AN INTELLIGENT INTERACTIVE CONFLICT SOLVER INCORPORATING AIR TRAFFIC CONTROLLERS' PREFERENCES USING REINFORCEMENT LEARNING

Ngoc Phu Tran, Duc-Thinh Pham, Sim Kuan Goh, Sameer Alam, Vu Duong
Air Traffic Management Research Institute,
School of Mechanical and Aerospace Engineering,
Nanyang Technological University, Singapore

Abstract

The increasing demand in air transportation is pushing the current air traffic management (ATM) system to its limits in the airspace capacity and workload of air traffic controllers (ATCOs). ATCOs are in an urgent need of assistant tools to aid them in dealing with increased traffic. To address this issue, the application of artificial intelligence (AI) in supporting ATCOs is a promising approach. In this work, we build an AI system as a digital assistant to support ATCOs in resolving potential conflicts. Our system consists of two core components: an intelligent interactive conflict solver (iCS) to acquire ATCOs' preferences, and an AI agent based on reinforcement learning to suggest conflict resolutions capturing those preferences. We observe that providing the AI agent with the human resolutions, which are acquired and characterized by our intelligent interactive conflicts solver, not only improves the agent's performance but also gives it the capability to suggest more human-like resolutions, which could help increase the ATCOs' acceptance rate of the agent's suggested resolutions. Our system could be further developed as personalized digital assistants of ACTOs to maintain their workloads manageable when they have to deal with sectors with increased traffic.

Introduction

Air traffic congestion is among the major concerns in the development of the future air traffic management (ATM) system, as the continuous growth of the air transportation demand [1] is leading to more traffic being introduced to the airspace while the airspace capacity remains unchanged. Traffic congestion not only causes delays but also results in more potential flight conflicts, which could pose a threat to the system safety. To improve the current system, new capacity will be necessarily created. However, airspace capacity design is heavily controlled by human factor, i.e. the workload of the air traffic controller (ATCO) in charge of the sector; therefore, additional airspace capacity must come with the development of assistant tools for the ATCOs, especially in giving conflict resolution advisories.

Conflict resolution advisories are necessarily given by the ATCOs (to the pilots) when potential conflicts are detected. A 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). Many mathematical models have been developed as conflict resolution advisory tools for the ATCOs [2-8]. Although these models had been reported with positive results, they suffer several common limitations. First, complete knowledge of the mapping from conflict scenarios to maneuvers is required and the input scenarios must be well standardized for the mathematical models to work properly. These make mathematical models highly complex, less flexible, and could only well perform on standardized scenarios. Second, these mathematical models do not possess self-learning ability, i.e. the ability to self-evolve when dealing with unseen and non-standard scenarios. More importantly, automated conflict resolutions generally get low acceptance rate from ATCOs [9,10], and one reason is that the resolutions suggested by mathematical models fail to incorporate the ATCOs' strategic preferences in resolving conflicts.

In this work, we attempt to build a conflict resolution advisory system that is capable of incorporating human preferences in its resolutions. Our system has two major components: (1) an interactive conflict solver (iCS) for acquiring and characterizing human resolutions; and (2) an artificial intelligent agent that learns to resolve conflict incorporating the characteristics of human resolution acquired by the iCS. The iCS presents conflict

978-1-7281-1893-2/19/$31.00 ©2019 IEEE

scenarios to ATCOs, observes ATCOs' behaviors when they are resolving conflicts, and characterizes ATCOs' resolutions. The AI agent is trained using reinforcement learning (RL) algorithm such that it could learn the characteristics of ATCOs' preferred resolutions provided by the iCS, and then suggest conflict resolutions that incorporate the learned ATCOs' preferences.

Methodology

In this section, we describe our approach to build the AI agent that suggests conflict resolutions incorporating ATCOs' preferences. First, we describe the conflict scenarios being considered in our experiments. Second, we explain the iCS for characterizing the resolutions provided by the ATCOs. Then, we present the reinforcement learning algorithm for the training of the agent such that the agent is able to suggest human-like resolutions.

Conflict Scenario

We define a conflict scenario as a traffic scenario that occurs within a circular area of interest (or interested airspace) of radius r, in which there is one pair of potential conflicts between an ownship and an intruder aircraft, in the presence of surrounding traffic (Figure 1). We assume no conflict among the surrounding aircraft; in other words, a conflict always occurs between the ownship and another aircraft in the interested airspace. Let n be the number of aircraft in the interested airspace when a potential conflict is being considered, $\mathbf{A}_i$ denote the locations of the aircraft at the moment the conflict scenario presented to the agent ($t_0 = 0$), and $\mathbf{B}_i$ the locations where the aircraft exit the interested airspace ($0 \le i < n$) (Figure 1). Consequently, $\mathbf{A}_i\mathbf{B}_i$ represent the aircraft's trajectories and $\overleftarrow{\mathbf{A}_i\mathbf{B}_i}$ are the initial headings of the aircraft. If the aircraft continue their journeys with this original flights plan, the ownship (following route $\mathbf{A}_0\mathbf{B}_0$) and the intruder (following route $\mathbf{A}_1\mathbf{B}_1$) are converging; they will simultaneously reach **P** and **Q**. Since the scenario is generated such that the distance **PQ** is less than d_{sep}, which is the safe separation to maintain, the two aircraft are losing their safe separation if none of them takes any maneuver. Here, **PQ** is called the closest point of approach (CPA) between the ownship and the intruder, also denoted by the CPA closure vector $\overleftarrow{d_1}$ from **P** to **Q**. Similarly, the CPAs between the ownship and the surrounding aircraft are denoted by $\overleftarrow{d_i}$ where $2 \le i < n$. Note that at the beginning, $\| \overleftarrow{d_1} \| < d_{\text{sep}}$ while $\| \overleftarrow{d_i} \| \ge d_{\text{sep}}$, $2 \le i < n$; this imposes the single initial conflict condition to the generated scenarios, which is the interest of this work.

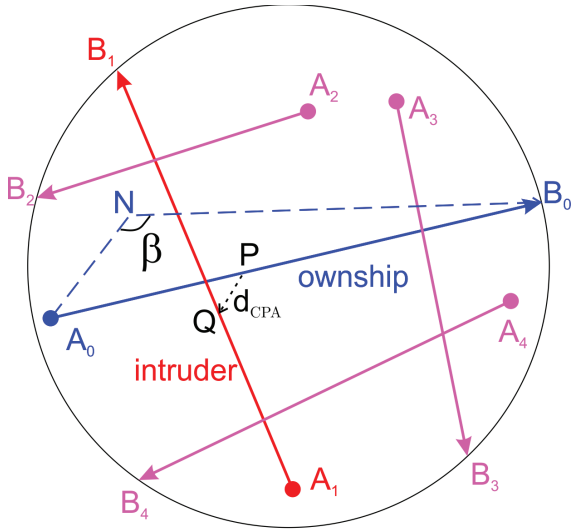

Figure 1. A traffic scenario with a potential conflict between the Ownship and the Intruder, in the Presence of Surrounding Aircraft

Assume that all aircraft are cruising at the same speed of v_c. At $t_0 = 0$, the ownship is at $\mathbf{A}_0$ and the intruder $\mathbf{A}_1$. The velocities of the ownship and the intruder are $\vec{u} = v_c(\overleftarrow{\mathbf{A}_0\mathbf{B}_0}/\| \overleftarrow{\mathbf{A}_0\mathbf{B}_0} \|)$ and $\vec{v} = v_c(\overleftarrow{\mathbf{A}_1\mathbf{B}_1}/\| \overleftarrow{\mathbf{A}_1\mathbf{B}_1} \|)$, respectively. At a time $t > 0$, the locations of the the ownship and the intruder are respectively given by $\vec{P}(t) = \overleftarrow{\mathbf{A}_0} + \vec{u}t$ *and* $\vec{Q}(t) = \overleftarrow{\mathbf{A}_1} + \vec{v}t$, and distance between them renders as $d_1(t) \equiv \| \overleftarrow{d_1} \| = \overleftarrow{W_0} + (\vec{u} - \vec{v})t$ where $\overleftarrow{W_0} = \overleftarrow{\mathbf{A}_0\mathbf{A}_1}$. Minimizing $d_1(t)$ yields the time to CPA as $t_{\text{CPA}} = -\overleftarrow{W_0} \cdot (\vec{u} - \vec{v})/\| \vec{u} - \vec{v} \|^2$, and the closure at CPA as $d_{1(\text{CPA})} = d_1(t_{\text{CPA}})$.

Maneuver and Interactive Conflict Solver (iCS)

To prevent the conflict, the ownship must take a maneuver $\mathbf{A}_0\mathbf{N}\mathbf{B}_0$ through the trajectory change point (TCP) **N**, as shown by the blue-dashed curve in Figure 1. The TCP **N** must be carefully decided such that the maneuver $\mathbf{A}_0\mathbf{N}\mathbf{B}_0$ should successfully resolve the primary conflict without giving rise to any secondary conflict with the surrounding aircraft. Here,

we consider heading change as the only maneuver for conflict resolution.

The ultimate goal of this work is to train the AI agent such that it is able to suggest human-like conflict resolution. Thus, it is necessary to collect conflict resolution performed by human as demonstrations for the agent during training. To accomplish this, we develop an interactive conflict solver that allows human to input the TCP for each conflict scenario.

Figure 2 shows the screenshot of the interactive conflict solver. The iCS is developed as a web-based application with an interactive interface that allows user to input resolutions. Conflict scenarios are generated and stored in the computer hard disk. The iCS loads the conflict scenarios and presents one by one to the ATCO. The conflict between the ownship and the intruder is indicated so that the ATCO does not need to perform conflict detection, but focus on providing a conflict resolution for every scenario. To input a resolution, the ATCO is to perform a mouse-click on the ownship's trajectory to add a TCP, and then drag the TCP to search for a good resolution. The separation status of the ownship is continuously updated and shown on the interface as the ATCO is dragging the TCP.

Figure 2. A Screenshot of the Interactive Conflict Solver

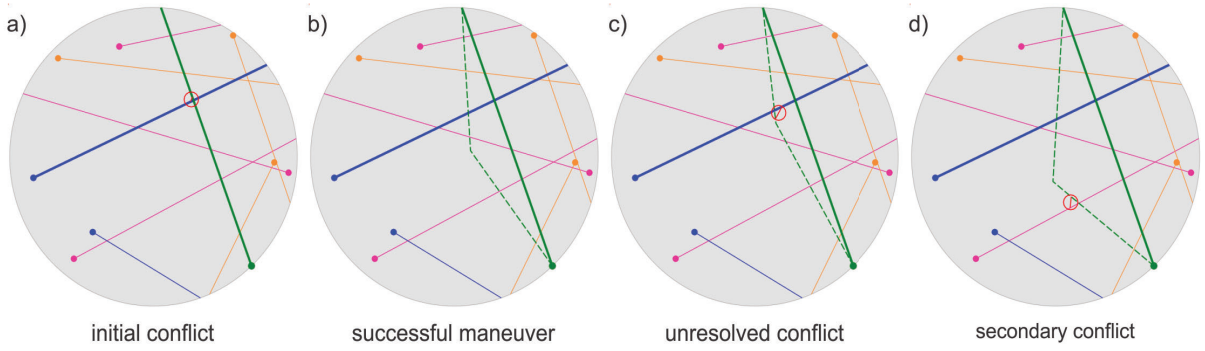

Figure 3. Different Possible Situations When User Is Searching for a Resolution using the iCS

Figure 3 presents the possible situations that could happen when the ATCO is interacting with the iCS during resolving conflict in a given scenario. Figure 3a shows the initial conflict between the ownship (green thick line) and the intruder (blue thick line), where the dots represent the current locations of all aircraft, and the red circle always indicates the loss of separation in the scenario. Figure 3b shows a

successful maneuver, the dashed green curve, after the ATCO has added a TCP to the flight plan of the ownship. Note that the successful maneuver eliminates all potential conflicts in the scenario; thus, there is no loss of separation indicator being seen in fig:iCS-exampleb. In Figure 3c, however, the location of the TCP is not able to constitute a successful maneuver, as the ownship is still in conflict with the intruder and the red indicator remains visible. In Figure 3d, although the initial potential conflict has been eliminated, the suggested maneuver is still invalid because it gives rise to a secondary conflict between the ownship and another aircraft, marked by the red indicator.

Once the ATCO has provided resolutions for the conflict scenarios, the resolutions, or more precisely the locations of the resolutions' TCP, are saved together with the conflict scenarios as training data for reinforcement learning. From the machine learning perspective, the provided resolutions could be considered as the labels of the input conflict scenarios, which encapsulate the preference of the ATCO in resolving conflict.

Reinforcement Learning for Conflict Resolution

In this work, we formulate the conflict resolution problem as a decision-making problem that is solved using reinforcement learning algorithm. Reinforcement learning has recently shown its great capability to successfully solve complex decision-making problems [11-16]. Similar to many complex decision-making problems reported in the literature, here, the conflict resolution problem has large state space and continuous action space; therefore, advanced treatments must be considered to guarantee the learning performance. For this reason, we employ the Deep Deterministic Policy Gradient (DDPG) algorithm for our problem, as this algorithm had been proven to have high performance on large state and continuous action spaces [17]. In the remaining of this section, we briefly describe the core components of reinforcement learning, and some consideration during the implementation of the DDPG algorithm in the conflict resolution problem. A comprehensive explanation of the algorithm could be found in [17].

In the reinforcement learning algorithm applied to our problem, the core components include a policy $\pi(s)$ and a state-action value function $\mathbf{V}(s,a)$ (Q-value). The policy $\pi(s)$ is the mapping from input conflict scenarios s to maneuvers, or action a, required to resolve the conflicts. Here, the agent's learning objective is to arrive at an optimal policy $\pi^*(s)$ that maps conflict scenarios to maneuvers which best capture the ATCO's preference. During training phase, the state-action value function $\mathbf{V}(s,a)$ is approximated and used to assess the similarity between the ATCO's resolutions and the agent's suggested ones. $\mathbf{V}(s,a)$ takes a conflict scenario s and an action a as its inputs, and returns a scalar value that represents the goodness of taking the given action at the given scenario. Given a scenario, an agent's action that is closer to the human decision is considered as better and its value is higher. During training, $\pi(s)$ and $\mathbf{V}(s,a)$ are alternatively updated until they both become stable and optimal.

In the training phase, all the scenarios in the train set of the data is presented to the agent, one by one. For a given scenario, the agent explores and tries a number of maneuvers, and receives a reward signal for every maneuver it has applied. The reward function $R(a)$ is designed such that it gives reward for a maneuver that resolve the conflict and penalizes a maneuver that fails to separate the aircraft. Because the action space is continuous, i.e. there are indefinite numbers of possible actions, it is impractical for the agent to try all of them. Instead, the agent tries a definite actions for each scenario, and data containing tuples of {state s, action a, reward r} are stored after every trial. Then, the policy $\pi(s)$ and the state-action value function $\mathbf{V}(s,a)$ are approximated by two deep neural networks, the actor network and the critic network respectively, employing the collected data tuples. The actor network, which acts in response to the current state, is to approximate the desired policy. The critic network, which criticizes the decision made by the actor network, is to approximate the real reward function (Figure 4). The model's performance is assessed by the similarity between the agent's suggested TCP and that from human in unseen scenarios of the test set.

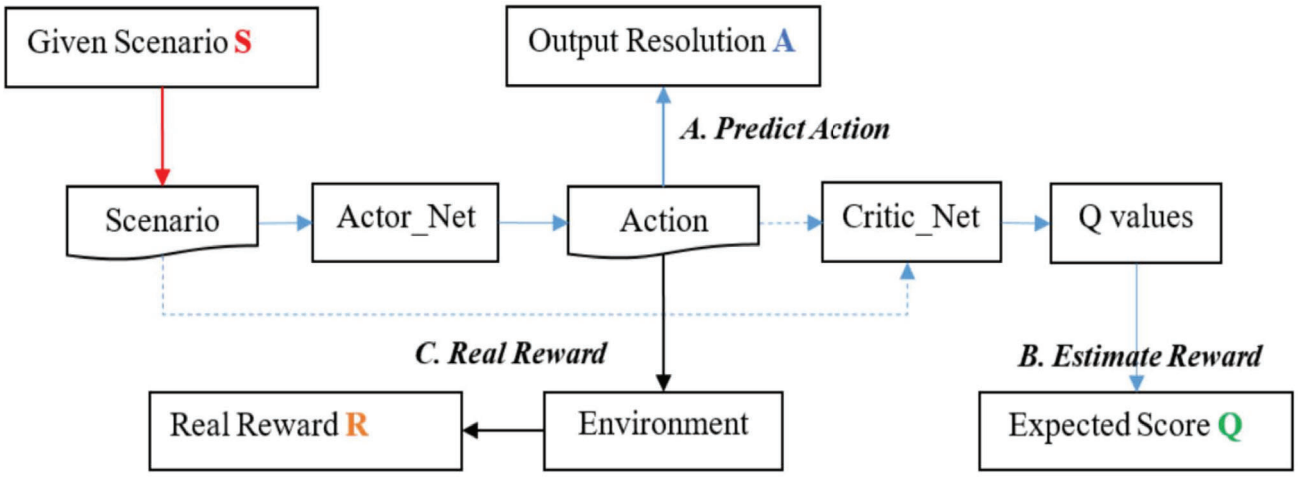

Figure 4. Components of the Reinforcement Learning Model for the Conflict Resolution Problem

In our experiment, the iCS presents 500 conflict scenarios to an ATCO and collects his resolutions for these scenarios. Each scenario has one potential conflict between the ownship and the intruder aircraft, and there are three other surrounding aircraft (see fig:scenario). Assume that the ATCO resolves all the conflicts employing a consistent strategy; therefore, the resolutions provided encapsulate the ATCO's preference in resolving conflicts. Among the 500 pairs of conflicts and captured resolutions, we randomly choose 400 pairs to form the train set of data, and the remaining 100 pairs being the test set used for model evaluation.

Results and Discussion

The most important indicator of the model's performance is the similarity between the agent's and the ATCO's resolutions. Since this similarity is assessed by the reward mechanism, the approximation of the reward (or the equivalent penalty) signal, i.e. $\mathbf{V}(s,a)$, is essential to the model's performance. Figure 5 presents the convergence of the approximated penalty and the actual penalty as the training is progressing. Note that the use of penalty instead of reward for model assessment does not alter the model's qualities in any manner. Figure 5 shows two qualities of the model when it is converging: (1) the approximated reward signal (the blue curve), i.e. the output of the critic network in Figure 4, converges to the actual one (the orange curve), and (2) they both become stable. We could see that the model well converges after 2,000 iterations. At the beginning of the training, the actual penalty starts from a very large value ($> 1{,}000$) because at that moment the agent is still in the exploration phase. The approximated penalty starts from a very small value because of the initialization of the critic network.

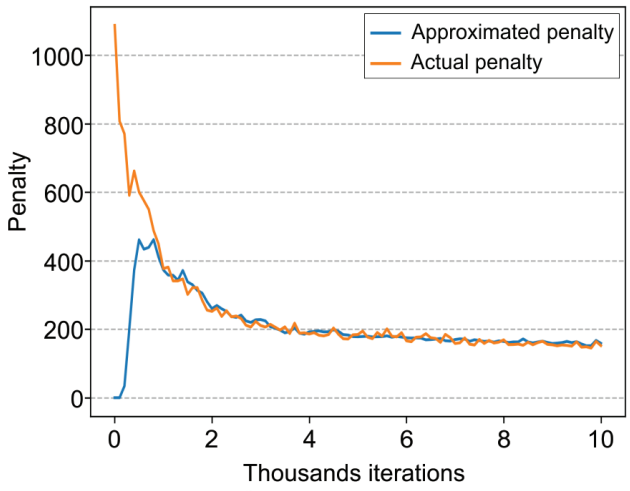

Figure 5. Convergence of the Learning Model

Figure 6 shows the agent's resolution for a unseen conflict scenario after model's convergence, as an example. The dashed white line is the originally planned trajectory of the ownship, and the solid thick white curve is the resolution suggested by the agent. It is shown that the TCP of the agent's resolution is very close to that provided by the ATCO, which is located by the start marker. In fig:sample, the heat map represents values of the penalty, where lower penalties (i.e. more desirable TCP) are located in the darker blue

regions and high penalties (i.e. low quality TCP) are at the darker red locations.

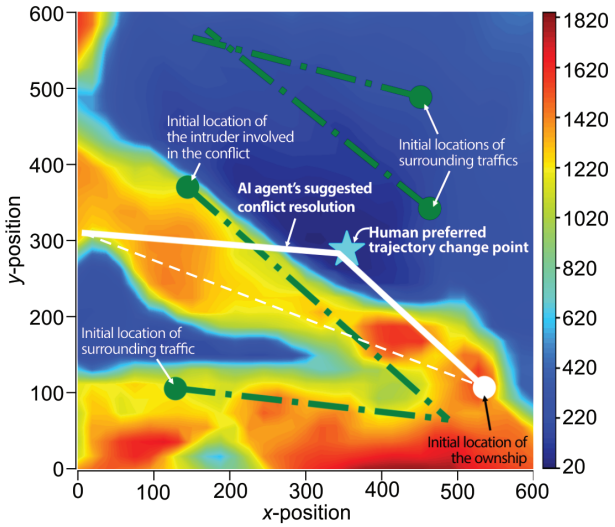

Figure 6. An Example of the Agent's Suggested Resolution Showing the Similarity between Agent's and Human Resolutions

After the model has converged, we allow the agent to resolve 100 unseen conflict scenarios in the test set, and the distribution of the penalties given to the agent is shown in Figure 7, where more than 65% of the resolutions suggested by the agent received very low penalties (i.e. lower than 100). The majority of suggested resolutions receive low penalties indicates the high capability of the agent of suggesting resolutions that capture the ATCO's preference.

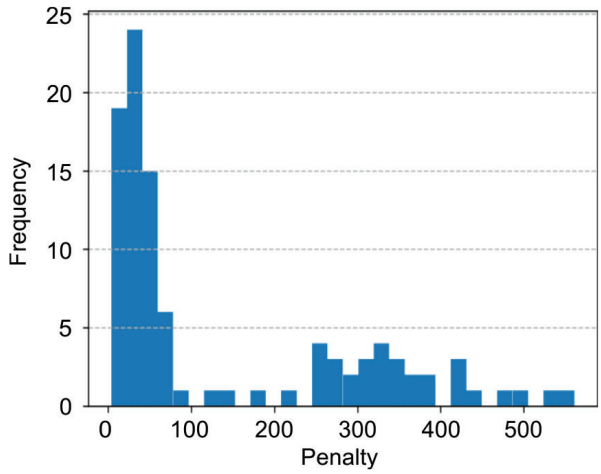

Figure 7. Penalties Distribution Performed on Test Set After Convergence

In Figure 8, we demonstrate the agent's suggested resolutions for six unseen conflict scenarios. It could be observed that in all scenarios, the trajectory change point is always located at the dark blue region, showing that the agent is able to limit the penalties to very low values by suggesting resolutions that are close to the preference provided by human.

Conclusion

We have proposed a framework for building an artificial intelligent agent capable of suggesting conflict resolution advisories that capture the preferences of ATCOs in resolving conflicts. We have developed an interactive conflict solver to acquire resolutions from ATCOs, and employed the collected data to train the agent using reinforcement learning algorithm. Our results have shown that more than 65% of the resolutions suggested by the agent are close to the resolutions performed by an ATCO. The obtained outcomes suggest that reinforcement learning is a promising approach for future resolution advisory systems, for its ability to learn from human experience.

We believe that the results of this work could be further improved. An improvement in the representation of conflict scenarios could help the agent better apprehends its environment and makes better decision. Also, a throughout investigation of the consistency in the strategies that ATCOs employ to resolve conflicts is necessary to improve the model's convergence and to make the agent's resolutions closer to that of ATCOs.

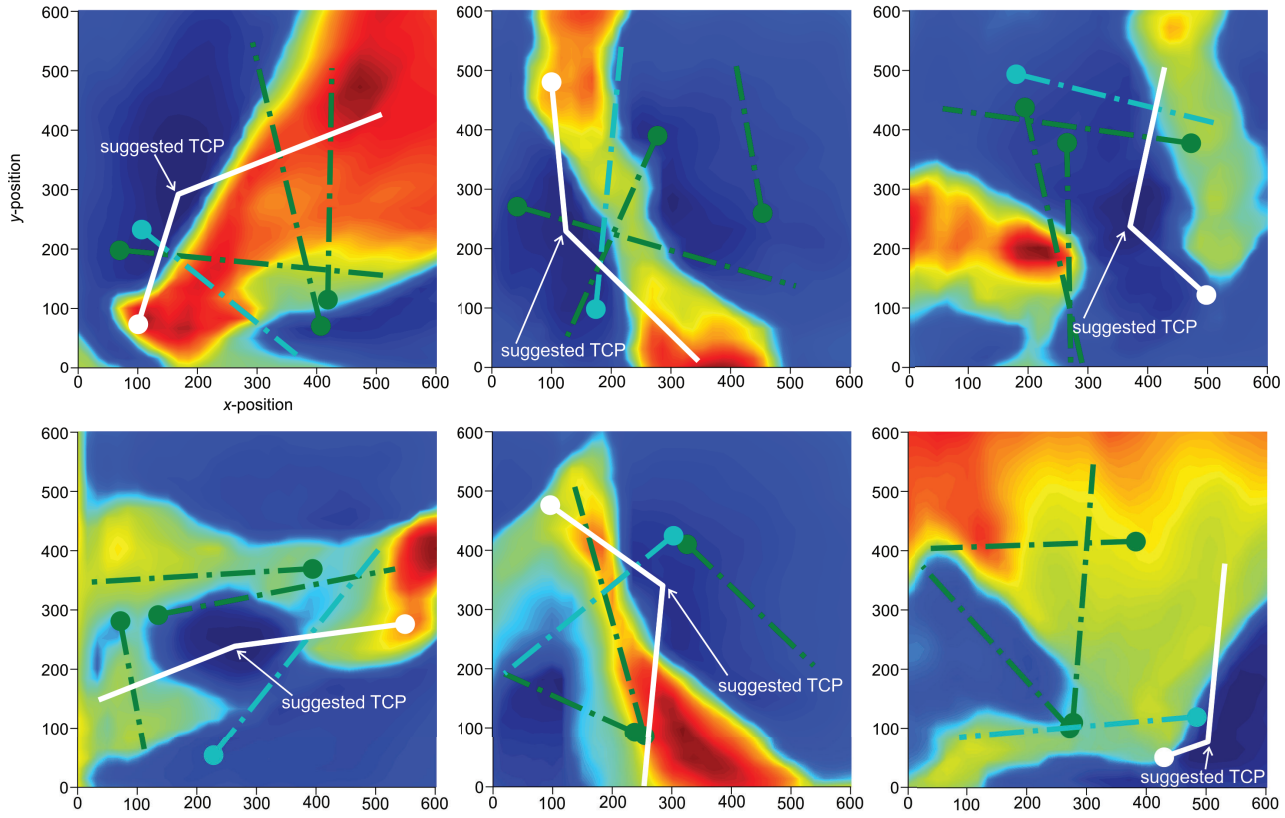

Figure 8. The Agent's Suggested Resolutions for Different Conflict Scenarios

References

[1] IATA, 20 year passenger forecast, https://www.iata.org/publications/store/Pages/20-year-passenger-forecast.aspx, [Accessed27-December-2018],2018.

[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, ISSN: 1524-9050. DOI: 10.1109/6979.898217.

[3] S. Hao, S. Cheng, and Y. Zhang, "A Multi-Aircraft Conflict Detection and Resolution Method for 4-Dimensional Trajectory-Based Operation," Chinese Journal of Aeronautics, vol. 31, no. 7, pp. 1579–1593, 2018, ISSN: 1000-9361. DOI: https : / / doi . org / 10 . 1016/j.cja.2018.04.017. [Online]. Available: http: / / www . sciencedirect . com / science / article / pii / S1000936118301705.

[4] M. Radanovic, M. A. Piera Eroles, T. Koca, and 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, ISSN: 0968090X. DOI: https://doi.org/10.1016/j.trc.2018.07.017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0968090X18302353.

[5] N. Yokoyama, "Decentralized Conflict Detection and Resolution Using Intent-Based Probabilistic Trajectory Prediction," in 2018 AIAA Guidance, Navigation, and Control Conference, ser. AIAA SciTech Forum. American Institute of Aeronautics and Astronautics, 2018. DOI: doi:10.2514/6.2018-185710.2514/6.20181857. [Online]. Available: https://doi.org/10.2514/6.2018-1857.

[6] V. P. Jilkov, J. H. Ledet, and X. R. Li, "Multiple Model Method for Aircraft Conflict Detection and Resolution in Intent and Weather Uncertainty," IEEE Transactions on Aerospace and Electronic Systems, pp. 1–1, 2018, ISSN: 0018-9251. DOI: 10.1109/TAES.2018.2867698.

[7] C. Allignol, N. Barnier, N. Durand, A. Gondran, and R. Wang, "Large Scale 3d En-Route Conflict Resolution," in ATM Seminar, 12th USA/Europe Air Traffic Management R&D Seminar.

[8] Z. Liu, K. Cai, X. Zhu, and Y. Tang, "Large Scale Aircraft Conflict Resolution Based on Location Network," in 2017 IEEE/AIAA 36th Digital Avionics Systems Conference (DASC), pp. 1–8, ISBN: 2155-7209. DOI: 10.1109/DASC.2017.8102134.

[9] C. Westin, B. Hilburn, C. Borst, and D Schaefer, "The Effect of Strategic Conformance on Acceptance of Automated Advice: Concluding the Mufasa Project," Proceedings of the SESAR Innovation Days, vol. 3, 2013.

[10] C. Westin, C. Borst, and B. Hilburn, "Automation Transparency and Personalized Decision Support: Air Traffic Controller Interaction with a Resolution Advisory System," IFAC-PapersOnLine, vol. 49, no. 19, pp. 201–206, 2016, ISSN: 2405-8963. DOI: https:// doi . org / 10 . 1016 / j . ifacol . 2016 . 10 . 520. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2405896316321103.

[11] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, Veness, M. G. Bellemare, A. Graves, M. Riedmiller, K. Fidjeland, G. Ostrovski, et al., "Human-Level Control Through Deep Reinforcement Learning," Nature, vol. 518, no. 7540, p. 529, 2015.

[12] M. Moravˇy, M. Schmid, N. Burch, V. Lis`Morrill, N. Bard, T. Davis, K. Waugh, M. Johanson, and M. Bowling, "Deepstack: Expert-level Artificial Intelligence in Heads-Up No-Limit Poker," Science, vol. 356, no. 6337, pp. 508–513, 2017.

[13] M. Campbell, A. J. Hoane Jr, and F.-h. Hsu, "Deep Blue," Artificial Intelligence, vol. 134, no. 12, pp. 57–83, 2002.

[14] J. Schaeffer, "A Gamut of Games," AI Magazine, vol. 22, no. 3, p. 29, 2001.

[15] N. Yakovenko, L. Cao, C. Raffel, and J. Fan, "Poker-cnn: A Pattern Learning Strategy for Making Draws and Bets in Poker Games Using Convolutional Networks.," in AAAI, 2016, pp. 360–368.

[16] 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.

[17] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous Control with Deep Reinforcement Learning," arXiv preprint arXiv:1509.02971, 2015.

Acknowledgements

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