IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 9 July 2022, accepted 17 August 2022, date of publication 26 August 2022, date of current version 1 September 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3201962

RESEARCH ARTICLE

Obstacle Avoidance for UAS in Continuous Action Space Using Deep Reinforcement Learning

JUEMING HU[1], XUXI YANG[2], WEICHANG WANG[3], PENG WEI[4], (Member, IEEE), LEI YING[5], (Fellow, IEEE), AND YONGMING LIU[1]

[1]School for Engineering of Matter, Transport and Energy, Arizona State University, Tempe, AZ 85287, USA
[2]Department of Aerospace Engineering, Iowa State University, Ames, IA 50011, USA
[3]Department of Computer and Energy Engineering, Arizona State University, Tempe, AZ 85287, USA
[4]Department of Mechanical and Aerospace Engineering, The George Washington University, Washington, DC 20052, USA
[5]Engineering and Computer Science Department, University of Michigan, Ann Arbor, Ann Arbor, MI 48109, USA

Corresponding author: Yongming Liu (yongming.liu@asu.edu)

This work was supported by the National Aeronautics and Space Administration (NASA) University Leadership Initiative Program under Contract NNX17AJ86A.

ABSTRACT Obstacle avoidance for small unmanned aircraft is vital for the safety of future urban air mobility (UAM) and Unmanned Aircraft System (UAS) Traffic Management (UTM). There are a variety of techniques for real-time robust drone guidance, but numerous of them solve in discretized airspace and control, which would require an additional path smoothing step to provide flexible commands for UAS. To deliver safe and computationally efficient guidance for UAS operations, we explore the use of a deep reinforcement learning algorithm based on Proximal Policy Optimization (PPO) to lead autonomous UAS to their destinations while bypassing obstacles through continuous control. The proposed scenario state representation and reward function can map the continuous state space to continuous control for both heading angle and speed. To verify the effectiveness of the proposed learning framework, we conducted numerical experiments with static and moving obstacles. Uncertainties associated with the environments and safety operation bounds are investigated in detail. Results show that the proposed model is able to provide accurate and robust guidance and resolve conflict with a success rate of over 99%.

INDEX TERMS Continuous control, deep reinforcement learning, UAS obstacle avoidance, uncertainty.

I. INTRODUCTION

Ranging from delivery drones to electrical vertical take-off and landing (eVTOL) passenger aircraft, modern unmanned aircraft systems (UAS) are able to accomplish a variety of tasks efficiently, including weather monitoring, surveillance, goods delivery, disaster relief, search and rescue, traffic monitoring, air transportation, and videography [1], [2]. Urban air mobility (UAM) is likely to occur in urban regions near airports or buildings. Thus, it is expected that UAS can use onboard detect and avoid systems to keep away from other traffic, terrain, hazardous weather, and natural and man-made obstacles without constant human intervention [2].

Numerous mathematical models of aircraft conflict resolution have been developed in the literature. Research efforts can be divided into centralized approaches and decentralized algorithms. Centralized algorithms can be based on semidefinite programming [3], nonlinear programming [4], [5], mixed-integer linear programming [6], [7], [8], [9], mixed-integer quadratic programming [10], sequential convex programming [11], [12], second-order cone programming [13], evolutionary techniques [14], [15], and particle swarm optimization [16]. In these centralized methods, the aim is to achieve the global optimum for all aircraft. The challenge is that with the increase in the number of aircraft, the computation cost of these methods grows exponentially. Among the decentralized methods, Markov Decision Process (MDP) allows the formalization of the conflict resolution problem. Reinforcement Learning (RL) is a promising solution to aircraft traffic management, but mostly uses traditional algorithms [17]. The collision avoidance systems (CAS) problem is modeled as a partially observable Markov Decision Process (POMDP) in the next-generation airborne collision avoidance system (ACAS X) and has been extended

The associate editor coordinating the review of this manuscript and approving it for publication was Zheng Chen.

to unmanned aircraft, named ACAS Xu [18]. Both ACAS X and ACAS Xu use Dynamic Programming (DP) to determine the expected cost of each action [19], [20]. Decomposition methods and DP are combined for optimizing collision avoidance with multiple threats in [21]. The traditional RL algorithms require a fine discretization scheme of state space and finite action space. Discretization potentially reduces safety by adding discretization errors and cannot provide flexible maneuver guidance for UAS. In addition, discretizing large airspace implies a high computation demand and can be time-consuming. Tree search based algorithms [22], [23] are also applied to CAS problems using MDP formulation which does not involve state discretization. But they typically require high onboard computation time to accommodate the continuous state space.

The large and continuous state and action spaces pose challenges to conflict resolution problems using reinforcement learning. Recently, Deep Reinforcement Learning (DRL) is studied to solve this challenge by applying the deep neural network to approximate the cost and the optimal policy functions. Development of DRL algorithms, e.g., Policy Gradient [24], Deep Q-Networks (DQN) [25], Double DQN [26], Dueling DQN [27], Deep Deterministic Policy Gradient (DDPG) [28], Asynchronous Advantage Actor-Critic (A3C) [29], and Proximal Policy Optimization (PPO) [30] has increased the potential of automation. In [31], DQN is used to perform corrections for an existing collision avoidance strategy to take into consideration the dense airspace. In [32], the feasibility of using algorithms based on DQN in UAV obstacle avoidance is verified. It is concluded in [33] that DQN can outperform value iteration both in terms of evaluation performance and solution speed when solving a UAV collision avoidance problem. The performance of the DQN algorithm in avoiding single aircraft to multiple aircraft is investigated in [34]. A novel deep multi-agent reinforcement learning framework based on PPO is proposed in [35] to detect and avoid conflicts among multiple aircraft in a high-density and dynamic sector under uncertainty. The DRL work mentioned above is in continuous state and discrete action space.

There has been less progress on utilizing DRL to solve UAS conflict resolution with continuous control. Authors in [36] proposed an approach inspired by Deep Q-learning and Deep Deterministic Policy Gradient algorithms and it can avoid conflicts, with a success rate of more than 81%, in the presence of traffic and various degrees of uncertainties. A generic framework that combines an autonomous obstacle detection module and an actor-critic based reinforcement learning (RL) module is developed to provide reactive obstacle avoidance policy for a UAV in [37]. Experiments in [30] test the performance of PPO on a set of benchmark tasks, including Atari game playing and simulated robotic locomotion, and show that PPO outperforms other online policy gradient approaches. PPO appears to be a favorable balance between sample complexity, simplicity, and wall-time. Thus, a PPO-based conflict resolution model is valuable for UAS traffic management, which is the major motivation of this study.

To the best of the authors' knowledge, this is the first study to develop a DRL approach based on PPO algorithm to enable the UAS to successfully navigate in continuous state and action spaces. Calculating in continuous space has the advantage of eliminating the requirement to discretize the state space or smooth the results for postprocessing. The proposed model with the optimal policy after offline training can be utilized for UAS real-time online trajectory planning. The main contributions of this paper are as follows:

- A PPO-based framework has been proposed for UAS to avoid both static and moving obstacles in continuous state and action spaces.
- A novel scenario state representation and reward function are developed and can effectively map the environment to maneuvers. The trained model can generate continuous heading angle commands and speed commands.
- We have tested the effectiveness of the proposed learning framework in the environment with static obstacles, the environment with static obstacles and UAS position uncertainty, and the deterministic and stochastic environments with moving obstacles. Results show that the proposed model can emit accurate and robust guidance and resolve conflict with a success rate of over 99%.

The remainder of this paper is organized as follows. The backgrounds of Markov Decision Process and Deep Reinforcement Learning is introduced in Section II. Section III presents the model formalization of UAS conflict resolution in continuous action space as a Markov Decision Process. In Section IV, we conduct numerical experiments to evaluate the effectiveness of the proposed approach to allow the UAS to learn to prevent conflicts. Section V summarizes this paper.

II. BACKGROUND

This section provides an overview of Markov Decision Process (MDP) and Deep Reinforcement Learning (DRL).

A. MARKOV DECISION PROCESS (MDP)

Since the 1950s, MDPs [38] have been thoroughly explored and benefitted a variety of fields [39], [40], [41], including robotics [42], [43], automatic control [44], manufacturing, and economics. In an MDP, an agent selects an available action a depending on the current state s at each time step. Then, the agent would move to a new state s' with certain transition probability at the next time step and receives a corresponding reward r.

Mathematically, an MDP is defined by a tuple, $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, r, \gamma)$, where $\mathcal{S}$ is a finite set of states consisting of all the possible states, $\mathcal{A}$ is the action space consisting of all the actions that the agent can select, $\mathcal{T}(s_{t+1}|s_t, a_t)$ is the transition function describing the probability of transferring to state s_{t+1} under the current state s_t and action a_t, $r(s_t, a_t, s_{t+1})$ is the reward function which determines the immediate reward (or expected immediate reward) for

transitioning to state s' by taking the action a at state s. For simplicity of notation, we denote r_t as the immediate reward at the time step t and R_t as the total discounted reward from time-step t forwards, and $\gamma \in [0, 1]$ is the discount factor which decides the importance of future rewards. With a small γ (close to 0), the agent cares for the immediate reward. With a large γ (close to 1), the agent considers future rewards with greater weight. Also, using a discount factor below 1 helps the cumulative reward to converge.

The policy π in an MDP can be stochastic or deterministic. A stochastic policy is a mapping from the state to a probability distribution over actions,

$$\pi : \mathcal{S} \rightarrow \text{Prob}(\mathcal{A}) \tag{1}$$

The output of a deterministic policy is one specific action,

$$\pi : \mathcal{S} \rightarrow \mathcal{A} \tag{2}$$

An MDP aims to find an optimal policy π^* that, starting from any initial state, maximizes the expected cumulative rewards:

$$\begin{aligned} \pi^* &= \arg\max_{\pi} \mathbb{E}[R_t|\pi] \\ &= \arg\max_{\pi} \mathbb{E}[\sum_{t'=t}^{T} \gamma^{t'-t} r(s_{t'}, a_{t'})|\pi] \end{aligned} \tag{3}$$

where T is the final time step when the agent reaches the terminal state.

An MDP uses Q-function and value function to evaluate the performance of the learning agent. The optimal Q-function $Q^*(s, a)$ denotes the maximum expected cumulative reward that the agent obtains as it starts from state s, selects action a and subsequently behaves optimally. Thus, $Q^*(s, a)$ indicates how good a certain action a is for the agent when it is at state s. The optimal value function $V^*(s)$ represents the maximum expected cumulative reward if the agent starts from state s, which is calculated as the maximum of $Q^*(s, a)$ over the available actions at state a:

$$V^*(s) = \max_a Q^*(s, a), \quad \forall s \in \mathcal{S} \tag{4}$$

B. DEEP REINFORCEMENT LEARNING

Reinforcement learning [45] is a promising technology to resolve the MDP problem. By leveraging deep learning, deep Reinforcement Learning (DRL) has recently shown great success in games such as Atari games [46], [47], game of GO [48], and Starcraft [49]. Deep reinforcement learning can generally be categorized into value-based learning [28], [46] and policy-based algorithm [29], [30], [50]. A policy-based DRL algorithm is utilized in this study to produce policies for the learning agent. The policy-based algorithm is more effective than a value-based DRL algorithm in high-dimensional or continuous action spaces and has the capacity to learn stochastic policies, which is beneficial when the environment contains uncertainty.

In the policy-based RL algorithm, function approximator such as the neural network is utilized to approximate the policy $\pi(s)$. The approximated policy takes the current state as the input and outputs the probability of each action for discrete action space or a distribution over actions for continuous action space. After each episode τ, the parameters of the function approximation are updated towards larger cumulative reward using gradient ascent:

$$\nabla_\theta J(\pi_\theta) = \mathop{\mathbb{E}}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) R_t \right] \tag{5}$$

where $J(\pi_\theta)$ represents the expected cumulative reward of the policy π parameterized by θ, $\pi_\theta(a_t|s_t)$ represents the probability of choosing action a_t at state s_t, and R_t represents the cumulative reward received by the agent for the remaining time steps in the episode τ. Eq. (5) aims to decrease the probability of selecting an action that results in a lower return and increase the probability of the actions that results in a higher reward. However, one challenge is that the variance of the cumulative reward could be high, which makes it hard for the algorithm to converge. To address this challenge, researchers develops an actor-critic algorithm [45] where a critic function, $V(s_t)$, is proposed for the approximation of the value function $V^{\pi_\theta}(s_t)$ under policy π_θ. After subtracting the critic function $V(s_t)$, the expectation of the gradient remains unchanged, and the variance of the policy is decreased significantly:

$$\nabla_\theta J(\pi_\theta) = \mathop{\mathbb{E}}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) (R_t - V(s_t)) \right] \tag{6}$$

where $V(s_t)$ is also updated to approximate $V^{\pi_\theta}(s_t)$.

III. MARKOV DECISION PROCESS FORMULATION

In this study, the UAS and intruders are treated as a point mass. The developed conflict resolution algorithm aims to discover the shortest route for a UAS to reach its destination without conflicts with other UAS and static obstacles. Leading the UAS to its goal is a discrete-time stochastic control process and is well suited to the formalization of a Markov Decision Process (MDP). The MDP formulation is introduced in the following subsections by describing its state representation, action space, terminal state, and reward function. The MDP formulation is developed in this work and the authors are not aware of any other work with the same formulation in the open literature. In this study, a deep reinforcement learning approach, proximal policy optimization (PPO) algorithm proposed in [30], is adopted. The details of PPO are also introduced in this Section.

A. STATE REPRESENTATION

The agent gains knowledge of the environment from the state of the formulated MDP. The state should provide an agent with all the information required to take optimal actions. The agent's state at time t is denoted by s_t. When querying the deep neural network, all of the parameters are normalized.

We divide s_t into two parts, that is $s_t = [s_t^0, s_t^1]$, where s_t^0 includes the information that is associated with the agent

itself and the destination, and s_t^1 includes the information that is associated with the environment, e.g., obstacles. We let w_t represent the information of the environment (including moving/static obstacles in this study) and set $s_t^1 = [w_t^1, w_t^2, \ldots, w_t^n]$. w_t^i indicates the information of obstacle i. To improve the efficiency of training the DRL algorithm, the state is transformed by following the robot-centric parameterization in [51], where the origin of the coordinate system is the agent's location and the x-axis points toward the agent's goal. Subscripts x and y denote x and y coordinates.

1) STATE REPRESENTATION FOR STATIC OBSTACLE AVOIDANCE

In the simulations of static obstacle avoidance,

$$s_t^0 = [d_g, v_x, v_y] \tag{7}$$

where d_g is the agent's distance to goal, v_x, v_y represent the agent's velocity. The features of the obstacle i include,

$$w_t^i = [P_y^i, d_i] \tag{8}$$

where P_y^i denotes the obstacle i's position in y-axis and d_i denotes the distance between the agent and the center of the obstacle i. P_y^i intends to encourage the agent to discover the global optimal solution. For instance, when the agent approaches the obstacle, a positive value of P_y^i indicates that the agent is on the right side of the line passing the center of the obstacle and the destination, and thus it is suggested to turn a small angle counterclockwise. We note that v and P are vectors in the transformed coordinate system.

2) STATE REPRESENTATION FOR MOVING OBSTACLE AVOIDANCE

Regarding moving obstacle avoidance, the position of the destination, (g_x, g_y), is added to s_t^0,

$$s_t^0 = [d_g, v_x, v_y, g_x, g_y] \tag{9}$$

The information of intruder i, w_t^i is represented by,

$$w_t^i = [P_x^i, P_y^i, V_x^i, V_y^i, d_i, V_{ref_x}^i, V_{ref_y}^i] \tag{10}$$

where P^i denotes the location of the intruder i, V^i denotes the velocity of the intruder i, d_i denotes the distance between the agent and the intruder i, and V_{ref}^i denotes the velocity of the agent relative to the intruder i. We note that v, g, P and V are vectors in the transformed coordinate system. UAS can obtain information about the moving obstacle in several ways. For example, some UAS may be equipped with ADS-B system and can communicate with the base station or neighboring UAS for state sharing. Another way is to use the radar system (both onboard and ground base) to detect and track moving obstacles. Finally, for some scenarios, UAS can use the onboard cameras, terrains, onboard avionics, and algorithms to locate and track the environments.

B. ACTION SPACE

1) ACTION SPACE FOR STATIC OBSTACLE AVOIDANCE AND STOCHASTIC INTRUDER AVOIDANCE

In the implementations of static obstacle avoidance and stochastic intruder avoidance, we define the action as the heading change of the controlled UAS at each time step. The action space is bounded as follows,

$$\mathcal{A}_h = [-30°, 30°] \tag{11}$$

Furthermore, the agent will choose an action $a_h \in \mathcal{A}_h$ at each time step t, and change its heading angle, ψ_t:

$$\psi_{t+1} = \psi_t + a_h \tag{12}$$

2) ACTION SPACE FOR DETERMINISTIC INTRUDER AVOIDANCE

As for deterministic intruder avoidance case, besides the heading angle change $a_h \in \mathcal{A}_h$, the UAS is also controlled by speed command. Both the heading angle and the speed are updated every second and remain unchanged during the interval. As UAS maneuvers are more flexible than manned aircraft and regulations on UAS speed change are unavailable, the action space of UAS speed command a_v is set to,

$$\mathcal{A}_v = [0m/s, 40\ m/s] \tag{13}$$

More specifically, the agent will take an action $a_v \in \mathcal{A}_v$, and change its speed $\tilde{v}$ accordingly at next time step $t+1$:

$$\tilde{v}_{t+1} = a_v \tag{14}$$

In real-world applications, however, it is rarely desirable for the controller of a UAS to make sharp turns. Therefore, a penalty for large heading or speed change due to the impact on power consumption could be considered in future research.

C. TERMINAL STATE

In the current study, a conflict occurs when the distance between the agent and the obstacle is less than the separation requirement. When the UAS operation is deterministic, a buffer zone is not necessary and the minimum separation distance is set to zero. In implementations of static obstacle avoidance with uncertainty and moving obstacle avoidance, the UAS position uncertainty is taken into account. The separation requirement is determined according to the operational safety bound proposed in [52]. With the UAS speed of 20 m/s and other UAS operation performance following the mean value shown in Table 3 in [52], the minimum separation distances for static obstacle avoidance is 75 m and 150 m for moving obstacle avoidance.

1) TERMINAL STATE FOR STATIC OBSTACLE AVOIDANCE

The terminal state for static obstacle avoidance consists of two different types of states:

- Conflict state: the distance between the agent and an obstacle is less than the separation requirement.
- Goal state: the agent is within 400 m from the destination.

2) TERMINAL STATE FOR MOVING OBSTACLE AVOIDANCE

The episode terminates only when the agent is within 200 m from the destination, which indicates the agent accomplishes the navigation task.

D. REWARD FUNCTION

In order to enable the agent to achieve its destination while avoiding conflicts, the reward function is designed to award task accomplishments while penalizing conflicts or failure to move towards the destination.

1) REWARD FUNCTION FOR STATIC OBSTACLE AVOIDANCE

As for the simulations of static obstacle avoidance, the reward function, $R(s, a)$, is developed in Eq. (15), where the learning agent receives a reward when it reaches the goal state and a penalty when it reaches the conflict state. The linear term of the reward function guides the UAS flying towards the destination. The constant penalty at each step emphasizes the shortest path rule.

$$R(s,a)=\begin{cases}10, & \text{if } s \text{ is goal state},\\ -0.001\ d_g-0.05-16, & \text{if } s \text{ is conflict state},\\ -0.001\ d_g-0.05, & \text{otherwise.}\end{cases} \tag{15}$$

2) REWARD FUNCTION FOR MOVING OBSTACLE AVOIDANCE

As for the simulations of intruder avoidance, the reward function, $R(s, a)$, is developed as shown in Eq. (16), similar to Eq. (15).

$$R(s,a)=\begin{cases}1000, & \text{if } s \text{ is goal state},\\ -c_g d_g-c_0+\sum_i c_{1i}(\arctan(c_{2i}(d_i-c_{3i}))-\frac{\pi}{2})-180, & \\ & \text{if } s \text{ is conflict state},\\ -c_g d_g-c_0+\sum_i c_{1i}(\arctan(c_{2i}(d_i-c_{3i}))-\frac{\pi}{2}), & \\ & \text{otherwise.}\end{cases} \tag{16}$$

In this reward function, $c_g, c_0, c_{1i}, c_{2i}, c_{3i}$ denote the coefficients of penalty terms and require to be balanced to assist the agent to avoid conflicts and reach the goal simultaneously. When the ownership is close to the intruder, the inverse tangent term of the reward function is activated to maintain the separation distance in an appropriate range. With the coefficients set in the stochastic intruder case (as shown in Section IV-B1), the relation between the distance and the inverse tangent term, $17(\arctan(0.1(d_i-12))-\frac{\pi}{2})$, is shown in Fig. 1. The agent starts to get a penalty when the distance approaches 250 m. This reward setting can help resolve conflicts with other intruders at a relatively early stage. We note that c_{2i}, c_{3i} can be tuned to fit different separation standards.

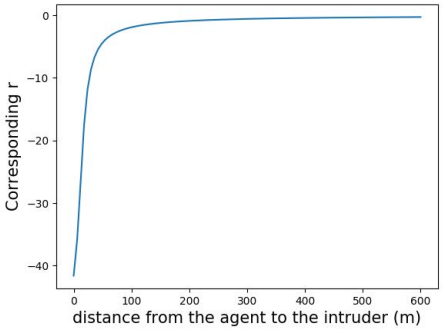


FIGURE 1. Reward that is based on the distance between the learning agent and the intruder.

E. PROXIMAL POLICY OPTIMIZATION ALGORITHM

One drawback of $\nabla_\theta J(\pi_\theta)$ shown in Eq. (6) which is proposed in [45], is that a single bad update may cause significant destructive effects and impede the model's final performance. A recently developed algorithm, called Proximal Policy Optimization (PPO) [30], addressed this issue by proposing a policy changing ratio which describes the difference between the previous policy and the new policy at time step t:

$$r_t(\theta)=\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \tag{17}$$

where θ_{old} and θ represent the network weights before and after update, respectively.

By limiting the policy changing ratio in the range of $[1-\epsilon, 1+\epsilon]$ abd setting ϵ to 0.2, the PPO loss functions for the actor and critic networks are determined as follows:

$$L_\pi(\theta)=-\mathop{\mathrm{E}}_{\tau\sim\pi_\theta}[\min(r_t(\theta)\cdot A_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\cdot A_t] - \beta\cdot H(\pi(\cdot|s_t)) \tag{18}$$

$$L_v=A_t^2 \tag{19}$$

$$A_t=R_t-V(s_t) \tag{20}$$

The advantage function A_t in Eq. (18) and Eq. (19) measures whether the action improves the policy's default behavior. Additionally, we include the policy entropy $\beta\cdot H(\pi(\cdot|s_t))$ in the actor loss function to encourage exploration by discouraging premature convergence to sub-optimal policies.

In the implementation, we utilize two layers Multilayer perceptron (MLP) with 64 hidden units each for both actor and critic networks. Tanh function is selected as the activation function for the hidden layers.

IV. NUMERICAL EXPERIMENTS

Numerical experiments are conducted in this section to determine the effectiveness of the proposed conflict resolution model in continuous action space. Two categories of collision avoidance are included: static obstacle avoidance, and moving obstacle avoidance. Regarding static obstacle avoidance, we explore the performance on obstacle with different shapes and sizes, and environment with uncertainty in UAS operation. Furthermore, we investigate the environment with

stochastic intruders under control of heading angle, and the environment with deterministic intruders under control of heading angle and speed. OpenAI Baselines [53] is utilized for the implementation of PPO algorithm. For each scenario, the number of training time steps is 30 million for the deep reinforcement learning model. Comparisons with other methods are not provided, since it can be difficult for classical optimization or control methods to handle problems with random origins — each specific case must be solved separately. While RL methods can learn a policy that guides the agent to the destination from different random points. Additionally, the comparison of PPO with other RL algorithms is conducted in [30] and PPO has shown better overall performance.

A. STATIC OBSTACLE AVOIDANCE

The simulation environment is a 4 km × 4 km two-dimensional free flight airspace. The speed of UAS is set to 20 m/s. In the training process, the learning agent starts from the four edges of the airspace boundary. The starting location is randomly sampled and the coordinates of the starting location are an integer array for simplification. The location of the goal is (2500, 2500). Here we consider two types of static obstacles: circular obstacle and rectangular obstacle, as illustrated in Fig. 2. The goal position is marked as the plus sign, and the no-passing zones are marked in blue. The RL learning curve is plotted in Fig. 3. The blue region is the raw episode reward data during training and the orange line is the episode reward mean of 100 episodes. Fig. 3 indicates that the episode reward increases and the optimal policy is found. For visualization of the performance of the developed conflict resolution model, a testing set of 160 trajectories with different starting positions are generated. Starting positions for testing are selected uniformly from the edges of the airspace boundary. The heading angles in 160 trajectories are collected and plotted every 15 time steps.

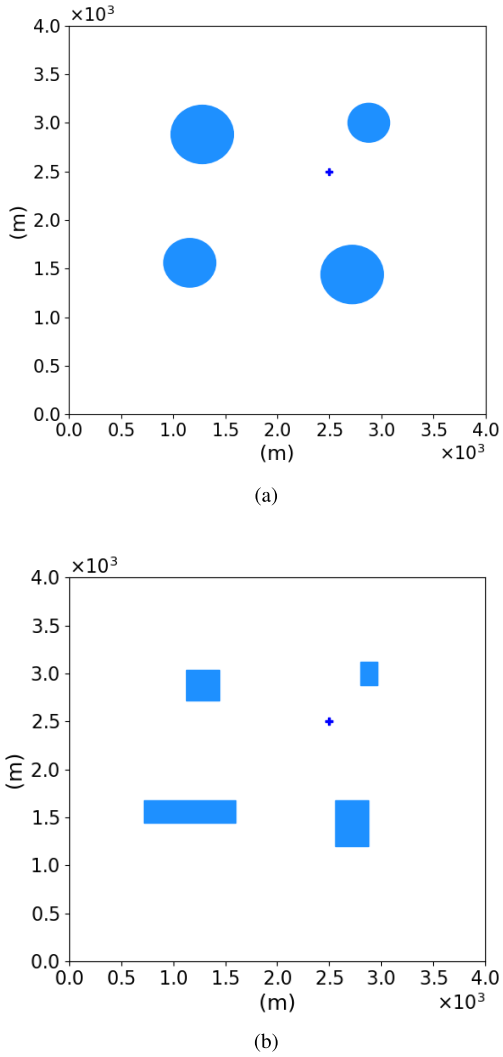


FIGURE 2. (a) Circular obstacle environment. (b) Rectangular obstacle environment. +: goal; blue: no-passing area.

1) CIRCULAR OBSTACLE AVOIDANCE

The static obstacles set up in this case study are illustrated in Fig. 2(a) and the testing result of 160 trajectories with different starting positions on the airspace boundary is demonstrated in Fig. 4. The agent's selected heading direction at current position is marked as a black arrow.

It can be seen from Fig. 4 that the agent selects the heading direction that points towards the destination and tends to bypass the no-passing regions. Moreover, the agent behaves optimally based on the relative position of the agent, obstacle, and goal. For example, near the lower-left obstacle, if the agent's position is above the line connecting the obstacle center and the goal, the UAS takes a small left turn to the higher semicircle to avoid the obstacle. Otherwise, the UAS bypasses the lower semicircle. No failure exists in the 160 testing trajectories in Fig. 4.

2) RECTANGULAR OBSTACLE AVOIDANCE

The simulation of the rectangular obstacle case differs from the previous circular obstacle case in the condition for determining whether the learning agent is at a conflict state. Fig. 2(b) illustrates the environment of the rectangular obstacle case and Fig. 5 shows the testing result of 160 trajectories.

The performance in Fig. 5 is similar to that in Fig. 4. The UAS learns to bypass the obstacle through one side of the obstacle depending on the relative position of the agent, obstacle, and goal. No failure happens among the test of 160 trajectories.

From Fig. 4 and Fig. 5, it can be seen that the proposed model has the capability to enable the UAS to navigate to the goal with the shortest path and avoid static obstacles meanwhile for different obstacle sizes or shapes.

3) CIRCULAR OBSTACLE AVOIDANCE WITH UNCERTAINTY

This case study is conducted to evaluate the effectiveness of handling uncertainty by the developed conflict resolution model. There exists randomness in almost every aspect of UTM (e.g., UAS operation, environment). Thus, uncertainty

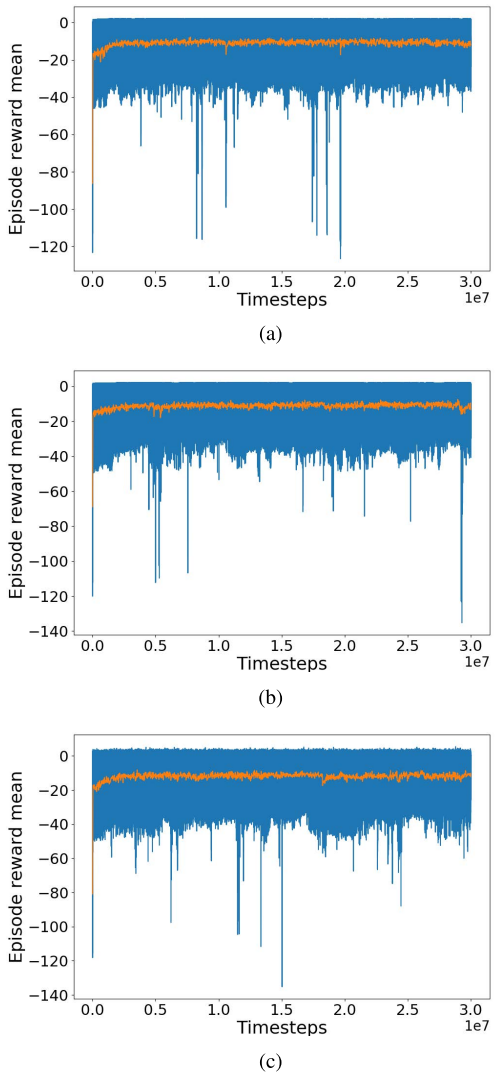


FIGURE 3. Episode reward mean for (a) circular obstacle avoidance, (b) rectangular obstacle environment, (c) circular obstacle avoidance with probabilistic agent's position.

quantification of aircraft operations is critical for future safety analysis (e.g., the deviation from a trajectory plan due to wind, true speed, positioning error) [52], [54], [55], [56], [57], [58]. Thus, to model the uncertainties in UAS operation, we form a circle, the center of which is the predicted UAS position without uncertainty. And the radius is the separation requirement, 75 m. With 90% probability, the UAS position is accurately located at the center of the circle; with a 10% probability, the UAS position will be located at a point around the circle with a uniform distribution. Such uncertainty is taken into consideration by adding to the position of the agent at the next time step after executing action a.

Fig. 6 shows the testing results. The agent's position with uncertainty is plotted In Fig. 6(a). Alternatively, the uncertainty of 75 m can be added to the obstacle. The red circles in Fig. 6(b) indicate the obstacle expansion due to the uncertainty of 75 m. As expected, the result in Fig. 6(a) shows that the UAS tries to keep 75 m away from the obstacles.

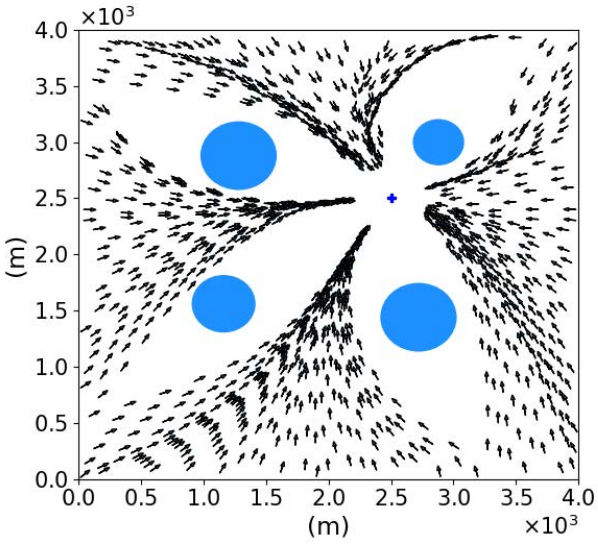


FIGURE 4. Results of circular obstacle avoidance. +: goal; blue: no-passing area; arrow: the selected heading direction.

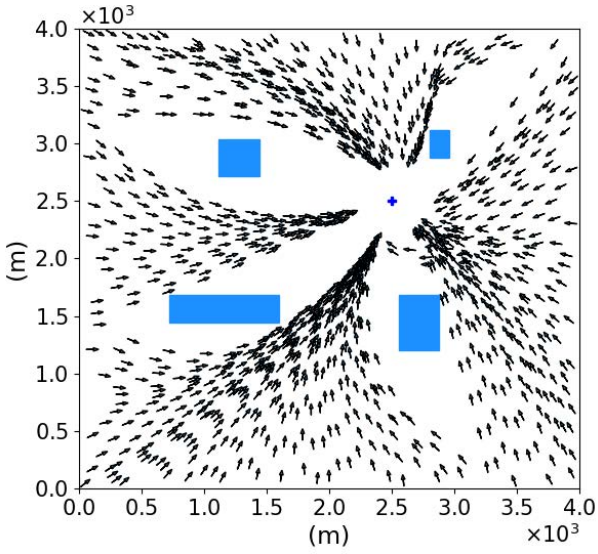
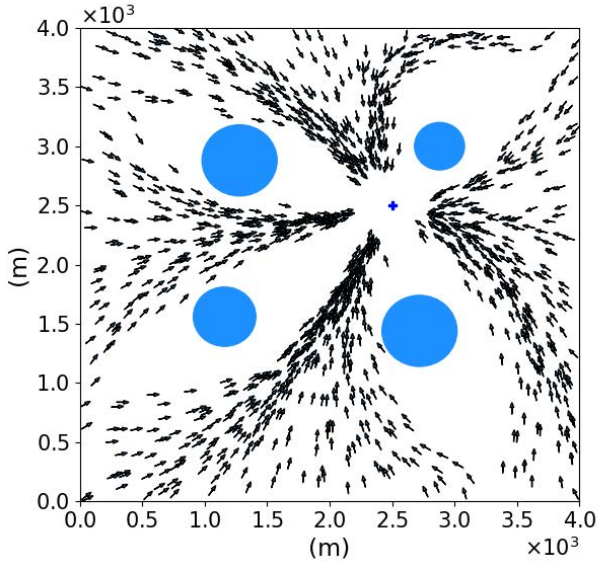


FIGURE 5. Results of rectangular obstacle avoidance. +: goal; blue: no-passing area; arrow: the selected heading direction.

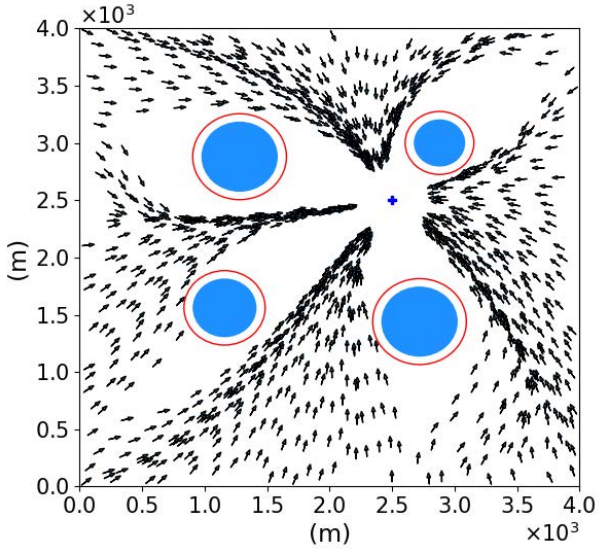
So both methods of representing uncertainty can work for the simulations of obstacle avoidance with uncertainty. One failure happens near the upper-left obstacle in Fig. 6(a). There are three failures near the lower-left obstacle in Fig. 6(b). The common among the failures is that the agent's origin is approximately on the line connecting the obstacle center and the goal. The possible reason is that the policy network gets stuck at the local optimum since the two trajectories next to it behave well.

B. MOVING OBSTACLE AVOIDANCE

In the simulation of the moving intruder aircraft avoidance case, the speed of intruders is set to 20 m/s. We conducted



(a)



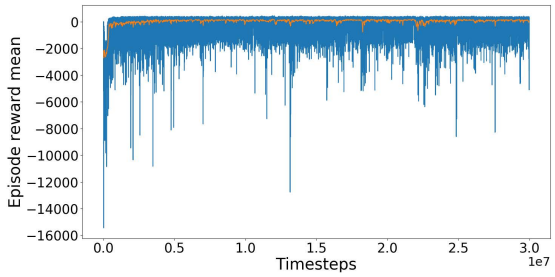
(b)

FIGURE 6. (a) Results with probabilistic agent's position. (b) Results with uncertainty added to obstacles. +: goal; blue: no-passing area; arrow: the selected heading direction; red: separation requirement due to uncertainty in UAS operation.

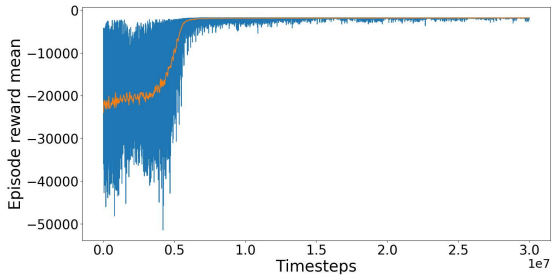
two case studies for moving obstacle avoidance: stochastic intruder case with control of heading angle and deterministic intruder case with control of heading angle and speed. In the stochastic intruder case, the scenario changes every episode. In detail, the intruder has a different origin and heading angle for each episode. But within one episode, intruders have fixed heading angles. The reward coefficients are listed in Table 1. The episode rewards during training are plotted in Fig. 7. The blue region is the raw episode reward data and the orange line is the episode reward mean of 100 episodes. For the visualization of the performance of the developed obstacle avoidance model, we generate a testing set of 500 episodes with the same the setting as the training process for each moving obstacle cases. Also, the minimum distance between the agent and the three intruders within each episode is recorded.

TABLE 1. Reward coefficient.

Coefficient	Stochastic-intruder	Deterministic-intruder
c_g	0.007	0.22
c_0	0.15	0.05
c_{1i}	17	3
c_{2i}	0.1	0.1
c_{3i}	12	12



(a)



(b)

FIGURE 7. (a)Episode reward mean of stochastic-intruder avoidance with control of heading angle. (b) Episode reward mean of deterministic-intruder avoidance with control of heading angle and speed.

1) STOCHASTIC-INTRUDER AVOIDANCE WITH CONTROL OF HEADING ANGLE

The origin and heading angle of the three intruders are assumed to follow a uniform distribution and the distribution range is shown in Table 2. The origin coordinate of agent is uniformly sampled from $(75 \sim 135, 0 \sim 25)$. The goal is located at $(100, 200)$. The agent moves at 20 m/s. The intruders are designed to pass the line connecting the UAS origin and the goal.

The demonstration of one scenario and UAS performance is shown in Fig. 8. Information related to the ownership is plotted in blue and black represents the information of intruders. The plus sign denotes the origin and the star sign is the goal for the agent. The centers of circles are the positions of aircraft which are plotted every 5 time steps and labeled with time step every 10 time steps. The radius of the circle

TABLE 2. Intruder information.

Intruder	Origin coordinate range	Heading angle range
1	$(5 \sim 35, 200)$	$[-90°, 0°]$
2	$(20 \sim 65, 20 \sim 65)$	$[0°, 90°]$
3	$(120 \sim 180, 120 \sim 180)$	$[-180°, -90°]$

represents the aircraft speed. In this scenario, the agent learns to go around the left side to avoid the three intruders. The result of the minimum distance between the agent and the three intruders within each episode is plotted in Fig. 9 by the blue dots. The orange line is the separation requirement of 150 *m*. All the blue dots are above the orange line, which represents that there is no failure case in Fig. 9 and the model succeeds to avoid the three intruders in 500 different testing scenarios.

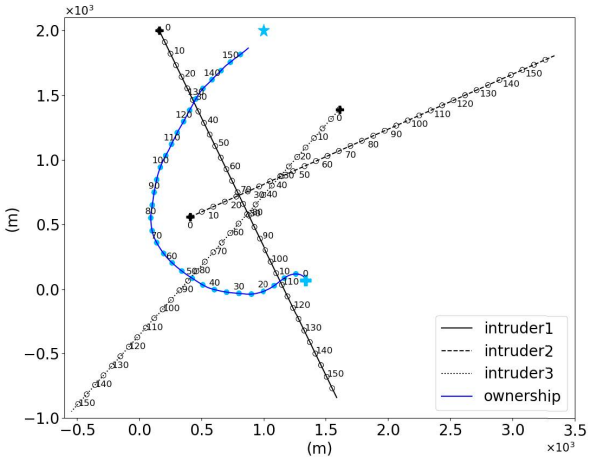


FIGURE 8. Demonstration of one scenario and UAS performance of stochastic-intruder avoidance with control of heading angle. Number: time step. Blue: ownership; black: intruders. Circle center: UAS position; circle radius: UAS speed. +: origin; *: goal.

2) DETERMINISTIC-INTRUDER AVOIDANCE WITH CONTROL OF HEADING ANGLE AND SPEED

We also investigate the possibility of utilizing the proposed reward function to generate heading angle change command and speed command. This investigation is valuable when changing the heading angle cannot efficiently resolve conflicts. Moreover, with an extra choice of changing speed, the UAS may result in less influence on flight plans of other aircraft and aerospace capacity. However, due to the larger action space, the training process needs more effort.

The origin and heading angle of the three intruders are listed in Table 3. The origin coordinate of agent is (100, 210) and the goal is located at (100, 0). Intruder 1 is designed to test if the ownership can fly at a suitable speed and the other two intruders are set to test the performance of the heading angle change command.

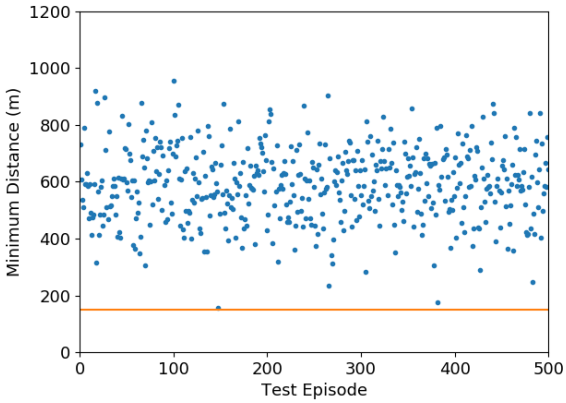


FIGURE 9. Minimum distance results of stochastic-intruder avoidance with control of heading angle. Orange line: separation requirement of 150 *m*. Blue dot: the minimum distance between the agent and the three intruders within each episode.

TABLE 3. Intruder information.

Intruder	1	2	3
Origin coordinate	$(90, 170)$	$(35, 155)$	$(-15, 115)$
Heading angle	$-90°$	$-0.2°$	$-0.2°$

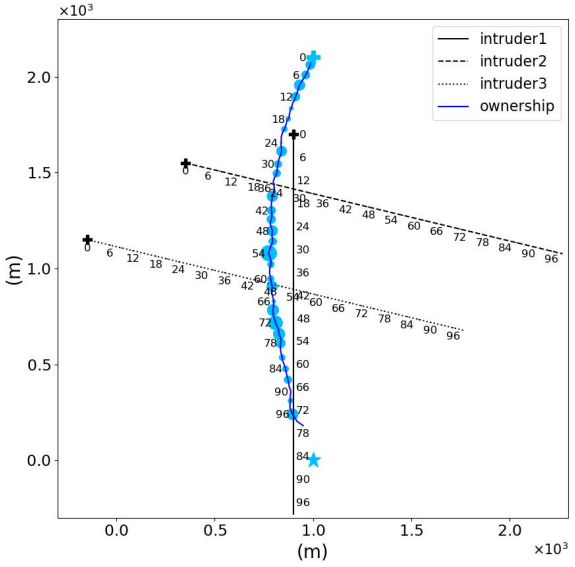


FIGURE 10. Demonstration of the scenario and UAS performance of deterministic-intruder avoidance with control of heading angle and speed. Number: time step. Blue: ownership; black: intruders. Circle center: UAS position; circle radius: UAS speed. +: origin; *: goal.

Similar to the result in Fig. 8, the demonstration of the scenario and UAS performance is shown in Fig. 10. Information related to the ownership is plotted in blue and black represents the information of intruders. The plus sign denotes the origin and the star sign is the goal for the agent. The centers of circles are the positions of aircraft which are plotted every 3 time steps and labeled with time step every 6 time steps. The radius of the circle represents the aircraft speed. It can be seen that the agent reduces speed from 12 to 24 time steps to

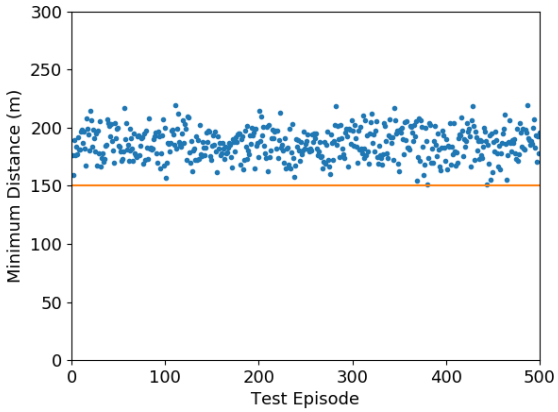


FIGURE 11. Minimum distance results of deterministic-intruder avoidance with control of heading angle and speed. Orange line: separation requirement of 150 *m*. Blue dot: the minimum distance between the agent and the three intruders within each episode.

keep a safe separation away from intruder 1. Also, the agent goes around the right side to avoid the approaching intruder 2 and after resolving the possible conflicts with intruder 3 at 66 time step, it flies towards the goal to save time. The result of the minimum distance is plotted in Fig. 11 by the blue dots. All the blue dots are above the orange line which indicates the separation requirement of 150 *m*. So, no failure exists in Fig. 11, indicating that the model succeeds to avoid the three intruders under the control of heading angle and speed.

V. CONCLUSION

In this study, we propose a method for utilizing deep reinforcement learning to enable the UAS to navigate successfully in urban airspace with continuous action space. This work proposes a thorough and holistic formulation of UAS motion planning as an MDP problem so the PPO algorithm can be applied. The formulation includes the definitions of the necessary components in RL — state, continuous action, transition, and reward function — to specify the problem. Both static and moving obstacles are simulated, and the trained UAS is capable of achieving the goal and resolving conflict simultaneously. We also investigate the performance on various static obstacle shapes and sizes, and under uncertainty in UAS operation. Stochastic intruders are included in the training process of the moving obstacle case studies. Moreover, we investigate the possibility of the proposed reward function to resolve conflict through heading angle and speed. Results show that the proposed model is able to provide accurate and robust guidance and resolve conflict with a success rate of over 99%.

To further improve the usability and efficiency of the proposed algorithm for real-world problems, in future work, we would model part of the intruders as agents and there could be cooperation among the multiple aircraft. Additionally, the two learning problems with static or moving obstacles can be integrated by combining reward functions for both static and dynamic obstacles. Tuning of coefficients in the reward functions needs to be done to ensure the proper weight of both static and dynamic rewards in the final function. More fundamental development of the RL algorithms to further increases the safety of UAS motion planning is also worth investigating in the future.

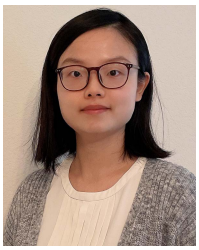
ACKNOWLEDGMENT

PI: Yongming Liu and Technical Officer: Anupa Bajwa. The support is gratefully acknowledged.

REFERENCES

[1] K. Balakrishnan, J. Polastre, J. Mooberry, R. Golding, and P. Sachs, "Blueprint for the sky: The roadmap for the safe integration of autonomous aircraft," Airbus UTM, San Francisco, CA, USA, 2018.
[2] P. Kopardekar, J. Rios, T. Prevot, M. Johnson, J. Jung, and J. E. Robinson, "Unmanned aircraft system traffic management (UTM) concept of operations," in *Proc. AIAA Aviation Aeronaut. Forum*, 2016, pp. 1–16.
[3] E. Frazzoli, Z.-H. Mao, J.-H. Oh, and E. Feron, "Resolution of conflicts involving many aircraft via semidefinite programming," *J. Guid., Control Dyn.*, vol. 24, no. 1, pp. 79–86, Jan. 2001.
[4] A. U. Raghunathan, V. Gopal, D. Subramanian, L. T. Biegler, and T. Samad, "Dynamic optimization strategies for three-dimensional conflict resolution of multiple aircraft," *J. Guid., Control, Dyn.*, vol. 27, no. 4, pp. 586–594, Jul. 2004.
[5] P. J. Enright and B. A. Conway, "Discrete approximations to optimal trajectories using direct transcription and nonlinear programming," *J. Guid., Control, Dyn.*, vol. 15, no. 4, pp. 994–1002, 1992.
[6] T. Schouwenaars, B. De Moor, E. Feron, and J. How, "Mixed integer programming for multi-vehicle path planning," in *Proc. Eur. Control Conf. (ECC)*, Sep. 2001, pp. 2603–2608.
[7] A. Richards and J. P. How, "Aircraft trajectory planning with collision avoidance using mixed integer linear programming," in *Proc. Amer. Control Conf.*, vol. 3, May 2002, pp. 1936–1941.
[8] L. Pallottino, E. M. Feron, and A. Bicchi, "Conflict resolution problems for air traffic management systems solved with mixed integer programming," *IEEE Trans. Intell. Transp. Syst.*, vol. 3, no. 1, pp. 3–11, Mar. 2002.
[9] A. Vela, S. Solak, W. Singhose, and J.-P. Clarke, "A mixed integer program for flight-level assignment and speed control for conflict resolution," in *Proc. 48h IEEE Conf. Decis. Control (CDC) Held Jointly 28th Chin. Control Conf.*, Dec. 2009, pp. 5219–5226.
[10] D. Mellinger, A. Kushleyev, and V. Kumar, "Mixed-integer quadratic program trajectory generation for heterogeneous quadrotor teams," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2012, pp. 477–483.
[11] F. Augugliaro, A. P. Schoellig, and R. D'Andrea, "Generation of collision-free trajectories for a quadrocopter fleet: A sequential convex programming approach," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2012, pp. 1917–1922.
[12] D. Morgan, S.-J. Chung, and F. Y. Hadaegh, "Model predictive control of swarms of spacecraft using sequential convex programming," *J. Guid., Control, Dyn.*, vol. 37, no. 6, pp. 1725–1740, Nov./Dec. 2014.
[13] B. Acikmese and S. R. Ploen, "Convex programming approach to powered descent guidance for Mars landing," *J. Guid., Control, Dyn.*, vol. 30, no. 5, pp. 1353–1366, Sep. 2007.
[14] D. Delahaye, C. Peyronne, M. Mongeau, and S. Puechmorel, "Aircraft conflict resolution by genetic algorithm and B-spline approximation," in *2nd ENRI Int. Workshop ATM/CNS (EIWAC)*, 2010, pp. 71–78.
[15] J. A. Cobano, R. Conde, D. Alejo, and A. Ollero, "Path planning based on genetic algorithms and the monte-carlo method to avoid aerial vehicle collisions under uncertainties," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2011, pp. 4429–4434.
[16] M. Pontani and B. A. Conway, "Particle swarm optimization applied to space trajectories," *J. Guid., Control, Dyn.*, vol. 33, no. 5, pp. 1429–1441, Sep. 2010.
[17] J. Sun and Y. Zhang, "A reinforcement learning-based decentralized method of avoiding multi-UAV collision in 3-D airspace," in *Proc. 3rd Int. Conf. Comput. Sci. Artif. Intell.*, Dec. 2019, pp. 77–82.
[18] M. J. Kochenderfer, J. E. Holland, and J. P. Chryssanthacopoulos, "Next-generation airborne collision avoidance system," Massachusetts Inst. Technol.-Lincoln Lab., Lexington, KY, USA, Tech. Rep., 2012.

[19] G. Manfredi and Y. Jestin, "An introduction to ACAS xu and the challenges ahead," in *Proc. IEEE/AIAA 35th Digit. Avionics Syst. Conf. (DASC)*, Sep. 2016, pp. 1–9.
[20] M. P. Owen, A. Panken, R. Moss, L. Alvarez, and C. Leeper, "ACAS Xu: Integrated collision avoidance and detect and avoid capability for UAS," in *Proc. IEEE/AIAA 38th Digit. Avionics Syst. Conf. (DASC)*, Sep. 2019, pp. 1–10.
[21] J. P. Chryssanthacopoulos and M. J. Kochenderfer, "Decomposition methods for optimized collision avoidance with multiple threats," *J. Guid., Control, Dyn.*, vol. 35, no. 2, pp. 398–405, Mar. 2012.
[22] X. Yang and P. Wei, "Autonomous on-demand free flight operations in urban air mobility using Monte Carlo tree search," in *Proc. 8th Int. Conf. Res. Air Transp. (ICRAT)*, 2018.
[23] X. Yang and P. Wei, "Scalable multi-agent computational guidance with separation assurance for autonomous urban air mobility," *J. Guid., Control, Dyn.*, vol. 43, no. 3, pp. 1473–1486, 2020.
[24] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2000, pp. 1057–1063.
[25] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," 2013, *arXiv:1312.5602*.
[26] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. 13th AAAI Conf. Artif. Intell.*, 2016, pp. 1–7.
[27] Z. Wang, T. Schaul, M. Hessel, H. van Hasselt, M. Lanctot, and N. de Freitas, "Dueling network architectures for deep reinforcement learning," 2015, *arXiv:1511.06581*.
[28] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.
[29] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1928–1937.
[30] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.
[31] S. Li, M. Egorov, and M. Kochenderfer, "Optimizing collision avoidance in dense airspace using deep reinforcement learning," 2019, *arXiv:1912.10146*.
[32] S. Yang, Z. Meng, X. Chen, and R. Xie, "Real-time obstacle avoidance with deep reinforcement learning three-dimensional autonomous obstacle avoidance for UAV," in *Proc. Int. Conf. Robot., Intell. Control Artif. Intell. (RICAI)*, 2019, pp. 324–329.
[33] B. Wulfe, "UAV collision avoidance policy optimization with deep reinforcement learning," Stanford Univ., 2017.
[34] C. W. Keong, H.-S. Shin, and A. Tsourdos, "Reinforcement learning for autonomous aircraft avoidance," in *Proc. Workshop Res., Educ. Develop. Unmanned Aerial Syst. (RED UAS)*, Nov. 2019, pp. 126–131.
[35] M. Brittain, X. Yang, and P. Wei, "A deep multi-agent reinforcement learning approach to autonomous separation assurance," 2020, *arXiv:2003.08353*.
[36] D.-T. Pham, N. P. Tran, S. Alam, V. Duong, and D. Delahaye, "A machine learning approach for conflict resolution in dense traffic scenarios with uncertainties," in *Proc. 13th USA/Eur. Air Traffic Manage. Res. Develop. Seminar*, 2019, pp. 17–21.
[37] Z. Ma, C. Wang, Y. Niu, X. Wang, and L. Shen, "A saliency-based reinforcement learning approach for a UAV to avoid flying obstacles," *Robot. Auto. Syst.*, vol. 100, pp. 108–118, Feb. 2018.
[38] R. Bellman, "A Markovian decision process," *Indiana Univ. Math. J.*, vol. 6, no. 4, pp. 679–684, Apr. 1957.
[39] R. A. Howard, *Dynamic Programming and Markov Processes*. Wiley, 1960.
[40] D. J. White, "A survey of applications of Markov decision processes," *J. Oper. Res. Soc.*, vol. 44, no. 11, pp. 1073–1096, 1993.
[41] E. A. Feinberg and A. Shwartz, *Handbook of Markov Decision Processes: Methods and Applications*, vol. 40. Springer, 2012.
[42] S. Koenig *et al.*, "Xavier: A robot navigation architecture based on partially observable Markov decision process models," in *Artificial Intelligence Based Mobile Robotics: Case Studies of Successful Robot Systems*. Citeseer, 1998, pp. 91–122.
[43] S. Thrun, "Probabilistic robotics," *Commun. ACM*, vol. 45, no. 3, pp. 52–57, 2002.
[44] M. Mariton, *Jump Linear Systems in Automatic Control*. New York, NY, USA: M. Dekker, 1990.
[45] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.
[46] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, and J. Veness, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.
[47] M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver, "Rainbow: Combining improvements in deep reinforcement learning," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 3215–3222.
[48] D. Silver, J. Schrittwieser, K. Simonyan, and I. Antonoglou, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, p. 354, Oct. 2017.
[49] O. Vinyals *et al.*, "StarCraft II: A new challenge for reinforcement learning," 2017, *arXiv:1708.04782*.
[50] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1889–1897.
[51] Y. F. Chen, M. Liu, M. Everett, and J. P. How, "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 285–292.
[52] J. Hu, H. Erzberger, K. Goebel, and Y. Liu, "Probabilistic risk-based operational safety bound for rotary-wing unmanned aircraft systems traffic management," *J. Aerosp. Inf. Syst.*, vol. 17, no. 3, pp. 171–181, Mar. 2020.
[53] P. Dhariwal, C. Hesse, O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Sidor, Y. Wu, and P. Zhokhov. (2017). *Openai Baselines*. [Online]. Available: https://github.com/openai/baselines
[54] Y. Liu and K. Goebel, "Information fusion for national airspace system prognostics: A NASA ULI project," in *Proc. 10th Annu. Conf. Prognostics Health Manage. Soc. (PHM)*. Philadelphia, PA, USA: Philadelphia Center City, 2018, pp. 24–27.
[55] J. Hu and Y. Liu, "UAS conflict resolution integrating a risk-based operational safety bound as airspace reservation with reinforcement learning," in *Proc. AIAA Scitech Forum*, 2020, p. 1372.
[56] Y. Pang, H. Yao, J. Hu, and Y. Liu, "A recurrent neural network approach for aircraft trajectory prediction with weather features from sherlock," in *Proc. AIAA Aviation Forum*, 2019, p. 3413.
[57] Y. Pang, N. Xu, and Y. Liu, "Aircraft trajectory prediction using LSTM neural network with embedded convolutional layer," in *Proc. Annu. Conf. PHM Soc.*, 2019, vol. 11, no. 1, pp. 1–10.
[58] Y. Pang, X. Zhao, H. Yan, and Y. Liu, "Data-driven trajectory prediction with weather uncertainties: A Bayesian deep learning approach," *Transp. Res. C, Emerg. Technol.*, vol. 130, Sep. 2021, Art. no. 103326.

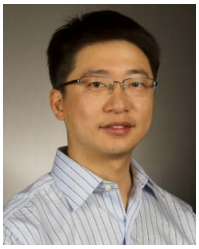


JUEMING HU received the bachelor's degree from Southeast University, China, in 2017, and the master's degree from Arizona State University, in 2018, where she is currently pursuing the Ph.D. degree with the School for Engineering of Matter, Transport and Energy. Her research interests include reinforcement learning, UAS traffic management, optimization, and operation research.

XUXI YANG received the bachelor's degree in applied mathematics from the Harbin Institute of Technology and the Ph.D. degree in aerospace engineering from Iowa State University under the supervision of Prof. Peng Wei. His research interests include deep reinforcement learning, machine learning, and decision theory, with applications in air traffic control/management (ATC/M) and UAS traffic management (UTM).

WEICHANG WANG received the bachelor's degree from the School of Electronic Information and Communications, Huazhong University of Science and Technology, and the M.Sc. and Ph.D. degrees from the School of Electrical, Computer and Energy Engineering, Arizona State University. His research interests include wireless communication networks, resource allocation in stochastic systems, multi-agent reinforcement learning, and the Internet of Things (IoT).

PENG WEI (Member, IEEE) is currently an Associate Professor with the Department of Mechanical and Aerospace Engineering, The George Washington University. By contributing to the intersection of control, optimization, machine learning, and artificial intelligence, he develops autonomy and decision support tools for aeronautics, aviation, and aerial robotics. His current research interests include safety, efficiency, scalability of decision making systems in complex, uncertain, and dynamic environments. His recent applications include air traffic control/management (ATC/M), airline operations, UAS traffic management (UTM), eVTOL urban air mobility (UAM), and autonomous drone racing (ADR). He is leading the Intelligent Aerospace Systems Laboratory (IASL).



LEI YING (Fellow, IEEE) received the B.E. degree from Tsinghua University, Beijing, China, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Illinois at Urbana–Champaign.

He currently is a Professor at the Electrical Engineering and Computer Science Department, University of Michigan, Ann Arbor. He has coauthored books *Communication Networks: An Optimization, Control and Stochastic Networks Perspective* (Cambridge University Press, 2014), *Diffusion Source Localization in Large Networks*, and *Synthesis Lectures on Communication Networks* (Morgan & Claypool Publishers, 2018). His research interests include the interplay of complex stochastic systems and big-data, including large-scale communication/computing systems for big-data processing, private data marketplaces, and large-scale graph mining.

Dr. Ying won the Young Investigator Award from the Defense Threat Reduction Agency (DTRA), in 2009, and NSF Career Award, in 2010. He was the Northrop Grumman Assistant Professor with the Department of Electrical and Computer Engineering, Iowa State University, from 2010 to 2012. His papers have received the Best Paper Award at IEEE INFOCOM 2015, the Kenneth C. Sevcik Outstanding Student Paper Award at ACM SIGMETRICS/IFIP Performance 2016, and the WiOpt'18 Best Student Paper Award; his papers have also been selected in ACM TKDD Special Issue "Best Papers of KDD 2016," Fast-Track Review for TNSE at IEEE INFOCOM 2018 (seven out of 312 accepted papers were invited), and Best Paper Finalist at MobiHoc 2019. He is an Associate Editor of the IEEE TRANSACTIONS ON INFORMATION THEORY and the *Performance Evaluation* (Elsevier).

YONGMING LIU is currently a Professor in aerospace and mechanical engineering with the School for Engineering of Matter, Transport and 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, imaging-based experimental testing, diagnostics, and prognostics.