Thirteenth USA/Europe Air Traffic Management Research and Development Seminar (ATM2019)

Optimizing Collision Avoidance in Dense Airspace using Deep Reinforcement Learning

Sheng Li
Aeronautics and Astronautics
Stanford University
Stanford, California, USA
lisheng@stanford.edu

Maxim Egorov
Airbus UTM
San Francisco, California, USA
maxim.egorov@airbus-sv.com

Mykel J. Kochenderfer
Aeronautics and Astronautics
Stanford University
Stanford, California, USA
mykel@stanford.edu

***Abstract*—New methodologies will be needed to ensure the airspace remains safe and efficient as traffic densities rise to accommodate new unmanned operations. This paper explores how unmanned free-flight traffic may operate in dense airspace. We develop and analyze autonomous collision avoidance systems for aircraft operating in dense airspace where traditional collision avoidance systems fail. We propose a metric for quantifying the decision burden on a collision avoidance system as well as a metric for measuring the impact of the collision avoidance system on airspace. We use deep reinforcement learning to compute corrections for an existing collision avoidance approach to account for dense airspace. The results show that a corrected collision avoidance system can operate more efficiently than traditional methods in dense airspace while maintaining high levels of safety.**

***Keywords*-collision avoidance, multi-agent systems, Markov decision process, deep reinforcement learning**

I. Introduction

Recent technological advances have enabled a number of new applications for unmanned aircraft ranging from aerial cargo delivery to autonomous vertical take-off and landing (VTOL) passenger aircraft. It is estimated that by the year 2035, the number of package delivery aircraft in the sky will increase by one to two orders of magnitude [1], while the number of passenger carrying VTOL operations is expected to increase at a similar pace [2]. This increase will lead to hundreds or even thousands of aircraft occupying relatively small volumes of airspace, and will require new methodologies to ensure safe and efficient operations.

It is unclear how traditional air traffic management (ATM) approaches for maintaining safety and efficiency in the airspace perform in the context of high-volume unmanned traffic. There has been tremendous interest in on-board collision avoidance systems (CAS), both in the context of manned commercial aviation [3], [4] and in the context of unmanned aircraft [5], [6]. For example, the Traffic-alert and Collision Avoidance System (TCAS) was designed for manned aviation and can accommodate densities of up to 0.3 aircraft/nmi^2 [7]. Its successor, the next-generation airborne collision avoidance system (ACAS X) formulates the CAS problem as a partially observable Markov decision process (POMDP) and is able to operate in even denser airspace [4], [8]. While ACAS X has been extended to unmanned aircraft [9] and resolving conflicts with multiple threats [10], its performance in ultra-dense airspace has not been deeply studied, with evaluations primarily focused on the much more common pairwise aircraft encounters [11], [12].

Collision avoidance has been studied in fields outside of aviation with applications ranging from robotics [13] to autonomous vehicles [14]. When communication networks exist, the problem can be solved using centralized path optimization [15], [16]. A number of decentralized approaches have also been developed to solve sequential multi-agent decision problems using deep reinforcement learning (DRL) [17]–[19], which can scale to large observation spaces and many agents. DRL has been extended to collision avoidance through approaches that learn interaction dynamics [20], explicitly model dynamic uncertainty [21], and learn policies end-to-end [22]. However, the performance of collision avoidance strategies typically degrades when the number of agents increases due to an exponential growth in the state space. Designing CAS policies for high airspace densities will require a new set of approaches, and this paper aims to explore one of them.

We formulate collision avoidance as a stochastic problem in the form of a multi-agent Markov decision process (MMDP) similar to [23] with a focus on resolutions in the horizontal plane. On top of decomposing the problem into pairwise encounters, we apply a DRL based approach to improve the collision avoidance in dense airspace. We combine the decentralized training approach that has shown to scale in multi-agent systems [19], [22] with a deep correction factor [24] to explicitly capture the properties of a multi-agent system and the requirements for collision avoidance. The contributions of this work are as follows: (1) an approach that adds corrections learned through DRL to an existing policy for further improving collision avoidance in dense airspace, (2) an analysis of how collision avoidance systems impact operations in the dense airspace, (3) recommendations for how to approach the dense airspace problem from the perspective of collision avoidance.

II. Problem Formulation

This section introduces the mathematical framework for collision avoidance using the Markov decision process (MDP) [25].

A. Markov Decision Process

An MDP is formally defined by the tuple $(S, A, T, R, \ )$, where S is the state space, A is the action space, T is the state transition function, R is the reward function, and $\ $ is the discount factor. In an MDP, an agent takes action $a_t \ 2 \ A$ at time t based on the state $s_t \ 2 \ S$, and receives a reward $r_t = R(s_t; a_t)$. At time $t + 1$, the state transits from s_t to s_{t+1} with a probability $\Pr(s_{t+1} \ j \ s_t; a_t) = T(s_{t+1}; s_t; a_t)$. The objective of the agent is to maximize the accumulated expected discounted reward $\sum_{t=0}^{1} \ ^t r_t$.

A solution to an MDP is a policy $\ : S \ ! \ A$ that defines what action to execute at a given state. An optimal policy $\ $ of an MDP can be represented by a state-action value function $Q \ (s; a)$ that satisfies the Bellman equation [26]:

$$Q \ (s; a) = R(s; a) + \sum_{s^\theta} T(s^\theta; s; a) \max_{a^\theta} Q \ (s^\theta; a^\theta), \quad (1)$$

where s is the current state and s^θ is a state reachable at the next time step by taking action a. In this work we use sigma-point sampling [27] and a generative model to formulate the transition function, which allows us to re-write the Bellman equation in a more general form:

$$Q \ (s; a) = \mathrm{E}_{s^\theta} \left[R(s; a) + \max_{a^\theta} Q \ (s^\theta; a^\theta) \right], \quad (2)$$

which represents the expected discounted reward for the next state s^θ. With Q , the corresponding optimal policy can be written as $\ (s) = \arg\max_a Q \ (s; a)$. While the optimal utility is given by $U \ (s) = \max_a Q \ (s; a)$.

B. Dynamics and Sensor Measurements

In this work we focus on co-altitude, horizontal encounters. The dynamics of the aircraft are described by its position coordinates $(x; y)$, speed v, heading angle $\ $ and turn rate $\_$, and are updated by

$$\begin{aligned} & + \ \_ \ t, \\ x \quad & x + v \cos \ \ t, \\ y \quad & y + v \sin \ \ t: \end{aligned} \quad (3)$$

The sensor model in our aircraft can be described by the following variables:

1) $\ $: Distance from the ownship to the intruder.
2) $\ $: Angle to the intruder relative to ownship heading direction.
3) $\ $: Heading angle of the intruder relative to the heading direction of the ownship.
4) v_{own}: Speed of the ownship.
5) v_{int}: Speed of the intruder.

An example encounter that includes the sensor measurements is illustrated in Fig. 1.

C. Action Space

The collision avoidance policy can issue the following advisories to resolve conflicts: $f$clear of conflict (COC), weak left (WL), weak right (WR), strong left (SL), strong right (SR),

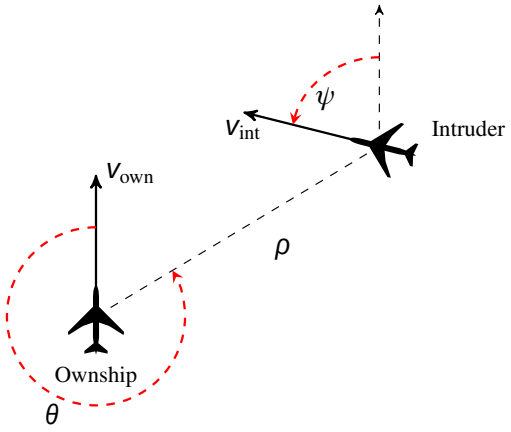

Figure 1. Sensor measurements for a co-altitude pairwise encounter [11].

MAINTAINg. These advisories can be transformed into turn rates that control the aircraft in the following way:

$$\begin{aligned} A = f & \text{COC} \ ! \ \text{free to fly towards the destination;} \\ & \text{WL} \ ! \ +5 \ =\text{s;} \ \text{WR} \ ! \ \ 5 \ =\text{s;} \\ & \text{SL} \ ! \ +10 \ =\text{s;} \ \text{SR} \ ! \ \ 10 \ =\text{s;} \ \text{MAINTAIN} \ ! \ 0g: \end{aligned}$$

The above discretization of turn rates was chosen to allow realistic control over the decision period considered in this paper.

III. CAS for Conventional Traffic Density

We use dynamic programming to compute pairwise conflict resolution policies similar to the approach for ACAS X [4]. Using the pairwise policies, we apply utility decomposition to approximate the optimal policies for multi-threat conflict resolution.

A. Pairwise Conflict Resolution

A co-altitude pairwise encounter is illustrated in Fig. 1, i.e. there is only one intruder within the sensing range of the ownship. We define the state space, state transition and the reward function for this pairwise encounter below.

1) State Space: The state space for a pairwise encounter is composed of a discrete set of locations, headings, and speeds of the intruder relative to the ownship. A single state s represented by the vector $[\ ; \ ; \ ; v_{\text{own}}; v_{\text{int}}]$, where each dimension of the state is discretized into finite grids.

2) State Transition: The state transition function comes from updating the dynamics of the ownship-intruder pair. We use sigma-point sampling to add noise to speed v and turn rate $\_$ in the dynamics model [28].

3) Reward Function: The objective of the policy is to resolve a conflict while maintaining safety and efficiency. To enforce this trade-off, we discourage aircraft being in close

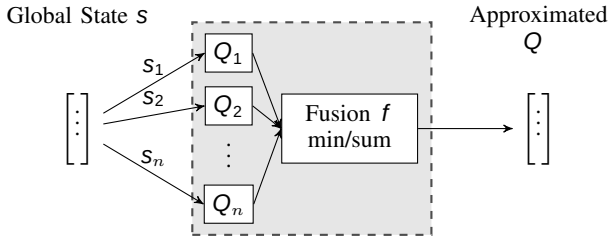

Figure 2. Utility decomposition [24].

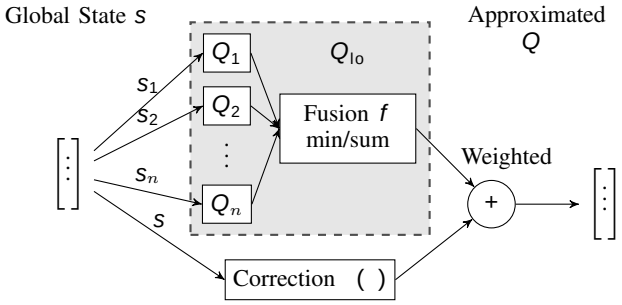

Figure 3. Utility decomposition with correction [24].

proximity to each other, and penalize large and frequent alerts. The reward function is

$$R(s;a) = \quad w \ \exp \ \frac{(\ (s) \quad _{\text{NMAC}})}{\text{NMAC}} \quad w_a \text{turnrate}(a)^2 \quad w_{\text{NMAC}} \ \ f \ (s) \ 6 \ \ _{\text{NMAC}} g \quad w_{\text{conflict}} \ \ f a \ 6= \text{COC} g, \tag{4}$$

where w penalizes close distance between the ownship and the intruder ($_{\text{NMAC}}$ is a predefined threshold for NMAC), w_a penalizes large magnitude of turn rate (the turn rate of COC is defined to be zero), w_{NMAC} penalizes the occurrence of NMAC, w_{conflict} penalizes alerts.

4) Value Iteration: As an MDP, the pairwise conflict problem can be solved using a dynamic programming approach known as value iteration [26]. The idea is to iteratively optimize the state-action value function $Q(s;a)$ for all s and a using the update

$$Q_{k+1}(s;a) \quad R(s;a) + \sum_{s^0} T(s^0;s;a) \max_{a^0} Q_k(s^0;a^0): \tag{5}$$

The result of value iteration is an optimal state-action value function $Q\ (s;a)$.

5) Policy: In the context of pairwise conflict resolution, $Q\ (s;a)$ acts as a numeric table for the ownship, which takes in the state and returns the evaluation on each action. We can extract an optimal policy $\ (s)$ by using the lookup $\ (s) = \arg\max_a Q\ (s;a)$.

B. Multi-threat Conflict Resolution

For conflicts with more than one intruder, the globally optimal solution would involve solving a single multi-agent MDP that takes all the intruders into consideration. However, this approach would be hard to scale since the dimension of the state space would grow exponentially with the number of intruders. Instead, we can combine simple sub-problems to approximate the complete multi-agent solution in a more efficient way.

We use utility decomposition [10], [28] to split a non-cooperative multi-threat conflict resolution problem into pairwise conflict resolution sub-problems. Let $Q_i\ (s_i;a)$ denote the optimal state-action value function of sub-problem i. We assume that the state of the full problem s contains the information needed by the state of each sub-problem s_i. The optimal state-action value function for the full problem $Q\ (s;a)$ can then be approximated by

$$Q\ (s;a) \quad f\ (Q_1(s_1;a);Q_2(s_2;a);::::;Q_n(s_n;a)), \tag{6}$$

where function f performs utility fusion [24]. We consider two approaches to utility fusion. A summation over state-action values which can be written as $Q\ (s;a) \quad \sum_i Q_i\ (s_i;a)$. Another approach is to considers the intruder with the lowest state-action value, i.e. the intruder with the highest threat level $Q\ (s;a) \quad \min_i Q_i\ (s_i;a)$. Taking the minimum value is considered a risk averse strategy [24]. Fig. 2 illustrates the mechanism of utility decomposition and approximation through utility fusion.

The policy of multi-threat conflict resolution can be extracted from the approximated state-action value function $Q\ (s;a)$ by choosing the action with maximum value. When this approach combined with the summation and the minimization approaches above, we refer to them as max-sum and max-min respectively. Prior work has shown that the max-min is superior over max-sum in terms of safety performance[24], [28]. We adopt max-min as our decomposition method.

IV. Collision Avoidance in Dense Airspace

In this section, we outline how collision avoidance can be further improved for operations in dense airspace over existing utility decomposition methods through policy correction.

A. Policy Correction

The formulation of policy correction can be derived from multi-fidelity optimization [24]. When a high-fidelity model (f_{hi}) is too expensive to evaluate, a surrogate model can be used. The surrogate combines a simpler low-fidelity model (f_{lo}) and an additive parametric correction term ( ) to approximate f_{hi} as $f_{\text{hi}} \quad f_{\text{lo}} + \ .$

In the context of multi-threat conflict resolution, the global optimal solution to the full problem $Q\ (s;a)$ is unfeasible to solve for. However, we can get a low-fidelity solution $Q_{\text{lo}}(s;a)$ using utility decomposition. We then add a parameterized correction term to approximate $Q\ (s;a)$ by

$$Q\ (s;a) \quad (1 \quad w_{\text{c}})Q_{\text{lo}}(s;a) + w_{\text{c}}\ (s;a; \ ), \tag{7}$$

where $\ (s;a; \ )$ is the correction term parameterized by , and w_{c} is the weight placed on the correction. Fig. 3 shows the mechanism of adding correction to utility decomposition.

B. Deep Correction Network

We use the deep Q-network (DQN) [29] to learn the parameters for the correction term $(s; a;)$. DQN uses a neural network to approximate the state-action value function of an MDP. It can be expressed as $Q(s; a;)$, where represents the weights of the neural network. The parameters of a DQN policy can be computed by minimizing the cost function J based on the temporal difference:

$$J( ) = \mathsf{E}_{s^0} \quad r + \max_{a^0} Q(s^0; a^0; \quad) \quad Q(s; a; \ )^{\ 2}, \quad (8)$$

where $r = R(s; a)$, and defines a fixed target network to be updated periodically with new parameters . The loss is minimized using experience samples $(s; a; r; s^0)$ that are collected during simulation. The update rule for is

$$\left[r + \max_{a^0} Q(s^0; a^0; \quad) \quad Q(s; a; \ ) \right] r \quad Q(s; a; \ ), \quad (9)$$

where is a configurable hyperparameter known as the learning rate.

By representing the correction as a neural network, we can learn it directly using DQN in a process known as deep correction. We use the utility decomposition policy as a fixed low-fidelity approximation for the optimal multi-threat policy. With some modification, the update rule becomes:

$$\left[r + \max_{a^0} \ (1 \quad w_c) Q_{\text{lo}}(s^0; a^0) + w_c \ (s^0; a^0; \quad) \right.$$
$$\left. (1 \quad w_c) Q_{\text{lo}}(s; a) + w_c \ (s; a; \ ) \right] r \quad (s; a; \ ): \quad (10)$$

We use a simulator to train the correction network using the modified update rule.

C. State Space

When training the deep correction network, we include additional information in the observation as input into the policy. We define the two approaches below.

1) Closest Intruders in Sectors: We coarsely model the sensing area of the aircraft as a circle divided into N sectors [30]. The aircraft observes the closest intruder in each sector, forming N pairwise encounters. We then extract N pairwise encounter states, which are referred to as sub-states. The state for the deep correction network is formed by concatenating the N sub-states in the sector ordering. Fig. 4 illustrates the sensing area being equally divided into four circular sectors, and the closest intruders in each sector are selected for the state. If an sector has no intruder, then the corresponding sub-state is set to empty (zeros).

This formulation encodes the approximate spatial locations of the most significant intruders into the state through predefined sector ordering to help the deep neural network better understand the state.

2) Closest Intruders: Another state formulation method considers the N closest intruders. The position, speed, and heading information of the N closest intruders is concatenated into a single observation sorted by their proximity in ascending order. If there are fewer than N intruders, we take all the existing intruders in to the state and leave the remaining entries of the state empty.

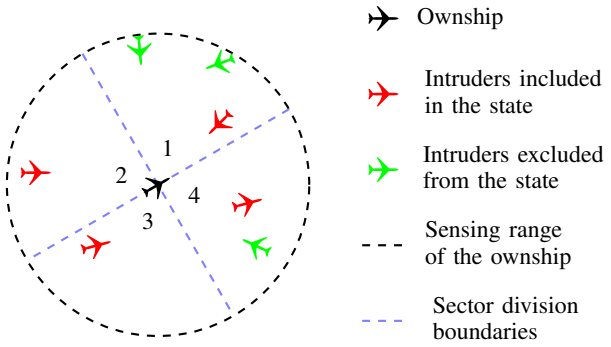

Figure 4. The sensing area equally divided into four sectors. The state is represented by the closest intruder from each sector.

Though a close distance does not necessarily indicate danger, they are highly related. This formulation encodes the ordering of threat level into the state, which could also help the deep neural network better understand the state. This method has a lower chance of having empty state entries than choosing the closest intruders in N sectors, which could be an advantage during training.

3) Destination Information: We add information about the final destination of the aircraft into the observation to encourage more efficient maneuvering. We refer to this additional information as augmented states. The augmented states include:

$_{\text{dest}}$ 2 [;]: The angle of the destination relative to the heading of the ownship.

$_{\text{dest}}$: The distance from the ownship to the destination.

$_{\text{dest, prev}}$: The distance from the ownship to the destination at previous time step.

By including augmented states in the observation, we provide the policy with information that can improve its efficiency.

D. State Transition through Simulation

To collect a large amount of experience samples efficiently, a simulator is developed for training the deep correction network. The simulator has one learning agent as the ownship. Intruders enter the sensing range of the learning agent at angles following the distribution obtained from one million random encounters generated by the Lincoln Laboratory Uncorrelated Encounter Model [31].

During training, the learning agent follows an -greedy policy with correction; while the intruders follow a multi-threat resolution policy using the max-min utility decomposition. We make the intruders' policy stochastic by turning the Q-values into a probability distribution using a softmax function.

The simulation has a predefined episode horizon. In each episode a random destination is given to the ownship. The end of an episode is marked by either the time step reaching the horizon or by the ownship reaching the destination.

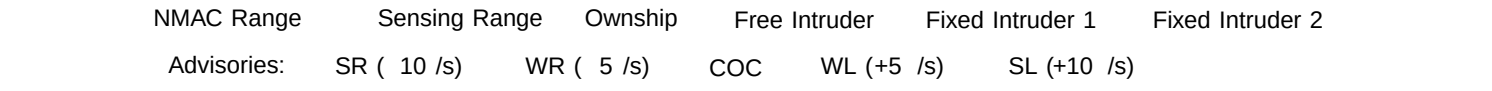

Figure 5. Sample policy slice visualization of different CAS for a four-aircraft encounter scenario. The position and heading of the xed intruders are xed. The free intruder can move to anywhere on the plane of encounter with its heading xed. The xed intruder 1 is at (600 m, 600 m). For the rst row of the policy slices, the xed intruder 2 is at (600 m, 450 m). For the second row of the policy slices, the xed intruder 2 is at (450 m, 450 m).

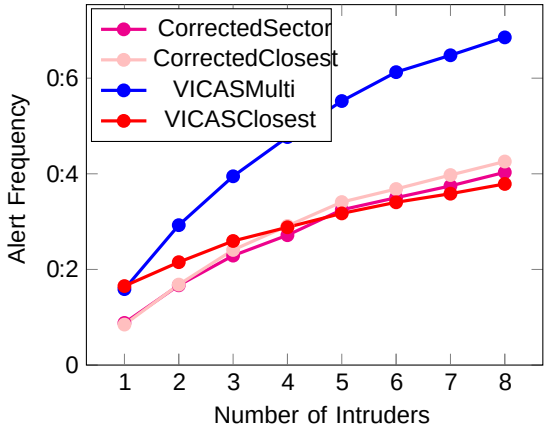

Figure 6. The probability of CAS issuing Non-COC advisories for different numbers of intruders.

E. Reward Function

The reward function for training the correction network is similar to that of the pairwise con ict resolution MDP. The major differences are the reward function for the correction network considers more intruders and it encourages travelling towards the destination. For a correction network that considers at most N intruders, its state s can be written as $s = [s_1; \cdots; s_N; s_{dest}; s_{dest, prev}]$. The reward function is

$$R_c(s,a) = \frac{1}{N}\sum_{i=1}^{N} w \exp\frac{(s_i) - {}_{NMAC})}{{}_{NMAC}} - w_{NMAC}\,1f\ (s_i) 6\ {}_{NMAC}g - w_a\,\text{turnrate}(a)^2 - w_{con\ ict}\,1f\ a \in COCg - w_{digression}(\ {}_{dest} - {}_{dest, prev}) - w_{deviation}\,|\ {}_{desi} - {}_{desi}| + w_{dest}\,1f\ {}_{dest} 6\ DCg, \quad (11)$$

where w , w_{NMAC}, w_a and $w_{con\ ict}$ have the same purposes as described in Section IV-E with adjustable values. The parameter $w_{digression}$ penalizes the digression of the ownship from the destination in terms of the distance from it to the destination, $w_{deviation}$ penalizes the deviation of the ownship's heading from the destination, w_{dest} rewards the ownship in reaching its destination (DC is the criterion judging whether the ownship being close enough to the destination). The reward function is constructed this way so that safety and ef ciency can be balanced.

F. Corrected Policy

With the correction network, the corrected policy $_c$ is extracted

$$_c(s) = \arg\max_a\ [(1 - w_c)Q_{lo}(s,a) + w_c\ (s,a)]\,, \quad (12)$$

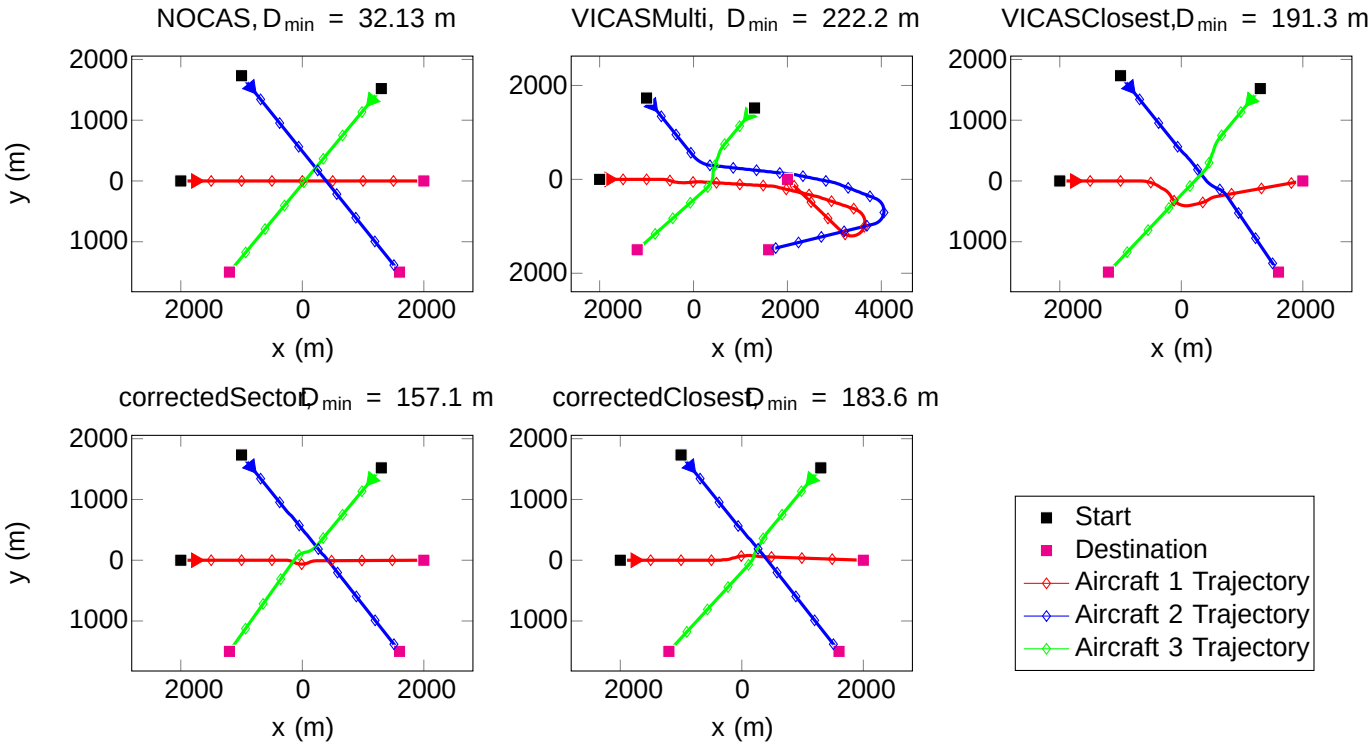

Figure 7. Sample trajectories of a three-aircraft encounter when using different CAS. Each mark on a trajectory indicates ten seconds D_{min} is the minimum distance among the three aircraft.

where Q_{lo} is obtained from utility decomposition.

V. Results

This section compares the safety and efficiency of the following systems:

- VICASMulti: A baseline method based on the CAS computed using value iteration (VICAS) that focuses on pairwise conflict resolution, resolving multi-threat conflicts with max-min utility decomposition.
- VICASClosest: A baseline method similar to VICASMulti, resolving multi-threat conflicts by considering the closest intruder.
- CorrectedSector: Using VICASMulti as the low-fidelity policy, adding the correction term with state space based on the closest intruders in four circular sectors.
- CorrectedClosest: Using VICASMulti as the low-fidelity policy, adding the correction term with state space based on the four closest intruders.

The sensing range of the aircraft is 1,000 m and the NMAC range is defined to be 150 m.

A. Policy Slices Visualization and Policy Sensitivity

One intuitive way of understanding what effects the correction term has on the low-fidelity policy is through visualization. Fig. 5 shows policy slices of different CAS in a four-aircraft encounter. Headings of all the aircraft as well as the positions of the ownship and the two fixed intruders are fixed. The position of the free intruder can be anywhere on the heat map. The heat maps show the advisory the CAS would issue to the ownship in response to the position of the free intruder.

Comparing the policy slices, the alert (non-COC) area of the ownship varies among different CAS. The general effects of the correction term are shaping the alert area more compact and more likely to issue COC. If a CAS is too sensitive, it could issue advisories too frequent. For example in the first row of the policy slices in Fig. 5, in an encounter situation where CorrectedClosest issues COC, VICASMulti issues WR instead. Intuitively, early responses are desirable. However, in a dense airspace, a more winding path means higher chances of encountering more intruders. Being less sensitive does not necessarily imply being less safe. When the intruders get closer, for example shown in the second row of the policy slices in Fig. 5, CorrectedClosest can still issue strong advisories to avoid the threats. A qualitative conclusion we may draw from this is the corrected CAS are less sensitive than the baselines, but still sensitive enough to remain safe. The corrected CAS show higher efficiency in terms of the alert frequency.

To quantify the sensitivity of the CAS, the alert frequency of each CAS given various number of intruders is estimated through a sampling based approach. Fig. 6 shows that the corrected CAS have lower sensitivity than the risk-averse VICASMulti, and have similar sensitivity with the VICASClosest, which only considers the closest intruder.

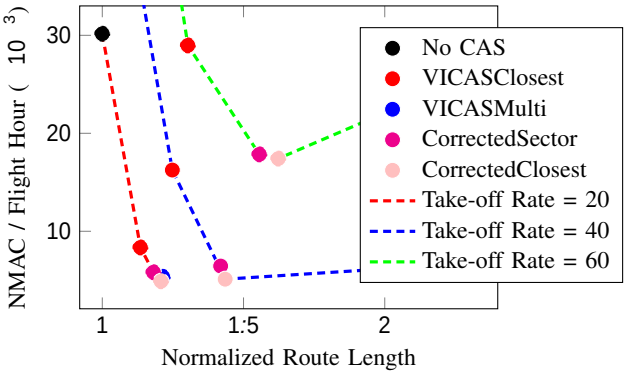

Figure 8. Pareto frontiers between safety and efficiency. The unit for take-off rates is flight / km^2-hr.

B. Trajectory Samples

Sample trajectories of a three-aircraft encounter are visualized in Fig. 7. The encounter is constructed so that when there is no CAS, an NMAC is inevitable. VICASMulti has the the maximum minimum distance between the aircraft with the most winding paths. The corrected CAS, on the other hand, produce the most efficient trajectories with fewer maneuvers. Qualitatively, the example illustrates that the correction term improves the efficiency of the the CAS, while maintaining safety.

C. Safety and Efficiency Evaluation through Airspace Simulations

We evaluate the safety and efficiency of the CAS in the airspace where aircraft take off at various rates. It is a more challenging scenario than the fixed-number encounters.

Airspace simulations are run in a 10 km 10 km airspace. The initial positions and destinations of the aircraft are uniformly sampled in the airspace. Each simulation runs for 5,000 s.

1) Safety: Safety is measured by number of NMACs per flight hour. Table I shows that the NMAC rates of CorrectedClosest are the lowest for all take-off rates. Note that VICASMuilti performs well until the take-off rate exceeds 40 flight / km^2-hr, where the NMAC rates increase dramatically.

2) Efficiency: Efficiency is measured by the ratio between the length of the actual taken path and the nominal distance from the start to the destination, i.e. the normalized route length. Listed in Table I, the normalized route length of VICASMulti dramatically increases when the take-off rate exceeds 40 flight / km^2-hr. This explains the increase in the NMAC rates. As aircraft follow overly winding paths generated by VICASMulti, the chance of NMAC increases. VICASClosest has the lowest the normalized route length at an expense of safety. The normalized route lengths of the corrected CAS increase at a rates proportional to the increase in the taking off rates, while maintaining low NMAC rates. This indicates that unlike VICASClosest, the corrected CAS are able to issue necessary advisories to stay safe; and unlike VICASMulti, efficiency is also considered at high take-off rates. We may say the safety and efficiency of the corrected CAS are balanced.

3) Safety versus Efficiency Trade-off: Fig. 8 shows the trade-off between safety and efficiency. Pareto frontiers are plotted for different take-off rates. We can see that the corrected CAS have the lowest NMAC rates with low normalized route lengths. The corrected CAS are corresponding with the best performing points lie at the bottom left corners of the Pareto frontiers.

4) Impact on Encounter Distribution: One way to capture the average decision burden on a CAS in a given airspace is to consider the number of intruders in a conflict [32]. We extend this notion to a distribution over the average number of intruders in all encounters within the airspace. This metric, called the encounter distribution, provides a general notion of CAS effectiveness in the airspace, as encounters with more intruders will have a higher likelihood of occurring in a dense airspace and are more likely to result in an NMAC.

We observe that a CAS will be less effective when attempting to resolve a conflict with more than one intruder. Therefore, we define an airspace to be dense with regards to the encounter distribution. Formally, a dense airspace is where the expectation of a multi-threat encounter is above some threshold value , $\mathsf{E}[N_{\text{intruder}} > 1]$, where N_{intruder} is the number of intruders in an encounter.

Fig. 9 illustrates the encounter distributions for different take-off rates. At the take-off rate of 5 flight / km^2-hr, the encounter distributions are not greatly impacted by the CAS. Over 50% of the encounters are pairwise. At the take-off rate of 40 flight / km^2-hr, the low efficiency of VICASMulti drives the encounter distribution towards higher numbers of intruders, which increases the average complexity of conflicts in the airspace. The corrected CAS impact the encounter distribution more than VICASClosest does, which can again be explained by the fact that the corrected CAS issue advisories more frequently than VICASClosest to stay safe.

By computing the total variation divergence between the encounter distribution for an airspace without an active CAS and one with an active CAS, we can quantify how a CAS changes the encounter structure of an airspace. Namely, given an encounter distribution for an airspace with no active CAS, P_{No}, and encounter distribution for an airspace with an active CAS, P_{CAS}, the impact of the active CAS on the airspace is

$$D_{\text{TV}}(P_{\text{No}}\ k\ P_{\text{CAS}}) = \frac{1}{2}\sum_{x2\Omega} jP_{\text{No}}(x) \quad P_{\text{CAS}}(x)j, \qquad (13)$$

where is the support of the encounter distribution. The impact on encounter distribution of a CAS is measured by D_{TV}. The results are listed in Table I.

D. Safety and Efficiency Evaluation through Stress Tests

A stress test is designed to further examine the safety and efficiency of the CAS [23]. In the stress test, the number of aircraft is fixed. They are randomly initialized in an annulus with an inner radius of 2,000 m and an outer radius of 4,000 m.

TABLE I. Performance metrics (as mean standard error) for different CAS and taking-off rates.

Metrics	CAS	Take-off Rates (flight / km^2-hr)				
		5	10	20	40	60
NMACs / Flight Hour (10 ^{3})	No CAS	12.07 0.82	16.57 0.40	30.16 0.39	57.57 0.82	85.52 0.85
	VICASMulti	4.83 0.08	4.92 0.25	5.35 0.17	8.18 2.74	92.32 6.36
	VICASClosest	4.97 0.21	6.17 0.15	8.35 0.22	16.25 0.30	28.97 0.57
	CorrectedSector	4.45 0.24	5.77 0.26	5.82 0.11	6.45 0.10	17.85 0.72
	CorrectedClosest	**2.99 0.07**	**4.72 0.14**	**4.90 0.34**	**5.11 0.12**	**17.42 0.35**
Normalized Route Length	No CAS	1.0	1.0	1.0	1.0	1.0
	VICASMulti	1.109 0.006	1.113 0.007	1.214 0.006	3.088 0.191	7.686 0.601
	VICASClosest	1.092 0.003	1.101 0.009	1.135 0.001	1.248 0.004	1.303 0.003
	CorrectedSector	1.095 0.002	1.114 0.008	1.181 0.003	1.419 0.003	1.556 0.008
	CorrectedClosest	1.108 0.003	1.140 0.008	1.208 0.006	1.435 0.003	1.623 0.005
D_{TV} (10 ^{2})	No CAS (ref)	-	-	-	-	-
	VICASMulti	3.142 0.107	3.407 0.091	6.139 0.063	30.638 0.034	41.389 0.189
	VICASClosest	1.106 0.117	1.660 0.087	2.756 0.066	5.749 0.049	7.411 0.086
	CorrectedSector	1.993 0.129	3.070 0.113	4.943 0.064	16.303 0.080	16.383 0.046
	CorrectedClosest	2.416 0.109	2.923 0.085	5.933 0.067	21.064 0.078	28.121 0.074

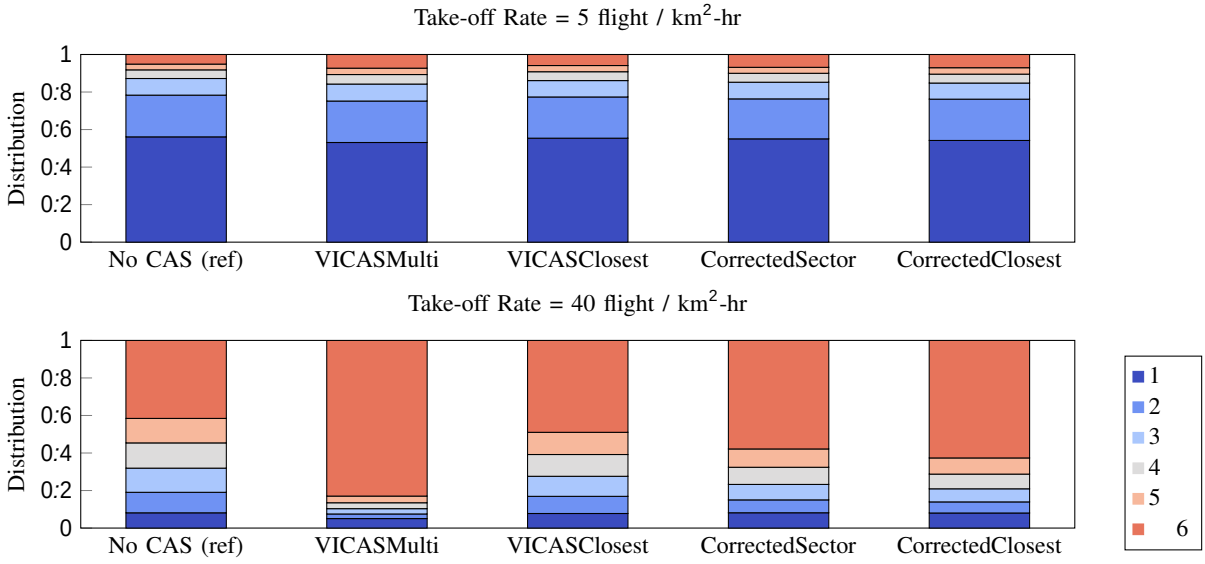

Figure 9. The encounter distribution for different take-off rates.

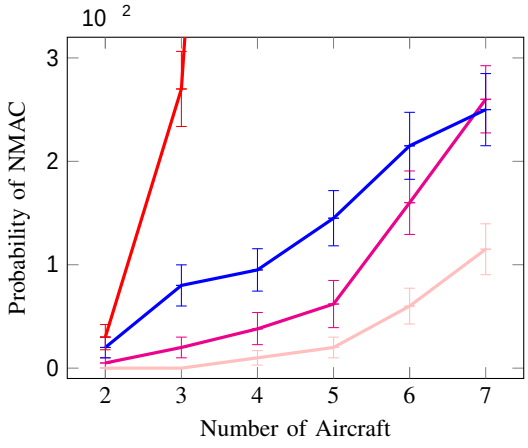

Figure 10. Probability of NMAC when resolving conflicts with different number of aircraft.

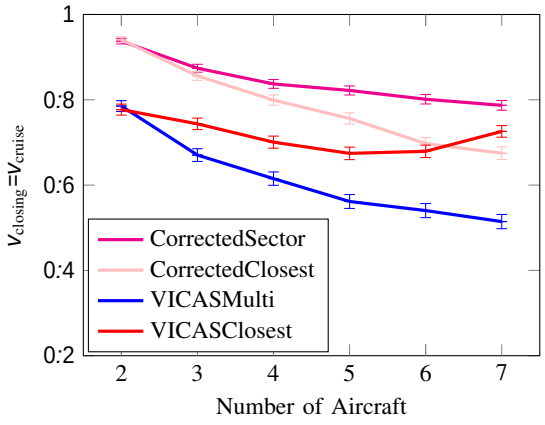

Figure 11. Closing speed / cruise speed when resolving conflicts with different number of aircraft.

TABLE II. NMAC severity (as mean standard error).

CAS	NMAC	Severity
No CAS	0.1894	0.0043
VICASMulti	0.3280	0.0129
VICASClosest	0.2017	0.0162
CorrectedSector	**0.1701**	**0.0090**
CorrectedClosest	0.2001	0.0256

Their initial headings are toward the center of the annulus to make sure the possibility of encountering.

Fig. 10 shows the probability of NMAC given different numbers of aircraft in the stress test. The corrected CAS are safer than the baselines, in which CorrectedClosest performs the best. VICASClosest shows extremely high probability of NMAC when the encounter is more complicated than pairwise.

The ratio between the speed of aircraft getting close to destinations and the cruise speed (speed efficiency, $v_{\text{closing}}/v_{\text{cruise}}$) is tracked in the stress tests as an additional indication of efficiency. Fig. 11 shows that the corrected CAS have superior speed efficiency.

We define the severity of an NMAC as

$$\text{Severity} = \max\{0, 1 - D_{\text{min}}/\text{NMAC Range}\}. \quad (14)$$

The NMAC severity is tracked in the stress tests. The NMAC severity is not strongly correlated with number of aircraft in an encounter. However, it differs among different CAS. Table II shows that CorrectedSector has the lowest NMAC severity among all the CAS, whereas VICASMulti has the highest NMAC severity.

VI. Conclusions and Further Work

In this paper, we assessed the safety and efficiency of CAS operation in dense airspace. We found that operating table-based CAS using utility decomposition is effective in low density airspace, but the performance can be further improved in dense airspace. We applied a correction term trained through deep reinforcement learning on top of the utility decomposition to better approximate an optimal policy for dense airspace. By adding the correction term, we successfully improved the safety and efficiency of CAS performance in both pairwise and multi-threat encounters. The correction term led to emergent behavior in which the CAS balanced its awareness of the risk from intruders and the goal of the operation. The corrected CAS demonstrated superior safety performance with relatively high efficiency and low impact on the encounter distribution of an airspace.

In the future, we could train CAS for multi-threat conflict resolution using deep reinforcement learning from scratch and try other deep reinforcement learning algorithms, such as trust region policy optimization [33] and proximal policy optimization [34]. The relationship between collision avoidance and flight planning could be examined as well, similar to have it has been done in traditional ATM [35]. In addition, a more sophisticated aircraft model could be used in future work.

References

[1] D. Jenkins, B. Vasigh, C. Oster, and T. Larsen, *Forecast of the Commercial UAS Package Delivery Market*. Embry-Riddle Aeronautical University, 2017.

[2] 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*, 2018.

[3] A. Mcfadyen, L. Mejias, P. Corke, and C. Pradalier, "Aircraft collision avoidance using spherical visual predictive control and single point features," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[4] M. J. Kochenderfer, J. E. Holland, and J. P. Chryssanthacopoulos, "Next generation airborne collision avoidance system," *Lincoln Laboratory Journal*, vol. 19, no. 1, pp. 17–33, 2012.

[5] D. Thipphavong, A. Cone, S. M. Lee, and C. Santiago, "Ensuring interoperability between uas detect-and-avoid and manned aircraft collision avoidance," in *USA/Europe Air Traffic Management Research and Development Seminar*, 2017.

[6] E. Mueller and M. J. Kochenderfer, "Multi-rotor aircraft collision avoidance using partially observable Markov decision processes," in *AIAA Modeling and Simulation Conference*, 2016.

[7] T. Williamson and N. A. Spencer, "Development and operation of the traffic alert and collision avoidance system (TCAS)," *Proceedings of the IEEE*, vol. 77, no. 11, pp. 1735–1744, 1989.

[8] M. J. Kochenderfer and J. P. Chryssanthacopoulos, "Robust airborne collision avoidance through dynamic programming," Massachusetts Institute of Technology, Lincoln Laboratory, Project Report ATC-371, 2011.

[9] G. Manfredi and Y. Jestin, "An introduction to ACAS Xu and the challenges ahead," in *Digital Avionics Systems Conference (DASC)*, 2016.

[10] J. P. Chryssanthacopoulos and M. J. Kochenderfer, "Decomposition methods for optimized collision avoidance with multiple threats," *Journal of Guidance, Control, and Dynamics*, vol. 35, no. 2, pp. 398–405, 2012.

[11] K. Julian, J. Lopez, J. S. Brush, M. Owen, and M. J. Kochenderfer, "Policy compression for aircraft collision avoidance systems," in *Digital Avionics Systems Conference (DASC)*, 2016.

[12] J. T. Davies and M. G. Wu, "Comparative analysis of ACAS Xu and DAIDALUS detect-and-avoid systems," National Aeronautics and Space Administration, Technical Memorandum NASA/TM-2018-219773, 2018.

[13] J. Van Den Berg, S. J. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *Robotics Research*, Springer, 2011, pp. 3–19.

[14] A. Mukhtar, L. Xia, and T. B. Tang, "Vehicle detection techniques for collision avoidance systems: A review.,"