aerospace

MDPI

Article

A Policy-Reuse Algorithm Based on Destination Position Prediction for Aircraft Guidance Using Deep Reinforcement Learning

Zhuang Wang [1], Yi Ai [1,2,*], Qinghai Zuo [1], Shaowu Zhou [1] and Hui Li [3]

[1] College of Air Traffic Management, Civil Aviation Flight University of China, Guanghan 618307, China
[2] Guangxi Key Laboratory of International Join for China-ASEAN Comprehensive Transportation, Nanning University, Nanning 530200, China
[3] College of Computer Science, Sichuan University, Chengdu 610065, China
* Correspondence: aiyi@cafuc.edu.cn

Abstract: Artificial intelligence for aircraft guidance is a hot research topic, and deep reinforcement learning is one of the promising methods. However, due to the different movement patterns of destinations in different guidance tasks, it is inefficient to train agents from scratch. In this article, a policy-reuse algorithm based on destination position prediction is proposed to solve this problem. First, the reward function is optimized to improve flight trajectory quality and training efficiency. Then, by predicting the possible termination position of the destinations in different moving patterns, the problem is transformed into a fixed-position destination aircraft guidance problem. Last, taking the agent in the fixed-position destination scenario as the baseline agent, a new guidance agent can be trained efficiently. Simulation results show that this method can significantly improve the training efficiency of agents in new tasks, and its performance is stable in tasks with different similarities. This research broadens the application scope of the policy-reuse approach and also enlightens the research in other fields.

Keywords: aircraft guidance; deep reinforcement learning; policy reuse

Citation: Wang, Z.; Ai, Y.; Zuo, Q.; Zhou, S.; Li, H. A Policy-Reuse Algorithm Based on Destination Position Prediction for Aircraft Guidance Using Deep Reinforcement Learning. *Aerospace* **2022**, *9*, 632. https://doi.org/10.3390/aerospace9110632

Academic Editor: Álvaro Rodríguez-Sanz

Received: 20 August 2022
Accepted: 20 October 2022
Published: 22 October 2022

1. Introduction

Aircraft guidance, especially in the scenario with a dynamic destination in three-dimensional space, is our research focus because it is widely used in reality. In air traffic control, aircraft can be guided to resolve conflicts or for follow-up flights. In aircraft carrier operations, aircraft need to be guided to land on the deck of a carrier moving at full speed. In air combat, fighters are guided to reach a position of advantage. The destinations have different movement patterns in different scenarios, so it is necessary to study a general method to solve the problem of aircraft guidance.

An aircraft guidance method is used to generate a trajectory or a set of instructions to guide an aircraft to a moving destination in a certain direction in three-dimensional continuous space. A series of advanced algorithms, such as optimal control techniques [1], geometry methods [2], model predictive control [3] and knowledge/rule-based decisions [4], have been investigated to guide and control aircraft. With the development of artificial intelligence, more and more scholars use intelligent algorithms for aircraft guidance.

Deep reinforcement learning (DRL) [5] is a type of artificial intelligence with the advantages of high decision-making efficiency and model or data independence. It has been utilized in many fields and has obtained great achievements of human-level or superhuman performance [6–8]. The theory of DRL is very suitable for solving sequential decision-making problems, including aircraft guidance.

Mainstream DRL algorithms, such as Deep Q Network (DQN) [9], Deep Deterministic Policy Gradient (DDPG) [10] and Proximal Policy Optimization (PPO) [11], have been

adopted to solve aircraft guidance problems. In [12], an RL model is built to improve the autonomy of gliding guidance for complex flight missions. An analytical terminal velocity prediction method considering maneuvering flight is studied to adjust the maneuvering amplitude intelligently in velocity control. In [13], a DQN algorithm is used to generate a trajectory to perform a perched landing on the ground. In this algorithm, the noise is added to the numerical model of airspeed in the training environment, which is more in line with the actual scenario. UAV autonomous landing on a moving platform is studied in [14]. DDPG is used as the training algorithm, and a reward function including the position, velocity and acceleration of the UAV is designed. UAV landing can be completed in both simulation and real flight scenarios. However, the orientation of the landing platform is not considered, and vertical velocity control is not included in the action set. A DRL-based guidance law is proposed in [15] to deal with maneuvering of high-speed targets. Based on the DQN algorithm, a relative-motion model is established and reward function is designed that can obtain continuous acceleration commands, make the LOS rate converge to zero rapidly, and hit the maneuvering target using only the LOS rate. In [16], an actor–critic model with a reward-shaping algorithm is proposed for guiding an aircraft to 4D waypoints at a certain heading. The trained agent guides an aircraft to move along the waypoint in three-dimensional space by outputting a discrete heading angle, horizontal velocity and vertical velocity. In the research field of air traffic control, there are also many DRL-based methods for aircraft guidance to avoid conflicts [17].

The authors of this article studied aircraft guidance based on the PPO algorithm [18]. A shaped reward function including instruction continuity and relative position is presented that can significantly improve convergence efficiency and trajectory performance. However, the reward-shaping parameters are only given qualitatively and lack detailed design and sufficient theoretical support, and thus need further improvement. In addition, a pre-trained algorithm that directly reuses an agent is proposed for guidance tasks with different kinds of moving destinations. Using this algorithm, an agent that will be used in a new task can be trained quickly based on the existing agent. Pre-training is a method of direct policy reuse that can only be used between scenarios with high similarity. Though the study preliminarily verifies the feasibility of using policy reuse to study aircraft guidance, the scope of application needs to be expanded.

A policy-reuse approach based on destination position prediction is proposed based on our previous research to solve the above problems. The main contributions of this article are:

1. A meticulous design of reward-shaping parameters is done based on theoretical properties, and the consistency of optimal policy before and after reward shaping is proved.
2. By predicting the position of the destination at the possible termination time, the old policy/agent can be reused for efficiently training new agents in multiple scenarios. The application scope of the policy-reuse method is broadened, and it can be effectively used in scenarios with low similarity to the old scenario.

The rest of this paper is organized as follows. Section 2 describes the training framework and process of aircraft guidance using DRL. Section 3 defines the DRL model for aircraft guidance. Section 4 designs the policy-reuse approach based on destination position prediction using DRL. Section 5 carries out simulations to demonstrate the effectiveness of the proposed algorithm. Section 6 concludes this paper.

2. Aircraft Guidance Problem Formulation

2.1. Problem Statement

The objective of aircraft guidance is to guide an aircraft from its current position to the target position, as shown in Figure 1. At any time t, the position of the aircraft is $(x_t^a, y_t^a, z_t^a, \theta_t^a, \psi_t^a, \phi_t^a)$, and the position of the destination is $(x_t^d, y_t^d, z_t^d, \psi_t^d)$. The superscripts a and d represent the aircraft and its destination, respectively. The subscripts t and t_f represent the current time and the final time, respectively. The term (x, y, z) is the three-

dimensional coordinates, and θ, ψ and ϕ are flight-path angle, heading angle and bank angle, respectively.

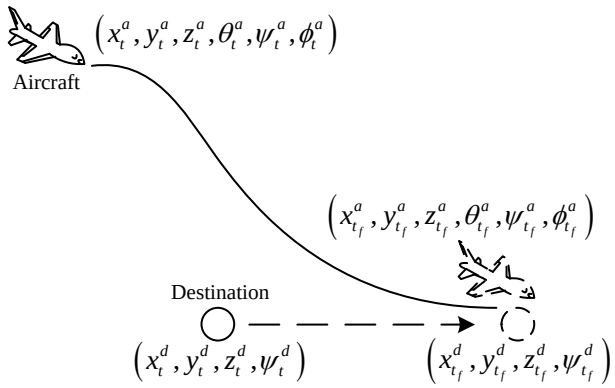

Figure 1. Aircraft guidance problem in three dimensional continuous space.

The aircraft needs to reach the destination within the specified time period without flying out of the airspace; that is, it must meet the space–time constraints:

$$\begin{cases} x \in [x_{min}, x_{max}], y \in [y_{min}, y_{max}], z \in [z_{min}, z_{max}] \\ t \leq t_{max} \end{cases} \tag{1}$$

If time or space is out of range, it is regarded as the failure of a guidance task. At time t_f with $t_f \leq t_{max}$, the conditions for successful guidance are:

$$\begin{cases} \sqrt{(x_{t_f}^a - x_{t_f}^d)^2 + (y_{t_f}^a - y_{t_f}^d)^2} \leq d_e \\ |z_{t_f}^a - z_{t_f}^d| \leq z_e \\ |\psi_{t_f}^a - \psi_{t_f}^d| \leq \psi_e \end{cases} \tag{2}$$

where d_e, z_e and ψ_e are the allowable ranges for distance, altitude difference and heading-angle difference between the aircraft and the destination, respectively.

2.2. DRL Training Framework

The generated aircraft guidance instructions can be used for automatic auxiliary decision-making or path planning. No matter what type of application, it belongs to the sequential decision-making problem, which is suitable for using DRL to solve. DRL combines reinforcement learning (RL) [19] and deep learning (DL) [20] and allows agents to learn directly from the environment through trial and error without a perfect knowledge of the environment in advance. Unlike supervised learning agents, DRL agents improve their abilities through continuous interaction with the environment. At each training time t, the agent receives a state S_t in a state space S and generates an action A_t from an action space A following a policy $\pi : S \times A \to \mathbb{R}$. Then, the agent receives a scalar reward R_t and transitions to the next state S_{t+1} according to the environment dynamics. The goal of an agent is to learn a policy π which defines the action that should be used in a state to maximize the future cumulative reward.

Through training in the environment, an agent has the ability of aircraft guidance and then can be validated in special aircraft guidance simulation software or realistic situations. The DRL training framework for aircraft guidance is shown in Figure 2.

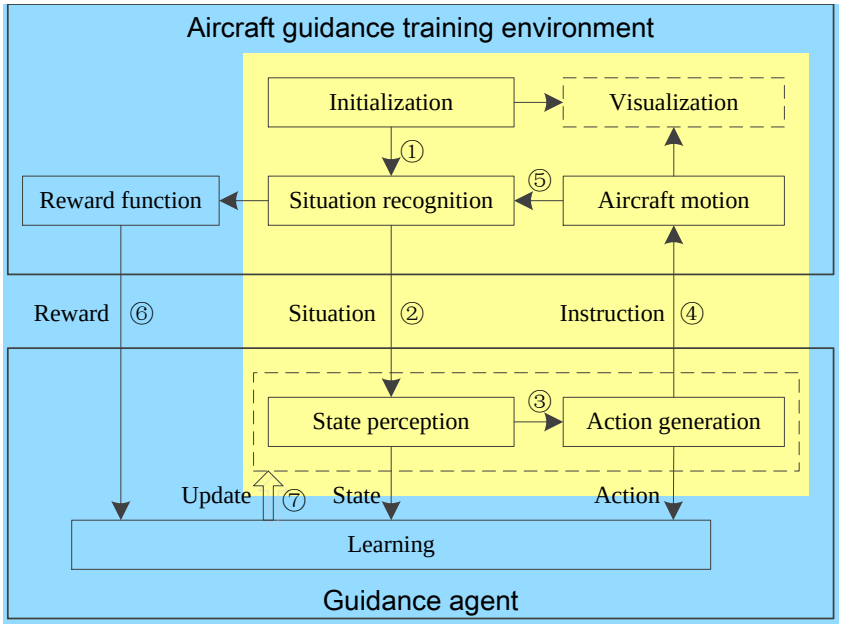

Figure 2. Aircraft guidance agent training process.

Aircraft guidance using DRL is an episodic task, and a training episode is composed of training steps. In each training step, first, the information recognized by radar or ADS-B is sent to the guidance agent as a tuple of state, see Steps 1 and 2 in Figure 2. Then, an action is generated by the agent through its neural networks and sent to the environment, as shown in Steps 3 and 4. Next, the next state and a scalar reward are calculated in the environment and then sent to the agent, as shown in Steps 5 and 6. Since agent training is an iterative process, all the steps except Step 1 are executed repeatedly in one episode. Last, for the update of neural networks, the tuple of the current state, action, reward and the next state in each step is used (see Step 7). Through episodic iterative training, the agent gradually has the ability of aircraft guidance.

3. DRL Model for Aircraft Guidance

Establishing the DRL model for aircraft guidance entails establishing a Markov decision process (MDP) model including state space, action space and reward function.

3.1. State Space and Action Space

The equations of motion for the aircraft are given by [21]:

$$\begin{cases} \dot{x} = v cos\theta cos\psi \\ \dot{y} = v cos\theta sin\psi \\ \dot{z} = v sin\theta \\ \dot{v} = g(n_x - sin\theta) \\ \dot{\theta} = \frac{g}{v}(n_z cos\phi - cos\theta) \\ \dot{\psi} = -\frac{g}{vcos\theta} n_z sin\phi \end{cases} \tag{3}$$

where n_x is the axial load factor and n_z is the normal load factor. In this article, we assume that the aircraft has real-time perception of its and its destination's correct position information. The state space of a guidance agent can be described with a vector:

$$S_t = (x_t^a, y_t^a, z_t^a, v_t^a, \theta_t^a, \psi_t^a, \phi_t^a, x_t^d, y_t^d, z_t^d, \psi_t^d, A_{t-1}) \tag{4}$$

where A_{t-1} is the last action, which is added to reflect the continuity of instructions.

Thrust, angle of attack and roll angle are used by an aircraft as control variables to change its flight path and are difficult for pilots to directly control. As an alternative, load factors can be utilized to control an aircraft. The maneuvering of an aircraft can be seen as a combination of some basic actions that can be used as the action space of the aircraft guidance agent to reduce the pressure on pilots. To build the action space, the continuous control variables are replaced with seven discrete control alternatives [22], including steady flight, max load factor left turn, max load factor right turn, max long acceleration, max long deceleration, max load factor pull up, and max load factor push over. Using this action space, only one of the seven actions needs to be selected for each step. Complex maneuvering can be generated through the combination of basic actions.

3.2. Reward Function

The evaluation of agent performance should be reflected in the design of the reward function, including guidance success rate, flight trajectory quality, instruction generation time and agent training efficiency. In aircraft-guidance agent training, the reward function of each step is:

$$R(S_t, A_t, S_{t+1}, A_{t+1}) = T(S_t) + F(S_t, A_t, S_{t+1}, A_{t+1}) \tag{5}$$

where $T(S_t)$ is the termination reward and $F(S_t, A_t, S_{t+1}, A_{t+1})$ is a potential-based shaping function [23].

The termination reward is obtained at the end of each training episode, which is defined as:

$$T(S_t) = \begin{cases} c_1, \text{ if successful arrived} \\ c_2, \text{ if out of the sector} \\ c_3, \text{ if no control times left} \\ 0, \text{ else} \end{cases} \tag{6}$$

The agent gets a positive reward when successfully guided, $c_1 > 0$. The agent gets a negative reward when the guidance task fails. To encourage the agent to guide an aircraft to explore the airspace, the penalty for aircraft flying out of airspace should be greater than the penalty for reaching the maximum time step, that is, $c_2 < c_3 < 0$. In the non-terminating step, the reward obtained by the agent is 0.

The shaping function has the form:

$$F(S_t, A_t, S_{t+1}, A_{t+1}) = \gamma\Phi(S_{t+1}, A_{t+1}) - \Phi(S_t, A_t) \tag{7}$$

where $\Phi(S_t, A_t)$ is a real-valued function over states and actions. The greater its value, the more valuable the state and action of the aircraft at that time. It is defined as:

$$\Phi(S_t, A_t) = C(S_t, A_t) + P(S_t) \tag{8}$$

where $C(S_t, A_t)$ is the continuous action reward function and $P(S_t)$ is the position reward function.

The continuous action reward function is defined as:

$$C(S_t, A_t) = \begin{cases} c_4, \text{ if current action is different from the previous one} \\ c_5, \text{ else} \end{cases} \tag{9}$$

To improve the smoothness of the trajectory, if the action is different from the previous one, a penalty is given. At each step, a smaller penalty is given to try to reduce the total time spent; that is, $c_4 < c_5 < 0$.

In different aircraft guidance tasks, evaluation of the relative position between the aircraft and the destination is different. In this paper, a general position reward function

is used without considering the requirements of specific tasks for a relative position; it is given by:

$$P(S_t) = c_6 D(S_t) + c_7 H(S_t) + c_8 A(S_t) \tag{10}$$

where $D(S_t)$ is the horizontal distance reward function, $H(S_t)$ is the direction reward function and $A(S_t)$ is the altitude reward function. They are defined as:

$$\begin{cases} D(S_t) = -\sqrt{(x_t^a - x_t^d)^2 - (y_t^a - y_t^d)^2} \\ H(S_t) = cos(\psi_t^a - \psi_t^d) \\ A(S_t) = -|z_t^a - z_t^d| \end{cases} \tag{11}$$

The terms c_6, c_7 and c_8 are weights of $D(S_t)$, $H(S_t)$ and $A(S_t)$, respectively. They satisfy:

$$\begin{cases} c_6 D(S_t)/c_7 H(S_t) = O(1) \\ c_6 D(S_t)/c_8 A(S_t) = O(1) \end{cases} \tag{12}$$

This makes the three functions of the same order so that the relative distance, relative direction and relative altitude have the same level of effect on training the agent.

If the sum of the shaping rewards of each step is greater than the positive termination reward value, the agent will guide the aircraft to seek a better trajectory and ignore the successful guidance. On the other hand, if the sum of the shaping rewards is less than the negative termination reward value, the agent will guide the aircraft to fail as soon as possible and get a relatively small penalty. Therefore, the design of each weight should satisfy:

$$c_3 < \sum_{t=1}^{t_{max}} \forall F(c_4, c_5, c_6, c_7, c_8) < c_1 \tag{13}$$

The original MDP of aircraft guidance is $M = (S, A, P, \gamma, R)$, where $R = T(S_t)$. Using reward shaping, the MDP is transformed into $M' = (S, A, P, \gamma, R')$, where $R' = T(S_t) + F(S_t, A_t, S_{t+1}, A_{t+1})$. It needs to be proven that each optimal policy in M' will also be an optimal policy in M.

The action-state value function of M is:

$$q_M(s, a) = E_\pi \left[\sum_{k=0}^{t_{f-1}} \gamma^k T_k | S_0 = s, A_0 = a \right] \tag{14}$$

The action-state value function of M' is:

$$\begin{aligned} q_{M'}(s, a) &= E_{\pi'} \left[\sum_{k=0}^{t_{f-1}} \gamma^k (T_k + F(S_k, A_k, S_{k+1}, A_{k+1})) | S_0 = s, A_0 = a \right] \\ &= E_{\pi'} \left[\sum_{k=0}^{t_{f-1}} \gamma^k (T_k + \gamma \Phi(S_{k+1}, A_{k+1}) - \Phi(S_k, A_k)) \right] \\ &= E_{\pi'} \left[\sum_{k=0}^{t_{f-1}} \gamma^k T_k \right] + E_{\pi'} \left[\sum_{k=1}^{t_f} \gamma^k \Phi(S_k, A_k) \right] - E_{\pi'} \left[\sum_{k=0}^{t_{f-1}} \gamma^k \Phi(S_k, A_k) \right] \\ &= E_{\pi'} \left[\sum_{k=0}^{t_{f-1}} \gamma^k T_k \right] + \gamma^{t_f} \Phi(S_{t_f}, A_{t_f}) - \Phi(S_0, A_0) \end{aligned} \tag{15}$$

In the initial step t_0, both the initial state S_0 and the default action A_0 are fixed values, so $\Phi(S_0, A_0)$ is a constant. At the final step t_f, $\Phi(S_{t_f}, A_{t_f})$ has no effect on the training result and A_{t_f} does not need to be generated. Thus, we get:

$$\gamma^{t_f}\Phi(S_{t_f}, A_{t_f}) - \Phi(S_0, A_0) = C \tag{16}$$

By substituting Equations (14) and (16) into Equation (15), we can get:

$$q_{M'}(s, a) = q_M(s, a) + C \tag{17}$$

For the optimal policy of M‘:

$$\begin{aligned} \pi^*_{M'}(s) &= \arg\max_{a\in A} q_{M'}(s, a) \\ &= \arg\max_{a\in A} q_M(s, a) + C \\ &= \arg\max_{a\in A} q_M(s, a) \\ &= \pi^*_M(s) \end{aligned} \tag{18}$$

Thus, the optimal policy in M‘ is also an optimal policy in M.

4. Policy-Reuse Algorithm Based on Destination Position Prediction

For different guidance tasks, the destinations have different movement patterns and parameters. It still takes a lot of time if the guidance agent for a new task is trained from scratch. For two different aircraft guidance tasks, their MDPs are $M = (S, A, R, \gamma, R)$ and $M' = (S', A', R', \gamma', R')$. The state space, action space and reward function of the two MDPs are the same, but their transition functions are different due to different destination movement patterns. The policy-reuse algorithm, whether used to solve the problem of different state/action spaces or different reward functions, assumes that the transition function is unchanged, which makes it difficult to reuse policies in aircraft guidance tasks.

In the scenario studied in this paper, the destination moves according to its own dynamic model, which makes the position of the destination unaffected by the instructions generated by the agent. At any time t, for the two different actions A_t and A_t‘ performed by the aircraft in state S_t, the destination positions in S_{t+1} and S_{t+1}‘ in tuple $[S_t, A_t, R_t, S_{t+1}]$ and $[S_t', A_t', R_t', S_{t+1}']$ are the same:

$$(x^d_{t+1}, y^d_{t+1}, z^d_{t+1}, \psi^d_{t+1}) = (x^d_{t+1}', y^d_{t+1}', z^d_{t+1}', \psi^d_{t+1}') \tag{19}$$

The sketch of destination position prediction is shown in Figure 3. An episode from $t = 0$ to $t = t_f$ is run in the moving destination scenario, and an action sequence is generated. Suppose there is an action sequence of the aircraft that is better than the current one and can make the aircraft arrive at the destination earlier at t_n, $0 < t_n < t_f$. The destination position at the possible termination time step t_n can be predicted by running an episode in advance. For each step n between 0 and t_n, if the destination position of t_n instead of n is taken as the target, then the problem is equivalent to guiding an aircraft to a fixed-position destination. That is, the aircraft is guided to the predicted position of the destination at t_n.

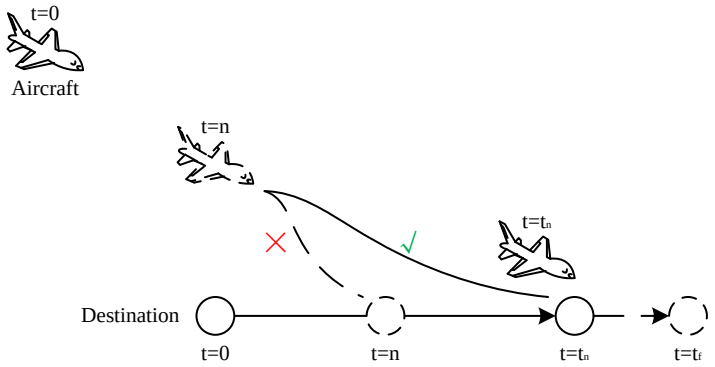

Figure 3. Destination position prediction.

A destination position prediction-based policy-reuse algorithm is proposed in this section. In a new guidance task, an existing policy/agent B trained on a fixed-position destination scenario is used to reduce the exploration space of a new agent Z. The destination positions in S_n and S_{n+1} in the state input of agent B are fixed and are the same as the destination position in S_{t_f}. For agent Z, the destination position in its state input S_n is constantly changing and cannot be directly used by agent B. Thus, if the destination position in S_n is the same as that in the possible termination step t_n:

$$(x_n^d, y_n^d, z_n^d, \psi_n^d) = (x_{t_n}^d, y_{t_n}^d, z_{t_n}^d, \psi_{t_n}^d) \tag{20}$$

Then, the problem of aircraft guidance with a moving destination is transformed into the problem of aircraft guidance with a fixed-position destination. The prediction is realized by running an episode in the moving destination scenario in advance to obtain the destination position at each time in the future. In the training process, by predicting the future destination position, the current destination position can be replaced by the destination position at the possible termination time. In this way, agent B can be used to generate actions that improve the training efficiency on the new task.

The policy-reuse algorithm based on destination position prediction is briefly illustrated in Figure 4. The objective of an aircraft guidance task with a moving destination is to train a guidance agent Z. The agent trained in the fixed-position destination scenario is taken as the baseline agent B. At each training step n, the destination position in S_n in the tuple is replaced by the destination position $(x_{t_n}^d, y_{t_n}^d, z_{t_n}^d, \psi_{t_n}^d)$ in the possible termination state S_{t_n} of the episode. The term $S_{n\_t_n}$ is used to denote the state obtained by replacing the destination position at step n with the predicted destination position at step t_n. Through this operation, the new guidance problem is transformed into a fixed-position destination guidance problem. A new action A_{n_r} can be generated by using the baseline agent B, and then a tuple $[S_{n\_t_n}, A_{n\_r}, R_{n\_r}, S_{n+1\_t_n}]$ is obtained. Since the destination position in the tuple is predicted rather than actually generated, it cannot be used to train an agent. The actual generated destination positions $(x_n^d, y_n^d, z_n^d, \psi_n^d)$ and $(x_{n+1}^d, y_{n+1}^d, z_{n+1}^d, \psi_{n+1}^d)$ in S_n and S_{n+1} are used to replace the destination positions in $S_{n\_t_n}$ and $S_{n+1\_t_n}$ and are saved as a new tuple $[S_{n\_r}, A_{n\_r}, R_{n\_r}, S_{n+1\_r}]$. The training efficiency is significantly improved by using this new tuple to train agent Z.

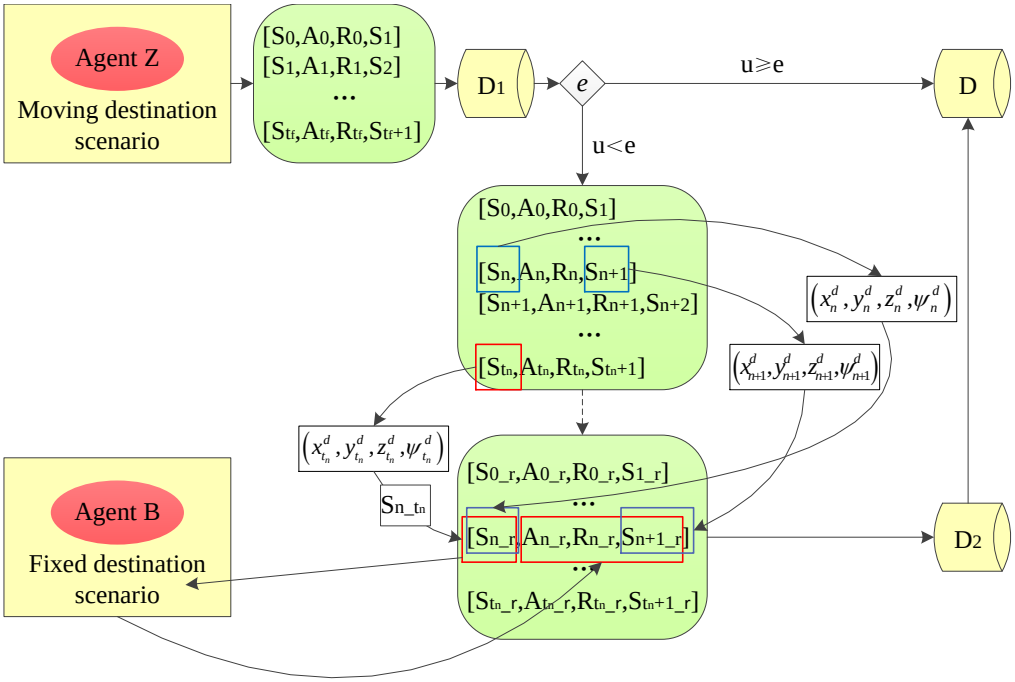

Figure 4. Illustration of policy-reuse algorithm based on destination position prediction.

The policy-reuse algorithm based on destination position prediction is shown in Algorithm 1. In the new guidance task, the input interface of agent Z includes aircraft position, destination position and the last action, and the output interface outputs one of the seven discrete actions, which are the same as those of agent B. At each time step t, the data generated in an episode using agent Z is stored in tuples. A random number u in [0, 1] is generated and compared with the current agent selection factor e. If u is greater than or equal to e, tuples are not updated; otherwise, agent B is used to update tuples.

As shown in Figure 3, the aircraft has the possibility of reaching the destination at any step from 1 to t_f, so a loop from 1 to t_f is operated to search for a possible termination step t_n. For each step t_n in the loop, the destination position in each tuple from t_0 to t_n is replaced with the destination position in S_{t_n} in step t_n, and it is saved as $S_{n\_t_n}$. The purpose of this process is to change the tuple into the tuple in the fixed-position destination scenario by replacing the destination position in the state so as to meet the use conditions of baseline agent B. With state $S_{n\_t_n}$ as input, a new tuple $[S_{n\_t_n}, A_{n\_r}, R_{n\_r}, S_{n+1\_t_n}]$ is generated using baseline agent B. In this tuple, A_{n_r} is the action performed by the aircraft, and R_{n_r} is the evaluation of this action, which can be used to train agent Z. However, the destination positions in $S_{n\_t_n}$ and $S_{n+1\_t_n}$ are not the positions in the actual scenario, which need further transformation. Since the destination positions in S_n and S_{n+1} are generated by destination movement, a new tuple $[S_{n\_r}, A_{n\_r}, R_{n\_r}, S_{n+1\_r}]$ is saved by replacing the destination positions in $S_{n\_t_n}$ and $S_{n+1\_t_n}$ with destination positions in S_n and S_{n+1}. In this tuple, the action is generated by the baseline agent B, and the destination position in the state is the actual trajectory of the destination in the moving-destination scenario. At any time in the loop, the update of tuples finishes when the aircraft successfully arrives at the destination. Finally, the DRL algorithm is used to train agent Z with the saved tuples. The agent selection factor e needs to be updated. When it is less than or equal to 0, the algorithm stops updating and makes it equal to 0.

Algorithm 1 Destination Position Prediction-Based Policy-Reuse Algorithm

Initialize agent Z to be trained
Select and load baseline agent B
Initialize agent selection factor $e = 1$
Initialize three memories D_1, D_2 and D
for each episode **do**
  Use agent Z to run a complete episode with t_f steps
  for every step t in $[0, t_f]$ **do**
    Save tuple $[S_t, A_t, R_t, S_{t+1}]$ into D_1
  end for
  Generate a random number u in [0,1]
  if $u \geq e$ **then**
    Transfer the data in D_1 to D and clear D_1
  else
    for every step t_n in $[1, t_f]$ **do**
      Clear D_2
      Transfer $[S_0, A_0, R_0, S_1]$ in D_1 to D_2 and save it as $[S_{0\_r}, A_{0\_r}, R_{0\_r}, S_{1\_r}]$
      for every step n in $[0, t_n]$ **do**
        Replace $(x_n^d, y_n^d, z_n^d, \psi_n^d)$ in S_{n_r} with $(x_{t_n}^d, y_{t_n}^d, z_{t_n}^d, \psi_{t_n}^d)$ in S_{t_n} and save it as $S_{n\_t_n}$
        Select action A_{n_r} in state $S_{n\_t_n}$ using the baseline agent B
        Execute action A_{n_r}, receive reward R_{n_r}, and transfer into the next state $S_{n+1\_t_n}$
        Replace $(x_{n+1\_t_n}^d, y_{n+1\_t_n}^d, z_{n+1\_t_n}^d, \psi_{n+1\_t_n}^d)$ in $S_{n+1\_t_n}$ with $(x_{n+1}^d, y_{n+1}^d, z_{n+1}^d, \psi_{n+1}^d)$ in S_{n+1} and save it as S_{n+1_r}
        Replace $(x_{n\_t_n}^d, y_{n\_t_n}^d, z_{n\_t_n}^d, \psi_{n\_t_n}^d)$ in $S_{n\_t_n}$ with $(x_n^d, y_n^d, z_n^d, \psi_n^d)$ in S_n and save it as S_{n_r}
        Overwrite tuple $[S_{n\_r}, A_{n\_r}, R_{n\_r}, S_{n+1\_r}]$ into D_2
        if guide succeeded **then**
          break
        end if
      end for
    end for
    Transfer the data in D_2 to D and clear D_2
    Update e by $e = e - k$
  end if
  Use the data in D to train agent Z
end for

5. Simulation and Results Analysis

5.1. Simulation Setup

In this section, aircraft guidance simulation is carried out for fixed-position destination and different movement pattern destination scenarios to verify the reward shaping and policy-reuse methods proposed in this paper. The aircraft guidance simulation parameters are shown in Table 1. The proposed algorithm does not limit the representation of position, and relative position or absolute position can be used. In this simulation, for the scenario with a fixed-position destination, the position of the destination is $(0, 0, 0, 0)$. Therefore, in the moving destination scenario, for the predicted possible termination step t_n, its positional information $(x_{t_n}^d, y_{t_n}^d, z_{t_n}^d, \psi_{t_n}^d)$ is transformed into $(0, 0, 0, 0)$. For the guided aircraft, at each time step n, its positional information $(x_n^a, y_n^a, z_n^a, \psi_n^a)$ is transformed into $(x_n^a - x_{t_n}^d, y_n^a - y_{t_n}^d, z_n^a - z_{t_n}^d, \psi_n^a - \psi_{t_n}^d)$.

Table 1. Aircraft guidance simulation parameters.

Parameter	Value
Simulation time-step	20 ms
Airspace	x $\in$ [−10 km, 10 km], y $\in$ [−10 km, 10 km], z $\in$ [0, 5 km]
Aircraft performance limitations	$\lvert\phi\rvert \leq 90^\circ$, v $\in$ [100 m/s, 250 m/s]
Aircraft initial position	(x_0^a, y_0^a, z_0^a) random initialization, $\theta_0^a = 0$, $\psi_0^a = 0$, $\phi_0^a = 0$
Destination initial position	$(x_0^d, y_0^d, z_0^d, \psi_0^d) = (0, 0, 0, 0)$

The PPO algorithm is used to train the agent in the simulation. The DRL parameters are shown in Table 2.

Table 2. Deep reinforcement learning parameters.

Parameter	Value
Training time-step	3 s
Discount factor	$\gamma = 0.99$
Learning rate	0.0005
Minibatch size	512
Termination reward function	$c_1 = 20$, $c_2 = -20$, $c_3 = -10$
Continuous action reward function	$c_4 = -0.1$, $c_5 = -0.01$
Position reward function	$c_6 = 0.0001$, $c_7 = 0.5$, $c_8 = 0.00001$

5.2. Simulation of Aircraft Guidance in Fixed-Position Destination Scenario

The aircraft guidance simulation of a fixed-position destination is carried out to verify the effect of reward functions and to train a baseline agent. The training success rates using four kinds of reward functions are shown in Figure 5.

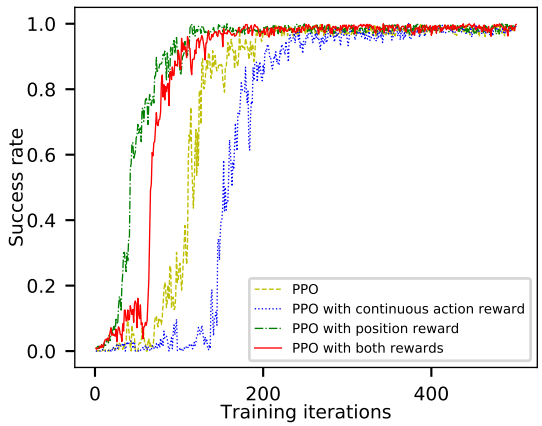

Figure 5. Training success rate using PPO with different reward functions.

The system converged after 200 training iterations using PPO only. The training speed of PPO with continuous-action reward function is the slowest, and the system converged after nearly 300 iterations. PPO with position reward function has the fastest training speed, and the system converged after about 100 training iterations. The system converged after 150 iterations using PPO with both reward functions. We found that during training, using different reward functions only makes a difference in training efficiency, and their success rates are stable at high levels after convergence.

Each well-trained agent is tested for 1000 simulations. The success rate, average number of control times and average computational time to generate an instruction are given in Table 3.

Table 3. Performance of different agents in fixed-position destination scenarios.

Algorithm	Success Rate	Average Number of Control Times	Average Computational Time
PPO	98.6%	11.31	2.98 ms
PPO with continuous-action reward function	99.4%	7.26	2.89 ms
PPO with position reward function	99.1%	11.52	2.92 ms
PPO with both reward functions	99.5%	7.22	2.91 ms

The success rate of using the standard PPO algorithm is 98.6%, and the average number of control times is 11.31. The number of control times is an important index for evaluating the performance of an agent in aircraft guidance. The lower the number of control times, the less pressure the pilot and air traffic controller will have, and the smoother the flight trajectory will be. Although the training efficiency of PPO with continuous-action reward function is decreased, the average number of control times is reduced from more than 11 to less than 8. The training speed of using position reward function is significantly improved, but the average number of control times does not improved; it is still more than 11. The training efficiency and flight trajectory quality are significantly improved by using PPO with both reward functions. The agent trained by DRL takes less than 3 ms to generate an instruction, which has high computational efficiency.

Typical trajectories are shown in Figure 6. Using the standard PPO algorithm, the aircraft can reach the destination under the guidance of the agent in most scenarios. However, the agent may output unnecessary actions, resulting in the flight trajectory not being smooth enough, as shown in Figure 6a. Using PPO with continuous-action reward function, although the training efficiency is decreased, the unnecessary instructions are fewer, and the flight trajectory is smoother, as shown in Figure 6b. Using PPO with position reward function, the flight trajectory is still not smooth, as shown in Figure 6c. Using PPO with both reward functions improves both training efficiency and flight trajectory quality, as shown in Figure 6d.

In fixed-position destination scenarios, from the perspective of DRL, there are multiple optimal policies for aircraft guidance if reward shaping is not adopted. However, from the perspective of aircraft guidance, although using different optimal policies will lead to success, their guidance processes are different. Using reward shaping, an optimal guidance policy that is more suitable for pilots and air traffic controllers can be obtained by further optimization within the scope of DRL optimal policies.

5.3. Simulation of Aircraft Guidance in Moving-Destination Scenarios

Scenarios with different moving destinations are set up for aircraft guidance agent training (1) from scratch, (2) with the pre-trained algorithm that reuses the baseline agent directly without any operations, and (3) with the proposed policy-reuse algorithm. In the uniform-motion destination scenario, the speed of the destination is set to 10 m/s, 20 m/s, 50 m/s and 100 m/s. In the curved-motion destination scenario, the speed is set to 20 m/s, and the turning radius is set to 500 m, 1000 m, 2000 m and 5000 m. The proposed policy-reuse algorithm needs to replace the stored data many times, which takes extra time. Therefore, the number of iterations for successful training cannot be used as an evaluation index. The training process chart in this section takes the training time as the abscissa and the success rate as the ordinate to compare the training efficiency of different algorithms.

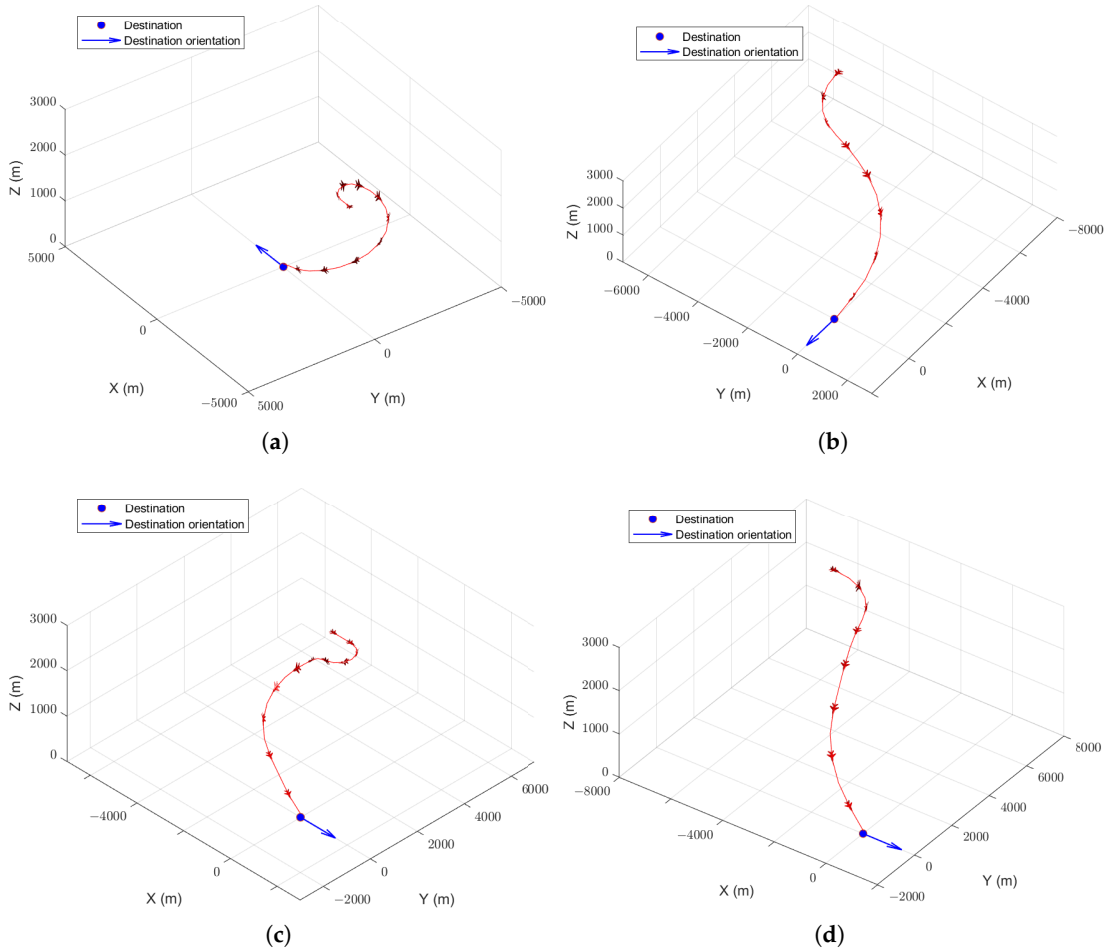

Figure 6. Trajectories in fixed-position destination scenario: (**a**) PPO, (**b**) PPO with continuous-action reward function, (**c**) PPO with position reward function and (**d**) PPO with both reward functions.

The success rates of the training processes in the uniform-motion destination scenario are shown in Figure 7. Using the pre-trained algorithm, a new agent can be trained quickly when the speed of the destination is slow. The training efficiency is reduced when the destination is at high speed, but it is still better than the training method from scratch. This is because the baseline agent will guide the aircraft to maneuver to the current position of the destination. Although having a dynamic destination impacts the training process, the agent will still guide the aircraft to explore the area near the destination, which is better than the random exploration of the training method from scratch. The training efficiency of the policy-reuse algorithm based on the destination position prediction is lower than that of the pre-trained algorithm when the speed of the destination is slow. This is because it needs to replace the training data many times and takes more time. However, the algorithm has good stability, and its performance does not decrease significantly with increases in the destination speed. Its efficiency is better than that of the pre-trained algorithm when the destination speed is high.

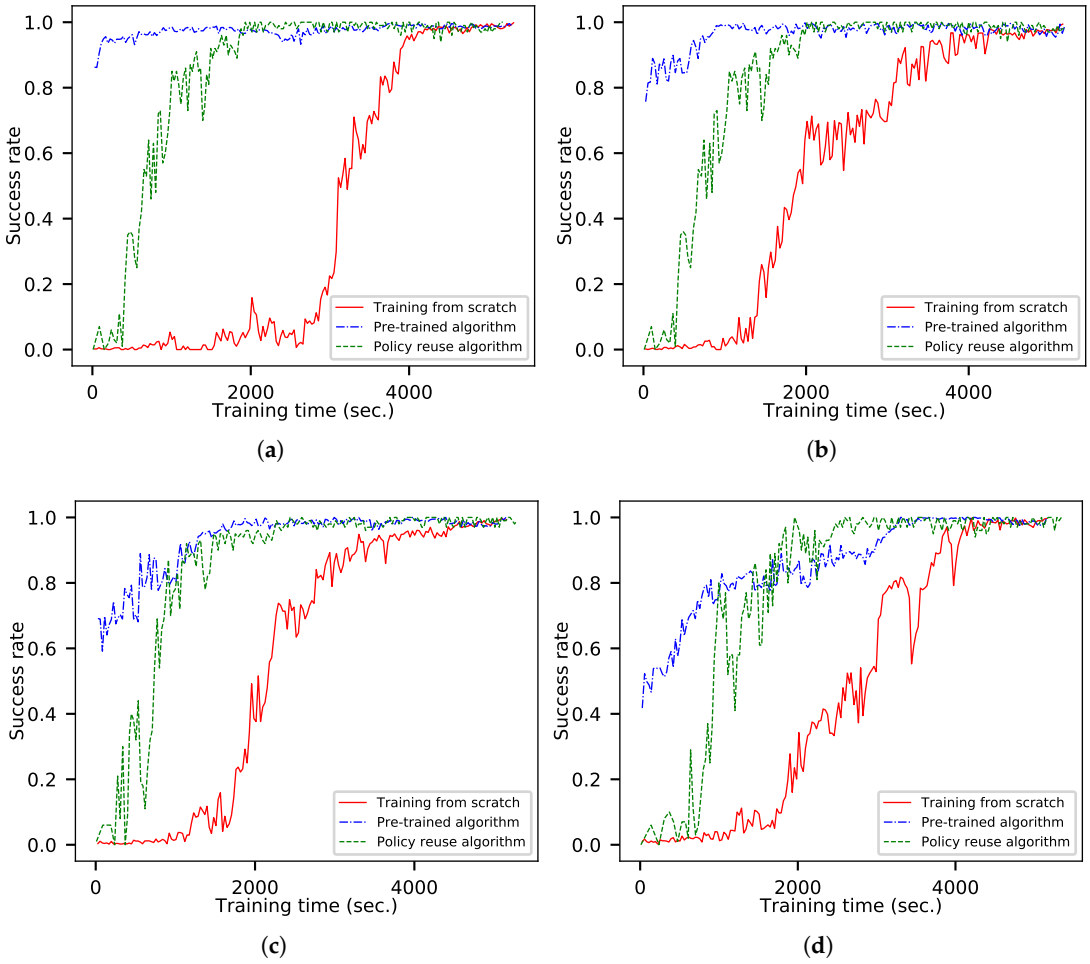

Figure 7. Policy reuse for agent training in uniform-motion destination scenario: destination speed of (**a**) 10 m/s, (**b**) 20 m/s, (**c**) 50 m/s and (**d**) 100 m/s.

Each well-trained agent is tested for 1000 simulations. The success rate is given in Table 4. The typical trajectories are shown in Figure 8. It can been seen that there is almost no difference in the performance of agents among various training algorithms; only training efficiency is affected. The faster the speed of the destination, the lower the similarity with the fixed-position destination guidance task. The pre-trained algorithm is sensitive to task similarity, and the time required for convergence obviously increases with the decrease of similarity. The policy-reuse algorithm based on destination position prediction is less affected by task similarity, and the convergence time does not increase significantly with the decrease of the similarity. This makes the proposed algorithm applicable to a wider range, and the baseline agent can be used for a wider range of destination speeds.

Table 4. Success rate in uniform-motion destination scenario.

Algorithm \ Speed	10 m/s	20 m/s	50 m/s	100 m/s
Training from scratch	99.6%	99.0%	99.4%	99.0%
Pre-trained algorithm	99.2%	99.4%	99.2%	99.1%
Policy-reuse algorithm	99.4%	99.2%	99.4%	99.1%

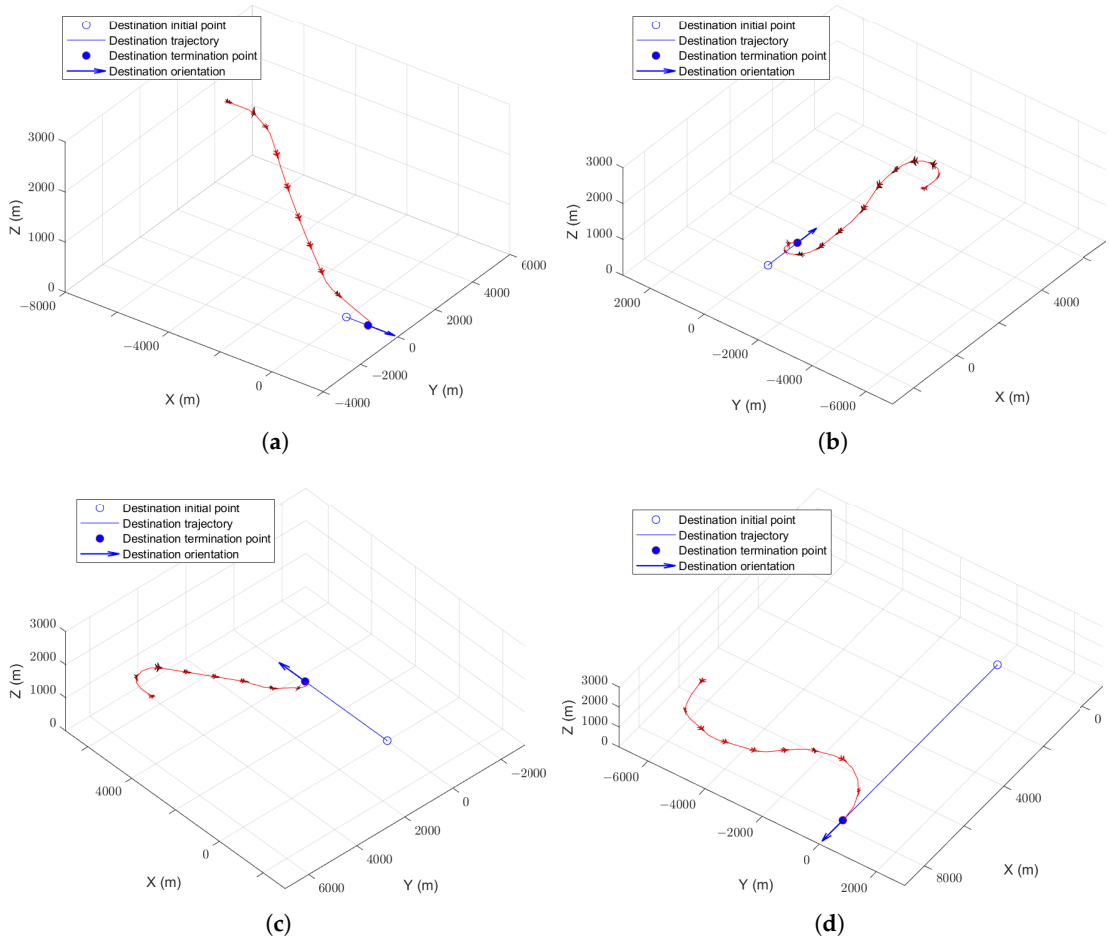

Figure 8. Trajectories in uniform-motion destination scenario: destination speed of (**a**) 10 m/s, (**b**) 20 m/s, (**c**) 50 m/s and (**d**) 100 m/s.

The aircraft guidance simulation in the curved-motion destination scenario is carried out to further verify the applicability of the proposed algorithm. The success rates of the training processes are shown in Figure 9. Under the premise that the destination speed is constant, a smaller turning radius, larger turning angle of the destination in unit time, and lower similarity with the baseline task resulted in more time required for training. Using the pre-trained algorithm, a new agent can be trained efficiently when the turning radius of the destination is large. When the turning radius is small, its performance is poor, but it is more efficient than training from scratch. The performance of the policy-reuse algorithm does not decrease significantly with the decrease of the turning radius of the destination, and it has stability with the change of similarity.

Each well-trained agent is tested for 1000 simulations. The success rate is given in Table 5. The typical trajectories are shown in Figure 10. It can been seen that the algorithm proposed in this paper can also be used if the destination moves in a curve. The smaller the turning radius of the destination, the better the performance of the algorithm compared to that of the pre-trained algorithm.

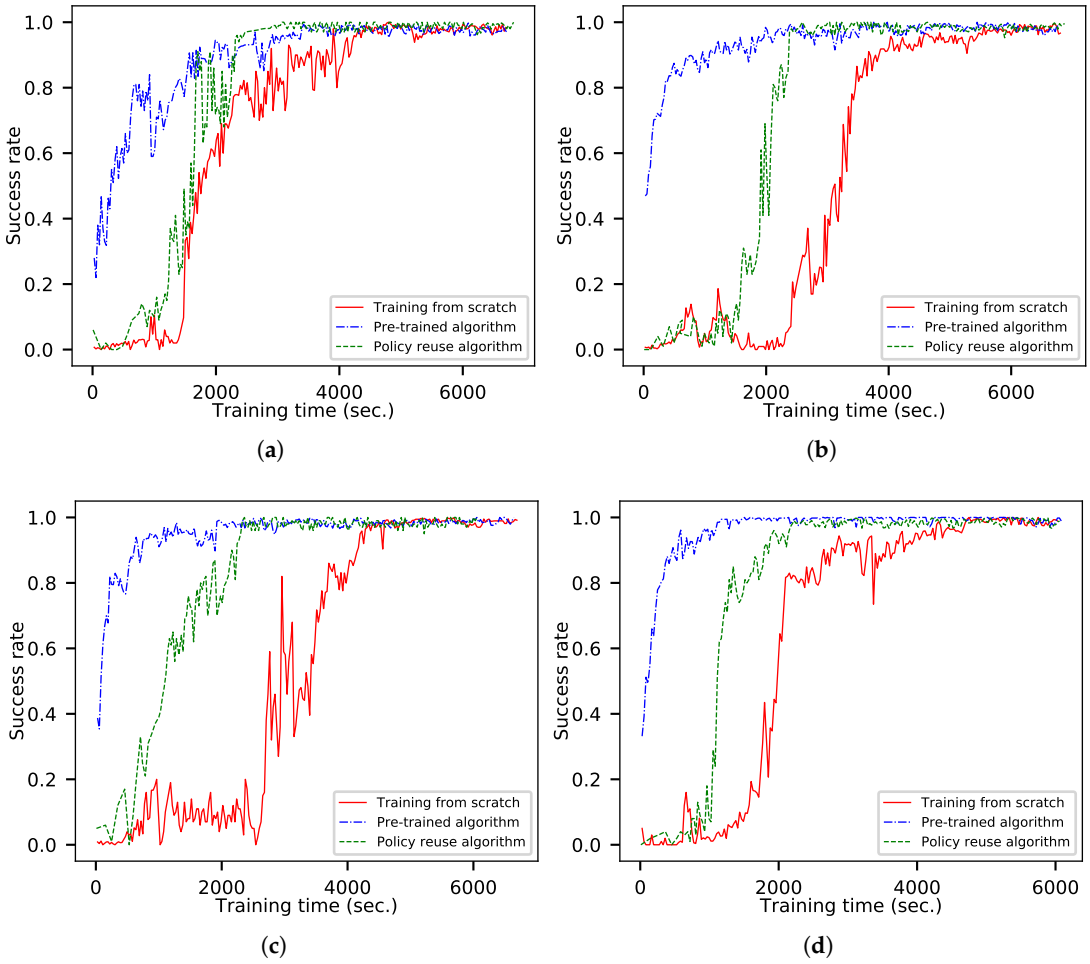

Figure 9. Policy reuse for agent training in curved-motion destination scenario: destination turning radius of (**a**) 500 m, (**b**) 1000 m, (**c**) 2000 m and (**d**) 5000 m.

Table 5. Success rate in curved-motion destination scenario.

Algorithm \ Turning Radius	500 m	1000 m	2000 m	5000 m
Training from scratch	99.1%	99.4%	99.4%	99.6%
Pre-trained algorithm	99.0%	99.4%	99.5%	99.7%
Policy-reuse algorithm	99.2%	99.5%	99.3%	99.7%

The simulation results of aircraft guidance in scenarios of different movement pattern destinations show that using the prior knowledge of the old policy/agent to guide the training of a new agent can effectively reduce the exploration space in the early stage of agent training, partially solve the problem of poor generalization of DRL, and improve the efficiency of agent training. Compared with the pre-trained algorithm using the old policy/agent directly, the proposed policy-reuse algorithm based on destination position prediction is not sensitive to the similarity between the old and new tasks, which expands the scope of policy reuse. The pre-trained algorithm has high efficiency when the similarity between the old and new tasks is high. On the contrary, the policy-reuse algorithm proposed in this paper is recommended when the similarity between the two tasks is low.

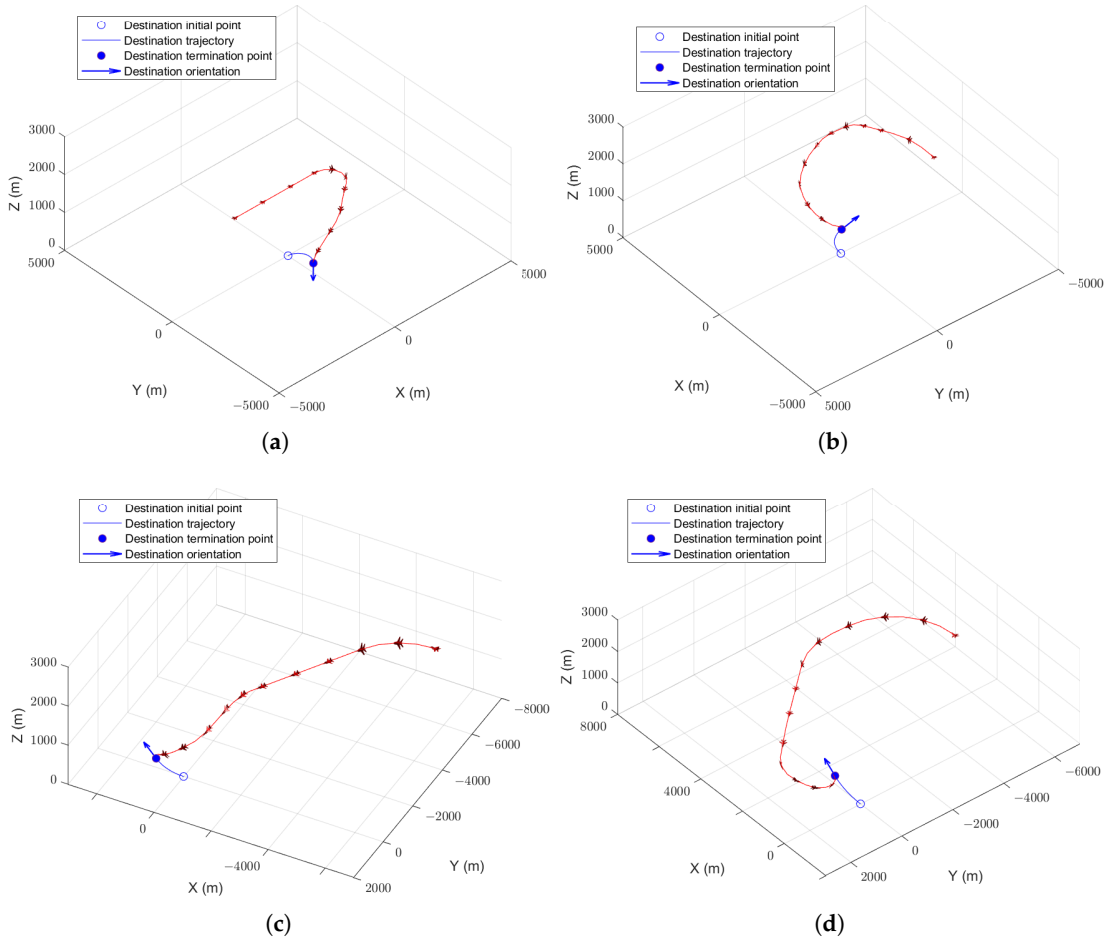

Figure 10. Trajectories in curved-motion destination scenario: destination turning radius of (**a**) 500 m, (**b**) 1000 m, (**c**) 2000 m and (**d**) 5000 m.

6. Conclusions

A policy-reuse approach based on destination position prediction for aircraft guidance using DRL is proposed in this article. A new agent can be trained efficiently using the existing policy/agent by substituting the prediction of the future position of the destination into the current state, which alleviates the slow agent training in the new guidance task caused by the poor generalization ability of DRL. For new aircraft guidance tasks, prior knowledge can be used to train new agents, which makes full use of previous research results. This research direction and achievement can also enlighten the research using DRL in other fields.

Aircraft guidance simulations using straight-moving destinations with different speeds and curved-motion destinations with different turning radii were carried out, and research on irregularly moving destinations should be carried out in the future. In addition, the structure of the agent, mainly the state space and the action space, need to be further optimized to make it more consistent with that of the actual guidance task.

Author Contributions: Conceptualization, H.L. and Z.W.; methodology, Z.W.; software, Y.A.; validation, Z.W., H.L. and Q.Z.; formal analysis, Q.Z.; resources, Y.A.; writing—original draft preparation, Z.W.; writing—review and editing, Y.A. and S.Z.; visualization, Q.Z.; supervision, Y.A.; project administration, Z.W.; funding acquisition, Y.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Natural Science Foundation of China (62203451) and Guangxi Key Laboratory of International jointly for China-ASEAN Comprehensive Transportation (21-220-21-01) and Fundamental Research Funds for the Central Universities (J2022-053).

Data Availability Statement: Not applicable.

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

References

1. Staffetti, E.; Li, X.; Matsuno, Y; Soler, M. Optimal control techniques in aircraft guidance and control. *Int. J. Aerosp. Eng.* **2019**, *2019*, 3026083. [CrossRef]
2. Lee, S.; Kim, M. Analysis of rendezvous guidance laws for autonomous aerial refueling for non-maneuvering and identical speed targets. *Aerosp. Sci. Technol.* **2022**, *121*, 107359. [CrossRef]
3. Eklund, J.M.; Sprinkle, J.; Sastry, S. Switched and symmetric pursuit/evasion games using online model predictive control with application to autonomous aircraft. *IEEE Trans. Control Syst. Technol.* **2011**, *20*, 604–620 [CrossRef]
4. You, D.I.; Shim, D.H. Design of an aerial combat guidance law using virtual pursuit point concept. *Proc. Inst. Mech. Eng. G J. Aerosp. Eng.* **2015**, *229*, 792–813 [CrossRef]
5. Arulkumaran, K.; Deisenroth, M.P.; Brundage, M.; Bharath, A.A. Deep Reinforcement Learning: A Brief Survey. *IEEE Signal Process. Mag.* **2017**, *34*, 26–38 [CrossRef]
6. Silver, D.; Hubert, T.; Schrittwieser, J.; Antonoglou, I.; Lai, M.; Guez, A.; Lanctot, M.; Sifre, L.; Kumaran, D.; Graepel, T. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science* **2018**, *362*, 1140–1144. [CrossRef] [PubMed]
7. Hwangbo, J.; Lee, J.; Dosovitskiy, A.; Bellicoso, D.; Tsounis, V.; Koltun, V.; Hutter, M. Learning agile and dynamic motor skills for legged robots. *Sci. Robot.* **2019**, *4*, 1–20. [CrossRef] [PubMed]
8. Degrave, J.; Felici, F.; Buchli, J.; Neunert, M.; Tracey, B.; Carpanese, F.; Ewalds, T.; Hafner, R.; Abdolmaleki, A.; de Las Casas, D. Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature* **2022**, *602*, 414–419. [CrossRef] [PubMed]
9. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjel, ; A.K.; Ostrovski, G. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533. [CrossRef] [PubMed]
10. Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; Wierstra, D. Continuous control with deep reinforcement learning. In Proceedings of the 4th International Conference on Learning Representations, San Juan, Puerto Rico, 2–4 May 2016.
11. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal policy optimization algorithms. *arXiv* **2017**, arXiv: 1707.06347.
12. Luo, Z.; Li, X.; Wang, L.; Shen, Q. Multiconstrained gliding guidance based on optimal and reinforcement learning method. *Math. Probl. Eng.* **2021**, *2021*, 6652232.
13. Waldock, A.; Greatwood. C.; Salama, F.; Richardson, T. Learning to perform a perched landing on the ground using deep reinforcement learning. *J. Intell. Robot. Syst.* **2018**, *92*, 685–704. [CrossRef]
14. Rodriguez-Ramos, A.; Sampedro, C.; Bavle, H.; De La Puente, P.; Campoy, P. A deep reinforcement learning strategy for UAV autonomous landing on a moving platform. *J. Intell. Robot. Syst.* **2019**, *93*, 351–366. [CrossRef]
15. Wu, M.; He, X.; Qiu, Z.; Chen. Z. Guidance law of interceptors against a high-speed maneuvering target based on deep Q-Network. *Trans. Inst. Meas. Control.* **2022**, *44*, 1373–1387. [CrossRef]
16. Zu, W.; Yang, H.; Liu, R.; Ji, Y. A multi-dimensional goal aircraft guidance approach based on reinforcement learning with a reward shaping algorithm. *Sensors* **2021**, *21*, 5643. [CrossRef] [PubMed]
17. Wang, Z.; Pan, W.; Li, H.; Wang, X.; Zuo, Q. Review of deep reinforcement learning approaches for conflict resolution in air traffic control. *Aerospace* **2022**, *9*, 294. [CrossRef]
18. Wang, Z.; Li, H.; Wu, Z.; Wu, H. A pretrained proximal policy optimization algorithm with reward shaping for aircraft guidance to a moving destination in three-dimensional continuous space. *Int. J. Adv. Robot. Syst.* **2021**, *18*, 1–13. [CrossRef]
19. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*, 2nd ed.; MIT Press: London, UK, 2018.
20. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
21. Virtanen, K.; Karelahti, J.; Raivio, T. Modeling air combat by a moving horizon influence diagram game. *J. Guid. Control Dyn.* **2006**, *29*, 1080–1091. [CrossRef]
22. Austin, F.; Carbone, G.; Falco M, Hinz, H.; Lewis, M. Game theory for automated maneuvering during air-to-air combat. *J. Guid. Control Dyn.* **1990**, *13*, 1143–1149. [CrossRef]
23. Ng, A.; Harada, D.; Russell, S. Policy invariance under reward transformations: Theory and application to reward shaping. In Proceedings of the 16th International Conference on Machine Learning, Bled, Slovenia, 27–30 June 1999.