One to Any: Distributed Conflict Resolution with Deep Multi-Agent Reinforcement Learning and Long Short-Term Memory

Marc Brittain*
Iowa State University, Ames, Iowa, 50011, USA

Peng Wei†
George Washington University, Washington, D.C., 20052, USA

A novel deep multi-agent reinforcement learning framework is proposed to identify and resolve conflicts between a variable number of aircraft in a high-density, stochastic, and dynamic en route airspace sector. With a growing demand in air transportation, ensuring these dynamic systems are both safe and efficient is essential. Air traffic is becoming denser and more complex, not only in traditional airspace, but also in low altitude airspace. Therefore, we need an autonomous air traffic control system to ensure safe separation requirements in these environments. We propose a Deep Distributed Multi-Agent Variable framework (D2MAV) that utilizes an actor-critic algorithm, Proximal Policy Optimization (PPO) that incorporates a Long Short-Term Memory (LSTM) network to encode a variable number of aircraft states into a fixed length vector. This allows the agents to have access to all aircraft information in the sector in a scalable, efficient way to achieve high traffic throughput under uncertainty. We train the agents using a centralized learning, decentralized execution scheme where one neural network is learned and shared by all agents in the environment. We evaluate our framework via simulation in the BlueSky air traffic control environment.

I. Introduction

A. Motivation

To guarantee air traffic control (ATC) safety and efficiency, with the fast global air traffic growth and expected high density air traffic is a critical challenge. The tactical decisions being made today by human air traffic controllers have experienced little change in en route sectors over the past 50 years [1]. The Advanced Airspace Concept (AAC), developed by Heinz Erzberger and his NASA colleagues, laid the foundation for autonomous air traffic control by developing tools such as the Autoresolver and TSAFE to augment human controllers for increased airspace capacity and operation safety in conflict resolution [2–4]. Inspired by Erzberger, we believe that a fully automated ATC system is the ultimate solution to handle the high-density, complex, and dynamic air traffic in the future en route and terminal airspace.

Many recent proposals for low-altitude airspace operations, such as the UAS Traffic Management (UTM) [5], U-space [6], and urban air mobility [7], require an autonomous air traffic control system to provide advisories or alerts to these intelligent aircraft, facilitate on-board autonomy or human operator decisions, and cope with high-density air traffic, while maintaining safety and efficiency [7–13]. According to [14], the key to these low-altitude airspace operations is to design the autonomous ATC with structured airspace to achieve the envisioned high throughput. Therefore, it is essential to design an autonomous air traffic control system that is able to provide real-time advisories to the aircraft to ensure safe separation both along air routes and at intersections. Furthermore, we require this autonomous ATC system to be able to manage multiple intersections and handle uncertainty in a tactical manner.

Designing such a system is a critical challenge. We need a model to comprehend the current air traffic situation to provide advisories to the aircraft in an efficient and scalable approach, without sacrificing any important information from the environment. One promising way to solve this problem is through reinforcement learning. The goal of reinforcement learning is to allow an agent to learn an optimal policy through interaction with an environment. The

*Ph.D. Candidate, Department of Aerospace Engineering.
†Assistant Professor, Department of Mechanical and Aerospace Engineering, AIAA Senior Member.

agent first perceives the state of the environment, selects an action based on the perceived state, and receives a reward based on this perceived state and action. By formulating the tasks of human air traffic controllers as a reinforcement learning problem, we can obtain dynamic real-time conflict resolution advisories to the aircraft with little computation overhead.

Artificial intelligence (AI) algorithms are achieving performance beyond humans in many real-world applications today. AlphaStar, an AI agent built by DeepMind in 2019, is able to defeat the world's top professionals in StarCraft II, which is a highly complex and strategic game [15]. This notable advance in the AI field demonstrated the theoretical foundation and computational capability to potentially augment and facilitate human tasks with intelligent agents and AI technologies. However, to utilize such techniques, we need a fast-time simulator to allow the agent to interact with. In our work, we use the BlueSky air traffic control simulator [16] as our environment for performance evaluation of our proposed framework.

In this paper, a deep multi-agent reinforcement learning framework is proposed to enable autonomous air traffic separation in en route airspace to mitigate conflicts, where each aircraft is represented by an agent. Our framework is able to handle a variable number of aircraft in the sector and does not scale with the number of intersections. To do this, we use Long Short Term Memory (LSTM) networks [17] to encode information about the environment into a fixed length vector. This provides critical information about the environment without having to select which aircraft to consider. Our proposed framework provides a promising potential solution to enable an autonomous air traffic control system.

B. Related Work

Ground transportation research has already explored the use of deep reinforcement learning in the form of traffic light control [18, 19]. These approaches place the agent at the intersection and allow the agent to control the traffic lights in manner that reduces delay. Our problem is similar to ground transportation in the sense we want to provide speed advisories to aircraft to avoid conflict, in the same way a traffic light advises cars to stop and go. The main difference with our problem is that we need to control the speed of each aircraft to ensure there is no along-route conflict. In our work, we represent each aircraft as an agent instead of the intersection to handle along route and intersection conflicts.

There have been many important contributions to the topic of autonomous air traffic control. One of the most promising and well-known lines of work is the Autoresolver designed and developed by Heinz Erzberger and his NASA colleagues [2–4]. It employs an iterative approach, sequentially computing and evaluating candidate trajectories, until a trajectory is found that satisfies all resolution conditions. The candidate trajectory is then output by the algorithm as the conflict resolution trajectory. The Autoresolver is a physics-based approach that involves separate components of conflict detection and conflict resolution. It has been tested in various large-scale simulation scenarios with promising performance.

Strategies for increasing throughput of aircraft while minimizing delay in high-density sectors are currently being designed and implemented by NASA. These works include the Traffic Management Advisor (TMA) [20] or Traffic Based Flow Management (TBFM), a central component of ATD-1 [21]. In this approach, a centralized planner determines conflict-free time slots for the aircraft to ensure separation requirements are maintained at the metering fix. The main difference with our work is that we are dealing with multiple intersections instead of merging points, so the aircraft maintain their route and do not change at the intersections. The second difference is that our method is a decentralized framework that can handle uncertainty. In TMA or TBFM, once the arrival sequence is determined and the aircraft are within the "freeze horizon" no deviation from the sequence is allowed, which could be problematic if one aircraft becomes uncooperative.

Conflict resolution techniques have also been investigated by multi-agent approaches [22, 23]. In this line of work, negotiation techniques are introduced to resolve identified conflicts in the sector. In our research, we do not impose any negotiation techniques, but leave it to the agents to learn negotiation techniques through learning and training. In previous work, we show that a decentralized formulation is able to resolve conflicts at an intersection, but this formulation only holds when the agent has access to the state information of the N-closest agents. Where the hyper-parameter N needs to be selected and tuned through experimentation, which limits the transferability of the network architecture to new environments. In our research, we allow the agent to have access to all aircraft state information and use an LSTM to encode the information into a fixed length vector to handle a variable number of agents.

Challenging games such as Go, Atari, Warcraft, and most recently Starcraft II have been played by AI agents with beyond human-level performance [15, 24–26]. These results show that a well-designed, sophisticated AI agent is capable of learning complex strategies under uncertainty. It was also shown in previous work that a hierarchical deep

reinforcement learning agent was able to avoid conflict and choose optimal route combinations for a pair of aircraft [27].

Recently, the field of multi-agent collision avoidance has seen much success in using a decentralized framework in ground robots [28, 29]. In this work, the authors develop an extension to the policy-based learning algorithm (GA3C) that proves to be efficient in learning complex interactions between many agents. We find that the field of collision avoidance can be adapted to conflict resolution by considering larger separation requirements, so our framework is inspired by the ideas set forth by [29].

In this paper, the deep multi-agent reinforcement learning framework is developed to solve the separation problem for a variable number of aircraft for autonomous air traffic control in en route dynamic airspace, where we avoid the computationally expensive forward integration method by learning a policy that can be quickly queried. The results show that our framework has very promising performance.

The structure of this paper is as follows: in Section II, the background of reinforcement learning, policy based learning, and multi-agent reinforcement learning will be introduced. In Section III, the description of the problem and its mathematical formulation of deep multi-agent reinforcement learning are presented. Section IV presents our designed deep multi-agent reinforcement learning framework to solve this problem. The numerical experiments and results are shown in Section V, and Section VI concludes this paper.

II. Background

A. Reinforcement Learning

Reinforcement learning, a branch of machine learning, is one type of sequential decision making where the objective is to learn a policy in a given environment with unknown dynamics. A reinforcement learning problem requires an interactive environment where an agent can select different actions that have an effect. If we let t represent the current time, then the components that make up a reinforcement learning problem are as follows:

- S - The state space S is a set of all possible states in the environment
- A - The action space A is a set of all actions the agent can select in the environment
- $r(s_t, a_t)$ - The reward function determines how much reward the agent is able to acquire for a given (s_t, a_t) transition
- $\gamma \in [0,1]$ - A discount factor determines how far in the future to look for rewards. As $\gamma \to 0$, immediate rewards are emphasized, whereas, when $\gamma \to 1$, future rewards are prioritized.

S contains all information about the environment and each element s_t can be considered a snapshot of the environment at time t. Based on s_t, the agent is able to select an action a_t which will effect the environment. The resulting change in the environment produces an updated state, s_{t+1} and a reward associated from making the transition from $(s_t, a_t) \to s_{t+1}$. How the state evolves from $s_t \to s_{t+1}$ given action a_t is dependent upon the dynamics of the environment, which is often unknown. The reward function needs to be carefully designed to reflect the objective of the environment.

From this formulation, the agent is able to derive an optimal policy in the environment by maximizing a cumulative reward function. Let π represent some policy and T represent the total time for a given environment, then the optimal policy can be defined as follows:

$$\pi^* = \arg\max_{\pi} E[\sum_{t=0}^{T} (r(s_t, a_t)|\pi)]. \tag{1}$$

By designing the reward function to reflect the objective in the environment, the optimal solution can be obtained by maximizing the total reward.

B. Policy-Based Learning

Reinforcement learning can be broken down into value-based and policy-based algorithms. In this work, we consider a policy-based reinforcement learning algorithm as these algorithms are able to learn stochastic policies, unlike value-based approaches. This is especially beneficial in non-communicating multi-agent environments, where there is uncertainty in other agent's actions. Proximal Policy Optimization (PPO) is a recent policy-based algorithm that uses a neural network to approximate both the policy (actor) and the value (critic) [30]. PPO improved upon previous approaches such as A3C [31] by limiting the change from the previous policy to the new policy and has been shown to lead to better performance [30]. If we let $r_t(\theta)$ denote the probability ratio and θ represent the neural network weights at time t, then:

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}. \tag{2}$$

We can then formulate the PPO loss function for the actor and critic as follows:

$$L_\pi(\theta) = E_t[\min(r_t(\theta)(A), clip(r_t(\theta), 1-\epsilon, 1+\epsilon)(A))] + \beta \cdot H(\pi(s_t)) \tag{3}$$

$$L_v = (A)^2, \tag{4}$$

where $A := R_t - V(s_t)$ and ϵ is a hyperparameter that determines the bound for $r_t(\theta)$. The second term in (2); $\beta \cdot H(\pi(s_t))$ is used to encourage exploration by discouraging premature convergence to sub-optimal deterministic policies. Here H is the entropy and the hyperparameter β controls the strength of the entropy regularization term. In (3), the critic is trained to approximate the future discounted rewards:

$$R_t = \sum_{i=0}^{k-1} \gamma^i r_{t+i} + \gamma^k V(s_{t+k}). \tag{5}$$

C. Multi-Agent Reinforcement Learning

While single agent reinforcement learning considers one agent's interaction with an environment, multi-agent reinforcement learning is concerned with a set of agents that share the same environment [32]. Fig. 1 shows the progression of a multi-agent reinforcement learning problem. Each agent has its own goals that it is trying to achieve in an environment that may be unknown to the other agents. The difficulty of learning useful policies greatly increases in these problems since the agents are both interacting with the environment and each other. One strategy for solving multi-agent environments is Independent Q-learning [33], where other agents are considered part of the environment and there is no communication between agents. This approach often fails since each agent is operating in the environment and in return, results in learning instability. This learning instability is caused by the fact that each agent is changing its own policy and how the agent changes this policy will influence the policy of the other agents [34].

III. Problem Formulation

In en route and terminal sectors, air traffic controllers are responsible for ensuring safe separation among aircraft. In our research, we used the BlueSky simulator as our environment [16]. We developed three challenging Case Studies with varying number and orientations of intersections with high-density air traffic to evaluate the performance of D2MAV.

The objective of the Case Studies is to maintain a safe separation between aircraft and resolve conflict for all aircraft in the sector by providing speed advisories. To obtain the optimal solution in this environment, the agents need to maintain safe separation and resolve conflict and every time step in the environment. The designed Case Studies are dynamic, where aircraft enter the sector stochastically, which provides a more difficult challenge for the agents since they need to develop a strategy instead of simply memorizing actions.

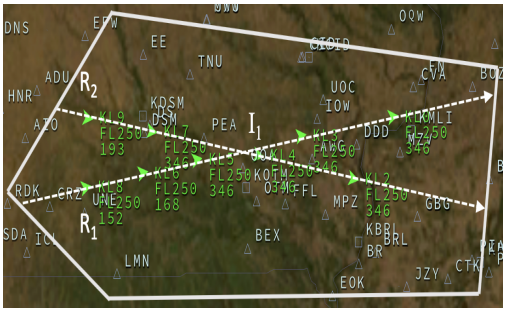
There are many settings we imposed to make the Case Studies feasible. For each simulation run, there is a fixed max number of aircraft. This is to allow comparable performance between simulation runs and to evaluate the final performance of the framework. In BlueSky, the performance metrics of each aircraft type impose different constraints on the range of cruise speeds. We set all aircraft to be the same type, Airbus A320, in these Case Studies. We also imposed a setting that all aircraft can not deviate from their route.

A. Multi-Agent Reinforcement Learning Formulation

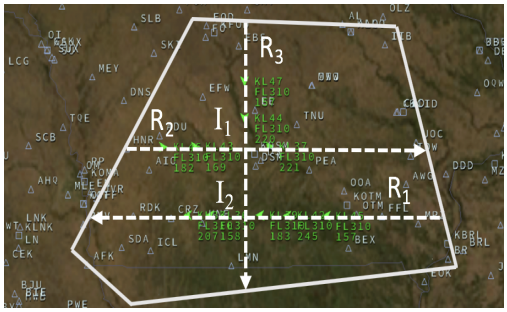
Here we formulate our conflict resolution Case Studies as a deep multi-agent reinforcement learning problem by representing each aircraft as an agent and define the state space, action space, termination criteria and reward function for the agents.

1. State Space

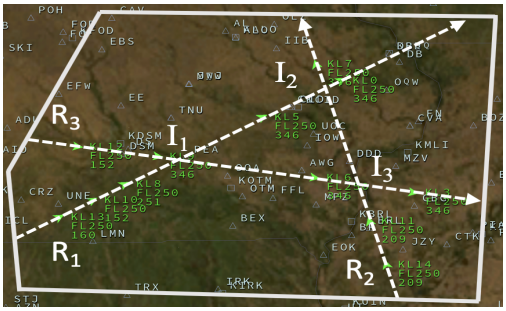
The state contains the information an agent needs to make decisions. Since this is a multi-agent environment, we need to incorporate communication and coordination between the agents. We assume that the position and dynamics of all intruders are available to each agent. We then process the intruder information, sorted by distance to ownship,



(a) Case Study A



(b) Case Study B



(c) Case Study C

Fig. 1 Case Studies designed in the BlueSky air traffic control environment.

through an LSTM network to encode all of the intruder's information into a fixed length vector. This allows the agent to have access to all intruder information without defining a max number of agents to consider. In other words, this approach is able to handle variable number of agents in the environment. By using the LSTM network to encode the intruder information, the LSTM network is trained to know which intruder's information is most important for the ownship to decide.

The state information includes distance to the goal, aircraft speed, aircraft acceleration, a route identifier, and the loss of separation distance, where the position of a given aircraft can be represented as (distance to the goal, route identifier). The intruder information is first processed through an LSTM network to create a fixed length vector. The intruder information includes the distance to the goal, aircraft speed, aircraft acceleration, a route identifier, distance from ownship to intruder, distance from ownship to intersection, and distance from intruder to intersection. Fig. 1 shows an example of a state in the BlueSky environment. The reason for not including the intersection information in the state for the ownship is that for a given route, the ownship may encounter many intersections, so the size of the state space would increase as the number of intersections increases. In addition, an intersection is simply a potential conflict point between two aircraft, so by leaving this information in the intruder information, we can handle this pairwise information in a scalable approach.

We also followed the same rules as in Brittain and Wei [23] for which aircraft are allowed to be in the state of the ownship: the aircraft on a conflicting route must have not reached the intersection and the aircraft must either be on the same route or on a conflicting route. We can then formulate the state and intruder state information for the agents as follows:

$$s_t^o = (d_{\text{goal}}^{(o)}, v^{(o)}, a^{(o)}, r^{(o)}, d^{\text{LOS}})$$

$$h_t^o(i) = (d_{\text{goal}}^{(i)}, v^{(i)}, a^{(i)}, r^{(i)}, d_o^{(i)}, d_{\text{int}}^{(o)}, d_{\text{int}}^{(i)}),$$

where s_t^o represents the ownship state and $h_t^o(i)$ represents the i intruder information that is available to the ownship at time t.

2. Action Space

Every 12 seconds, radar in en route airspace sends updates about aircraft positions [35], therefore our AI agents were only allowed to select an action every 12 seconds in simulation time. The actions available were: increase desired speed (accelerate), decrease desired speed (decelerate), or hold the current speed (no acceleration). The action space for the agents can be defined as follows:

$$A_t = [a_-, 0, a_+],$$

where a_- is to decelerate (decrease speed), 0 is no acceleration (hold current speed), and a_+ is positive acceleration (increase speed).

3. Terminal State

The episode terminated when all aircraft had exited the sector ($N_{\text{aircraft}} = 0$).

4. Reward Function

Cooperation was encouraged by defining identical reward functions for all agents. We define a conflict as the distance between any two aircraft is less than 3 nautical miles ($d^{\text{LOS}} = 3$). The reward needed to be designed to reflect the goal of this paper: safe separation and conflict resolution. We were able to capture our goals in the following reward function for the agents:

$$r_t = \begin{cases} -1 & \text{if } d_o^c < d^{\text{LOS}} \\ -\alpha + \delta \cdot d_o^c & \text{if } d_o^c < 10 \text{ and } d_o^c \geq d^{\text{LOS}} \\ 0 & \text{otherwise} \end{cases},$$

where d_o^c is the distance from the ownship to the closest aircraft in nautical miles, and α and δ are small, positive constants to penalize agents as they approach the loss of separation distance. By defining the reward to reflect the distance to the closest aircraft, this allows the agent to learn to select actions to maintain safe separation requirements.

IV. Solution Approach

We designed and developed a novel deep multi-agent reinforcement learning framework called the Deep Distributed Multi-Agent Variable framework (D2MAV). In this section, we introduce and describe the framework, then we explain why this framework is needed to solve the Case Studies.

To formulate this environment as a deep multi-agent reinforcement learning problem, we utilized a centralized learning with decentralized execution framework with one neural network where the actor and critics share layers of the same neural network, further reducing the number of trainable parameters. By using one neural network that is shared by all agents, we can train a model that improves the joint expected return of all agents in the sector, which encourages cooperation. Our reinforcement learning algorithm is a policy-based approach, PPO, which is shown to perform well across a range of challenging environments [30]. Fig. 2 shows an illustration of the the neural network architecture.

With this framework, we can implement the neural network's policy to all aircraft at the beginning of each episode. Each aircraft then follows this policy until termination. Since the environment is stochastic, we collect the experiences (state, action, reward, terminal) of five episodes using the same policy to update the neural network. In this formulation, each agent has identical neural networks, but since they are evolving different states, their actions can be different.

V. Numerical Experiments

A. Environment Setting

We used BlueSky air traffic control simulator to test the performance of the D2MAV framework*. By design, when restarting the simulation, all objectives were the same: maintain safe separation and sequencing, resolve conflicts, and minimize delay. Aircraft initial positions and available speed changes did not change between simulation runs.

There were many different parameters that needed to be tuned and selected. We implemented the PPO concept with two hidden layers consisting of 256 nodes. The encoding for the intruder aircraft state information consisted of a LSTM

*Code is available at https://github.com/marcbrittain

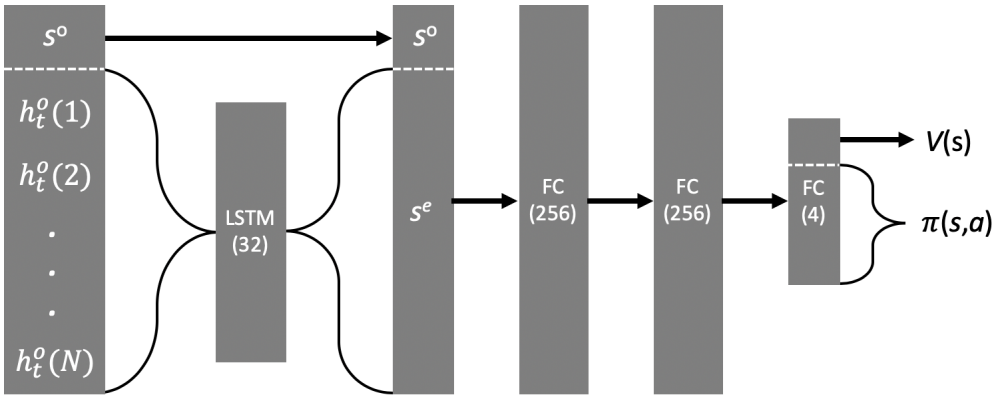


Fig. 2 Illustration of the neural network architecture for PPO with shared layers between the actor and critic. The intruder information is first encoded into a fixed length vector by the LSTM layer before being concatenated to the ownship state information. The concatenated information is then sent through two fully connected layers (FC) of 256 nodes. The policy and value is then obtained from the output fully connected layer of 4 nodes.

layer with 32 nodes. We used the ReLU activation function for all hidden layers except the LSTM for which we used tanh. The output of the actor used a softmax activation function and the output of the critic used a linear activation function. Other key parameter values included: learning rate $lr = 0.0001$, $\gamma = 0.99$, $\epsilon = 0.2$, $\alpha = 0.1$, $\delta = 0.005$, $\beta = 0.0001$, and we used the Adam optimizer for both the actor and critic loss [36].

B. Case Studies

In this work, we developed three challenging Case Studies: A, B, and C as shown in Fig. 1. In our D2MAV framework, the single neural network is distributed to each aircraft as they enter the sector. Each agent is then able to select its own desired speed, which greatly increases the complexity of this problem since the agents need to learn how to cooperate to maintain safe separation requirements.

In each Case Study, aircraft enter the airspace following a uniform inter-arrival distribution from 3 to 6 minutes. This introduces stochasticity into the environment, which increases the difficulty of the problem. In stochastic environments, the agents cannot simply memorize actions, they need to learn a strategy. The episode terminated when all aircraft had exited the sector, so the optimal solution in the Case Studies is to have all aircraft reach their goal. Here a goal is defined as an aircraft exiting the sector without conflict. It is important to note that the difficulty of the Case Studies is based on the inter-arrival times of the aircraft. The inter-arrival time controls the density of the airspace, therefore, if the AI agents develop a strategy based on stochastic inter-arrival times there is no limit on the total number of aircraft to send through the sector.

C. Algorithm Performance

In this section, we analyze the performance of the D2MAV framework on the Case Studies. We allowed the AI agents to train for 100k episodes and then evaluated the final policy for 200 episodes. The performance of the agents during training is shown in Fig. 3. We can see how early on how the agents quickly converge to a sub-optimal policy with increasing performance throughout training. Training for 100k episodes equates to around 5 days of training using an NVIDIA Titan Xp (12GB) graphics card, although we suspect that faster training time can be achieved through further code optimization. We calculate the mean and standard deviation along with the median to evaluate the performance of the final policy as shown in Table 1. We also include a random baseline agent that always selects an action randomly following a uniform distribution. The mean score of the random agent is shown in the random column of Table 1.

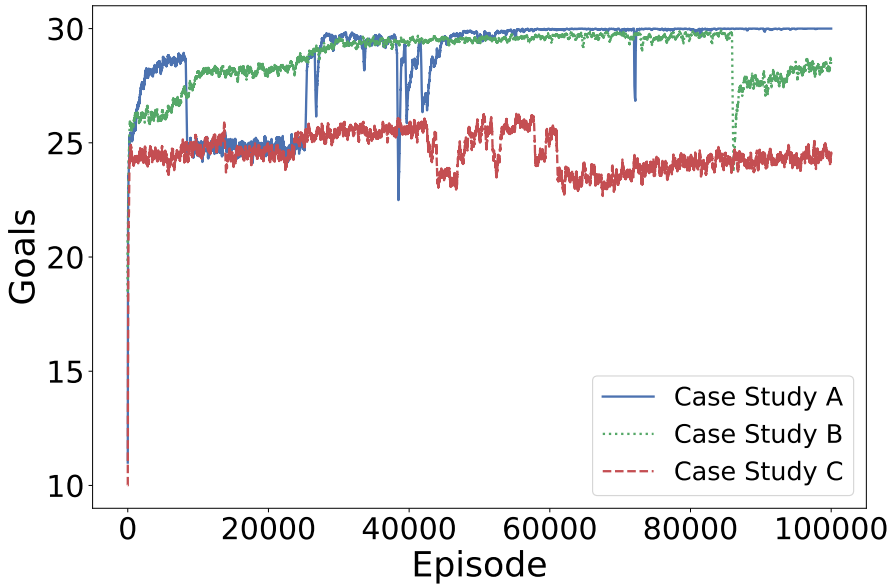


Fig. 3 Performance of the D2MAV framework during training. Results are smoothed with a 200-episode rolling average for clarity.

Table 1 Performance of the policy tested for 200 episodes.

Case Study	Mean	Median	Random
A	30.0 ± 0.0	30.0	12.19 ± 2.832
B	29.25 ± 1.190	30.0	19.74 ± 3.318
C	25.88 ± 2.761	26.0	12.73 ± 3.012

We can see from Table 1 that we obtained near optimal scores on all case studies throughout the 200 episode evaluation phase. This equates to resolving conflict 100%, 98%, and 86% of the time, respectively, for Case Study A, B, and C, both at the intersections and along route. Given that this is a stochastic environment, we speculate that there could be cases where there is an orientation of the aircraft where the 3 nautical mile loss of separation distance can not be achieved, and in such cases we would alert a human ATC to resolve this type of conflict. The median score removes any outliers from our testing phase, and we can see the median score is optimal for Case study A and Case Study B. In comparison with the baseline random agent, we can see that our D2MAV framework was able to achieve a significantly better score than that of the random agent.

While encoding the agents based on distance to the ownship through an LSTM is a promising approach, we suspect that important information may be lost for aircraft that are far from the ownship. One approach to handle this problem is to change the way the intruder information is sorted before processing it through the LSTM. For example, the intruder information could be sorted based on the expected time to the intersection. We leave it to future work to investigate and resolve these types of encounters.

VI. Conclusion

A novel deep multi-agent reinforcement learning framework is proposed to separate en route aircraft as a core component in an autonomous air traffic control system in a structured en route sector. We formulate three Case Studies as deep reinforcement learning problems with the action of selecting a desired speed. We then solve the problem using the D2MAV framework, which is shown to be capable of solving distributed sequential decision making problems with variable number of agents and uncertainties.

According to our knowledge, the major contribution of this research is that we are the first research group to investigate the feasibility and performance of autonomous aircraft separation with a deep multi-agent reinforcement learning framework that incorporates Long Short Term Memory networks to handle variable scalability to enable an automated, safe and efficient en route sector. The promising results from our numerical experiments encourage us to conduct future work on improving the efficiency of learning in complex environments.

Acknowledgements

This research is partially funded by the National Science Foundation under Award No. 1718420, NASA Iowa Space Grant under Award No. NNX16AL88H, and the NVIDIA GPU Grant program.

References

[1] Council, N. R., et al., *Autonomy research for civil aviation: toward a new era of flight*, National Academies Press, 2014.

[2] Erzberger, H., "Automated conflict resolution for air traffic control," 2005.

[3] Farley, T., and Erzberger, H., "Fast-time simulation evaluation of a conflict resolution algorithm under high air traffic demand," *7th USA/Europe ATM 2007 R&D Seminar*, 2007.

[4] Erzberger, H., and Heere, K., "Algorithm and operational concept for resolving short-range conflicts," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, Vol. 224, No. 2, 2010, pp. 225–243.

[5] Kopardekar, P., Rios, J., Prevot, T., Johnson, M., Jung, J., and Robinson, J. E., "Unmanned aircraft system traffic management (UTM) concept of operations," 2016.

[6] Undertaking, S. J., "U-space Blueprint," *SESAR Joint Undertaking. Accessed September*, Vol. 18, 2017.

[7] Mueller, E. R., Kopardekar, P. H., and Goodrich, K. H., "Enabling airspace integration for high-density on-demand mobility operations," *17th AIAA Aviation Technology, Integration, and Operations Conference*, 2017, p. 3086.

[8] Google, "Google UAS Airspace System Overview," , 2017. URL `https://utm.arc.nasa.gov/docs/GoogleUASAirspaceSystemOverview5pager[1].pdf`.

[9] Air, A. P., "Revising the airspace model for the safe integration of small unmanned aircraft systems," *Amazon Prime Air*, 2015.

[10] Kopardekar, P. H., "Safely Enabling Civilian Unmanned Aerial System (UAS) Operations In Low-Altitude Airspace By Unmanned Aerial System Traffic Management (UTM)," 2015.

[11] Balakrishnan, K., Polastre, J., Mooberry, J., Golding, R., and Sachs, P., "Blueprint for the sky," *The roadmap for the safe integration of autonomous aircraft. Airbus A*, Vol. 3, 2018.

[12] Holden, J., and Goel, N., "Fast-forwarding to a future of on-demand urban air transportation," *San Francisco, CA*, 2016.

[13] Uber, "Uber Elevate - The Future of Urban Air Transport," , 2018. URL `https://www.uber.com/info/elevate`.

[14] Hunter, G., and Wei, P., "Service-oriented separation assurance for small UAS traffic management," *2019 Integrated Communications, Navigation and Surveillance Conference (ICNS)*, IEEE, 2019, pp. 1–11.

[15] Vinyals, O., Ewalds, T., Bartunov, S., Georgiev, P., Vezhnevets, A. S., Yeo, M., Makhzani, A., Küttler, H., Agapiou, J., Schrittwieser, J., et al., "Starcraft ii: A new challenge for reinforcement learning," *arXiv preprint arXiv:1708.04782*, 2017.

[16] Hoekstra, J. M., and Ellerbroek, J., "Bluesky atc simulator project: an open data and open source approach," *Proceedings of the 7th International Conference on Research in Air Transportation*, FAA/Eurocontrol USA/Europe, 2016, pp. 1–8.

[17] Hochreiter, S., and Schmidhuber, J., "Long short-term memory," *Neural computation*, Vol. 9, No. 8, 1997, pp. 1735–1780.

[18] Liang, X., Du, X., Wang, G., and Han, Z., "Deep reinforcement learning for traffic light control in vehicular networks," *arXiv preprint arXiv:1803.11115*, 2018.

[19] Genders, W., and Razavi, S., "Using a deep reinforcement learning agent for traffic signal control," *arXiv preprint arXiv:1611.01142*, 2016.

[20] Erzberger, H., and Itoh, E., "Design principles and algorithms for air traffic arrival scheduling," 2014.

[21] Baxley, B. T., Johnson, W. C., Scardina, J., and Shay, R. F., "Air Traffic Management Technology Demonstration-1 Concept of Operations (ATD-1 ConOps), Version 3.0," 2016.

[22] Wollkind, S., Valasek, J., and Ioerger, T., "Automated conflict resolution for air traffic management using cooperative multiagent negotiation," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2004, p. 4992.

[23] Brittain, M., and Wei, P., "Autonomous Separation Assurance in An High-Density En Route Sector: A Deep Multi-Agent Reinforcement Learning Approach," *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, IEEE, 2019, pp. 3256–3262.

[24] Amato, C., and Shani, G., "High-level reinforcement learning in strategy games," *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*, International Foundation for Autonomous Agents and Multiagent Systems, 2010, pp. 75–82.

[25] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M., "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[26] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al., "Mastering the game of Go with deep neural networks and tree search," *nature*, Vol. 529, No. 7587, 2016, p. 484.

[27] Brittain, M., and Wei, P., "Autonomous aircraft sequencing and separation with hierarchical deep reinforcement learning," *Proceedings of the International Conference for Research in Air Transportation*, 2018.

[28] Chen, Y. F., Liu, M., Everett, M., and How, J. P., "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning," *2017 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2017, pp. 285–292.

[29] Everett, M., Chen, Y. F., and How, J. P., "Motion planning among dynamic, decision-making agents with deep reinforcement learning," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2018, pp. 3052–3059.

[30] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O., "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[31] Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K., "Asynchronous methods for deep reinforcement learning," *International conference on machine learning*, 2016, pp. 1928–1937.

[32] Bu, L., Babu, R., De Schutter, B., et al., "A comprehensive survey of multiagent reinforcement learning," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, Vol. 38, No. 2, 2008, pp. 156–172.

[33] Tan, M., "Multi-agent reinforcement learning: Independent vs. cooperative agents," *Proceedings of the tenth international conference on machine learning*, 1993, pp. 330–337.

[34] Matignon, L., Laurent, G. J., and Le Fort-Piat, N., "Independent reinforcement learners in cooperative markov games: a survey regarding coordination problems," *The Knowledge Engineering Review*, Vol. 27, No. 1, 2012, pp. 1–31.

[35] Belobaba, P., Odoni, A., and Barnhart, C., *The global airline industry*, John Wiley & Sons, 2015.

[36] Kingma, D. P., and Ba, J., "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.