drones

Article

A Penetration Method for UAV Based on Distributed Reinforcement Learning and Demonstrations

Kexv Li, Yue Wang *, Xing Zhuang, Hao Yin, Xinyu Liu and Hanyu Li

School of Mechatronical Engineering, Beijing Institute of Technology, Beijing 100081, China
* Correspondence: jackwy@bit.edu.cn

Abstract: The penetration of unmanned aerial vehicles (UAVs) is an essential and important link in modern warfare. Enhancing UAV's ability of autonomous penetration through machine learning has become a research hotspot. However, the current generation of autonomous penetration strategies for UAVs faces the problem of excessive sample demand. To reduce the sample demand, this paper proposes a combination policy learning (CPL) algorithm that combines distributed reinforcement learning and demonstrations. Innovatively, the action of the CPL algorithm is jointly determined by the initial policy obtained from demonstrations and the target policy in the asynchronous advantage actor-critic network, thus retaining the guiding role of demonstrations in the initial training. In a complex and unknown dynamic environment, 1000 training experiments and 500 test experiments were conducted for the CPL algorithm and related baseline algorithms. The results show that the CPL algorithm has the smallest sample demand, the highest convergence efficiency, and the highest success rate of penetration among all the algorithms, and has strong robustness in dynamic environments.

Keywords: UAV penetration; demonstrations; distributed reinforcement learning; asynchronous advantage actor-critic

Citation: Li, K.; Wang, Y.; Zhuang, X.; Yin, H.; Liu, X.; Li, H. A Penetration Method for UAV Based on Distributed Reinforcement Learning and Demonstrations. *Drones* **2023**, *7*, 232. https://doi.org/10.3390/drones7040232

Academic Editors: Kaiyu Qin, Haitao Nie, Yikang Yang, Xue Li, Jinliang Shao, Lei Shi and Mengji Shi

Received: 26 February 2023
Revised: 21 March 2023
Accepted: 23 March 2023
Published: 27 March 2023

1. Introduction

The autonomous penetration of UAVs refers to the ability of UAVs to break through the enemy's defenses, which is an essential function of military UAVs. In the actual battlefield environment, once a UAV is found by the enemy, it will be intercepted and destroyed continuously. Therefore, one of the key factors for UAVs to successfully perform tasks is being able to break through the enemy's multi wave interception. For this reason, many scholars have done intensive research on UAV penetration.

Based on the Dubins curve, Bares P et al. designed a 3D path following algorithm, so that the UAV can avoid static obstacles while meeting the kinematic and dynamic constraints [1]. Based on the performance index of minimum energy, Han S C et al. designed an optimal/suboptimal proportional guidance method to enable the UAV to avoid linear motion obstacles. However, when approaching the target, this method has the problem of divergent overload instructions [2]. Based on nonlinear model predictive control theory, Singh L designed a flight trajectory that can automatically avoid incoming missiles, however, it is practical when the overload of the aircraft itself can reach infinity, the computational efficiency of this algorithm is limited [3]. Based on model prediction theory, Gagnon E et al. designed corresponding penetration trajectories for several situations in an unknown environment, but the assumed obstacle situation is relatively simple [4]. Based on the given midpoint, Watanabe Y et al. designed the optimal avoidance trajectory with the least energy, but in the complex obstacle type, this method can hardly give the required midpoint [5]. Based on the orthogonal configuration method, Smith NE et al. estimated the flight range of the intruder and designed the avoidance corridor, but the result turned out to be probabilistic [6].

In general, research in the field of UAV penetration has been emerging, and the methods used are becoming more and more intelligent. With the development of computer hardware and artificial intelligence technology, machine learning methods such as deep learning and reinforcement learning are gradually being applied to the field of UAV penetration. This type of method is usually based on a large number of sample data generated by confrontation, iterating the neural network repeatedly and constantly updating the internal parameters of the network to obtain autonomous strategies for UAV penetration. The above methods greatly enhance the autonomy and rapidity of UAV penetration, but the disadvantage is that the sample demand is too large. For UAV penetration, generating a large amount of real data for confrontation requires significant time and money. In addition, the process of sample generation requires UAVs to constantly engage in confrontation, resulting in incalculable equipment losses, which is almost unacceptable for costly UAVs. Aiming at the problem of excessive sample demand, this paper proposes a combination policy learning method to reduce the sample demand in UAV penetration. Based on the combination of demonstrations and distributed reinforcement learning, this paper is dedicated to achieve online penetration with less sample requirements, which can provide some ideas for online penetration of UAV.

The structure of this paper is shown below.

(a) Introduction. This section mainly describes the research status of UAV penetration strategies, summarizes the existing problems, and then leads to the theme and purpose of this paper.
(b) Problem Description. This section first introduces the application scenario of UAV penetration in detail, and then describes the modeling of the UAV guidance system. On this basis, it describes and models the UAV penetration problem from the perspective of Reinforcement Learning.
(c) CPL Algorithm for UAV penetration. This section derives the theoretical formula of the proposed algorithm. The related algorithms mainly include a pre-training algorithm—Adversarial Inverse Reinforcement Learning (AIRL), distributed reinforcement learning algorithm—Asynchronous Advantage Actor-Critic (A3C), and the Combination Policy Learning (CPL) algorithm, formed by combining the former two.
(d) Environmental results and discussion. This section mainly analyzes and discusses the experimental results. For the CPL algorithm and other related algorithms, this section conducts training and testing experiments and compares the results, so as to conduct a detailed analysis of the sample requirements, convergence efficiency, success rate of penetration, and other indicators of each algorithm.
(e) Conclusions and future work. This section mainly summarizes the full text and prospects for future work in this paper.

2. Problem Description

2.1. UAV Penetration

As shown in Figure 1, the application scenario of this paper is: during the flight of the UAV, it is found and continuously tracked by the enemy in state 1, then it is intercepted by interceptor 1 in state 2, which is launched by the enemy. If the UAV successfully evades interceptor 1 at this time, during the subsequent flight, the enemy will continue to launch interceptor 2, so as to conduct the second interception in state 3.

In the above process, the problem to be solved in this paper is to make a real-time penetration decision during the two times of interception, guiding the UAV to break through the enemy's defense.

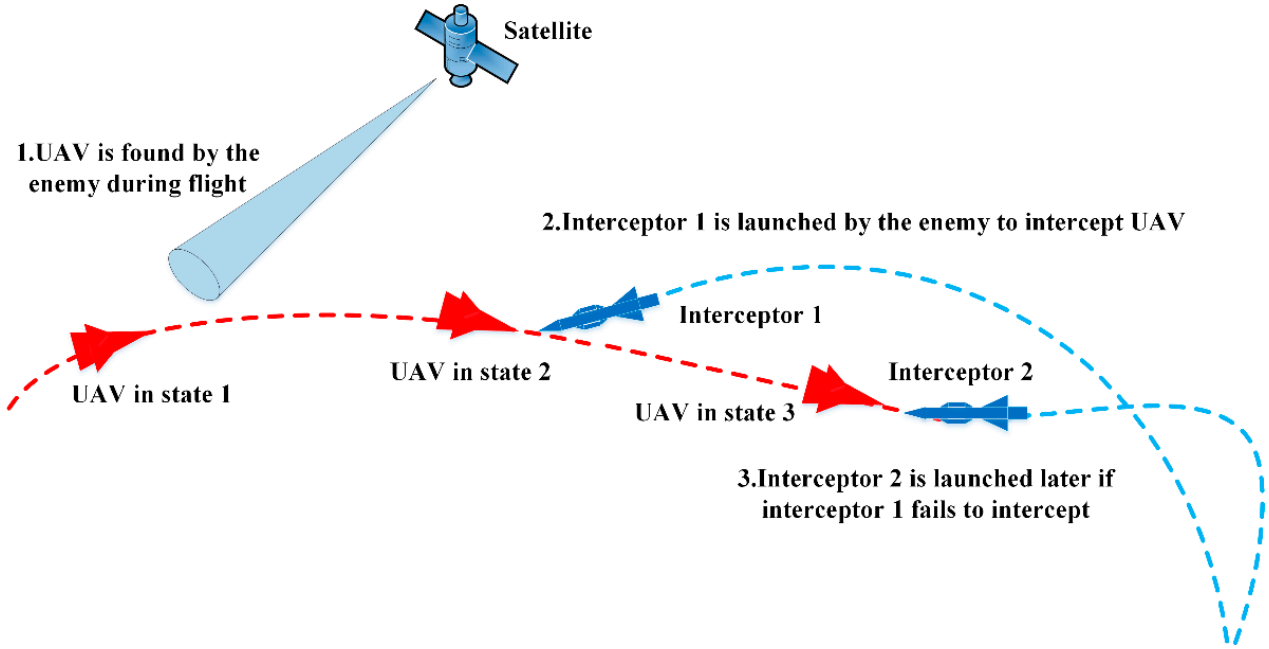

Figure 1. The application scenario for the online penetration of UAV.

2.2. Modeling of the UAV Guidance System

Before the design of UAV penetration algorithm, the first step is to build a six degree of freedom UAV model, providing simulation support for subsequent experiments. Taking the glider as an example, a six degree of freedom UAV model is built with its parameters derived from real data. The main reference for the establishment of dynamic and kinematic models is Ref. [7], which will not be repeated here. This paper focuses on its guidance system modeling, which will provide a basis for the action space design of the subsequent algorithm.

Based on the current flight state and mission, the task of the guidance system is to generate the corresponding guidance command, which needs to meet a variety of terminal constraints. The current state [8] includes speed v, speed inclination θ, speed azimuth σ, longitude λ, latitude ϕ, and altitude h. The terminal constraint is shown in Formula (1).

$$x_f = \left(v_f, \theta_f, \lambda_f, \phi_f, h_f\right) \tag{1}$$

where x_f is the constraint vector, v_f is the speed constraint, θ_f is the speed inclination constraint, λ_f is the longitude constraint, ϕ_f is the latitude constraint, and h_f is the altitude constraint.

In this paper, the longitudinal overload command is defined as n_y^* and the lateral overload command is defined as n_z^*. During the guidance process of UAV, it is necessary to eliminate heading errors in a timely manner. Heading errors are defined as the included angle between the line-of-sight azimuth from the current position of the aircraft to the target and speed azimuth. This paper defines σ_{LOS} as the line-of-sight azimuth from the current position to the target, then the heading errors are equal to $\sigma_{LOS} - \sigma$. For longitudinal guidance and lateral guidance, the methods for obtaining the optimal lateral overload and optimal lateral overload are shown in Formula (2).

$$\begin{cases} u_y^* = n_y^* = k(C_h L_R - C_\theta) + 1 \\ u_z^* = n_z^* = \frac{\sigma_{LOS} - \sigma}{k\left(L_{Rf} - L_R\right)} \end{cases} \tag{2}$$

where u_y^* is the optimal overload required longitudinally, u_z^* is the optimal overload required laterally, $k = \frac{g_0}{v^2} \approx \frac{g}{v^2}$, L_R is the current range, and L_{Rf} is the range constraint

of the glide section. C_h and C_θ are the guidance coefficients obtained based on optimal control [9], and their calculation method is shown in Formula (3).

$$\begin{cases} C_h = \frac{6\left[\left(L_R - L_{Rf}\right)\left(\theta_f - \theta\right) - 2h + 2h_f\right]}{k^2\left(L_R - L_{Rf}\right)^3} \\ C_\theta = \frac{2\left[L_R L_{Rf}\left(\theta - \theta_f\right) - {L_{Rf}}^2\left(2\theta + \theta_f\right) + {L_R}^2\left(2\theta_f + \theta\right) + 3\left(L_R + L_{Rf}\right)\left(h_f - h\right)\right]}{k^2\left(L_R - L_{Rf}\right)^3} \end{cases} \tag{3}$$

For the glider, if the overload command is taken as the control command, the method for calculating the control angles [10] of attack α and bank β is shown as Formula (4).

$$\begin{cases} \frac{\rho v^2 S_m C_L(M_a, \alpha)}{2g_0} = \sqrt{{n_y^*}^2 + {n_z^*}^2} \\ \beta = arctan\left(\frac{n_z^*}{n_y^*}\right) \end{cases} \tag{4}$$

where ρ is the atmospheric density at the current altitude, S_m is the aircraft reference area, C_L is the lift coefficient determined by the Mach number and angle of attack, and g_0 is the gravitational acceleration at sea level. According to the first equation in Formula (4), the angle of attack α can be obtained by inverse interpolation, and the second equation in Formula (4) can directly calculate the angle of bank β.

2.3. Reinforcement Learning for UAV Penetration

For the penetration process of UAV, the state of the next moment depends on the state and action of the current moment. Furthermore, the penetration process of UAV can be modeled as a Markov process [11]. On this basis, the decision-making problem of UAV penetration can be solved by reinforcement learning method. This method can update its own network iteratively by constantly interacting with the environment, in order to learn a good strategy for UAV penetration. The first step of this method is to design the corresponding action space, state space, and reward function, which are related to the UAV's own model, combat tasks, and battlefield environment, etc.

In this paper, the upper control command of the guidance system is overload and the lower command is the angles of attack and bank. Moreover, when using the Reinforcement Learning algorithm to control UAV flight, the command output by the algorithm can be either overload (longitudinal overload and lateral overload) or attitude angle (angles of attack and bank). Both commands can replace the command of the guidance system. (1) The overload control method can quickly achieve the required overload for UAVs, and is widely used in aircraft control systems that require rapid maneuvers; (2) The attitude angle control method has good stability margins, but excessive stability margins lead to insufficient maneuverability. The purpose of this paper is to study the maneuvering penetration methods of UAVs, so it is necessary for UAVs to have high maneuverability. Based on the above considerations, this paper selects overload command as the action variable of the Reinforcement Learning Network.

According to the current battlefield situation, the Reinforcement Learning Network outputs the values of longitudinal overload command and lateral overload command that control the UAV flight, thus replacing the overload command value of the original guidance system, controlling the UAV to fly along the new trajectory and avoid interception.

In addition, the longitudinal and lateral limit overloads of the 6-DOF glider UAV constructed in this paper are 10G, respectively. Therefore, the action vector a is shown in Formula (5).

$$a = \left(n_y^*, n_z^*\right) \tag{5}$$

The value ranges of longitudinal overload command n_y^* and lateral overload command n_z^* are: $n_y^* \in (0, 10)$, $n_z^* \in (0, 10)$, which can ensure the smooth flight of the UAV.

For the state space of the reinforcement learning network, the purpose of the network is to guide the UAV to break through the defense of the interceptor, so the situation information that the network needs to observe should include: the position and speed

information of both the UAV and the interceptor. The network judges the next action of the UAV according to the appeal information. The state vector s of UAV penetration network is shown in Formula (6).

$$s = \left(B_{UAV}, L_{UAV}, H_{UAV}, v_{x,UAV}, v_{y,UAV}, v_{z,UAV}, B_{INT}, L_{INT}, H_{INT}, v_{x,INT}, v_{y,INT}, v_{z,INT}\right) \tag{6}$$

where, B_{UAV}, L_{UAV}, and H_{UAV} are the longitude, latitude, and altitude of the UAV, respectively, $v_{x,UAV}$, $v_{y,UAV}$, and $v_{z,UAV}$ are the component velocities of the UAV on the x, y, and z axes in the *ECEF* coordinate system, B_{INT}, L_{INT}, and H_{INT} are the longitude, latitude and altitude of the interceptor, respectively, and $v_{x,INT}$, $v_{y,INT}$, and $v_{z,INT}$ are the component velocities of the interceptor on the x, y, and z axes in the *ECEF* coordinate system.

In a real scenario, (1) the position and speed information of the UAV can be directly obtained from the UAV's navigation system. For example, both inertial navigation systems and GPS navigation systems can obtain the position and speed information of the UAV; (2) the position and speed information of the interceptor is integrated by our detection system (seeker, ground radar, space-based satellite, etc.).

The reward function of the reinforcement learning network is designed based on the state space. Since the purpose of the network is to guide UAV to break through the interceptor's defense, the main return focus on the successful penetration of UAV is, when the UAV successfully penetrates, it gives the agent a positive reward. Under the assumption that the interceptor is equipped with an antipersonnel warhead, the sign of successful penetration is that the UAV has passed the interceptor and the relative distance between the above two is always greater than 400 m. However, if there is only main return, the UAV will face the problem of sparse reward, which leads to difficult convergence of the network. In addition to the main return, this paper adds intermediate return to guide the UAV to explore in the process of penetration.

For the intermediate return, the farther the relative distance between the UAV and the interceptor is, the easier it is for the UAV to evade the interceptor successfully. The greater the relative distance between the two, the greater the intermediate return. In addition, the greater the relative velocity between the UAV and the interceptor, the more difficult it is for the interceptor to intercept the UAV. Therefore, the greater the relative velocity between the two, the greater the probability of successful penetration, and the greater the intermediate return.

Combined with the above analysis and various performance parameters of the UAV model constructed in this paper, the reward function r is shown in Formula (7), whose value at each time is controlled between $[-1,1]$ for easy convergence.

$$r = \begin{cases} 1 & , Situation\ 1 \\ -1 & , Situation\ 2 \\ \frac{\arctan(dR-400)}{100} + \frac{e^{dV}}{50} & , Situation\ 3 \end{cases} \tag{7}$$

where *Situation 1* represents the successful penetration of UAV, *Situation 2* represents the failure of UAV penetration, *Situation 3* represents that the UAV is in the process of penetration, dR represents the relative distance between UAV and interceptor, and dV represents the relative velocity between UAV and interceptor.

The interaction process between the network and the environment in training is shown in Figure 2: (1) the network first obtains the state information from the environment, including the position and speed information of the UAV and Interceptor; (2) the network outputs the corresponding action command (the longitudinal overload command and lateral overload command) according to the above state information, for controlling UAV flight; (3) the UAV executes the above action command, the current state is changed, and the corresponding reward value of this action command is obtained. Through continuous interaction with the environment, the network gradually collects samples and updates iterations, so as to continuously improve the reward value, and finally converges to obtain a satisfied network of UAV penetration.

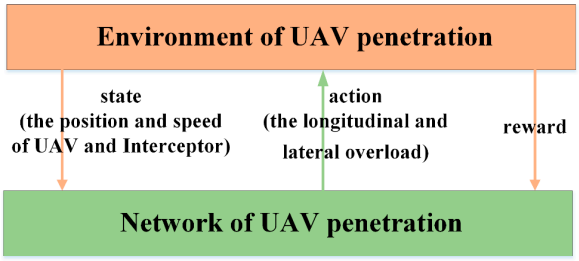

Figure 2. Interaction between the environment and network in UAV penetration.

3. CPL Algorithm for UAV Penetration

3.1. Pre-Training Algorithm

According to the demonstrations of UAV penetration, the initial policy of UAV penetration is learned by pre-training. In this process, considering the high-dimensional and continuous characteristics of the UAV penetration environment, the adversarial inverse reinforcement learning (AIRL) algorithm [12] is selected as the pre-training network. The specific illustrations of AIRL are as follows.

The general inverse reinforcement learning methods, such as generative adversarial imitation learning (GAIL) [13] and generative adversarial network guided cost learning (GAN-GCL) [14], are trajectory-centric. Moreover, these methods need to estimate a complete trajectory, whose estimation variance is larger compared with estimating a single state-action pair. The AIRL algorithm directly estimates a single state-action pair [12], so its estimation variance is smaller.

The AIRL algorithm introduces a sampling distribution, which naturally turns the reinforcement learning problem into a generative adversarial network (GAN) optimization problem. The AIRL algorithm iterates between the introduced sampling distribution and the reward function, where the discriminator is the reward and the generator is the sampling distribution. The original discriminator D_θ is shown as Formula (8).

$$D_\theta(s_t, a_t) = \frac{exp(f_\theta(s_t, a_t))}{exp(f_\theta(s_t, a_t)) + \pi(a_t|s_t)} \tag{8}$$

where $\pi(a_t|s_t)$ is the policy of the sampling distribution to be updated and $f_\theta(s_t, a_t)$ is the learned function. The partition function is ignored in the above Formula (8). In practice, the normalization of the probability value can be guaranteed by Softmax function or Sigmoid function. It is proved that in the optimal case, $f^*(s_t, a_t) = log\pi^*(a_t|s_t) = A^*(s_t, a_t)$ gives the advantage function of the optimal policy [12]. However, the advantage function is a highly entangled Reward Function minus a baseline value. The AIRL algorithm proposes to disentangle the advantage function to obtain the reward function.

Theoretically, when the reward is state-only, it is more likely to be robust to dynamics, and the reward shaping will reduce its robustness [12]. Therefore, it is necessary to multi-parameterize a shaping term function h_ϕ, whose parameters are ϕ. The discriminator $D_{\theta,\phi}$ in the AIRL algorithm is shown as Formula (9).

$$D_{\theta,\phi}(s_t, a_t, s_{t+1}) = \frac{exp\left(f_{\theta,\phi}(s_t, a_t, s_{t+1})\right)}{exp\left(f_{\theta,\phi}(s_t, a_t, s_{t+1})\right) + \pi(a_t|s_t)} \tag{9}$$

where $f_{\theta,\phi}$ is limited to the reward approximator g_θ and the shaping term h_ϕ, which is shown as Formula (10).

$$f_{\theta,\phi}(s_t, a_t, s_{t+1}) = g_\theta(s_t, a_t) + \gamma h_\phi(s_{t+1}) - h_\phi(s_t) \tag{10}$$

Among them, additional fitting of h_ϕ is needed.

The pseudo code of pre-training algorithm (AIRL) is shown as Algorithm 1.

Algorithm 1. Pre-training algorithm (AIRL) Algorithm.

Obtain expert demonstrations τ_i^E
Initialize policy π and discriminator $D_{\theta,\phi}$
for step t in $\{1, \ldots, N\}$ **do**
    Collect trajectories $\tau_i = (s_0, a_0, \ldots, s_T, a_T)$ by executing π.
    Train $D_{\theta,\phi}$ via binary logistic regression to classify expert data τ_i^E from sample τ_i.
    Update reward $r_{\theta,\phi}(s_t, a_t, s_{t+1}) \leftarrow logD_{\theta,\phi}(s_t, a_t, s_{t+1}) - log(1 - D_{\theta,\phi}(s_t, a_t, s_{t+1}))$
    Update π with respect to $r_{\theta,\phi}$ using any policy optimization method.
end for

3.2. A3C Algorithm

Combining the results of the pre-training algorithm, the distributed reinforcement learning algorithm can further reduce the sample demand, thus improving the speed of training. As an excellent distributed reinforcement learning algorithm, the asynchronous advantage actor-critic (A3C) algorithm [15] adopts asynchronous training, which is based on the advantage actor-critic (A2C) algorithm [16]. The A3C algorithm greatly improves the convergence efficiency, and is quite suitable for UAV penetration training, which needs to reduce the time and capital cost as much as possible. Therefore, the A3C algorithm is initially selected as the distributed reinforcement learning algorithm for UAV penetration.

As shown in Figure 3, in the A3C algorithm, the neural network model includes multiple thread networks for interacting with the environment and a public network in the parameter server. The thread networks are mainly responsible for interacting with the environment, whose number is at most equal to the number of CPU cores of the computer [17]. The actor-learner in each thread network interacts with the environment, respectively [18]. Each thread network does not interfere with each other, but interacts with the environment independently. each actor-learner calculates the gradient of loss function in its thread network, and uploads the gradient to the parameter server, thus iteratively updating the public network in the parameter server [19]. Every once in a while, these thread networks will update their network parameters to the same parameters as the public network, guiding the subsequent environment interaction [20–24].

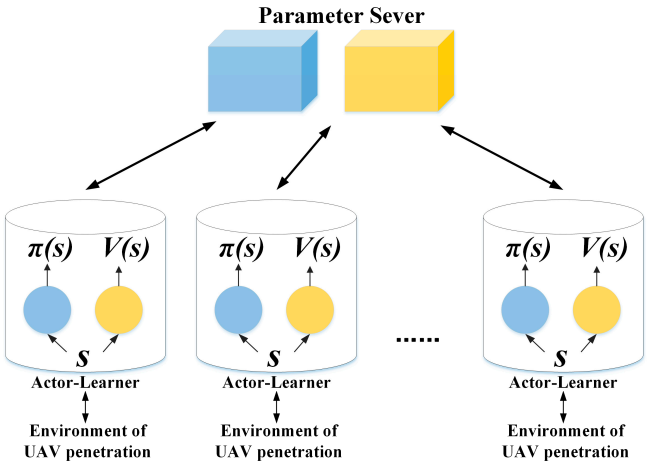

Figure 3. Architecture of the A3C algorithm.

The pseudo code of the A3C algorithm is shown in Algorithm 2. Since the training process of each actor-learner is the same, the algorithm can be interpreted from the perspective of an actor-learner [25,26]. In each learning cycle, through asynchronous communication, each actor-learner will first synchronize its network parameters from the parameter server [27]. Based on the updated thread network, the actor-learner will make decisions and interact with the environment at most t_{max} times [28]. The experience of interactive exploration with the environment will be collected and used to train its own

thread network, and the gradients $d\theta$ and $d\theta_v$ will be obtained, respectively [29]. After the actor-learner interacts with the environment to T_{max} times, the actor-learner will submit the sum of all accumulated gradients to the parameter server, so that it can update the network parameters θ and θ_v in the parameter server asynchronously.

Algorithm 2. Asynchronous Advantage Actor-Critic (A3C).

Initialize total number of exploration steps T_{max}, number of exploration steps in each cycle t_{max}
Initialize thread step counter $t = 1$
while $T \leq T_{max}$ **do**
  Initialize network parameter gradient: $d\theta = 0$,$d\theta_v = 0$
  Keep synchronization with the parameter server: $\theta' = \theta$,$\theta'_v = \theta_v$
  $t_{start} = t$
  Set the initial state of each exploration cycle to s_t
  while the end state is reached or $t - t_{start} == t_{max}$ do
    Select decision behavior a_t based on decision strategy $\pi(s_t|\theta')$
    Executing a_t in the environment and get reward r_t and the next state s_{t+1}
  end while
  if the end state is reached, **then**
    $r = 0$
  else
    $r = V(s_t|\theta'_v)$
  end if
  for $i = t-1, t-2, \ldots, t_{start}$ **do**
    Update discount rewards $r = r_i + \gamma r$
    Cumulative parameter gradient θ', $d\theta = d\theta + \nabla_{\theta'} log\pi(s_i|\theta')(r - V(s_i|\theta'_v))$
    Cumulative parameter gradient θ'_v, $d\theta_v = d\theta_v + \partial\left(r - V(s_i|\theta'_v)\right)^2/\partial\theta'_v$
  end for
  Asynchronous update θ and θ_v based on gradient $d\theta$ and $d\theta_v$
end while

3.3. CPL Algorithm

The exploration space of UAV penetration is huge, so if the A3C algorithm is used simply to explore from scratch, until a better strategy for UAV penetration is obtained, the sample size required in this process may be quite large, and the network may not even converge [30,31]. Therefore, this paper proposes to conduct pre-training before the training of the A3C algorithm. The pre-training algorithm gets the policy obtained from the expert demonstrations, then the policy will be the initial policy of the A3C algorithm, so as to reduce invalid exploration and improve training efficiency [32]. The policy obtained from pre-training does not need to be optimal; it only needs to be good enough compared with the conventional initial policy, which is usually obtained by the random generation method [33].

When pre-training is combined with the reinforcement learning process, the conventional initial policy of reinforcement learning is usually directly replaced by the policy obtained from pre-training, which is also called the direct replacement method. However, the initial policy for UAV penetration obtained through expert demonstrations is still of great reference value. If the direct replacement method is simply used, the initial policy will soon be covered with the exploration of the network, resulting in its inability to maximize its value. Therefore, this paper proposes a combination policy learning (CPL) algorithm, which can retain the performance of the initial policy to the greatest extent in the subsequent training.

In the CPL algorithm, the action a follows a combined strategy. That is, a is obtained by combining the initial policy π_{ini} and the target policy π_{tar}, as shown in Formula (11).

$$a = \rho a_{ini}(s) + a_{tar}(s) \tag{11}$$

where ρ is equal to the reciprocal of the number of iterations, which is shown in Formula (12).

$$\rho = \frac{1}{E} \tag{12}$$

where E is the number of iterations.

The training process includes two parts of the policies: one is the Initial Policy obtained from pre-training, which will be fixed after initialization; the other is the Target Policy that will be trained in the later learning process. That is, the initial policy obtained from the pre-training is fixed, and only the target policy is iterated. Formula (11) illustrates that the action consists of the above two policies, and the weight of the initial policy decreases with the increase of the training times. Moreover, at the initial stage of training, the initial policy can play a greater guiding role; later, with the increase of training times, the target policy gradually takes shape; at the end of training, the action decision depends more on the target policy, which is the ultimate policy. In this way, the CPL algorithm can maintain the performance of the initial policy as much as possible.

Combining the above CPL algorithm, the complete training process is as follows.

(a) Initialize all networks in the A3C algorithm in the way of combination policy learning;
(b) Let the actor-learner interact with the environment. The action is the combination of the initial policy and target policy, i.e., formula (11), and the samples are stored in the form of $(s_t, a_{tar}, s_{t+1}, r_t)$;
(c) Sample from the replay buffer and get $(s_t, a_{tar}, s_{t+1}, r_t)$, then update the Target Policy π_{tar};
(d) Repeat steps (b) and (c) above until the UAV penetration strategy converges to near optimal.

Compared with the general A3C algorithm, the CPL algorithm has two differences: (1) its initial policy is not randomly generated, but obtained by pre-training according to expert demonstrations; (2) its action is the combination of initial policy and target policy, but not the entire action a_{tar} of the target policy. The pseudo code of the CPL algorithm shown in Algorithm 3.

The structure of the CPL algorithm is shown in Figure 4. First, the demonstrations of the UAV penetration are used to obtain the pre-training Network, which will be the initial policy of the CPL algorithm. In the subsequent training of reinforcement learning, the action of UAV penetration is composed of the fixed initial policy and learned target policy. Target policy is in the public network of A3C algorithm. After iterative updating, the target policy gradually converges, and the final decision network of UAV penetration is obtained.

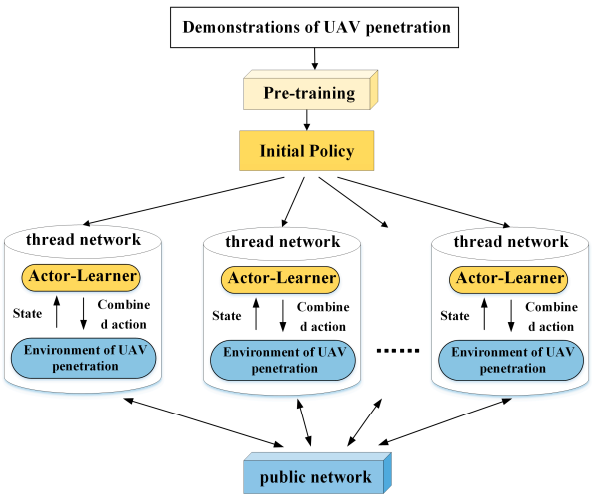

Figure 4. Architecture of the CPL algorithm.

Algorithm 3. Combination Policy Learning (CPL) Algorithm.

Obtain expert demonstrations τ_i^E
Initialize policy π_{ini} and discriminator $D_{\theta,\phi}$
for step t in $\{1, \ldots, N\}$ **do**
    Collect trajectories $\tau_i = (s_0, a_0, \ldots, s_T, a_T)$ by executing π_{ini}
    Train $D_{\theta,\phi}$ via binary logistic regression to classify expert demonstrations τ_i^E from sample τ_i
    Update reward $r_{\theta,\phi}(s_t, a_t, s_{t+1}) \leftarrow logD_{\theta,\phi}(s_t, a_t, s_{t+1}) - log(1 - D_{\theta,\phi}(s_t, a_t, s_{t+1}))$
    Update π_{ini} with respect to $r_{\theta,\phi}$ using any policy optimization method
end for
Obtain the initial policy π_{ini} from the above training, and obtain its corresponding neural network parameters θ_{ini}
Initialize total number of exploration steps T_{max}, number of exploration steps in each cycle t_{max}
while $T \leq T_{max}$ **do**
    Initialize network parameter gradient: $d\theta = 0$,$d\theta_v = 0$
    Keep synchronization with the parameter server: $\theta' = \theta_{ini}$,$\theta'_v = \theta_{ini}$
    $t_{start} = t$
    Set the initial state of each exploration cycle to s_t
    while the end state is reached or $t - t_{start} == t_{max}$ **do**
$a_t = \rho a_{ini}(s_t) + a_{tar}(s_t)$
        Take decision behavior a_t in the environment and get reward r_t and the next state s_{t+1}
    end while
    if the end state is reached, **then**
        $r = 0$
    else
        $r = V(s_t|\theta'_v)$
    end if
    for $i = t-1, t-2, \ldots, t_{start}$ **do**
        Update discount rewards $r = r_i + \gamma r$, where γ is the discount factor
        Cumulative parameter gradient θ', $d\theta = d\theta + \nabla_{\theta'} log\pi(s_i|\theta')(r - V(s_i|\theta'_v))$
        Cumulative parameter gradient θ'_v, $d\theta_v = d\theta_v + \partial\left(r - V(s_i|\theta'_v)\right)^2/\partial\theta'_v$
    end for
    Asynchronous update θ and θ_v based on gradient $d\theta$ and $d\theta_v$
end while

4. Environmental Results and Discussion

4.1. Training Experiment

According to the above analysis, this paper compares the pre-training algorithm, A3C algorithm, pre-training-A3C algorithm, and CPL algorithm, to analyze the performance of four algorithms in the UAV penetration environment. The pre-training-A3C algorithm is a combination of the A3C algorithm and pre-training algorithm using the direct replacement method. Specifically, the average reward of the above four algorithms in the training process is shown in Figure 5 and the success rate of penetration in the training process is shown in Figure 6.

In the training process, the UAV model and Interceptor model are first exported to the Dynamic Link Library form, then the above models are applied by invoking the logic of the Dynamic Link Library using Python language. The maximum number of episodes for each algorithm and interactions per episode are 1000 and 2000, respectively. This paper sets the seed to change from 0 to 9, that is, each algorithm conducts 10 groups of training, and then averages the results of these 10 trainings, so as to obtain the average reward value represented by the dark curve in Figure 5 and the success rate of penetration represented by the dark curve in Figure 6.

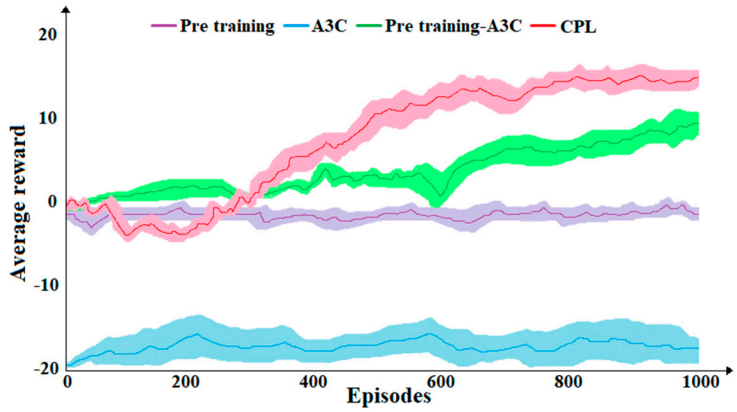

Figure 5. Average reward of all algorithms in training.

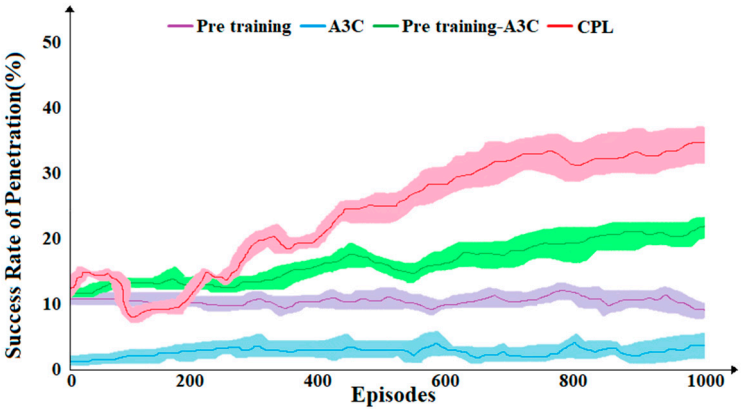

Figure 6. Success rate of penetration of all algorithms in training.

In Figure 5, (1) when the A3C algorithm is used for training alone, the value of the average reward is almost unchanged, indicating that the network has hardly learned valuable strategies. (2) When the pre-training algorithm generated by expert demonstrations is applied separately, the network can stably output the UAV penetration strategy. The average reward value of the pre-training algorithm is higher than that of the A3C algorithm, indicating that the strategy obtained by pre-training can output a more reasonable strategy. However, the average reward value is not high enough, and there are better penetration strategies that are worth exploring. (3) When the A3C algorithm is simply combined with the pre-training algorithm by using the direct replacement method, the pre-training-A3C algorithm can explore on the basis of the original pre-training strategy, quickly learning a better strategy, which rises its average reward value. However, the above strategy has a low utilization of expert demonstrations and its learning speed is low. When 1000 episodes are over, the average reward is still in the rising stage, so the algorithm has not converged in the end. (4) When using the CPL algorithm proposed in this paper, the algorithm converges when the episodes are about 700, and its convergence result is the best among the above algorithms. This is because this algorithm can not only explore beyond pre-training, but also retain the weight of pre-training to a large extent at the early stage of training, which can guide the algorithm to explore in a more valuable direction.

The changing trend of the success rate of penetration in Figure 6 is in accordance with that of the average reward in Figure 5, which also shows the correctness and rationality of reward shaping. From the training results in Figures 5 and 6, the CPL algorithm proposed in this paper has the highest convergence speed and the best convergence performance, thus verifying its feasibility and effectiveness. However, the above results are only the

performance in the training process. To objectively verify the apply effect of each algorithm, a test experiment should be conducted after the training.

4.2. Test Experiment

After fixing the network parameters of each algorithm obtained from the above training, this paper randomly sets the positions of both launch point and target point so as to randomly generate 500 different initial environments, and conduct 500 UAV penetration test experiments for each algorithm (pre-training; A3C; pre-training-A3C; CPL). The statistical data of these 500 experiments are shown in Table 1. For the convenience of the description, this paper defines the numbers of test as NT, numbers of penetrating interceptor 1 as NP1, numbers of penetrating interceptor 2 as NP2, and success rate of penetration as SRP. SRP is obtained by dividing NP2 by NT. the success rate of penetration (SRP) of each algorithm is further shown in Figure 7.

Table 1. Statistics of the 500 test experiments.

Algorithm	NT	NP1	NP2	SRP
Pre-training	500	178	163	32.6%
A3C	500	19	11	2.2%
Pre-training-A3C	500	234	197	39.4%
CPL	500	273	234	46.8%

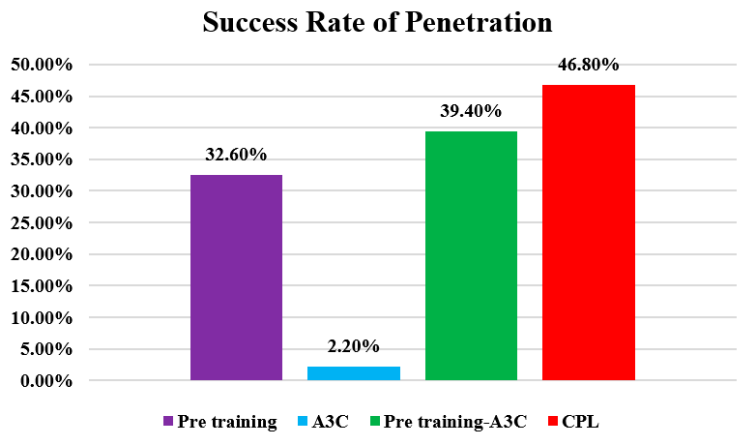

Figure 7. The success rate of penetration (SRP) of each algorithm in the test experiments.

Table 1 shows that (1) the pre-training algorithm successfully penetrated Interceptor 1 a total of 178 times, Interceptor 2 a total of 163 times, and the final success rate of penetration was 32.6%, which shows that the penetration strategy obtained through pre-training played a role. (2) The A3C algorithm successfully penetrated Interceptor 1 a total of 19 times, Interceptor 2 a total of 11 times, and the final success rate of penetration was 2.2%. This group of data shows the worst penetration performance among all algorithms, indicating that it is difficult to learn a good penetration strategy when using the reinforcement learning algorithm alone. (3) The pre-training-A3C algorithm successfully penetrated Interceptor 1 a total of 234 times, Interceptor 2 a total of 197 times, and the final success rate of penetration was 39.4%. Compared with the previous two groups, the performance of this algorithm was significantly improved, indicating that the reinforcement learning algorithm finally learned a better penetration strategy with pre-training. (4) The CPL algorithm successfully penetrated Interceptor 1 a total of 273 times, Interceptor 2 a total of 234 times, and the final success rate of penetration was 46.8%. The performance of this algorithm is the best among all algorithms, indicating that the CPL algorithm proposed in this paper performs best in the UAV penetration environment. In addition, the success rate of penetration for each algorithm in Figure 7 is consistent with Table 1, which also shows that the CPL

algorithm has the strongest penetration ability in all algorithms. The above results verify the effectiveness and robustness of CPL algorithm.

Three groups of experimental results were randomly selected from the above 500 test results, as shown in Figure 8. It can be seen from Figure 8 that only the CPL algorithm can completely break through the defense of Interceptor 1 and Interceptor 2 in Test environments 1, 2, and 3, while the other three algorithms cannot completely break through the defense of the two interceptors, which also shows the excellent performance of the CPL algorithm in UAV penetration.

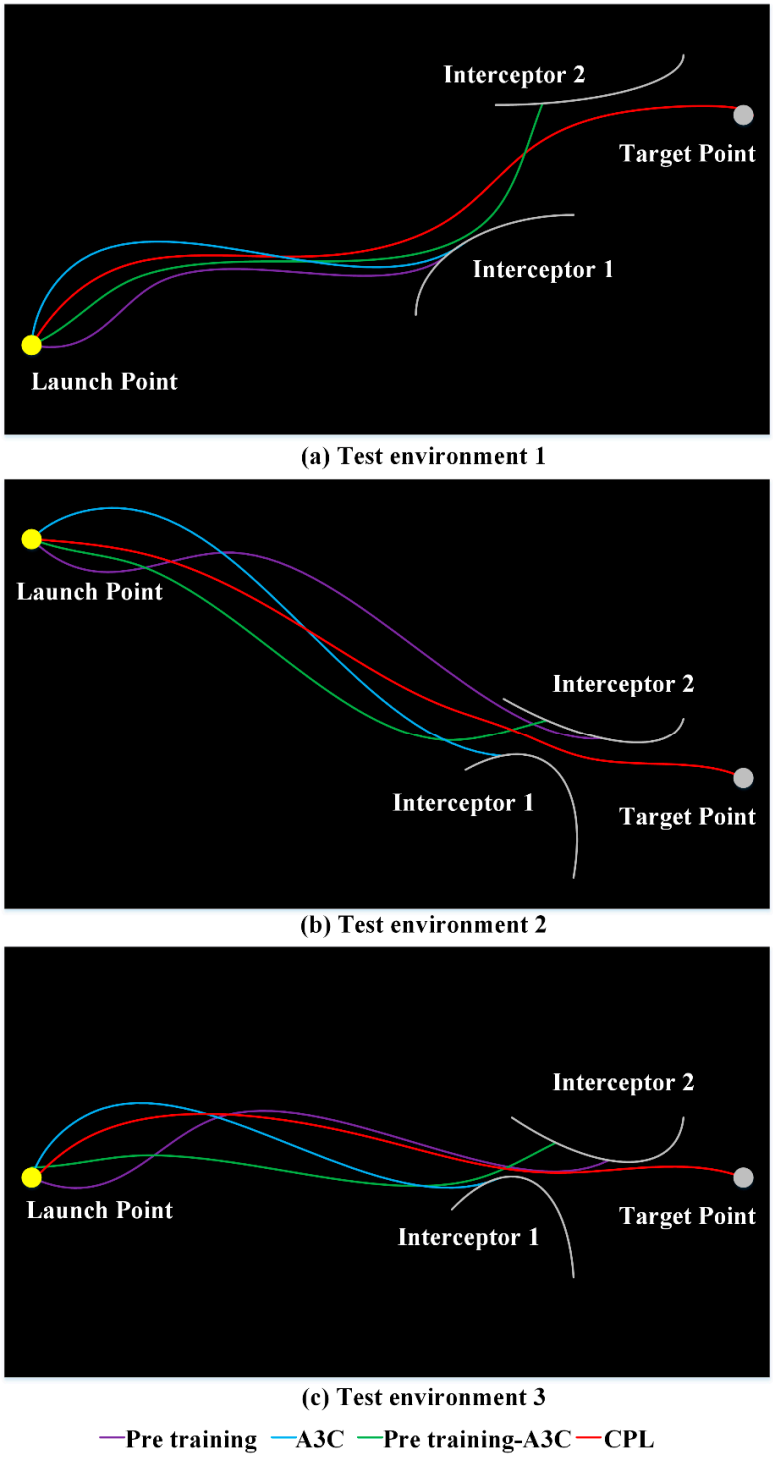

Figure 8. Test results of the algorithms in different environments.

5. Conclusions and Future Work

In order to reduce the sample demand of UAV penetration training, this paper first conducts pre-training based on expert demonstrations and uses the obtained network as the initial policy of the A3C algorithm. Then, based on the pre-training algorithm and A3C algorithm, the CPL algorithm is designed. The CPL algorithm innovatively retains the guidance role of the demonstrations more in the early stage of training, and that of reinforcement learning network more in the late stage of training. The training experiment shows that the convergence speed and result of the CPL algorithm have been greatly improved, compared with the pre-training algorithm, A3C algorithm, and pre-training-A3C algorithm. The test experiment shows that the CPL algorithm has the best performance among all algorithms in the randomly generated complex and unknown environment, which verifies the effectiveness and robustness of the CPL algorithm.

In the future, our work will be devoted to studying a new sampling method of the reinforcement learning algorithm, which will assist the network in exploring more valuable strategies, and thus achieve better performance of UAV penetration.

Author Contributions: K.L. wrote the manuscript, proposed the algorithm innovation, verified the experiments, and conducted data analysis; Y.W. designed the experimental process; H.Y. designed the architecture of the paper; X.Z. was responsible for the project management; X.L. and H.L. provided the UAV model and interceptor model. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

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

References

1. Jiang, L.; Nan, Y.; Zhang, Y.; Li, Z.H. Anti-Interception Guidance for Hypersonic Glide Vehicle: A Deep Reinforcement Learning Approach. *Aerospace* **2022**, *9*, 21. [CrossRef]
2. Han, S.C.; Bang, H.; Yoo, C.S. Proportional Navigation-Based Collision Avoidance for UAVs. *Int. J. Control Autom. Syst.* **2009**, *7*, 553–565. [CrossRef]
3. Singh, L. Autonomous missile avoidance using nonlinear model predictive control. In Proceedings of the AIAA Guidance Navigation, and Control Conference and Exhibit, Providence, RI, USA, 16–19 August 2004; p. 4910.3. [CrossRef]
4. Gagnon, E.; Rabbath, C.; Lauzon, M. Receding horizons with heading constraints for collision avoidance. In Proceedings of the AIAA Guidance Navigation, and Control Conference and Exhibit, San Francisco, CA, USA, 15–18 August 2005; p. 6369. [CrossRef]
5. Watanabe, Y.; Calise, A.; Johnson, E.; Evers, J. Minimum-effort guidance for vision-based collision avoidance. In Proceedings of the AIAA Atmospheric Flight Mechanics Conference and Exhibit, Keystone, CO, USA, 21–24 August 2006; p. 6641. [CrossRef]
6. Smith, N.E.; Cobb, R.; Pierce, S.J.; Raska, V. Optimal collision avoidance trajectories via direct orthogonal collocation for unmanned/remotely piloted aircraft sense and avoid operations. In Proceedings of the AIAA Guidance Navigation, and Control Conference, National, Harbor, MD, USA, 13–17 January 2014; p. 0966. [CrossRef]
7. Acton, J.M. Hypersonic boost-glide weapons. *Sci. Glob. Secur.* **2015**, *23*, 191–219. [CrossRef]
8. Li, G.H.; Zhang, H.B.; Tang, G.J. Maneuver characteristics analysis for hypersonic glide vehicles. *Aerosp. Sci. Technol.* **2015**, *43*, 321–328. [CrossRef]
9. Li, S.J.; Lei, H.M.; Shao, L.; Xiao, C.H. Multiple Model Tracking for Hypersonic Gliding Vehicles with Aerodynamic is Modeling and Analysis. *IEEE Access* **2019**, *7*, 28011–28018. [CrossRef]
10. Liu, X.D.; Zhang, Y.; Wang, S.; Huang, W.W. Backstepping attitude control for hypersonic gliding vehicle based on a robust dynamic inversion approach. *Proc. Inst. Mech. Eng. Part I-J Syst. Control Eng.* **2014**, *228*, 543–552. [CrossRef]
11. Karabag, O.; Bulut, O.; Toy, A.O. Markovian Decision Process Modeling Approach for Intervention Planning of Partially Observable Systems Prone to Failures. In Proceedings of the 4th International Conference on Intelligent and Fuzzy Systems (INFUS), Izmir, Turkey, 19–21 July 2022; Springer International Publishing Ag: Izmir, Turkey, 2022; pp. 497–504. [CrossRef]
12. Sackmann, M.; Bey, H.; Hofmann, U.; Thielecke, J. Modeling driver behavior using adversarial inverse reinforcement learning. In Proceedings of the 33rd IEEE Intelligent Vehicles Symposium (IEEE IV), Aachen, Germany, 5–9 June 2022; IEEE: Aachen, Germany, 2022; pp. 1683–1690. [CrossRef]
13. Li, W.X.; Hsueh, C.H.; Ikeda, K. Imitating Agents in A Complex Environment by Generative Adversarial Imitation Learning. In Proceedings of the IEEE Conference on Games, Osaka, Japan, 24–27 August 2020; pp. 702–705.

14. Chen, H.; Wang, Y.H.; Lagadec, B.; Dantcheva, A.; Bremond, F. Joint Generative and Contrastive Learning for Unsupervised Person Re-identification. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Virtual, 19–25 June 2021; pp. 2004–2013. [CrossRef]
15. Tiong, T.; Saad, I.; Teo, K.T.K.; Bin Lago, H. Autonomous Valet Parking with Asynchronous Advantage Actor-Critic Proximal Policy Optimization. In Proceedings of the IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 26–29 January 2022; pp. 334–340.
16. Maeda, R.; Mimura, M. Automating post-exploitation with deep reinforcement learning. *Comput. Secur.* **2021**, *100*, 13. [CrossRef]
17. Clark-Turner, M.; Begum, M. Deep Reinforcement Learning of Abstract Reasoning from Demonstrations. In Proceedings of the 13th Annual ACM/IEEE International Conference on Human-Robot Interaction (HRI), Chicago, IL, USA, 5–8 March 2018; Assoc Computing Machinery: Chicago, IL, USA, 2018; p. 372. [CrossRef]
18. Nair, A.; McGrew, B.; Andrychowicz, M.; Zaremba, W.; Abbeel, P. Overcoming Exploration in Reinforcement Learning with Demonstrations. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; IEEE Computer Society: Brisbane, Australia, 2018; pp. 6292–6299.
19. Liang, X.D.; Wang, T.R.; Yang, L.N.; Xing, E.R. In CIRL: Controllable Imitative Reinforcement Learning for Vision-Based Self-driving. In Proceedings of the 15th European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; Springer International Publishing: Munich, Germany, 2018; pp. 604–620. [CrossRef]
20. Hu, Y.D.; Gao, C.S.; Li, J.L.; Jing, W.X.; Li, Z. Novel trajectory prediction algorithms for hypersonic gliding vehicles based on maneuver mode on-line identification and intent inference. *Meas. Sci. Technol.* **2021**, *32*, 20. [CrossRef]
21. Wang, Z.; Cheng, X.X.; Li, H. Hypersonic skipping trajectory planning for high L/D gliding vehicles. In Proceedings of the 21st AIAA International Space Planes and Hypersonics Technologies Conference, Xiamen, China, 6–9 March 2017; p. 2135.
22. Wan, J.Q.; Zhang, Q.; Zhang, H.; Liu, J.N. A Midcourse Guidance Method Combined with Trajectory Prediction for Antinear-Space-Gliding Vehicles. *Int. J. Aerosp. Eng.* **2022**, *2022*, 4528803. [CrossRef]
23. Bhatta, P. Nonlinear Stability and Control of Gliding Vehicles. PhD Thesis, Princeton University, Princeton, NJ, USA, 2006.
24. Hu, Y.D.; Gao, C.S.; Jing, W.X. Joint State and Parameter Estimation for Hypersonic Glide Vehicles Based on Moving Horizon Estimation via Carleman Linearization. *Aerospace* **2022**, *9*, 20. [CrossRef]
25. Li, M.J.; Zhou, C.J.; Shao, L.; Lei, H.M.; Luo, C.X. A Trajectory Generation Algorithm for a Re-Entry Gliding Vehicle Based on Convex Optimization in the Flight Range Domain and Distributed Grid Points Adjustment. *Appl. Sci.* **2023**, *13*, 21. [CrossRef]
26. Yu, W.B.; Chen, W.C.; Jiang, Z.G.; Liu, M.; Yang, D.; Yang, M.; Ge, Y.J. Analytical entry guidance for no-fly-zone avoidance. *Aerosp. Sci. Technol.* **2018**, *72*, 426–442. [CrossRef]
27. Zhang, J.B.; Xiong, J.J.; Lan, X.H.; Shen, Y.N.; Chen, X.; Xi, Q.S. Trajectory Prediction of Hypersonic Glide Vehicle Based on Empirical Wavelet Transform and Attention Convolutional Long Short-Term Memory Network. *IEEE Sens. J.* **2022**, *22*, 4601–4615. [CrossRef]
28. King, L. Model Predictive Control and Reinforcement Learning Control for Hypersonic Gliding Vehicles and Spacecraft Docking. Available online: https://www.osti.gov/biblio/1863484 (accessed on 24 March 2022).
29. Tripathi, A.K.; Patel, V.V.; Padhi, R. Autonomous Landing of Fixed Wing Unmanned Aerial Vehicle with Reactive Collision Avoidance. In Proceedings of the 5th IFAC Conference on Advances in Control and Optimization of Dynamical Systems (ACODS), Hyderabad, India, 18–22 February 2018; Elsevier: Hyderabad, India, 2018; pp. 474–479. [CrossRef]
30. Xu, J.J.; Dong, C.H.; Cheng, L. Deep Neural Network-Based Footprint Prediction and Attack Intention Inference of Hypersonic Glide Vehicles. *Mathematics* **2023**, *11*, 24. [CrossRef]
31. Zhang, J.B.; Xiong, J.J.; Li, L.Z.; Xi, Q.S.; Chen, X.; Li, F. Motion State Recognition and Trajectory Prediction of Hypersonic Glide Vehicle Based on Deep Learning. *IEEE Access* **2022**, *10*, 21095–21108. [CrossRef]
32. Zhao, K.; Cao, D.Q.; Huang, W.H. Maneuver control of the hypersonic gliding vehicle with a scissored pair of control moment gyros. *Sci. China-Technol. Sci.* **2018**, *61*, 1150–1160. [CrossRef]
33. Chai, R.; Tsourdos, A.; Savvaris, A.; Chai, S.; Xia, Y. Trajectory Planning for Hypersonic Reentry Vehicle Satisfying Deterministic and Probabilistic Constraints. *Acta Astronaut.* **2020**, *177*, 30–38. [CrossRef]