This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Physics Informed Deep Reinforcement Learning for Aircraft Conflict Resolution

Peng Zhao and Yongming Liu

***Abstract*— A novel method for aircraft conflict resolution in air traffic management (ATM) using physics informed deep reinforcement learning (RL) is proposed. The motivation is to integrate prior physics understanding and model in the learning algorithm to facilitate the optimal policy searching and to present human-explainable results for display and decision-making. First, the information of intruders' quantity, speeds, heading angles, and positions are integrated into an image using the solution space diagram (SSD), which is used in the ATM for conflict detection and mitigation. The SSD serves as the prior physics knowledge from the ATM domain which is the input features for learning. A convolution neural network is used with the SSD images for the deep reinforcement learning. Next, an actor-critic network is constructed to learn conflict resolution policy. Several numerical examples are used to illustrate the proposed methodology. Both discrete and continuous RL are explored using the proposed concept of physics informed learning. A detailed comparison and discussion of the proposed algorithm and classical RL-based conflict resolution is given. The proposed approach is able to handle arbitrary number of intruders and also shows faster convergence behavior due to the encoded prior physics understanding. In addition, the learned optimal policy is also beneficial for proper display to support decision-making. Several major conclusions and future work are presented based on the current investigation.**

***Index Terms*— Conflict resolution, deep reinforcement learning, air traffic management.**

I. Introduction

THE increased airspace density is putting a greater burden on pilots, controllers, and conflict resolution systems such as Center TRACON Automation System (CTAS) [1]. In addition, the recent trends for integration of unmanned aerial systems (UAS) into the national airspace system (NAS) will cause additional complexity for safety. Thus, efficient and scalable conflict resolution method is in critical need to accommodate this challenge. First, conflict resolution in a dense airspace requires the algorithm to be efficient, which makes real-time/in-time decision for safety assurance. Next, the scalability refers to that the algorithm should be robust to the number of air vehicles. This is especially true for UAS which usually includes a large number of unmanned aerial vehicles. In addition, the algorithm should have good resilience to adapt for changing environments and non-cooperative air vehicles. Most existing reinforcement learning methods have to learn from the environment with fixed number of intruders in which the dimension is known as a prior and cannot be changed. Retraining is generally required for scenarios with different number of intruders. As the number of intruders increases, the computational complexity increases quickly for training. To resolve this problem, an image-based deep reinforcement learning method is proposed for conflict resolution. The image-based deep learning largely improve the scalability issue as the algorithm can handle arbitrary number of aircraft as images rather than aircraft states are used. Another major difference is that the proposed model uses both the true observation data, including air vehicles' position, speed and heading angle (e.g., in classical reinforcement learning), and the imaginary data, which is informed by the prior physics knowledge for the conflict detection and resolution. The added data is not a simple manipulation of the real observations, but includes the physics modeling of the prediction and conflict detection using a specified safety bound model. The observations and the physics-informed knowledge are encoded in the images to facilitate the final policy searching. Thus, We name this approach as physics-informed deep reinforcement learning for aircraft conflict resolution hereafter.

Manuscript received May 13, 2020; revised October 11, 2020, January 22, 2021, and April 13, 2021; accepted April 27, 2021. This work was supported by the National Aeronautics and Space Administration (NASA) University Leadership Initiative Program (Project Officer: Dr. Anupa Bajwa, Principal Investigator: Dr. Yongming Liu) under Contract NNX17AJ86A. The Associate Editor for this article was D. Sun. *(Corresponding author: Yongming Liu.)*

The authors are with the School for Engineering of Matter, Transport & Energy, Arizona State University, Tempe, AZ 85281 USA (e-mail: pzhao28@asu.edu; yongming.liu@asu.edu).

Digital Object Identifier 10.1109/TITS.2021.3077572

A. Related Work

Many studies have been done for aircraft conflict resolution and a review can be found in [2]. One group of the most widely used methods is based on the advanced autoresolver investigated by NASA Ames Research Center. A logic of iterating the generated maneuvers to resolve the conflict is presented in [3]. A path-stretch algorithm is proposed in [4], [5] to avoid a conflict in its present course with a turn back to a downstream waypoint. An algorithm for computing horizontal resolution trajectories with a constraint on the bank angle is described in [6], where a set of maneuvers are generated to achieve or exceed the specified minimum separation. The Tactical Separation-Assisted Flight Environment (TSAFE) is designed [7] to alert air traffic controllers of imminent conflicts. An updated version [8] is presented to unify the resolution to three types of separation assurance problems that includes separation conflicts, arrival sequencing, and weather-cell avoidance. A similar encounter-based simulation architecture is presented in [9] for conflict detect-and-avoid modelling.

1558-0016 © 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on May 25,2021 at 13:07:53 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Another major category of methods is based on the force field (also known as voltage potential) methods. By analogy with the positively charged particles pushing away from each other, the authors [10] proposed an approach to calculate the paths of aircraft for conflict resolution. Based on closest point of approach, an improved force field method is proposed to increase the minimum distance between aircraft [11]. The force field method resolves multi-aircraft conflicts by calculating resolution maneuvers for each individual conflict pair separately [12]. To resolve the coordination issues in the unknown behavior in multi-aircraft conflicts, an optimized method was proposed in [13] from the display perspective to assist the operator for conflict resolution, which is subsequently known as solution space diagram (SSD) method and is further extended in [14], [15]. These methods can provide a visualized way for operators to read the conflict resolution method from display and assist them to perform maneuvers. However, lacking of dynamic behavior can cause more conflicts, large path deviation, and long conflict duration [12]. The image produced by the SSD method is implemented to model and predict controllers' deconflict decisions using a supervised learning algorithm [16]. The model trained by this method is individual and scenario sensitive, which is difficult to extend to a general situation. An Optimal Reciprocal Collision Avoidance (ORCA) algorithm [17] was proposed using geometric approach to resolve conflict in a decentralized way. This algorithm guarantees a conflict free solution for the agents with no speed constraint (e.g., robots in horizontal). The algorithm was modified to apply to speed constrained aircraft [18] without guaranteeing conflicts being resolved.

Reinforcement learning based methods for air traffic management have been investigated recently due to the rapid development in artificial intelligence. Many studies implemented reinforcement learning to avoid air traffic congestion [19]–[22]. The others focus on collision avoidance or conflict resolution. The collision avoidance problem is formulated as Markov Decision Process (MDP) problem in [23], where the generic MDP solvers are used to generate avoidance strategies. Then a dynamic programming approach is proposed in [24] to update the TCAS system that is broadly used in the NextGen (next generation air traffic management system). These methods only tackle the conflict problem between aircraft pair due to the dimensionality problem in multi-aircraft scenarios. To solve this problem, two approaches are proposed in [25] to compress the lookup table. Another method involves decomposing a large multi-agent Markov decision process and fusing their solutions [26]. However, these methods use decentralized training and centralized execution schemes, which still train the policy with single intruder. Thus, the trained policy cannot account for the risk from other intruders. These methods also cannot be extended to continuous action due to the space for heuristic search could be extremely large. To improve the responding to multi-threats, a method is proposed in [27] to improve methods developed in [26] by training corrections for the pair-wise policy. Many methods were also proposed where multiple intruders are directly used to train the policy [28]–[30]. In these methods, fixed number of the nearest intruders are observed to formulate the state for training. Ignorance of the other intruders may lead to risk condition especially in dense airspace. In addition, most of these methods are assumed that the intruders are flying at a fixed velocity and only the own aircraft takes actions to resolve conflicts, which is not practical. Multi-agent reinforcement learning that uses a centralized learning and decentralized execution scheme is proposed in [31]. The learned conflict resolution policies are not in conformance with the preference of air traffic controller in practice. To solve this problem, an interactive conflict solver that combine the controller's demonstrations and AI agent is proposed in [32]. An hierarchical deep reinforcement learning method is also implemented in [33] to solve the conflict resolution problem while performing the air traffic sequencing. However, most of these existing methods train policies with respect to specific number of intruders, which limit the scalibility for different scenarios in practice.

In recent years, many approaches have been developed for reinforcement learning with neural network function approximation. The leading contenders are deep Q-learning [34], "vanilla" policy gradient methods [35], trust region policy optimization (TRPO) [36], and proximal policy optimization (PPO) method [37]. The PPO method is developed on the basis of TRPO method and outperforms the other methods on efficiency and robustness. Therefore, we use PPO in this proposed method to train the deconflict policy.

B. Proposed Method

A novel method is proposed which integrates the observations and the physics-informed knowledge in images to learn aircraft conflict resolution policy. This is also beneficial for future visual decision-making support from the pilot's or controller's screen. Pixels of the screen carry the intruders' information including number of aircraft, positions, speeds and heading angles that is integrated by the solution space diagram (SSD) method. Most importantly, SSD provides prior physics understanding in identifying the potencial conflict and guidance for de-conflict action determination. This is intrinsically different than existing learning-based methods where direct physical observations are used. A convolution neural network (CNN) is used to extract hidden information from the SSD-based image for deep reinforcement learning to learn the resolution policy. The deep reinforcement learning is based on the Poximal Policy Optimization (PPO) algorithm [37].

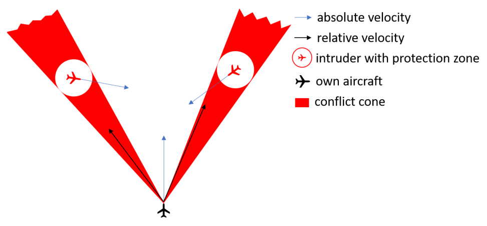
Most existing reinforcement learning methods for conflict resolution assume the fixed state dimension (such as ACAS X), which have to train the agent in an environment with certain number of intruders. The proposed method does not suffer from this problem as the SSD images can present arbitrary number of intruders. The proposed method provides a new physics-informed learning scheme for the aircraft to learn resolution from prior knowledge and current environment, which fuses different sources of information to assist the decision-making.

C. Limitation

No uncertainties are included in the current formulation and further study for probabilistic failure probability (i.e., risk)

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on May 25,2021 at 13:07:53 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.



(a)

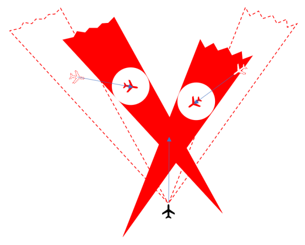

(b)

Fig. 1. Illustration of SSD method for conflict detection. (a) Conflicts are detected if the relative velocities within the conflict cone. (b) Conflicts are detected if the absolute velocity vector's end is in the conflict cones that are translated from (a) along their own corresponding intruder's velocity vector.

constraints needs to be included. Only horizontal deconflict is considered and combined vertical-horizontal deconflict needs further investigation.

D. Summary of Contributions

- A physics-informed learning method is proposed to integrate prior domain knowledge and advanced data analytic method for conflict resolution. The prior knowledge encoded in SSD as images to allow flexible training with reduced samples and faster convergence.
- The proposed learning-based method improves the limitation of classical SSD method in ATM domain, in which only static SSD is used for de-conflict and did not consider the dynamic environment changes.
- A meta-control logic is used to add the flight intent information to ensure the small deviation from the original flight plan while performing the conflict resolution.

II. Resolution Policy Learning for Single Aircraft

A. Prior Domain Knowledge

Solution space diagram (SSD) method was first proposed as an alternative metric to predict workload for an air traffic controller [38], and further extended as a visual aid for possible airborne separation task on navigational display in the cockpit [39].

The SSD method as a conflict resolution method is briefly illustrated in Fig. 1a and Fig. 1b. As shown in Fig. 1a, the circle around an intruder represents the protection zone whose radius is the minimum separation between aircrafts. Loss of separation leads to a conflict. The conflict cone is constructed by two lines that is from the own aircraft and is tangent to the protection zone. It is easy to demonstrate that if the vector of relative velocity of the own aircraft is within the conflict cone, conflict will occur if the corresponding intruder keeps its current velocity. To resolve this potential conflict, the relative velocity vector should be moved out of the conflict zone. However, in the multi-intruders scenario, it will be difficult to make an easy decision. This problem can be resolved by translating the relative velocity to the absolute velocity of the own aircraft, which is obtained by summing the relative velocity and the intruder's absolute velocity. The conflict cone is also translated by adding the corresponding intruder's velocity vector to the coordinates of each point, as illustrated in Fig. 1b. This translation moves the end of the relative velocity vector to the end point of the own aircraft's absolute velocity vector. Therefore, it indicates a potential conflict if the end point of the absolute velocity is within the conflict cone area. Then, the multi-aircraft conflicts resolution needs to move the end point of the own aircraft's absolute velocity vector out of the conflict cone. Different predefined rules leads to different deconflict velocity vector. For example, the conflict can be resolved by taking the shortest way out or by only changing heading angle [12].

In this paper, the SSD method is employed as an image generation model to encode the information of intruders into an image, rather than a conflict resolution method that directly resolves conflict. Generation of SSD image is not computational demanding. Thus, there is negligible impact on computational complexity of training process or operation.

B. Deep Reinforcement Learning

The deep reinforcement learning methods are categorized into two groups, value-based methods and policy-based methods. With value-based methods, the agent learns from the environment to maintain an estimate of the optimal action-value function, from which the optimal policy is obtained. Policy-based methods directly learn the optimal policy without having to maintain a separate value function estimate. The policy-based methods estimate the policy gradient as [37], [40]:

$$\hat{g} = \hat{\mathbb{E}}_t[\nabla_\theta \log \pi_\theta(a_t|s_t)\hat{A}_t] \tag{1}$$

where π_θ is a stochastic policy and $\hat{A}_t$ is an estimator of the advantage function at time t. The expectation $\hat{\mathbb{E}}_t$ indicates the empirical average over a finite batch of samples. The gradient is rewritten as the following after importance sampling,

$$\hat{g} = \nabla_\theta \hat{\mathbb{E}}_t\left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta old}(a_t|s_t)}\hat{A}_t\right] \tag{2}$$

where the θ_{old} is the vector of policy parameters before update. The estimator $\hat{g}$ is obtained by differentiating the objective:

$$L^{CPI}(\theta) = \hat{\mathbb{E}}_t\left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta old}(a_t|s_t)}\hat{A}_t\right] \tag{3}$$

where the superscript refers to conservative policy iteration [41]. To keep the policy from an excessively large policy

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on May 25,2021 at 13:07:53 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

update, PPO method uses the following objective function:

$$L^{CLIP}(\theta)=\hat{\mathbb{E}}_t\left[\min(k_t(\theta)\hat{A}_t, clip(k_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)\right] \quad (4)$$

where $k_t(\theta)$ denotes the probability ratio $k_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta old}(a_t|s_t)}$, ϵ is a hyperparameter.

This objective is further augmented by adding an entropy bonus and value function error terms, which is maximized in each iteration:

$$L_t^{CLIP+VF+S}(\theta)=\hat{\mathbb{E}}_t\left[L_t^{CLIP}(\theta)-c_1L_t^{VF}(\theta)+c_2S[\pi_\theta](s_t)\right] \quad (5)$$

where c_1,c_2 are coefficients, and S denotes an entropy bonus, and L_t^{VF} is a squared-error loss $(V_\theta(s_t) - V_t^{targ})^2$

C. Action Space

Main maneuvers for conflict resolution in horizontal deconflict includes heading angle change and speed control. Altitude changing is also an efficient way to resolve conflict, but is not considered in this study. In practice, combination of the heading angle change and speed control is not implemented by the pilot and controller due to high workload. A fully automated system may use both these maneuvers simultaneously to make the resolution more efficient. In this work, we first investigate the heading angle maneuvers and speed control separately (i.e., related to the current ATM practice), and then combine them together (i.e., related to a fully automated system in the future).

1) Discrete Action Space: Discrete action space refers to a set $\mathbb{D}$ that contains finite number of fixed actions. For example, $\mathbb{D} = [-10°, 0°, 10°]$ represents using heading angle change of $\pm10°$ or $0°$ to resolve conflicts. Note that the action space is not confined to heading angle change, speed change, and simultaneous speed change and heading change are also applicable as long as the actions are finite and fixed.

2) Continuous Action Space: Continuous action space refers to a continuous space from which the agent chooses deconflict actions. In this work, two kinds of continuous space are considered, which are heading angle and speed. For example, the heading angle action space $(-\pi/2, \pi/2)$ allows the agent choose any heading angle change from $-\pi/2$ to $\pi/2$, the speed control action space $(-50knots, 50knots)$ implies the agent can choose any value from $-50knots$ to $50knots$ to add to its current speed.

D. Reward Function

The reward function of the proposed method is formulated to balance the objectives of safety assurance and minimized disruption. Safety is the most critical consideration in the proposed method. The reward with respect to safety is expressed as:

$$R_c = \begin{cases} -1 & \text{if conflict} \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

where conflict is defined as the condition that the distance between two aircraft is less than a specific value (5 nautical miles as specified by International Civil Aviation Organization (ICAO)).

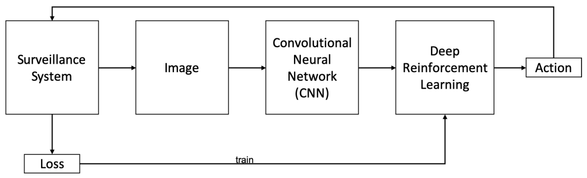

Surveillance System
Image
Convolutional Neural Network (CNN)
Deep Reinforcement Learning
Action
Loss
train

Fig. 2. Flow chart of the proposed conflict resolution algorithm.

The heading-change action will cause the aircraft to deviate from its original flight plan. A term of the reward function to minimize disruption is defined as:

$$R_h = 0.001\cos(\theta) \quad (7)$$

where θ is the angle between the action direction and the intention direction. This term rewards the action following the intention and penalizes the action of deviation.

The speed change action makes no spatial deviation from the original flight plan, but deviating from the cruise speed is fuel consuming. A term of the reward function is defined to reward moderate speed change:

$$R_s = 0.001e^{-(\frac{v-v_0}{v_u-v_l})^2} \quad (8)$$

where v is the current speed, v_0 is the original speed, v_l and v_u are the lower bound and upper bound of the speed respectively.

The total reward is expressed as:

$$R = \begin{cases} R_c + R_h & \text{heading change} \\ R_c + R_s & \text{speed change} \\ R_c + R_d + R_s & \text{simultaneous heading and speed} \end{cases} \quad (9)$$

E. Network Architecture

The network architecture for learning is illustrated in Fig. 2. Information from the surveillance systems is processed first to generate images based on SSD method. Next, convolutional neural network is used to extract features and output the state vector for learning. The deep reinforcement learning process is performed to select an action and interact with the environment. Then a loss calculated by the PPO method is used to train the learning network. This process is iterated until the result converges to a predefined level.

1) Architecture of Convolutional Neural Network (CNN): The SSD-based image contains information of number, speeds, headings of the intruders, and indications for potential conflict identification criteria, which are critical for conflict resolution. These information should be extracted for the deep reinforcement learning algorithm to learn the resolution policy. Two hidden layers using convolutional neural network are used for this purpose in this work, which is illustrated in Fig. 3. The main parameters are listed in Table I. As shown in Fig. 3, the input image is 80 × 80 pixels, which are produced by the SSD method. The first layer output 4 × 38 × 38 pixels, followed by the second layer that outputs 16 × 9 × 9 pixels. The final output is a 1 × 1296 vector as the input for the

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on May 25,2021 at 13:07:53 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

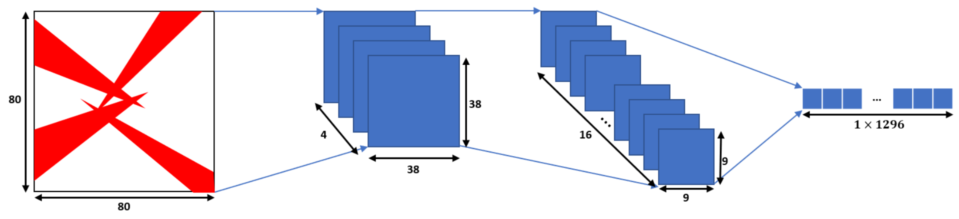
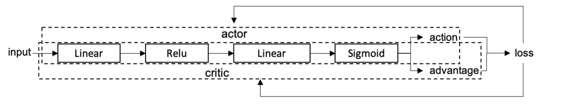


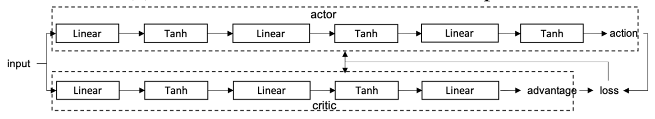
Fig. 3. Illustration of two layers convolution neural network for features extraction. The output of this neural network is an vector which is the state of the following reinforcement learning.

TABLE I
HYPERPARAMETERS FOR CNN

	Layer 1	Layer 2
Input Depth	1	4
Output Depth	4	16
Kernel Size	6	6
Stride	2	4
Padding	0	0
Bias	False	True



(a) Architecture for discrete action space



(b) Architecture for continuous action space

Fig. 4. Architectures of deep reinforcement learning neural network.

following neural network for deep reinforcement learning. Through this process, the vector contains all the intruders' information extracted by the neural network.

2) *Architecture of Deep Reinforcement Learning Neural Network:* The output of CNN is the input of the deep reinforcement learning neural network that consists of several layers and activations. We denote the output of CNN at time t as s_t that is the state for reinforcement learning. The deep reinforcement learning neural network takes the state as input and output the policy $\pi(a_t|s_t;\theta_{\pi_{old}})$. The action is taken following the policy and the environment will return a reward r_t and the next-state s_{t+1}. A T time steps trajectory is collected in this manner, which is denoted as $\{(s_t, r_t), (s_{t+1}, r_{t+1}), \cdots, (s_T, r_T)\}$. Fig. 4 presents two architectures of neural network used to train policies of discrete action and continuous action, respectively. Fig. 4a shows the neural network for discrete action, where a hidden layer is connected and followed by the output layer that is the probabilities of the actions. Each node of the output layer corresponds to an action, associated with probability of this action being selected. The probability distribution of the actions is discrete, and the action is selected by sampling from the probability distribution. The selected action is used for the agent to interact with the environment and a loss is calculated using the PPO method described previously. Note that even though there are not two separate networks used to respectively estimate the value function and policy in the network for discrete action space, it also belongs to actor-critic style (see [40]).

The training network architecture for continuous action space is shown in Fig. 4b, where the output of the CNN network is connected to an actor-critic network. The actor-critic network is widely used for continuous action space, where the critic network is trained to approximate the value function while the actor network is trained to approximate the policy. The activation (tanh) in the output layer of the actor network is used to generate an continuous action. The actor network is trained using Monte-Carlo method that has no bias but suffers from large variance. The critic network is trained using boot-strap (also named as "temporal difference") method that has small variance but introduces bias [42]. Combining these two networks makes the training process to be more efficient and robust.

The advantage function used for both the two architectures is expressed as [40]:

$$\hat{A}_t = \sum_{i=t}^{T-1} \gamma^{i-t} r_i + \gamma^{T-t} V(s_T) - V(s_t) \tag{10}$$

where $V(s)$ is the state value.

F. Meta Control Logic for Returning to Intention

In this work, we apply a meta control logic to let the aircraft return to its own waypoint after deconflict. The logic is performed on the basis of conflict detection. As we have mentioned previously, it indicates a potential conflict if the end point of the absolute velocity of the own aircraft is in the conflict cone area. In this situation, the aircraft must perform the deconflict behavior to avoid the risk. Therefore, the meta control logic is designed as: aircraft keeps heading to its intention until the end point of its velocity vector inserts the conflict cone, then a resolution velocity is selected. At each time step of resolution, the intention velocity is monitored. The intention velocity refers to a vector pointing to the next waypoint from the current position. The magnitude of the intention velocity would be the same with the current speed. Once the end point of the intention velocity moves out of the conflict cone, the aircraft will choose the intention velocity to return to its original flight plan. This process is illustrated by the following pseudo-code and Fig. 5, where the green arrow represents the intention velocity and the black arrow denotes the resolution velocity.

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on May 25,2021 at 13:07:53 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Algorithm 1 Meta Controller

```
Result: $v$: velocity of next step
while run do
    $obs.$ ← observe airspace
    $states \leftarrow SSDProcesse(obs.)$
    velocity for resolution: $v_r \leftarrow Policy(states)$
    velocity for return: $v_i$ ← intention velocity
    conflict detection for $v_i$
    if conflict then
        $v \leftarrow v_r$ (black arrow in Fig. 5)
    else
        $v \leftarrow v_i$ (green arrow in Fig. 5)
    end
end
```

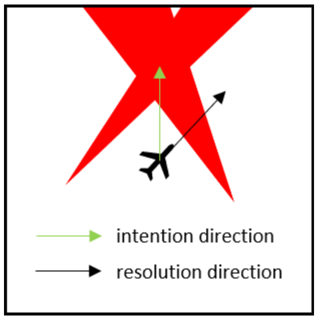


Fig. 5. Illustration of meta control logic. The resolution action is selected if a potential conflict is detected (the end of the green vector is within the red area).

The above process is initiated when the en-route phase starts. The algorithm will perform in the entire en-route phase to detect and resolve potential conflict. SSD image can be used to identify whether a potential conflict exists in the current direction. Once detected, the conflict resolution process will be triggered. If the conflict is resolved, the aircraft will head forward its next waypoint following the above returning process. This horizontal conflict resolution terminates when the en-route phase ends. If the algorithm fails and a conflict is happening in a short time (e.g., 2 minutes or less), the TCAS system or other conflict/collision avoidance system performing in vertical dimension will take charge of decision making and the algorithm will be suspended.

In the process of resolution, the controller will not know in advance when the resolution maneuvers will be completed. The conflict resolution using reinforcement learning (rather than optimization) is a one-step method that gives a strategy with respect to the current state. The benefit is that it can cope with uncertainty of the environment. Therefore, the method will not predict when the resolution maneuvers will be completed. The aircraft will inform the controller when no conflict is detected after resolution.

III. Multi-Agent Coordination

Cooperatively resolving conflicts by all affected aircraft is safer and more practical. This cooperation can be arranged by a computation center (ground center or one of the effected aircraft). When conflicts are detected, all affected aircraft need to communicate with the center to upload their conflict information. The center will determine a minimum-pilot-disturbance cooperative strategy. This strategy can be beneficial as it reduces the usage of limited communication resource, reduces the pilot disturbance, and reduces the risk due to the uncertainty of cooperative operations.

Aircraft uses SSD method to detect potential conflict intuitively by identifying whether the end point of its absolute velocity is within the other aircraft's conflict cone. Once a conflict is detected, all the aircraft that involve in the conflict and the associated conflict should be coupled together, for example, aircraft a_1 detects a potential conflict with a_2, while a_3 also detects a conflict with a_2, then all the three aircraft should be requested to cooperate. Once all affected aircraft have been identified, the computation center will determine the aircraft and the order to perform conflict resolution according to the following criteria.

1) : Aircraft may not respond to the cooperation request due to communication failures or human errors. In this situation, the other aircraft is responsible to avoid conflict with the non-cooperative one. Therefore, the aircraft that has conflict with the non-cooperative aircraft must take actions.

2) : If the conflict is detected between an aircraft pair, which is the most common case, the one that has a greater q-value should take actions. In reinforcement learning, the q-value is the estimated reward of the future in the episode if the agent takes a specific action at the current state. In the studied scenario, according to the definition of reward function in the previous section, the q-value here reflects the possibility of successful conflict resolution. Therefore, it is safer to require the aircraft with greater q-value to take actions.

3) : Fig. 6 gives two typical scenarios of multiple aircraft conflict. In the Fig. 6a, there exists at least one aircraft subset in which each of the aircraft has no conflict with others. The subsets can be easily identified by iterating all the affected aircraft to find their no-conflict sets. Among all the subsets, the one that contains maximum number of aircraft is used for efficiency. For example, in Fig. 6a, the maximum no-conflict set is $\{a_1, a_2\}$. Following this, aircraft in this subset will keep their original flight plans and the other takes actions for conflict resolution. If more than one aircraft are needed to take actions, they should act cooperatively. A time interval is assigned for each aircraft to take actions in rotation. The increasing order of the q-value should be used since smaller q-value indicates more risk of conflict according to the cost function, and thus the immediate actions should be taken. While the rotating aircraft is taking its action, the other aircraft need to keep their current velocity until their turns for taking actions. This method resolves conflicts with minimum pilot disturbance.

Fig. 6b presents an example where no-conflict subset does not exist, although this scenario rarely happen in practice. In this condition, only one aircraft is allowed to keep its original flight plan. The largest and/or heaviest aircraft has the priority to stay in its flight plan since high action cost and low agility. The other aircraft need to take actions in a time interval alternatively, using the same manner as we have demonstrated in the above paragraph.

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on May 25,2021 at 13:07:53 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

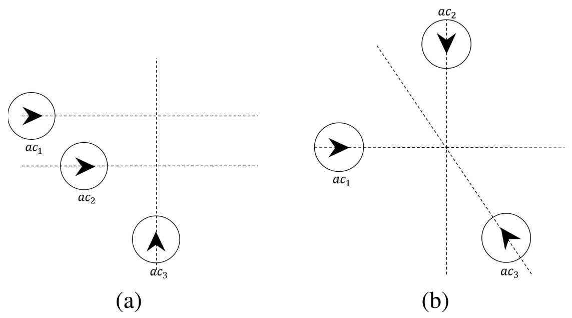


Fig. 6. Two typical scenarios of multiple aircraft conflict.

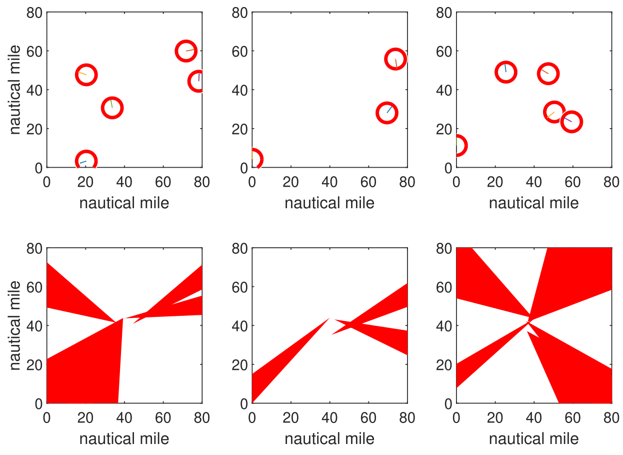


Fig. 7. Samples of environment image showing intruders' positions, protection zones and SSD representation.

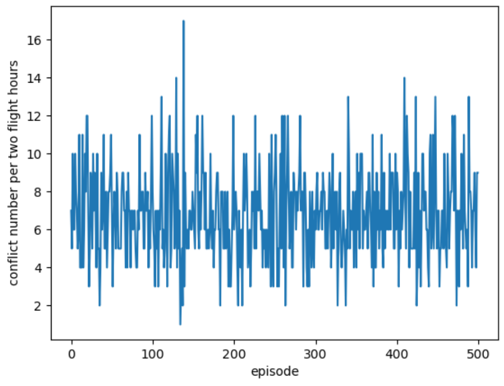


Fig. 8. Number of conflicts if no resolution is performed in the constructed environment.

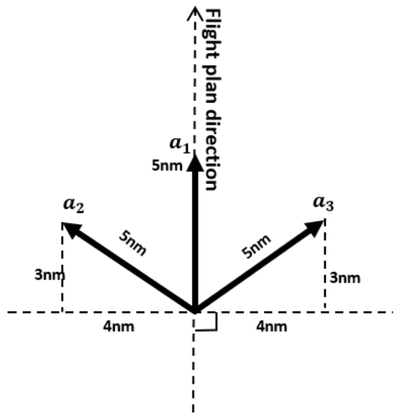


Fig. 9. Discrete action space.

Short interval can decreases conflict risk since each aircraft can have many opportunities to adjust its actions to avoid conflict. But the short interval also increases the pilot workload.

IV. Numerical Examples

A. Training Data

The environment is developed using Pygame. A 80×80 nautical miles surveillance area is simulated where the own aircraft is located at the center. Every $0 \sim 150$ seconds, an intruder is created on a random location of the border and then flies along a line with random direction. The intruder is removed if it flies beyond the scope. The intruder's speed is randomly set as $400 \sim 500$ knots according to the typical cruise speed of commercial aircraft. A time step in the simulation corresponds to 40 seconds in practice, and an episode consists of 200 time steps. Environment reset is not needed at the beginning of each episode, due to the randomly set of intruders. The training is terminated after 500 episodes that is sufficient for convergence.

As presented in the first row of Fig. 7, three images are sampled from the dynamic environment to illustrate the simulated airspace. The second row images are the corresponding SSD airspace representations which are the inputs of the training network.

Fig. 8 gives the conflict number per two flight hours if the own aircraft keeps its original flight plan in this airspace.

B. Action Space

Several categories of action space are studied, including discrete heading angle action space, continuous heading angle action space, speed action space, and simultaneous heading change and speed control action space, which are depicted in the following:

1) Discrete Heading Angle Action Space: It is easy for pilots or controllers to perform conflict resolution using actions selected from discrete action space. An example action space is shown in Fig. 9, which includes three actions a_1, a_2, a_3. These actions are to move 5 nautical miles over a time interval in different directions illustrated in Fig. 9. The three actions are used because the current deconflict system also uses few number of actions and we want to be consistent with the current practice. More actions will lead to increased action space and training time.

2) Continuous Heading Angle Action Space: Continuous action space leads to a more smooth resolution path. Though the continuous action is not practical for the pilots or controllers to operate manually, the automated operation of manned or unmanned aircraft can implement the continuous action space to resolve conflict. In this work, we use the continuous action space where the selected direction should have an angle with the original direction no greater than $\frac{\pi}{2}$. An additional reward (Eq. 7) is added to make sure the deviation between the selected direction and the intention

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on May 25,2021 at 13:07:53 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

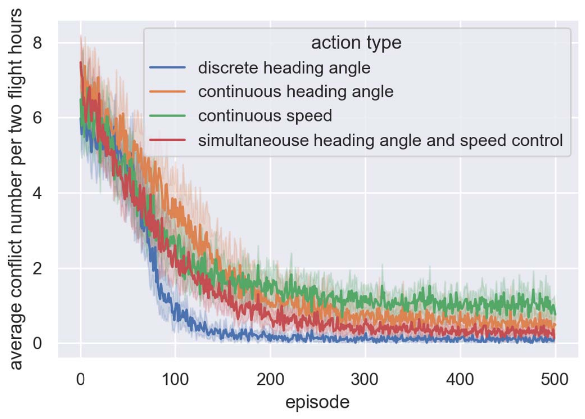


Fig. 10. Convergence speed comparison between different action types.

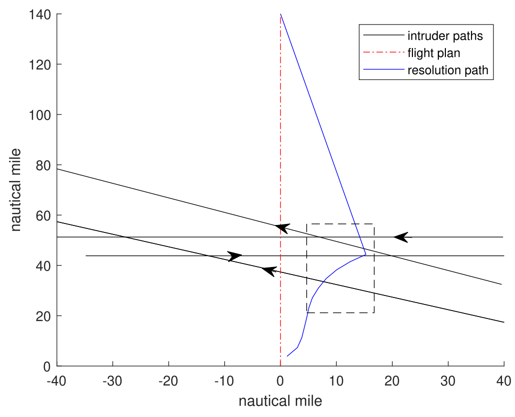


Fig. 11. Paths of the intruders and the own aircraft's resolution.

direction is small if not necessary. The reward decreases with the angle between the two directions increases, and should be much smaller than 1 to prevent affecting the deconflict policy learning. This reward can keep the aircraft from deviating its flight plan significantly when performing conflict resolution.

3) Speed Control: Another conflict resolution method is to change the aircraft's speed while keeping its original direction. In this example we use continuous action space, which is 300 $\sim$ 600 knots (the intruders' speeds are randomly chosen from 400 $\sim$ 500 knots).

4) Simultaneous Heading Angle and Speed Control: Combining the heading angle changing and speed control extends the action space from 1 dimension to 2 dimensions, which provides more options for conflict resolution.

The results of convergence behavior for training the neural network using the above four action space categories are presented in Fig. 10. Each of the curve converges to a low level, indicating the average conflict number per two flight hours reduces in the training process. The discrete action space performs best among the four categories, but limited actions restrict feasibility of the agent. Continuous action space allows the agent performing conflict resolution in a broader range. The agent using speed control action space keeps the original direction and avoids conflicts by changing speed. However, both the continuous action space and speed control performs lower comparing with the discrete action space. The action space of simultaneous heading change and speed control shows better performance than any single control method and closes to the level of discrete action space.

C. Path of Deconflict and Returning

Implementation in practice needs the algorithm to include the aircraft's intention. The aircraft should return to its original flight plan when conflicts are resolved. To achieve this objective, a high level control mechanism is used. This mechanism consists of three part: conflict detection, conflict resolution, and flight plan return. In practice, aircraft always performs conflict detection. Once conflicts are detected, the conflict resolution function is triggered. If no conflict is detected when resolution is completed, the flight-plan-return logic is implemented.

Fig. 11 presents an example, where four intruders have potential conflicts with the own aircraft in its flight plan. The blue line presents the path of the aircraft to resolve these conflicts using continuous heading change action. The dynamic resolution processes in the dashed rectangle are shown in Fig. 12, where the blue circle area represents the intruder's protection zone, the green arrow indicates the velocity for intention, and the black arrow is the velocity for resolution. The agent selects the resolution action from time step 1 to time step 8. At time steps 9 to 12, the potential conflicts have been resolved, thus the return action is selected. Note that the continuous changes in heading is difficult for the controllers to observe and high workload for the pilots to perform. The continuous action space may be applicable for the unmanned aircraft system traffic management (UTM) or the future fully automated systems.

D. Multi-Agent Coordination

The proposed method can be extended to coordinate multiple aircraft in conflict resolution. This section uses three multiple aircraft encounter examples illustrated in Fig. 13, Fig. 14, and Fig. 15 to verify the effectiveness of the proposed method. The 3 aircraft encounter example is also used to compare the proposed method with the existing CSORCA method and value-based learning method.

As mentioned in Section III, in most conditions, some of the aircraft keep their original flight plans according to the minimum pilot disturbance policy, while the others have the responsibilities for conflict resolution. In the 3-aircraft example, at least two of the three aircraft have to take actions in order to resolve conflicts. The aircraft 3 ("ac3") is selected to keep its original flight plan according to the minimum pilot disturbance policy. The other two aircraft take discrete deconflict actions described in Section II. Aircraft 1 and aircraft 2 detect the potential conflict using the method mentioned in the previous part. Once the potential conflict has been resolved, the return action should be taken to return to its intention. In this process, the aircraft 1 takes action first while aircraft 2 keeps its current velocity. In the second time step, the aircraft 2 takes action and the aircraft 1 keeps its current velocity. This process is going on until the potential conflicts

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on May 25,2021 at 13:07:53 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

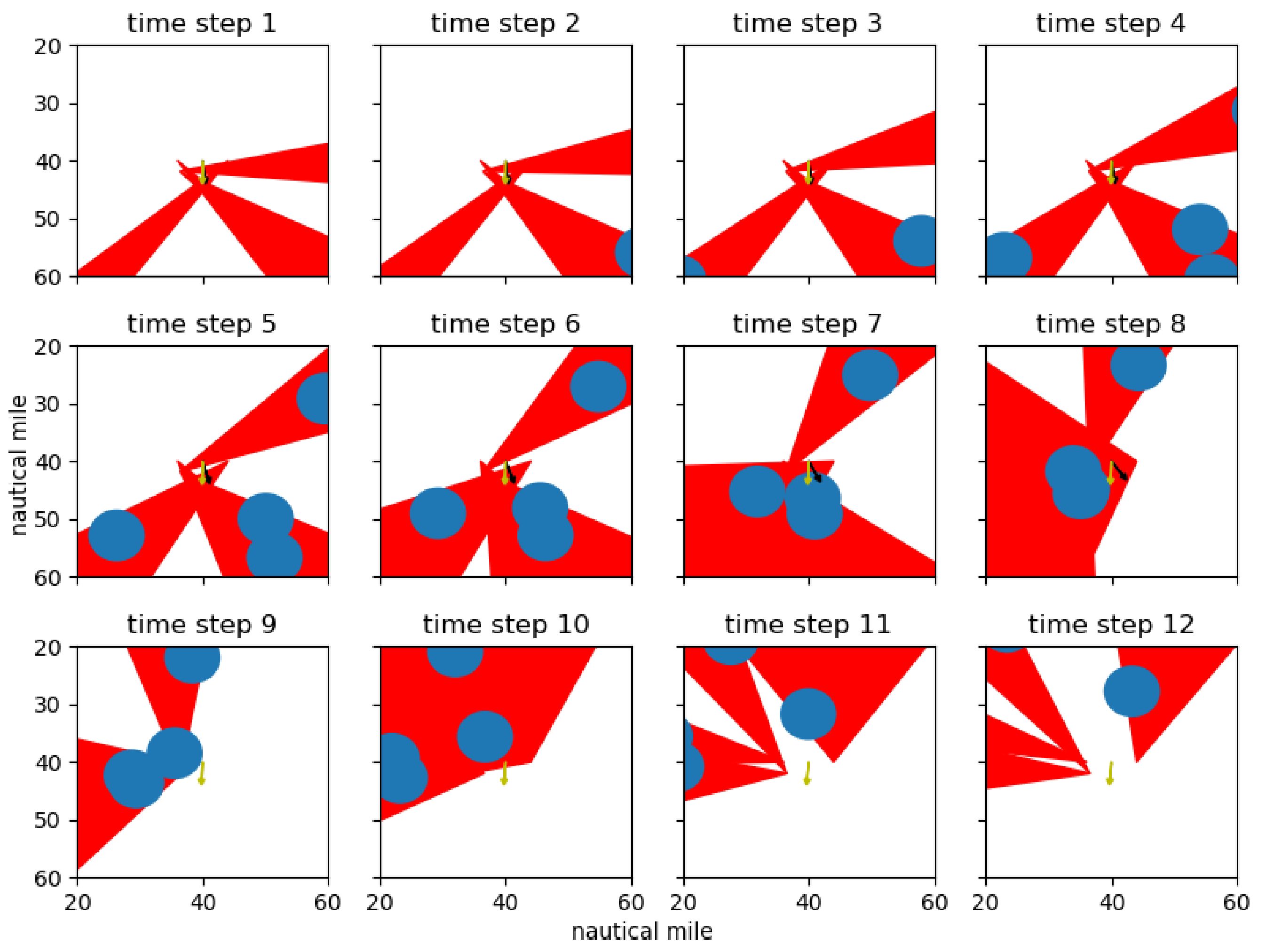


Fig. 12. An example of the conflict resolution process.

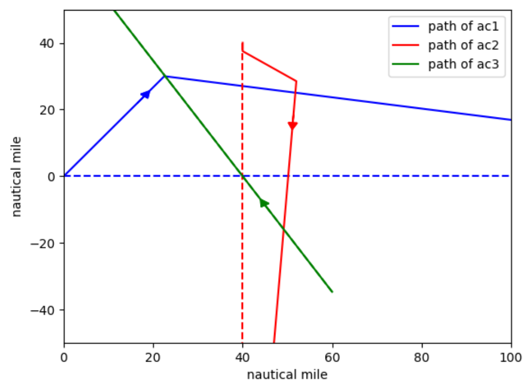


Fig. 13. Result of 3 aircraft deconflicting paths using the proposed method.

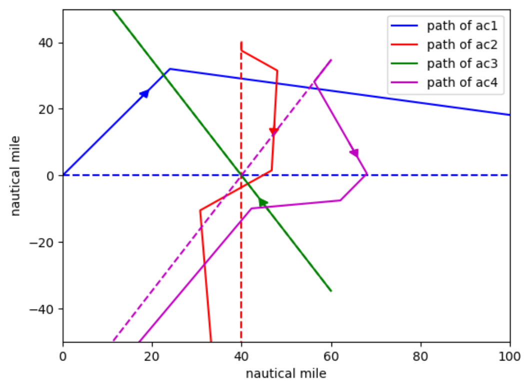


Fig. 14. Result of 4 aircraft deconflicting paths using the proposed method.

are resolved. Then the return actions are taken. Fig. 13 shows the paths generated by the deconflict and return actions. Fig. 14 shows the deconflict paths of the 4-aircraft encounter scenario, where the aircraft 3 is selected as the non-disturbance aircraft. Fig. 15 presents the result of 5-aircraft scenario in which the aircraft 3 and aircraft 5 are requested to keep their original plans according to the rule of minimum-pilot-disturbance.

As a comparison, the value-based method developed in [26] was tested in the same scenario with the 3-aircraft example. The result of deconflict paths are presented in Fig. 16. Although the max-sum utility function combines multiple affected aircraft to perform conflict resolution cooperatively, the deconflict policy is trained in pair-wise scenario. This leads the agent to make decisions mainly depending on the proximity distance between aircraft rather than the entire

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on May 25,2021 at 13:07:53 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

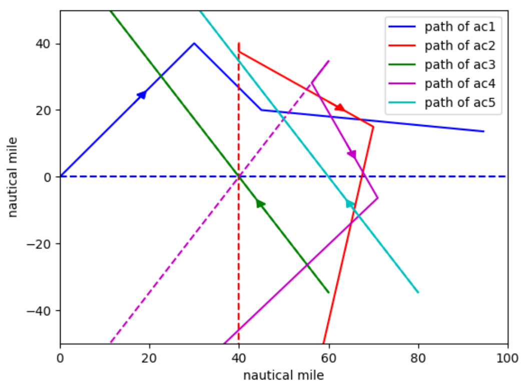


Fig. 15. Result of 5 aircraft deconflicting paths using the proposed method.

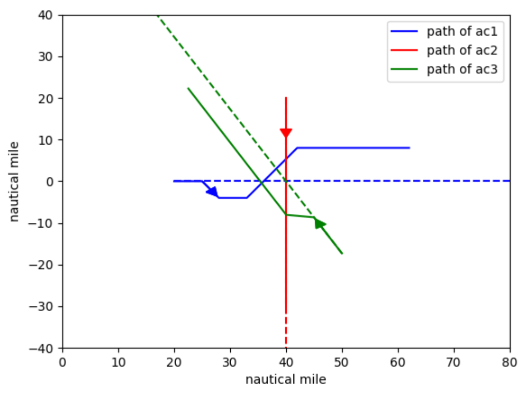


Fig. 16. Result of deconflict path of the value-based method.

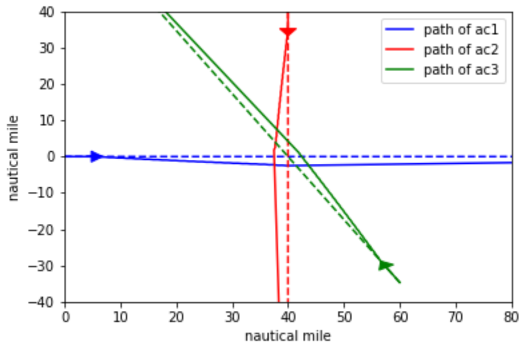


Fig. 17. Result of deconflict path of the CSORCA method.

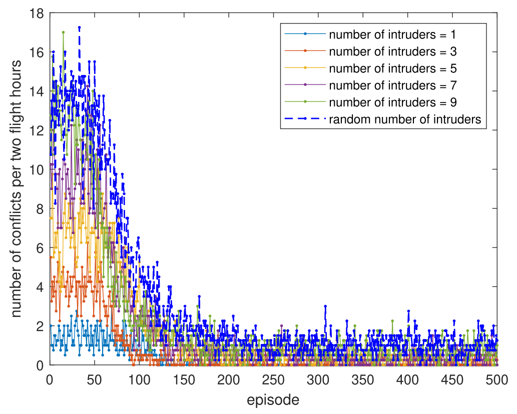


Fig. 18. Sample efficiency with different number of intruders.

airspace awareness. This explains the aircraft 1 (blue curve) turns right first and then turns left to avoid a new conflict.

Comparing with the value-based method, the proposed method has several other benefits. First, the intention information is accounted for. The proposed method guides the aircraft to return to the intention if no conflict is detected in the future path. Second, the proposed method is also applicable to continuous action space. The value-based method, which uses search heuristic method to find deconflict policy, cannot search the optimal policy in continuous action space. Third, the proposed method selects a subset of aircraft to perform deconflict to reduce disturbance of pilots, which is not applicable in the value-based method.

The constant speed optimal reciprocal collision avoidance (CSORCA) method developed in [18] is also implemented using the same scenario. The deconflict path of this method is presented in Fig. 17. It can be seen that the deconflict paths do not deviate much from the original flight plans, which is beneficial for fuel saving. However, as mentioned previously, this method cannot guarantee to find a resolution, causing a relative high probability of conflict in implementation [18]. This will be discussed in the next section. Another drawback is that the strong assumption is enforced that all the affected aircraft must take the same conflict resolution strategy. If one aircraft does not follow this assumption in practice, the ORCA might lead to a dangerous situation.

V. Discussion

A. Scalability

The scalability of the proposed method is studied using the example of discrete actions. Fig. 18 shows the sample efficiency of the proposed method by comparing with the convergences of learning using different number of intruders. The policy with fixed number of intruders is trained by keeping a specific number of intruders in the view (80 $\times$ 80 nautical miles). The policy with random number of intruders is trained by randomly generating a number of aircraft (i.e., 1-5 aircraft) in random time steps (i.e., 1-5 time steps). This setting increases the randomness of aircraft number in the view. It is observed from Fig. 18 that the algorithm all converges around 150 episode despite different initial conflict rates. The initial conflict rates typically increases as the aircraft density increases. This demonstrates that samples for the training are efficient irrespective of the number of intruders, which is due to that the learning is based on the SSD image rather than raw state vector. The dimension of the raw state vector depends on the number of intruders, but the image keeps the same size irrespective of the number of intruders.

The trained policy with random number of intruders is tested in several scenarios where the average number of intruders ranges from 1 to 20. The average number of conflicts per two hours flight presented in Fig. 19, which indicates the scalibility of the learned policy. The policy is trained using cases having 1-5 aircrafts and it is shown that the conflict rate with the

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on May 25,2021 at 13:07:53 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

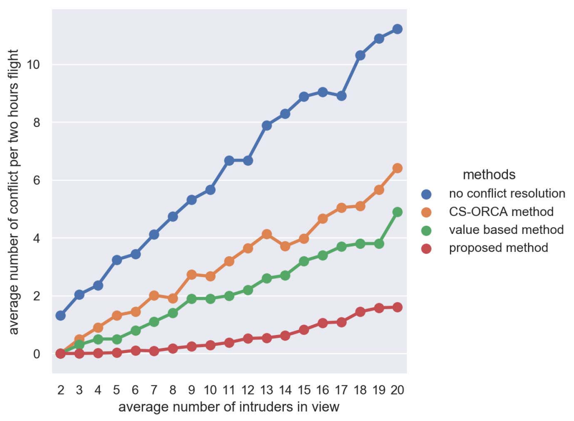


Fig. 19. Performance of resolving different number of intruders using the same policy trained by random number of intruders, in comparison with no conflict resolution, ORCA method and value based method.

proposed physics-informed learning is almost zero (see green line in Fig. 19). The trained policy is used for testing cases where the number of intruders increases to 20. We observe that the conflict rates are close to zero until 15 and increases slightly for 20 intruders. If the proposed physics-informed learning is not used, it is observed that the conflict rate increases significantly. The increase is almost linear, which is the theoretical increase rate with no resolution action. The value-based method and constant speed ORCA (CS-ORCA) method are compared with the proposed method and the results are shown in Fig. 19. As shown in the figure, the proposed method outperforms the value-based method and CS-ORCA method as the number of intruders increases. It should be noted that the increase of conflict rate for large number of intruders is partially due to the significantly increased airspace density and the resolution may not always be successful. Other deconflict actions, such as vertical conflict resolution or collision avoidance should be used to ensure safety. The current field of view is 80-by-80 nautical miles and 20 aircraft in this view is a very high density in the current NAS and is not likely to happen in real life.

B. Comparison With Inputting Adjacent Frames

The proposed method uses the preprocessed frame where the information (speed, direction and position information broadcasted by ADS-B systems) of intruders are encoded by SSD-based method. These information can also be represented by multiple adjacent frames. Feeding multiple adjacent frames into the neural network so that it can detect motion is a typical process in reinforcement learning trained by images. For example, this procedure has been commonly used for reinforcement learning to play Atari Games [34], [43]. In this section, we compare the performances resulted from the two different inputs, i.e., single frame with the information encoded by SSD-based method and raw multiple adjacent frames. As illustrated in Fig. 20a, the red circles directly provide the positions and protection zones of the intruders, and their velocities are obtained by differentiating two adjacent frames. The Fig. 20b presents the corresponding SSD-based image.

We compare the performances of the two different inputs (i.e., multiple adjacent frames and the SSD-based single frame) using the discrete action example described previously. In the training process, an aircraft enters the surveillance area (80×80 nautical miles area around the own aircraft) at a random time step from 1 to 5, and pops out if it flies beyond the scope of the area, this ensures the policy is trained by random number of aircraft. Two adjacent frames (the previous frame and the current frame) are feed into two channels of the convolutional neural network. The CNN extracts the dynamic information from the two adjacent frames for training. The proposed method input the SSD-based single frame. The other settings and hyperparameters are all identical for the two methods for a fair comparison. Thus, the only difference between the two methods is on the inclusion of prior physics knowledge in the SSD-based image. The convergence behaviors of the two methods are presented in Fig. 21. Several experiments are performed to plot the mean and confident interval. A much better performance of convergence is observed using the proposed method, including faster convergence speed, lower mean value and smaller deviation.The difference of the number of conflicts at the end of training may appear to be very small in Fig. 21 (e.g., 0.2 vs. 0.01). However, this difference represents a more than magnitude change of probability of conflict which is significant from a safety point of view. It appears that the embeded prior physics knowledge using SSD enhances the performance in the current investigation.

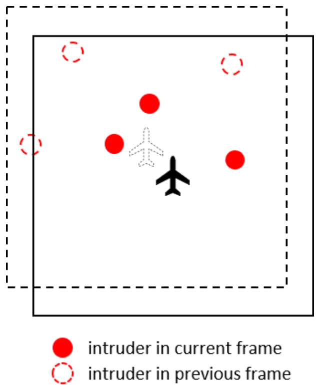
Lower performance of using adjacent frames than SSD encoded image as input might due to the synchronization problem between the state and action. As has been mentioned, the velocities of intruders extracted by differentiating the previous frame and the current frame are the average velocities of the last interval. Therefore, the environment state used for making decision includes the intruders' average velocities of the last interval. Changes in the current velocities are not respected in the current decision, which might mislead the training process and increase conflict risk. Contrarily, the SSD-based single frame encodes the real-time velocity and position information (broadcasted by ADS-B system) of each intruder, ensuring the own aircraft responds to the current velocities of intruders. This explains the lag of convergence of the method using adjacent frames as input. In practice, this situation might be improved by reducing the time interval of sampling images, but noises will dominate the observations (position and velocity) if the interval is too small and thus compromise the safety. In addition, high frequency of taking actions will increase workload of pilots and controllers.

C. Look-Ahead Time and Action Frequency

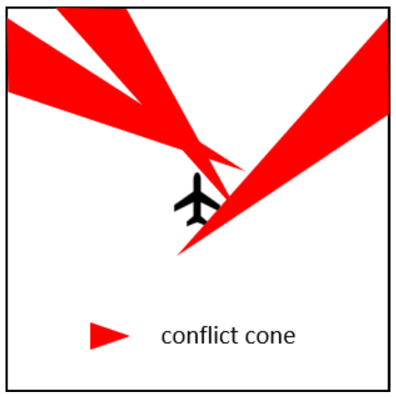
The proposed method focuses on short-term conflict resolution. A 80×80 pixels image is used in all the above experiments to represent 80×80 nautical miles airspace centered by the own aircraft, which is projected time of 5 minutes for the own aircraft to fly out the area if the speed is assumed to be 450 knots. The conflict detection and resolution are triggered once the intruders enter this airspace, the look-ahead time in this situation is 5 minutes. The Fig. 22 presents convergence

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on May 25,2021 at 13:07:53 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.



(a) 2-frames input



(b) SSD-based single frame input

Fig. 20. Illustration of information encoding manipulation of reinforcement learning method using multiple frames versus the SSD based single frame. The former uses at least two adjacent frames to encode dynamic information while the SSD based method only uses one frame to include the dynamic.

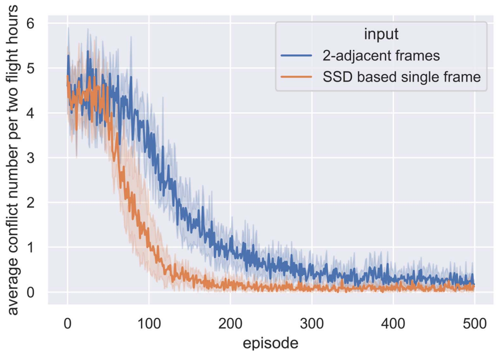


Fig. 21. Convergence speed comparison between different inputs.

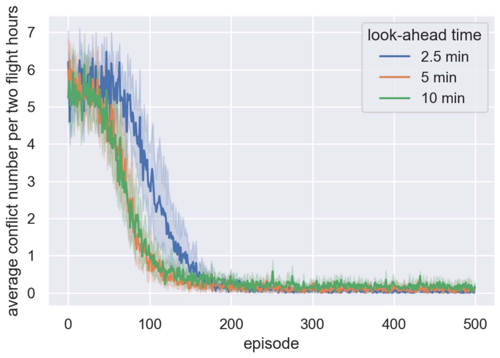


Fig. 22. Convergence speed comparison between different look-ahead times.

performance using different look-ahead times of 2.5 minutes (corresponds to 40 × 40 nautical miles surveillance area), 5 minutes and 10 minutes (corresponds to 160 × 160 nautical miles surveillance area), indicating that the algorithm can learn deconflict policies for all the 3 look-ahead times. The 2.5 minutes look-ahead time converges with a very little delay than the others in the training, this might be explained by less choices of deconflict strategies existing in the case of short-term look-ahead time and thus is more difficult to learn. The 5 minutes and 10 minutes look-ahead time cases behave the same in training. This might because the strategy spaces are sufficient for the agents to learn. In practice, potential conflict with too short look-ahead time will be taken charge of by airborne TCAS system, which is beyond the scope of this research. A long look-ahead time deconflict strategy can increase the false alarm rate, which should be avoided to reduce the pilot's workload.

The time steps of all the above experiments are assumed to be 40 seconds, i.e., the time interval between two adjacent actions is 40 seconds. Convergence behaviors of different time steps (20 seconds, 40 seconds, and 60 seconds) are studied in Fig. 23, which shows no significant difference. It should be noted that a short time-step will increase the pilot or controller's workload, and a long time-step will increase the risk due to uncertainties in each time-step. Therefore, a proper time step should be set in practice.

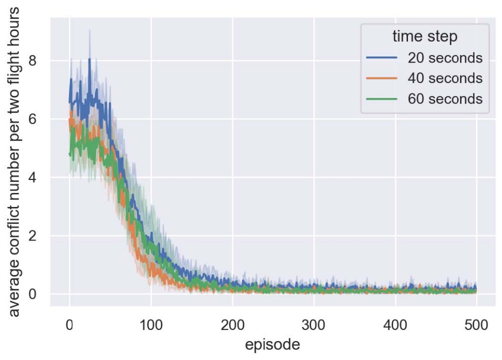


Fig. 23. Convergence speed comparison between different time intervals.

D. Action Smoothness

The reinforcement learning based controller may exhibit non-smoothed behaviors. Oscillatory actions have been observed in many tasks [44]–[48], especially in continuous control implementation. Action smoothness is a challenge but critical work for transferring RL-based controller from simulation to real-world application. Attempts to solve this problem mainly focuse on reward-engineering to induce the desired behavior [45], [46]. In our work, we also use the reward-engineering method (see Eq.7) to induce smooth actions. It is recognized that reward-engineering for action

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on May 25,2021 at 13:07:53 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

smoothness needs careful parameters tuning and provides no guarantees for the desired behavior [44]. Therefore, action smoothness would be an import future work for the implementation of this work.

VI. Conclusion and Future Work

A novel physics-informed deep reinforcement learning for conflict resolution in air traffic management is proposed in this study. Based on the solution space diagram (SSD) method, intruders' information at each time step are integrated into an image. This image is used by an convolutional network to extract the information that is the input into the deep reinforcement learning network for learning the resolution policy. Several numerical examples are studied, including both heading angle changing and speed control. Extensive discussion for the scalability, convergence rate, single/multi agent learning, and flight intent effect are presented. Several conclusions can be drawn based on the proposed study:

- This work provides a mechanism for the aircraft to learn conflict resolution policy from the simulation environment. The physics-informed deep learning largely improve the scalability issue as the algorithm can handle arbitrary number of aircraft as images rather than aircraft states are used.
- The discrete action space performs better than the continuous action space on convergence speed and robustness.
- The current study shows that the heading angle changing method performs better than the speed control method on convergence speed and robustness.
- Conflict resolution that simultaneously using heading angle and speed control actions has better performance than any single control action.
- The embedded physics knowledge shows significantly improvement of scalability and only needs a small number of intruders to train the policy. Traditional reinforcement learning method fails to resolve the conflict when using the policy to a larger number of intruders.

The future work is listed as the following:

- Action smoothness would be an import future work for the implementation of this work.
- Detailed investigation of scalable multi-agent reinforcement learning (MARL) for large number of aircraft conflict resolution would be a future work.
- Further study for probabilistic failure probability constraints needs to be conducted.

3-dimensional

- deconflict strategy should be studied in the future.

Acknowledgment

The works related to this research were performed at the Prognostic Analysis and Reliability Assessment Laboratory, Arizona State University. The support is gratefully acknowledged.

References

[1] H. Erzberger, "CTAS: Computer intelligence for air traffic control in the terminal area," NASA Ames Res. Center, Moffett Field, CA, USA, Tech. Rep. 92N33080, 1992.

[2] J. K. Kuchar and L. C. Yang, "A review of conflict detection and resolution modeling methods," *IEEE Trans. Intell. Transp. Syst.*, vol. 1, no. 4, pp. 179–189, Dec. 2000.

[3] H. Erzberger, "Automated conflict resolution for air traffic control," NASA Ames Res. Center, Moffett Field, CA, USA, Tech. Rep. 20050242942, 2005.

[4] R. Bach, Y.-C. Chu, and H. Erzberger, "A path-stretch algorithm for conflict resolution," NASA-Ames Research Center, Moffett Field, CA, USA, Tech. Rep. NASA/CR-2009-214574, 2009.

[5] R. Bach, C. Farrell, and H. Erzberger, "An algorithm for level-aircraft conflict resolution," NASA, Washington, DC, USA, Tech. Rep. CR-2009-214573, 2009.

[6] H. Erzberger and K. Heere, "Algorithm and operational concept for resolving short-range conflicts," *Proc. Inst. Mech. Eng., G, J. Aerosp. Eng.*, vol. 224, no. 2, pp. 225–243, Feb. 2010.

[7] R. A. Paielli, H. Erzberger, D. Chiu, and K. R. Heere, "Tactical conflict alerting aid for air traffic controllers," *J. Guid., Control, Dyn.*, vol. 32, no. 1, pp. 184–193, Jan. 2009.

[8] H. Erzberger, T. A. Lauderdale, and Y.-C. Chu, "Automated conflict resolution, arrival management and weather avoidance for ATM," in *Proc. 27th Int. Congr. Aeronaut. Sci.*, 2010, pp. 1–20.

[9] M. Refai, M. Abramson, S. Lee, and G. Wu, "Encounter-based simulation architecture for detect-and-avoid modeling," in *Proc. AIAA Scitech Forum*, Jan. 2019, p. 1476.

[10] M. S. Eby, "A self-organizational approach for resolving air traffic conflicts," *Lincoln Lab. J.*, vol. 7, no. 2, p. 239–254, Sep. 1995.

[11] K. Zeghal, "A review of different approaches based on force fields for airborne conflict resolution," in *Proc. Guid., Navigat., Control Conf. Exhibit*, Boston, MA, USA, Aug. 1998, p. 4240.

[12] S. Balasooriyan, "Multi-aircraft conflict resolution using velocity obstacles," M.S. thesis, Delft Univ. Technol., Delft, The Netherlands, 2017. [Online]. Available: https://repository.tudelft.nl/islandora/object/uuid%3Acf361aff-a7ec-444c-940a-d711e076d108

[13] J. Ellerbroek, M. Visser, S. B. J. van Dam, M. Mulder, and M. M. van Paassen, "Design of an airborne three-dimensional separation assistance display," *IEEE Trans. Syst., Man, Cybern., A, Syst. Humans*, vol. 41, no. 5, pp. 863–875, Sep. 2011.

[14] S. M. Abdul Rahman, M. Mulder, and R. van Paassen, "Using the solution space diagram in measuring the effect of sector complexity during merging scenarios," in *Proc. AIAA Guid., Navigat., Control Conf.*, Aug. 2011, p. 6693, doi: 10.2514/6.2011-6693.

[15] J. G. d'Engelbronner, C. Borst, J. Ellerbroek, M. M. van Paassen, and M. Mulder, "Solution-space–based analysis of dynamic air traffic controller workload," *J. Aircr.*, vol. 52, no. 4, pp. 1146–1160, 2015, doi: 10.2514/1.C032847.

[16] S. J. van Rooijen, J. Ellerbroek, C. Borst, and E. van Kampen, "Toward individual-sensitive automation for air traffic control using convolutional neural networks," *J. Air Transp.*, vol. 28, no. 3, pp. 105–113, Jul. 2020, doi: 10.2514/1.D0180.

[17] J. van den Berg, S. J. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *Robotics Research*, C. Pradalier, R. Siegwart, and G. Hirzinger, Eds. Berlin, Germany: Springer, 2011, pp. 3–19.

[18] N. Durand, "Constant speed optimal reciprocal collision avoidance," *Transp. Res. C, Emerg. Technol.*, vol. 96, pp. 366–379, Nov. 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0968090X18314232

[19] S. Ghosh, S. Laguna, S. H. Lim, L. Wynter, and H. Poonawala, "A deep ensemble multi-agent reinforcement learning approach for air traffic control," 2020, *arXiv:2004.01387*. [Online]. Available: https://arxiv.org/abs/2004.01387

[20] A. M. F. Crespo, L. Weigang, and A. G. De Barros, "Reinforcement learning agents to tactical air traffic flow management," *Int. J. Aviat. Manage.*, vol. 1, no. 3, pp. 145–161, 2012.

[21] T. Kravaris *et al.*, "Resolving congestions in the air traffic management domain via multiagent reinforcement learning methods," 2019, *arXiv:1912.06860*. [Online]. Available: https://arxiv.org/abs/1912.06860

[22] K. Malialis, S. Devlin, and D. Kudenko, "Resource abstraction for reinforcement learning in multiagent congestion problems," 2019, *arXiv:1903.05431*. [Online]. Available: https://arxiv.org/abs/1903.05431

[23] S. Temizer, M. Kochenderfer, L. Kaelbling, T. Lozano-Perez, and J. Kuchar, "Collision avoidance for unmanned aircraft using Markov decision processes," in *Proc. AIAA Guid., Navigat., Control Conf.*, Toronto, ON, Canada, Aug. 2010, p. 8040, doi: 10.2514/6.2010-8040.

[24] M. J. K. J. P. Chryssanthacopoulos, "Robust airborne collision avoidance through dynamic programming," Lincoln Lab., Massachusetts Inst. Technol., Cambridge, MA, USA, Tech. Rep. ATC-371, Jan. 2011.

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on May 25,2021 at 13:07:53 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

[25] K. D. Julian, J. Lopez, J. S. Brush, M. P. Owen, and M. J. Kochenderfer, "Policy compression for aircraft collision avoidance systems," in *Proc. IEEE/AIAA 35th Digit. Avionics Syst. Conf. (DASC)*, Sep. 2016, pp. 1–10.

[26] H. Y. Ong and M. J. Kochenderfer, "Markov decision process-based distributed conflict resolution for drone air traffic management," *J. Guid., Control, Dyn.*, vol. 40, no. 1, pp. 69–80, Jan. 2017, doi: 10.2514/1.G001822.

[27] S. Li, M. Egorov, and M. Kochenderfer, "Optimizing collision avoidance in dense airspace using deep reinforcement learning," 2019, *arXiv:1912.10146*. [Online]. Available: https://arxiv.org/abs/1912.10146

[28] M. Ribeiro, J. Ellerbroek, and J. Hoekstra, "Improvement of conflict detection and resolution at high densities through reinforcement learning," in *Proc. ICRAT*, 2020, pp. 1–4.

[29] H. Wen, H. Li, Z. Wang, X. Hou, and K. He, "Application of DDPG-based collision avoidance algorithm in air traffic control," in *Proc. 12th Int. Symp. Comput. Intell. Design (ISCID)*, vol. 1, Dec. 2019, pp. 130–133.

[30] Z. Wang, H. Li, J. Wang, and F. Shen, "Deep reinforcement learning based conflict detection and resolution in air traffic control," *IET Intell. Transp. Syst.*, vol. 13, no. 6, pp. 1041–1047, Jun. 2019. [Online]. Available: https://digital-library.theiet.org/content/journals/10.1049/iet-its.2018.5357

[31] M. Brittain and P. Wei, "Autonomous air traffic controller: A deep multi-agent reinforcement learning approach," 2019, *arXiv:1905.01303*. [Online]. Available: https://arxiv.org/abs/1905.01303

[32] P. N. Tran, D.-T. Pham, S. K. Goh, S. Alam, and V. Duong, "An interactive conflict solver for learning air traffic conflict resolutions," *J. Aerosp. Inf. Syst.*, vol. 17, no. 6, pp. 271–277, 2020, doi: 10.2514/1.I010807.

[33] M. Britaain and P. Wei, "Autonomous aircraft sequencing and separation with hierarchical deep reinforcement learning," in *Proc. Int. Conf. Res. Air Transp.*, Catalonia, Spain, Jun. 26–29, 2018.

[34] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[35] V. Mnih *et al.*, "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1928–1937.

[36] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel, "Trust region policy optimization," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1889–1897.

[37] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*. [Online]. Available: https://arxiv.org/abs/1707.06347

[38] P. Hermes, M. Mulder, M. M. van Paassen, J. H. L. Boering, and H. Huisman, "Solution-space-based complexity analysis of the difficulty of aircraft merging tasks," *J. Aircr.*, vol. 46, no. 6, pp. 1995–2015, Nov. 2009, doi: 10.2514/1.42886.

[39] J. Ellerbroek, K. C. R. Brantegem, M. M. van Paassen, and M. Mulder, "Design of a coplanar airborne separation display," *IEEE Trans. Human-Mach. Syst.*, vol. 43, no. 3, pp. 277–289, May 2013.

[40] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," 2018, *arXiv:1506.02438*. [Online]. Available: https://arxiv.org/pdf/1506.02438.pdf

[41] S. Kakade and J. Langford, "Approximately optimal approximate reinforcement learning," in *Proc. 19th Int. Conf. Mach. Learn.*, 2002, pp. 267–274.

[42] D. Silver, R. S. Sutton, and M. Müller, "Temporal-difference search in computer go," *Mach. Learn.*, vol. 87, no. 2, pp. 183–219, May 2012, doi: 10.1007/s10994-012-5280-0.

[43] V. Mnih *et al.*, "Playing atari with deep reinforcement learning," 2013, *arXiv:1312.5602*. [Online]. Available: https://arxiv.org/abs/1312.5602

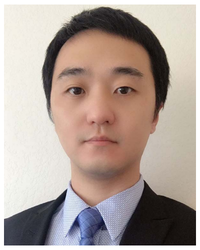
[44] S. Mysore, B. Mabsout, R. Mancuso, and K. Saenko, "Regularizing action policies for smooth control with reinforcement learning," 2020, *arXiv:2012.06644*. [Online]. Available: https://arxiv.org/abs/2012.06644

[45] A. R. Mahmood, D. Korenkevych, G. Vasan, W. Ma, and J. Bergstra, "Benchmarking reinforcement learning algorithms on real-world robots," in *Proc. Conf. Robot Learn.*, 2018, pp. 561–591.

[46] W. Koch, "Flight controller synthesis via deep reinforcement learning," 2019, *arXiv:1909.06493*. [Online]. Available: https://arxiv.org/abs/1909.06493

[47] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel, "Benchmarking deep reinforcement learning for continuous control," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1329–1338.

[48] F. Sadeghi, A. Toshev, E. Jang, and S. Levine, "Sim2real view invariant visual servoing by recurrent control," 2017, *arXiv:1712.07642*. [Online]. Available: https://arxiv.org/abs/1712.07642



Peng Zhao received the Ph.D. degree from the School of Electronic and Information Engineering, Beihang University, China, working on integrity monitoring of satellite navigation systems application in civil aircraft. He is currently a Post-Doctoral Researcher with the School for Engineering of Matter, Transport & Energy, Arizona State University. His research interests include information fusion for real-time national air transportation system prognostics under uncertainty, collision avoidance and resolution methods, and unmanned air traffic management systems.

Yongming Liu is currently a Professor of aerospace and mechanical engineering with the School for Engineering of Matter, Transport & Energy, Arizona State University. He heads the Prognostic Analysis and Reliability Assessment Laboratory (PARA). His research interests include fatigue and fracture of engineering materials and structures, probabilistic computational mechanics, risk assessment and management to multi-physics damage modeling and structural durability, multi-scale uncertainty quantification and propagation, and imaging-based experimental testing, diagnostics, and prognostics.

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on May 25,2021 at 13:07:53 UTC from IEEE Xplore. Restrictions apply.