Reinforcement learning for two-aircraft conflict resolution in the presence of uncertainty

Pham, Duc-Thinh; Tran, Ngoc Phu; Goh, Sim Kuan; Alam, Sameer; Duong, Vu

2019

Pham, D.-T., Tran, N. P., Goh, S. K., Alam, S., & Duong, V. (2019). Reinforcement learning for two-aircraft conflict resolution in the presence of uncertainty. Proceedings of the 2019 IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF), 1-6. doi:10.1109/RIVF.2019.8713624

https://hdl.handle.net/10356/144709

https://doi.org/10.1109/RIVF.2019.8713624

Reinforcement Learning for Two-Aircraft Conflict Resolution in the Presence of Uncertainty

Duc-Thinh Pham*, Ngoc Phu Tran†, Sim Kuan Goh‡, Sameer Alam§, Vu Duong¶
Air Traffic Management Research Institute, School of Mechanical and Aerospace Engineering
Nanyang Technological University, Singapore
Email: {*dtpham | †phutran | ‡skgoh | §sameeralam |¶vu.duong}@ntu.edu.sg

***Abstract*—Recently, the advances in reinforcement learning have enabled an artificial intelligent agent to solve many challenging problems (e.g. AlphaGo) at unprecedented levels. However, the robustness of reinforcement learning in safety critical operation remains unclear. In this work, the applicability of reinforcement learning in Air Traffic Control was explored. We focus on building an algorithm to automate flight conflict resolution which is an ultimate goal of air traffic control. For that purpose, a simulator, that provides learning environment for reinforcement learning, was developed to simulate a variety of air traffic scenarios. We propose a variant of reinforcement learning approach to resolve conflict in an airspace and investigate the performance of the method in achieving that. Reinforcement learning model, specifically deep deterministic policy gradient, was adopted to learn the conflict resolution with continuous action spaces. Experimental results demonstrate that our proposed method is effective in resolving conflict between two aircraft even in the presence of uncertainty. The accuracy of our model is $\approx 87\%$ at different uncertainty levels. Our findings suggest that reinforcement learning is a promising approach for conflict resolution.**

***Index Terms*—reinforcement learning, air traffic control, deep deterministic policy gradient, conflict resolution**

I. INTRODUCTION

In air transportation, air traffic control (ATC) is one of the core operations in air traffic management. The roles of ATC include preventing flight conflict, managing air traffic flow and providing useful information to ensure efficient flights. However, the ultimate function of an air traffic control (ATC) system is to maintain a safe separation distance, both vertically and horizontally, between any two aircraft at all time. To accomplish this crucial task, Air Traffic Controllers (ATCOs) are warned of any potential loss of separation by tools such as Medium-Term Conflict Detection (MTCD) and Short-Term Conflict Alert (STCA). Once a potential conflict warning is raised, the ATCOs must take appropriate actions to resolve the conflict. Some advanced ATC systems are equipped with conflict resolution tools to provide the ATCOs with resolution advisory. As air traffic is continuously growing with the increasing air passenger demand [1], conflict resolution advisory systems are gaining in importance in aiding controllers to deal with complex conflict scenarios in increasingly busy airspace.

Many approaches in literature have been proposed for conflict resolution systems [2]–[4]; some of them are able to resolve very complex conflict scenarios. Many mathematical models for automatic conflict resolution have been developed while artificial intelligent (AI) solutions are less explored; thus, in this study, we explore the capability of AI in suggesting automated solutions for conflict resolution problem. Following the recent breakthroughs in machine learning, e.g. deep learning, AI gets more attention and closer to one of its prime goal: producing fully autonomous agents. Those agents are able to interact with the environments to make decisions like human with the ability to learn from experiences.

Throughout the literature, reinforcement learning (RL) has been displaying its advantages in learning human behaviors and strategies from stock trading to playing game. Furthermore, the combination of deep learning and reinforcement learning has increased the potential of automation for decision-making problems that were previously intractable because of their high-dimensional state and action spaces. In 2015, Mnih et. al. [5] introduced Deep Q-Network model which could learn to play a range of Atari 2600 video games at a superhuman level, directly from raw image pixels. Secondly, AlphaGo [6], that defeated a human world champion in Go used neural networks that were trained using supervised and reinforcement learning, in combination with a traditional heuristic search algorithm. These two outstanding works in deep reinforcement learning (DRL) field strengthen the belief of community in DRL's ability for problem solving at human level. DRL algorithms are applied in robotics to learn control policies directly from real-world camera [7], [8] and also able to develop agents with ability to adapt to unseen complex visual environment [9]. Differentiating from those works dealing with discrete action space, [10] has tackled the problem of continuous action space by introducing a general-purpose continuous DRL framework, the actor-critic deterministic policy gradient algorithms.

Numerous pioneering works have also adopted machine learning for air traffic management. These works include but not limited to trajectory prediction, delay prediction, and conflict detection [11]–[15]. To the best of our knowledge, nevertheless, no previous study which applies machine learning or reinforcement learning method for conflict resolution has been observed. Loss of separation, or conflict, between any two aircraft occurs when the distance between them is smaller than a standard separation (i.e. 5 nautical miles laterally and 1000 feet vertically) during en-route flight. In this phase, when a potential loss of separation is detected from predicted trajectories, the controller is responsible for giving resolution

978-1-5386-9313-1/19/$31.00 ©2019 IEEE

advisory, or maneuver, to one or both aircraft to resolve the conflict. A maneuver is an action or a series of actions for the pilot to take, including heading change and speed change for lateral conflict resolutions, or flight level change (climb or descend) for vertical conflict resolutions. In general, heading change is practically more preferable for maneuvering since it requires less effort to monitor and verify by the controllers on the radar screen.

In this work, we attempt to apply reinforcement learning for lateral conflict resolution of two aircraft using heading change in the presence of uncertainty, in strategic (planning) phase. Because the maneuver advisory in this phase is being implemented by the aircraft as soon as they enter the airspace sector, our problem could be considered as a strategic planning problem rather than a continuous control problem, which is concerned in the next phase (tactical phase) of air traffic control.

The procedure for training the AI agent is as follows. Conflict scenarios involving two aircraft are generated and presented to the AI agent. The agent, which is driven by reinforcement learning algorithm, learns to resolve these conflicts by applying a number of maneuvers. The agent receives a reward for every maneuver it tried as performance feedback, and the value of the reward depends on the quality of the maneuvers: positive rewards for maneuvers that successfully resolve the conflicts and negative rewards, or penalties, for invalid maneuvers that are unable to separate the aircraft. The learning objective is to maximize the reward and the agent is well trained when it stably gains high rewards for resolving unseen scenarios.

The major contribution of this work is formulating the conflict resolution in strategic phase as a decision-making problem, which can be solved employing reinforcement learning algorithm. To accomplish this, we give special considerations to the following subtasks.

1) Model the planning problem as a single-step game, in which the agent makes a single action at the beginning of every conflict scenario, unlike the continuous control problem reported in [10]. Here, our action space is continuous, which consists of an infinite number of possible actions.
2) Develop a learning environment for flight conflict detection and resolution problem that possesses the following characteristics.
   - The reward function is carefully designed such that not only the conflict status of the scenario but also the quality (e.g. deviation, maneuverability...) of the maneuvers are considered.
   - The deviations of executed maneuvers from the agent's suggested actions are modelled as the results of environmental uncertainty. Higher levels of uncertainty result in greater deviations and cause more negative impacts on the agent's learning progress.
   - The movement and distance of flight trajectories are essential to the representation of scenarios. Thus, we consider 1-D numerical vector for scenarios' state presentation rather than using matrix density of the 2-D visual snapshot of scenarios. This state vector must be carefully designed in order to guarantee the convergence of the training.
3) The learning model is designed to handle multi-dimensional actions with different physical scales and units (e.g. time and distance).

II. Learning Environment

An air traffic simulator is developed to simulate air traffic scenarios and provide an interface to interact with AI agent for training and testing.

A. Conflict scenarios

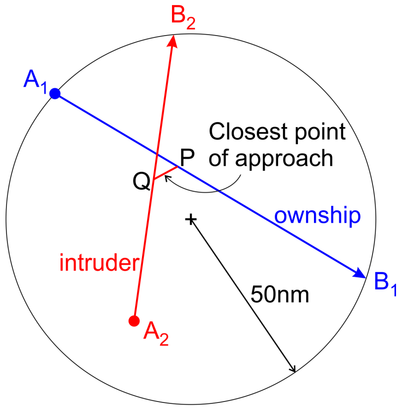

Fig. 1. A conflict scenario involving two aircraft. A_1B_1 is the ownship and A_2B_2 is the intruder. PQ is the closest distance between two aircraft.

In this study, any conflict scenario involves two aircraft called ownship (in blue color) and intruder (in red color) cruising at the same speed v_c in a round-shaped airspace of radius $r = 50$ nautical miles (Fig. 1). Assume that at the moment a potential conflict is detected, the ownship is located at A_1 on the sector border, heading directly to B_1; while the intruder is located at A_2, heading directly to B_2. From that moment, if the two aircraft continue their journeys following the original paths, they will be converging such that the ownship and the intruder will simultaneously reach P and Q. Since the scenario is generated such that the distance PQ is less than 5 nautical miles, the two aircraft is losing their safe separation if none of them takes any maneuver.

Assume that at $t_0 = 0$, the ownship is at A_1 while the intruder at A_2. The velocities of the ownship and the intruder are $\overrightarrow{u} = v_c(\overrightarrow{A_1B_1}/\|\overrightarrow{A_1B_1}\|)$ and $\overrightarrow{v} = v_c(\overrightarrow{A_2B_2}/\|\overrightarrow{A_2B_2}\|)$, respectively. At time $t > 0$, the positions of the the ownship and the intruder are respectively given by $\overrightarrow{P}(t) = \overrightarrow{A_1} + \vec{u}t$ and $\overrightarrow{Q}(t) = \overrightarrow{A_2} + \vec{v}t$, and distance between them renders as $d(t) = \overrightarrow{W_0} + (\vec{u} - \vec{v})t$ where $\overrightarrow{W_0} = \overrightarrow{A_1A_2}$. Minimizing $d(t)$ yields the time to closest point of approach (CPA) as $t_{CPA} = -\overrightarrow{W_0} \cdot (\vec{u} - \vec{v})/\|\vec{u} - \vec{v}\|^2$.

In our experiments, conflict scenarios are randomly generated such that $d_{CPA} \equiv d(t_{CPA}) < 5$ nm and $240 \leq t_{CPA} \leq 480$ seconds, given that the common speed of the aircraft $v_c = 400$ knots (nautical miles per hour). This configuration implies that potential loss of separation between two aircraft is foreseen 4-8 minutes in advanced. Note that in all generated scenarios, the ownship's start point is always located on the border of the circular sector.

B. Maneuver

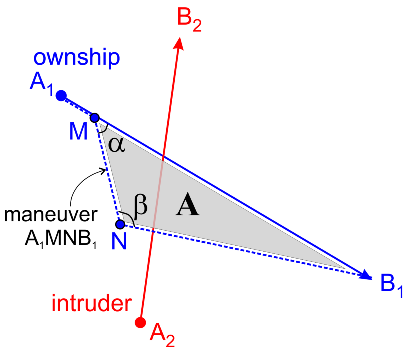

Fig. 2. An example of maneuver for resolving conflict. The ownship makes a heading change $\alpha°$ at point M at $t = t_1$, continues in the new heading MN during t_2 seconds, and heading back towards original end point at return point N. A maneuver is fully defined by a set of three parameters (t_1, α, t_2).

A maneuver, e.g. maneuver A_1MNB_1 in Fig. 2, is defined as a series of actions performed by the ownship: deviate from original path at time t_1 seconds and at location M (measuring from $t_0 = 0$ at A_1) by changing the heading an angle α, and then keep heading along vector $\overrightarrow{MN}$ in t_2 seconds before heading back towards B_1 at return point N. Thus, a maneuver is fully defined by a set of three parameters (t_1, α, t_2). In this study, we assume that any applied maneuver modifies the path of the ownship whilst leaves the intruder's path unchanged, and maneuver parameters always take values from the ranges $0 \leq t_1, t_2 \leq 480$ seconds and $|\alpha| \leq 30°$.

C. Reward function

In reinforcement learning, reward is an environmental feedback that is given to the AI agent for every action it takes. The design of reward function is crucial as it governs not only the convergence time but also the quality of the convergent point of the learning model. In this work, the reward function is designed such that the agent earns highest reward for suggesting a maneuverable resolution that successfully resolves the conflict while maintains a minimal deviation of the maneuver from the original flight path. With this in mind, three independent components that constitute the total reward are considered: (1) separation status of the two aircraft, (2) maneuverability of the resolution, and (3) deviation of the resolution from original path. To ensure the total reward given to the agent is always positive for convenient interpretation, the agent receives a total reward of 5 prior to each action, and it will be punished by a negative reward for every poor action. In particular, total reward given to the agent is $R = 5 + (R_1 + R_2 + R_3)$, taking into account the evaluation of the maneuver as follows.

Separation status: $R_1 = -3$ if the maneuver fails to resolve the conflict, otherwise $R_1 = 0$.

Maneuverability: $R_2 = -2$ if the maneuver's return point N (see Fig. 2) falls outside the circular sector or the return angle $\beta < 120°$; otherwise $R_2 = 0$.

Trajectory deviation: The deviation of the resolution maneuver from the original path is evaluated by the area $\mathbf{A}$ between the resolution path and the original one (see Fig. 2). Let l_1 and l_2 are the lengths of MB_1 and MN, respectively, l_1 and l_2 could be determined from the three parameters (t_1, α, t_2) of the maneuver, and the deviation area is calculated by $\mathbf{A} = l_1 l_2 sin(\alpha)/2$. One can observe that the maximum deviation area that could happens is the area $\mathbf{A_{max}}$ of the equilateral triangle that takes the sector border as its circumcircle: $\mathbf{A_{max}} = (3\sqrt{3}r^2)/4$ where r is the radius of the sector. Employing this, we compute the punishment for large deviation as $R_3 = -\mathbf{A}/\mathbf{A_{max}}$. It should be noted that the punishment for invalid maneuvers—maneuvers that are unmaneuverable or fail to separate the aircraft—is much heavier than punishment for large deviation of valid ones.

$$R_{conflict} = \begin{cases} -3 & CPA_{closure} \leq 5\text{nm} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

$$R_{Maneuver} = \begin{cases} -2 & ||\overrightarrow{ON}|| \geq 50\text{nm or } \beta > 120° \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

$$R_{deviation} = \frac{2 * l_1 l_2 sin(\alpha)}{3\sqrt{3}r^2} \tag{3}$$

$$R_{final} = 5 + R_{conflict} + R_{Maneuver} + R_{deviation} \tag{4}$$

D. Environmental uncertainty

In this study, we consider uncertainty during the implementation of a maneuver, as actual maneuver of the aircraft is always deviated from the controller's command due to some noise. We use three Gaussian noise with different levels of variance to model the noise for each parameter of the action. The variance is reflected by the "uncertainty level": *0* - no uncertainty, *10% of parameter value* - normal uncertainty and *20% of parameter value* - high uncertainty. Moreover, for each level of uncertainty, we study two modes: *0% confidence* (worst case) or *1% confidence* for remaining conflict. In the Air Traffic Management research, *worst case* is always preferred as safety management does not accept any tolerance. However, other setup is introduced here to show the ability of our model in working with probabilistic conflict. In the presence of uncertainty, the separation status of the scenario after applying the maneuver does not simply take value of positive (still have conflict) or negative (conflict eliminated); instead, we compute the percentage of having loss of separation over all possible states of the scenario that resulted from taking the maneuver. If this percentage of loss of separation is below a certain threshold, we consider it as negative, otherwise it is positive.

E. State of scenario

The AI agent does not see the conflict scenarios like human do; instead, it ”perceives” scenarios through state vectors, or the numerical representations of scenarios. A good representation of scenarios helps the agent to better incorporate important features of the scenarios into its decision-making during maneuver computation. At the moment, we represent the scenarios by 1-D numerical vectors, including the following features: separation status, time to CPA, locations of A_1, B_1, A_2, B_2.

III. AI AGENT

The AI Agent is designed to perform the conflict resolution automatically with high *successful rate*. Successful rate is defined as the percentage of conflict scenarios which are successfully resolved. We also target on tackling the large action space for maneuvering an aircraft for given conflict scenario which also is belongs to a continuous and large state space. Thus, firstly our approach is designed to work on large and continuous action space by introducing neural networks as approximators for spaces. Although there is no guarantee for convergence of approximators for actor and critic model, they are mentioned [10] as a necessary approach for generalizing on large action and state spaces. Secondly, off-policy approach is considered in our work to work with large scenario space. Finally, to incorporate the different levels of uncertainty and eliminate any assumption on modelling the environment, model-free reinforcement learning is selected. Based on those characteristics, our AI agent is adapted from deep deterministic policy gradient (DDPG) [10] a variant of actor-critic model with small modifications in actor model to target this challenge. The DDPG algorithm has two models:

1) Actor model: A neural network for learning the mapping deterministically from state to action.
2) Critic model: Another neural network for estimating Q values for all (state, action) pairs.

The Algorithm 1 shows the *Single-Step DDPG Algorithm* for conflict resolution. Two modifications are made to DDPG model to fit our problem formulation. First, since our action has three parameters: (time duration, turning angle, time duration), the dimensionless output (t_1, α, t_2) from the activation function $tanh$ of the actor model within the range $([-1,1],[-1,1],[-1,1])$ must be transformed to the physical spaces within the range $([0,480]$ second, $[-30,30]$ degree, $[0,480]$ second$)$. Second, as we design our problem as a Sing-Step Markov Decision Process, in which the AI agent performs only a single action and receives feedback once for every scenario it encounters, the discounted reward for future action is of little interest. Thus, we set γ to zero in the formula $y_i = r_i + \gamma Q'(s_{s+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$ (line 12 of Algorithm 1), as γ governs the expected gain from future steps in the context of multi-step problem.

Algorithm 1 Single-Step DDPG Algorithm

```
 1: Randomly initialize weight $\theta^Q$ for Critic Net $Q(s,a|\theta^Q)$
 2: Randomly initialize weight $\theta^\mu$ for Actor Net $\mu(s|\theta^\mu)$
 3: Initialize target networks $Q'$ and $\mu'$ with weight from $\theta^{Q'} \leftarrow \theta^Q$, $\theta^{\mu'} \leftarrow \theta^\mu$.
 4: Initialize replay buffer R
 5: for episode = 1, M do
 6:     Initialize a random process $\mathcal{N}$ for action exploration.
 7:     Receive scenario $s$ from Environment
 8:     Select action $a = \mu(s|\theta^\mu) + \mathcal{N}$ according to current policy.
 9:     Execute action $a$, observe reward $r$ and new state $s'$
10:     Store transition $s, a, r, s'$ in R
11:     Sample a random K experiences $s_i, a_i, r_i, s_{i+1}$ from R
12:     Set $y_i = r_i$
13:     Update critic by minimizing the loss:
            $$L = \frac{1}{K}\sum_i (y_i - Q(s_i, a_i|\theta^Q))^2$$
14:     Update actor policy using sampled policy gradient:
            $$\nabla_{\theta^\mu}(J) \approx \frac{1}{K}\sum_i \nabla_{\theta^\mu} Q(s_i, \mu(s_i)|\theta^Q)\nabla_{\theta^\mu}\mu(s_i|\theta^\mu)$$
15:     Update the target networks:
            $$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}$$
            $$\theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}$$
16: end for
```

We also incorporate recent advancements in deep reinforcement learning such as replay memory, batch normalization and soft target update to improve model's stability. Replay memory stores past experiences for batch training and it could improve the independence of samples in the input batch. Batch normalization is employed to deal with different scales and ranges of the input scenarios, and soft target update is a technique for greater learning's stability.

IV. EXPERIMENT SETUP

The training and testing processes are described in Fig. 3, and general computation process in Algorithm 1. Batch training is used to train our model with samples generated from the interactions between ”current agent” (AI agent*) and the environment. At each iteration, the environment randomizes a conflict scenario, presents it to the ”current agent”, and then evaluates the agent's action by a reward. Agent's experiences, in the form of *(Current State* ***S****, Action* ***A****, Reward* ***R****, Next State* ***S'****, Done* ***D****)*, are pushed to the replay buffer. As soon as the replay buffer is filled with at least 5000 samples, batches (S, A, R, S', D) are randomly sampled and provided to the AI agent. Tuples (s_i, a_i, r_i) are used to train critic model in similar manner to training a supervised learning model. Then, the predicted action $\mu(s_i|\theta^\mu)$ is used in combination with critic regression $Q(s_i, \mu(s_i|\theta^\mu)|\theta^Q)$ to compute policy gradient for actor model training. Finally, trained models interact with environment to perform the prediction for testing and collection of new data. Here, we use two fully connected neural networks for actor and critic approximations.

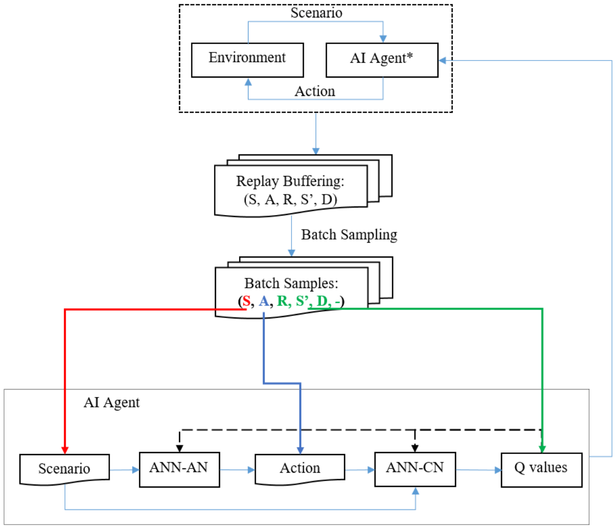

Fig. 3. Flowchart with detail information of AI Agent for Training and Testing.

TABLE I
LIST OF EXPERIMENTS TO ASSESS THE MODEL PERFORMANCE

Number of Flight	Uncertainty	Level	Mode
2	False	0	0
2	True	0.1	0 (Worst Case)
2	True	0.1	0.01
2	True	0.2	0 (Worst Case)
2	True	0.2	0.01

Table I presents the settings of five experimental runs to assess the performance of our algorithm under different circumstances. There are 4 parameters which are considered for the experiments: number of flight (=2), uncertainty (True/False) with 3 levels and 2 modes (mentioned in II-D). For each uncertainty setting, the reinforcement learning model is trained by allowing the agent to interact with the environment and learn from the experiences gained from these interactions. After every 1000 iterations, 1000 random unseen scenarios are solved by the model in order to evaluate its performance in terms of average reward, accuracy (or successful rate) and reward approximation. In each assessment, the average reward is calculated as the total reward the agent earned for solving a thousand scenarios, divided by 1000. Similarly, the successful rate, or accuracy, is equal to the number of scenarios successfully resolved by the agent, divided by 1000. The goodness of the reward approximation is an important performance indicator as it reflects the quality of the critic model, which in turn acts as the guideline for training the actor model. It should be noted that the model's convergence is defined as the stability of the average rewards earned.

V. RESULTS AND DISCUSSION

Fig. 4 presents the convergence of our model under different uncertainty conditions. It could be observed that at all conditions, the model converges after about 80 thousand iterations. The environmental uncertainty, especially in the worst case conditions, has obvious impact on the learning of the model as it significantly reduces the model's converging speed. It is interesting to observe that, although the environmental disturbance does slow down the convergence, it has little impact on the convergent values of the model, as one could see that at all conditions, the model converges to an average reward of approximately 4.65 out of 5. Here, it should be highlighted that this is a very encouraging convergent reward achieved by the agent showing its great balancing strategy, because the maximum reward of 5 could never be accomplished due to the trade-off constraint: The maneuver must be always deviated from the original trajectory while we penalize the agent for generating maneuver's deviation. Fig. 4 suggests that our setting of the learning parameters provides the AI agent with the capability to maintain the quality of the conflict resolutions despite the impact of the defined environmental disturbances.

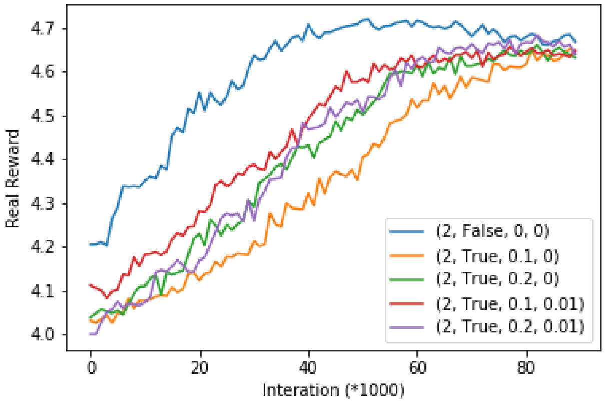

Fig. 4. Convergence of the model in estimating the reward

In Fig. 5, it is shown that our algorithm can achieve the successful rate of $\mathbf{87\% \pm 2\%}$ in resolving two-aircraft conflicts. An interesting observation is the similarity in shapes of the curves presented in Fig. 5 (model's accuracy) and Fig. 4 (average reward). This similarity could be explained as the consequence of the definition of the scoring mechanism, which highly correlates the reward earned and the maneuver's feasibility when the agent resolves conflicts. Another important thing to consider when investigating the model quality is the error in approximating the real reward (by the critic model). Fig. 6 presents this approximating error from the 5000th to 90000th iteration. The first 5000 iterations are not shown because this is the warming up phase which carries little information and very large error in this phase severely affects the scale of the chart. We could observe from Fig. 6 that the error converges to a relatively small value of 0.1 ± 0.05 out of 5 ($\approx \mathbf{2\%}$), which is highly acceptable.

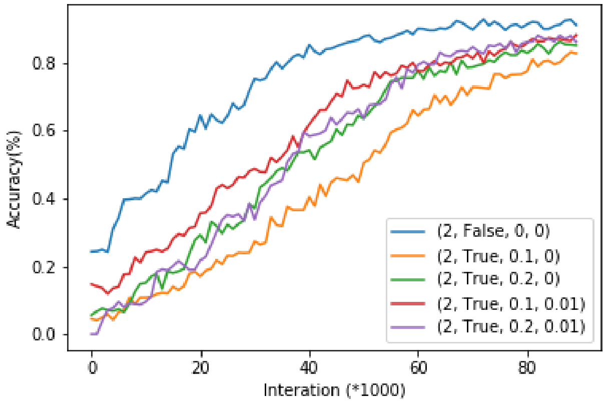

Fig. 5. Model performance evaluated by accuracy (successful rate) at different uncertainty settings

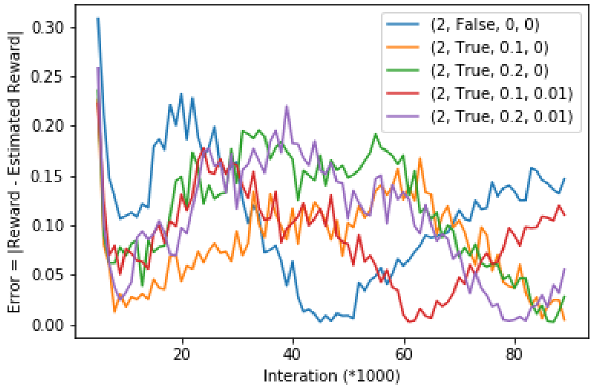

Fig. 6. Error in approximating reward function of the critic model

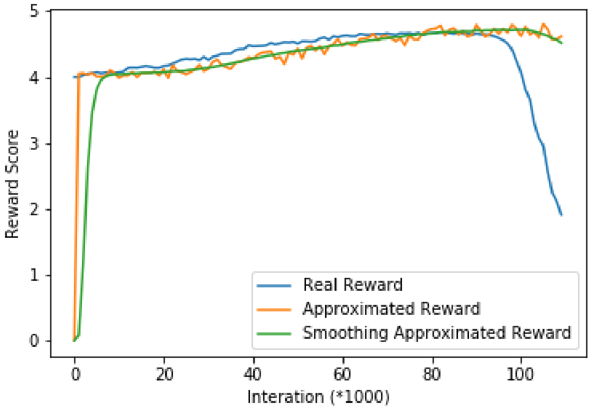

Fig. 7. Model divergence after a certain number of iterations

However, after a certain number of iterations ($> 100,000$), we surprisingly observe the divergence of our model (Fig. 7). The achieved reward of the model rapidly drops even though the estimated rewards (both target and running critic model) are still high. This phenomenon could be caused by a complex mechanism that needs further and throughout investigation to explain.

VI. Conclusion

The contributions of this work include (1) the formulation of conflict resolution problem as a reinforcement learning problem; (2) the development of an air traffic scenario simulator as the environment for reinforcement learning; and (3) the development of an AI agent employing deep deterministic policy gradient algorithm to learn the control actions for conflict resolution. Our results show that reinforcement learning is a promising approach for two-aircraft conflict resolution with the accuracy $\approx \mathbf{87\%}$ in the presence of environmental uncertainty.

As this work is possibly one of the earliest attempts to apply reinforcement learning approach to conflict resolution problem, our learning model obviously has limitations and could be further improved. Possible future considerations include but not limited to the improvement of the convergence of DDPG model for control actions, the investigation of model performance in the scenarios involving more than two aircraft, and the enhancement of state representation of the conflict scenario to help the agent better ”perceive” its learning environment, which could lead to higher model performance.

VII. Acknowledgement

This research has been partially supported under Air Traffic Management Research Institute (NTU-CAAS) Grant No. M4062429.052

References

[1] “IATA forecasts passenger demand to double over 20 years,” https://www.iata.org/pressroom/pr/Pages/2016-10-18-02.aspx, october 2016, accessed: 2018-10-02.

[2] J. K. Kuchar and L. C. Yang, “A review of conflict detection and resolution modeling methods,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, no. 4, pp. 179–189, 2000.

[3] C. Allignol, N. Barnier, N. Durand, and J.-M. Alliot, “A new framework for solving en-routes conflicts,” in *ATM 2013, 10th USA/Europe Air Traffic Management Research and Development Seminar*, 2013, Conference Proceedings, pp. pp 1–9.

[4] M. Radanovic, M. A. Piera Eroles, T. Koca, and J. J. Ramos Gonzalez, “Surrounding traffic complexity analysis for efficient and stable conflict resolution,” *Transportation Research Part C: Emerging Technologies*, vol. 95, pp. 105–124, 2018.

[5] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, no. 7540, p. 529, 2015.

[6] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, “Mastering the game of go with deep neural networks and tree search,” *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[7] S. Levine, C. Finn, T. Darrell, and P. Abbeel, “End-to-end training of deep visuomotor policies,” *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1334–1373, 2016.

[8] S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen, “Learning hand-eye coordination for robotic grasping with large-scale data collection,” in *International Symposium on Experimental Robotics*. Springer, 2016, pp. 173–184.

[9] Y. Duan, J. Schulman, X. Chen, P. L. Bartlett, I. Sutskever, and P. Abbeel, “Rl^2: Fast reinforcement learning via slow reinforcement learning,” *arXiv preprint arXiv:1611.02779*, 2016.

[10] 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,” *arXiv preprint arXiv:1509.02971*, 2015.

[11] A. de Leege, M. van Paassen, and M. Mulder, "A machine learning approach to trajectory prediction," in *AIAA Guidance, Navigation, and Control (GNC) Conference*, Conference Proceedings, p. 4782.

[12] J. Zhang, H. Jiang, X. Wu, and X. Tang, *4D trajectory prediction based on BADA and aircraft intent*, 2014, vol. 49.

[13] W. Dai, Y. Hu, and Y. Zhao, "A study on the mechanism and prediction of flight delay with ads-b surveillance data," 2018.

[14] Z. Wang, M. Liang, and D. Delahaye, "A hybrid machine learning model for short-term estimated time of arrival prediction in terminal manoeuvring area," *Transportation Research Part C: Emerging Technologies*, vol. 95, pp. 280–294, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0968090X18310167

[15] D. D. Zhengyi Wang, Man Liang, "Short-term 4d trajectory prediction using machine learning methods," 2017. [Online]. Available: https://hal-enac.archives-ouvertes.fr/hal-01652041