, *9*, 3117. [\[CrossRef\]](#)
6. Jia, Q.; Chen, G.; Sun, H.; Zheng, S. Path planning for space manipulator to avoid obstacle based on A* algorithm. *J. Mech. Eng.* **2010**, *46*, 109–115. [\[CrossRef\]](#)
7. Majeed, A.; Hwang, S.O. A Multi-Objective Coverage Path Planning Algorithm for UAVs to Cover Spatially Distributed Regions in Urban Environments. *Aerospace* **2021**, *8*, 343. [\[CrossRef\]](#)
8. Yuan, Y.; Zhang, X.; Hu, X.A. Algorithm for optimization of apple harvesting path and simulation. *Trans. CSAE* **2009**, *25*, 141–144.
9. Zhang, Q.; Chen, B.; Liu, X.; Liu, X.; Yang, H. Ant colony optimization with improved potential field heuristic for robot path planning. *Trans. Chin. Soc. Agric. Mach.* **2019**, *15*, 642733.
10. Wang, Y.; Chen, H.; Li, H. 3D path planning approach based on gravitational search algorithm for sprayer UAV. *Trans. Chin. Soc. Agric. Mach.* **2018**, *49*, 1–7.
11. Tang, Z.; Xu, L.; Wang, Y.; Kang, Z.; Xie, H. Collision-Free Motion Planning of a Six-Link Manipulator Used in a Citrus Picking Robot. *Appl. Sci.* **2021**, *11*, 1336. [\[CrossRef\]](#)
12. Szczepanski, R.; Bereit, A.; Tarczewski, T. Efficient Local Path Planning Algorithm Using Artificial Potential Field Supported by Augmented Reality. *Energies* **2021**, *14*, 6642. [\[CrossRef\]](#)
13. Gu, S.; Holly, E.; Lillicrap, T.; Levine, S. Deep reinforcement learning for robotic manipulation with asynchronous Off-Policy updates. *arXiv* **2016**, arXiv:1610.00633.
14. Wen, S.; Chen, J.; Wang, S.; Zhang, H.; Hu, X. Path planning of humanoid arm based on deep deterministic policy gradient. In Proceedings of the 2018 IEEE International Conference on Robotics and Biomimetics (ROBIO), Kuala Lumpur, Malaysia, 12–15 December 2018.
15. Kim, M.; Han, D.K.; Park, J.H.; Kim, J.S. Motion planning of robot manipulators for a smoother path using a twin delayed deep deterministic policy gradient with hindsight experience replay. *Appl. Sci.* **2020**, *10*, 575. [\[CrossRef\]](#)
16. Prianto, E.; Park, J.H.; Bae, J.H.; Kim, J.S. Deep Reinforcement Learning-Based Path Planning for Multi-Arm Manipulators with Periodically Moving Obstacles. *Appl. Sci.* **2021**, *11*, 2587. [\[CrossRef\]](#)
17. Chen, C.; Hu, Z.H.; Wang, L. Scheduling of AGVs in Automated Container Terminal Based on the Deep Deterministic Policy Gradient (DDPG) Using the Convolutional Neural Network (CNN). *Mar. Sci. Eng.* **2021**, *9*, 1439. [\[CrossRef\]](#)
18. Xu, X.; Chen, Y.; Bai, C. Deep Reinforcement Learning-Based Accurate Control of Planetary Soft Landing. *Sensors* **2021**, *21*, 8161. [\[CrossRef\]](#) [\[PubMed\]](#)
19. Yu, X.; Wang, P.; Zhang, Z. Learning-Based End-to-End Path Planning for Lunar Rovers with Safety Constraints. *Sensors* **2021**, *21*, 796. [\[CrossRef\]](#)
20. Wang, C.S.; Guo, C.W.; Tsay, D.M.; Perng, J.W. PMSM Speed Control Based on Particle Swarm Optimization and Deep Deterministic Policy Gradient under Load Disturbance. *Machines* **2021**, *9*, 343. [\[CrossRef\]](#)
21. Kim, J.-H.; Huh, J.-H.; Jung, S.-H.; Sim, C.-B. A Study on an Enhanced Autonomous Driving Simulation Model Based on Reinforcement Learning Using a Collision Prevention Model. *Electronics* **2021**, *10*, 2271. [\[CrossRef\]](#)
22. Sun, Y.; Yuan, B.; Zhang, T.; Tang, B.; Zheng, W.; Zhou, X. Research and Implementation of Intelligent Decision Based on a Priori Knowledge and DQN Algorithms in Wargame Environment. *Electronics* **2020**, *9*, 1668. [\[CrossRef\]](#)
23. Schaul, T.; Quan, J.; Antonoglou, I.; Silver, D. Prioritized experience replay. *arXiv* **2015**, arXiv:1511.05952.
24. Xie, J.; Shao, Z.; Li, Y.; Guan, Y.; Tan, J. Deep reinforcement learning with optimized reward functions for robotic trajectory planning. *IEEE Access* **2019**, *7*, 105669–105679. [\[CrossRef\]](#)
25. Zheng, C.; Gao, P.; Gan, H.; Tian, Y.; Zhao, Y. Trajectory planning method for apple picking manipulator based on stepwise migration strategy. *Trans. Chin. Soc. Agric. Mach.* **2020**, *51*, 15–23.
26. Sun, H.; Zhang, W.; Yu, R.; Zhang, Y. Motion Planning for Mobile Robots—Focusing on Deep Reinforcement Learning: A Systematic Review. *IEEE Access* **2021**, *9*, 69061–69081. [\[CrossRef\]](#)
27. Chen, P.; Lu, W. Deep Reinforcement Learning Based Moving Object Grasping. *Inf. Sci.* **2021**, *565*, 62–76. [\[CrossRef\]](#)
28. Zheng, J. Simulation for Manipulator Trajectory Planning Based on Deep Reinforcement Learning. Master's Thesis, University of Electronic Science and Technology of China, Chengdu, China, 2020.
29. Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; Wierstra, D. Continuous control with deep reinforcement learning. *arXiv* **2015**, arXiv:1509.02971.
30. Yin, J.; Wu, C.; Yang, S.X.; Mittal, G.S.; Mao, H. Obstacle-avoidance path planning of robot arm for tomato-picking robot. *Trans. Chin. Soc. Agric. Mach.* **2012**, *43*, 171–175.
31. Cai, J.R.; Zhao, J.W.; Thomas, R.; Macco, K. Path planning of fruits harvesting robot. *Trans. Chin. Soc. Agric. Mach.* **2007**, *38*, 102–105, 135.
32. Hou, Y.; Liu, L.; Wei, Q.; Xu, X.; Chen, C. A novel DDPG method with prioritized experience replay. In Proceedings of the 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Banff, AB, Canada, 5–8 October 2017; pp. 316–321.
33. Hester, T.; Vecerik, M.; Pietquin, O.; Lanctot, M.; Schaul, T.; Piot, B.; Horgan, D.; Quan, J.; Sendonaris, A.; Osband, I.; et al. Deep q-learning from demonstrations. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018.