An Interactive Conflict Solver for Learning Air Traffic Conflict Resolutions

Phu N. Tran,* Duc-Thinh Pham,† Sim Kuan Goh,‡ Sameer Alam,§ Vu Duong ¶
Air Traffic Management Research Institute (ATMRI),
Nanyang Technological University, Singapore 637460

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, specifically in resolving potential conflict. Since current automated conflict resolutions are not in conformance with the thinking or preferences of individual ATCOs, consequently, they are unlikely accepted by the ATCOs. In this work, we build an artificial intelligence 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' demonstrations, and an AI agent. The AI Agent is based on reinforcement learning to suggest conflict resolutions. 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. That could help to 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.

I. 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 system safety. To improve the current system, the new capacity will be necessarily created. However, airspace capacity design is heavily controlled by the human factor, i.e. the workload of the air traffic controller (ATCO) in charge of the sector; therefore, any additional airspace capacity must come with solutions to keep the workload of ATCOs manageable. As the main role of ATCOs is to maintain the safe

*Research Fellow, Air Traffic Management Research Institute (ATMRI), phutran@ntu.edu.sg
†Research Fellow, ATMRI, dtpham@ntu.edu.sg
‡Research Fellow, ATMRI, skgoh@ntu.edu.sg
§Program Director, ATMRI, sameeralam@ntu.edu.sg
¶Director, ATMRI, vu.duong@ntu.edu.sg

separation between aircraft, one way to manage their workload is to provide the ATCOs with assistant tools that could suggest them conflict resolution advisories during their deconflict tasks.

Conflict resolution advisories are necessarily given by the ATCOs to the pilots when potential conflicts have been 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). To assist ATCOs in conflict resolution tasks, many automated conflict resolution models had been developed during the past few decades. A comprehensive summary of early mathematical methods for automated conflict resolution could be found in [2]. More recently, different approaches have been proposed to improve the performance of automated conflict resolver [3–12]. 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. In our research, we observed that data-driven and machine-learning-based approaches are capable to overcome these limitations.

With the availability of aviation data and the significant advancements in computational power, data-driven and machine-learning-based methods have recently become a very promising approach to many challenging problems in air traffic management, such as taxi-out time prediction [13, 14], aircraft sequencing [15], trajectory prediction [16], aircraft performance parameter predicting [17], air traffic flow extraction [18], flight delay prediction [19, 20]. The conflict resolution problem in air traffic management, in particular, could be formulated as a decision-making problem, and then solved using machine learning algorithms that focus on learning to make decisions. Machine learning for decision making has recently achieved remarkable advancements and demonstrated remarkable results in tasks that previously required human participation or supervision (e.g. DeepBlue in chess competitions [21], Poker-CNN [22] and DeepStack [23] for the poker game, etc.). Those learning techniques have also been employed to build automated systems for air traffic control. For example, in one of the earliest applications of machine learning in conflict resolution, Brittain et al. developed a deep distributed multi-agent reinforcement learning (DD-MARL) framework to resolve conflicts at intersections and merging points with the success rate of 99.97% and 100%, respectively [24]. Reinforcement learning was also employed for collision avoidance at the non-towered airports [25], which could issue high-level recommendations to the pilots to anticipate the aircraft.

Although there are continuous improvements in automated tools for conflict resolution, there remains a challenge that automated conflict resolutions generally get a low acceptance rate from ATCOs [26, 27]. The major reason is that those automated conflict resolutions tools do not gain sufficient trust from the ATCOs because the suggested resolutions are poor in strategic conformance to the preferences of individual ATCOs in resolving conflicts. In the study by Erzberger [3], for example, ATCOs' preference is highly appreciated as an important factor; however, there was no

kind of direct involvement of the ATCOs in the development of the algorithms. To improve the acceptability, in our opinion, the development of the automated tools must be transparent to the ATCOs, and one way to accomplish this is to directly involve the ATCOs in the loop.

In this work, we attempt to build a conflict resolution advisory system that enables the direct involvement of ATCOs in the development of the advisory tool. The system consists of two components: (1) an interactive conflict solver (iCS) for generating conflict scenarios, acquiring and characterizing human resolutions, and (2) an artificial intelligence agent that learns to resolve conflict incorporating the characteristics of human resolution acquired by the iCS. The iCS presents conflict scenarios to ATCOs, observes ATCOs' behaviors when they are resolving conflicts, and characterizes ATCOs' resolutions. The AI agent is trained using a 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' demonstrations.

II. Methodology

A. Interactive Conflict Solver (iCS)

We develop an interactive conflict solver that facilitates the participation of ATCOs in the development of automated conflict resolution. iCS is a web-based application with an interactive graphical user interface. Its functionalities include generating conflict scenarios, acquiring ATCOs' actions while they are performing separation, and characterizing/evaluating the resolutions provided by ATCOs. Insights about human-provided conflict resolution are used to train the intelligent agent, which is described later.

1. Conflict scenario

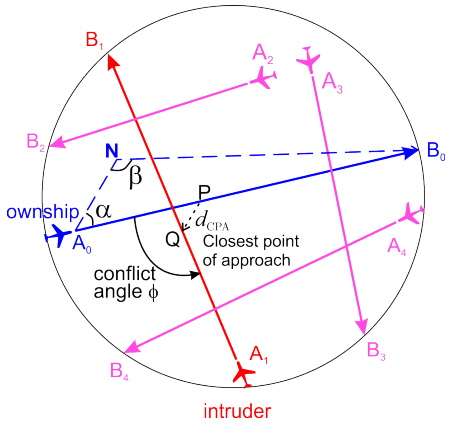

Fig. 1 A traffic scenario with a potential conflict between the ownship and the intruder aircraft, in the presence of surrounding aircraft

In this research, we define a conflict scenario as a traffic scenario that occurs within a circular area of interest (or

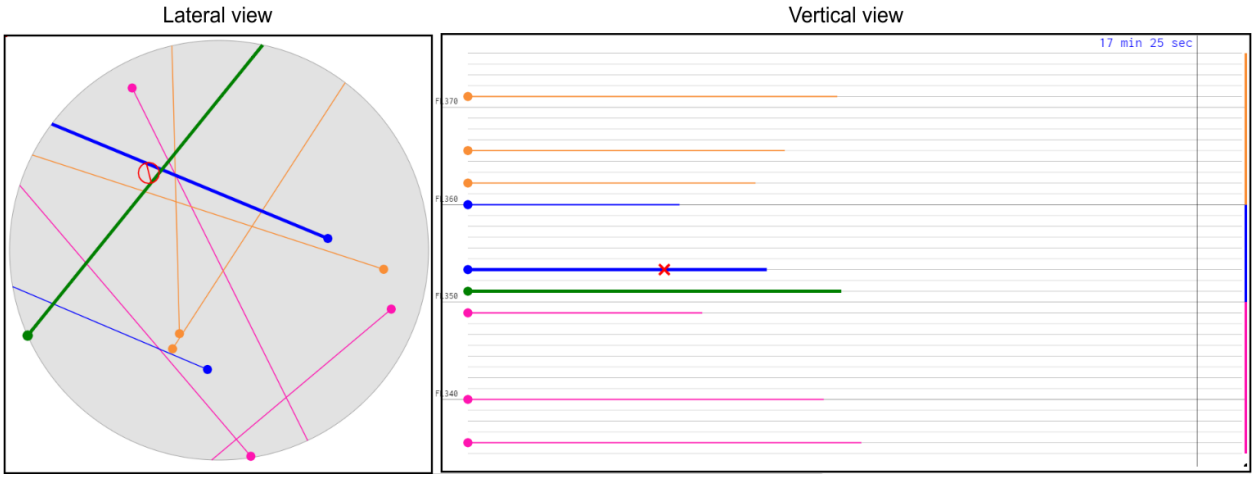

Fig. 2 A screenshot of the interactive conflict solver

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 interesting airspace ($0 \leq i < n$) (Figure 1). Consequently, $\mathbf{A}_i\mathbf{B}_i$ represents the aircraft's trajectories and $\overrightarrow{\mathbf{A}_i\mathbf{B}_i}$ are the initial headings of the aircraft. If the aircraft continue their journeys with this original flight 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 $\overrightarrow{d_1}$ from **P** to **Q**. Similarly, the CPAs between the ownship and the surrounding aircraft are denoted by $\overrightarrow{d_i}$ where $2 \leq i < n$. Note that at the beginning, $\|\overrightarrow{d_1}\| < d_{\text{sep}}$ while $\|\overrightarrow{d_i}\| \geq d_{\text{sep}}$, $2 \leq i < n$; this imposes the single initial conflict condition to the generated scenarios, which is the interest of this work.

The detection of a potential conflict between two aircraft is as follows. Assume that the ownship and the intruder aircraft are cruising at speeds of v_{ow} and v_{in}, respectively. 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 $\overrightarrow{u} = v_{\text{ow}}(\overrightarrow{\mathbf{A}_0\mathbf{B}_0}/\|\overrightarrow{\mathbf{A}_0\mathbf{B}_0}\|)$ and $\overrightarrow{v} = v_{\text{in}}(\overrightarrow{\mathbf{A}_1\mathbf{B}_1}/\|\overrightarrow{\mathbf{A}_1\mathbf{B}_1}\|)$, respectively. At a time $t > 0$, the locations of the the ownship and the intruder are respectively given by $\overrightarrow{P}(t) = \overrightarrow{\mathbf{A}_0} + \vec{u}t$ and $\overrightarrow{Q}(t) = \overrightarrow{\mathbf{A}_1} + \vec{v}t$, and distance between them renders as $d_1(t) \equiv \|\overrightarrow{d_1}\| = \overrightarrow{W_0} + (\vec{u} - \vec{v})t$ where $\overrightarrow{W_0} = \overrightarrow{\mathbf{A}_0\mathbf{A}_1}$. Minimizing $d_1(t)$ yields the time to CPA as $t_{\text{CPA}} = -\overrightarrow{W_0} \cdot (\vec{u} - \vec{v})/\|\vec{u} - \vec{v}\|^2$, and the closure at CPA as $d_{1(\text{CPA})} = d_1(t_{\text{CPA}})$. Similar computation applies for the detection of potential conflict between the ownship and any of the surrounding aircraft.

The conflict scenarios generation is governed by two factors: time to CPA and conflict angle. The conflict angle is

defined as the angle ϕ being shown in Figure 1, where $\phi = 0$ leads to a perfect in-trail conflict and $\phi = 180$ degrees a perfect head-on conflict. In the conflict scenarios generation process, given the cruising speeds of the ownship and the intruder aircraft, the initial locations and headings of the two aircraft are generated such that the times to CPA of the ownship fall within the desired look-ahead time (4-8 minutes in this work), and the conflict angle ϕ covering the range 0-180 degrees. From this setting, the generated conflicts include in-trail, crossing, and head-on ones. To generate a conflict scenario, in particular, time to CPA and conflict angle are randomized satisfying the mentioned time to CPA and conflict angle ranges.

In this work, we generate conflict scenarios in an interested area that does not have any structured air route. Such an approach brings greater diversity to the conflict scenarios and allows us to investigate the learning model performance when conflict scenarios do not follow any structured pattern. This becomes useful when one considers the free route airspace, where pilots have some delegation to choose the preferred flight routes rather than following the predefined airways. In addition, in the generated scenarios, there is only a conflict pair between an ownship and an intruder aircraft, and all aircraft are cruising at the same speed. We find such assumptions useful because conflict involving multiple aircraft is extremely rare in reality, and speeds of aircraft during the cruising phase are usually comparable in many situations. These assumptions also help reduce the complexity of the learning model. Furthermore, to make the scenarios the most generic ones, we do not simulate any particular aircraft type. This is to make the model in this study as generic as possible, which would allow the customization of the interactive conflict solver and the learning model into different situations when one wishes to integrate the model of this work into an existing simulation environment in the future development.

2. Maneuver

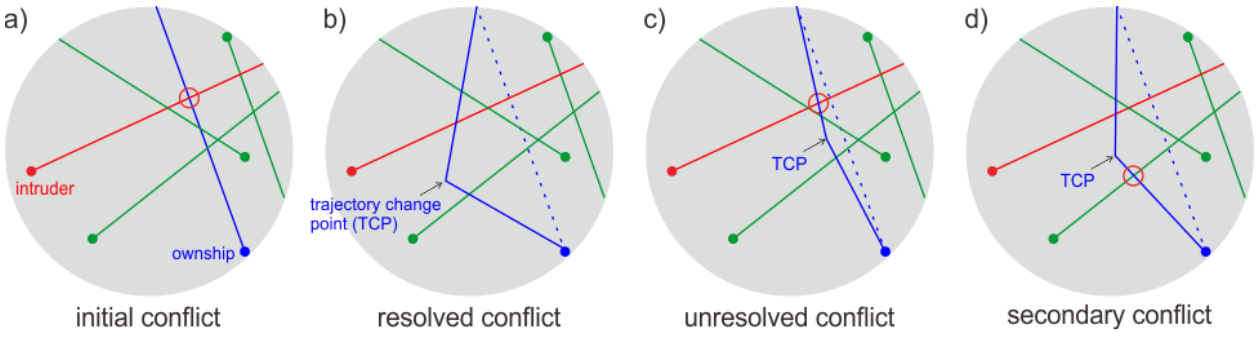

Fig. 3 Possible situations of conflict resolutions using the iCS: a) The initial conflict; b) No more conflict; c) Unresolved conflict, and d) Secondary conflict

In this work, we define the maneuver taken by the ownship aircraft as the path $\mathbf{A}_0\mathbf{N}\mathbf{B}_0$ shown in Figure 1. In particular, the ownship changes its current heading by an angle α, travels a distance $\mathbf{A}_0\mathbf{N}$, and heads back to $\mathbf{B}_0$ be performing turning an angle β (see Figure 1). Here, it is observed that any choice of the two values of the angle α and

the distance $\mathbf{A}_0\mathbf{N}$ is equivalent to a choice of the location of the trajectory change point (TCP) $\mathbf{N}$. The TCP $\mathbf{N}$ 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.

3. Interactive Conflict Solver (iCS)

One important aim of this work is to investigate the direct involvement of ATCOs in the development of automated conflict advisory tools. To facilitate this, we develop an interactive conflict solver that could acquire conflict resolutions from ATCOs. The iCS is a web-based application with an interactive graphical user interface that allows users 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 user. The conflict between the ownship and the intruder aircraft is indicated so that the ATCO does not need to perform conflict detection, but focuses on providing a conflict resolution for every scenario. To input a resolution, the ATCO performs a mouse-click on the ownship's trajectory to add a TCP, and then drags 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 shows the screenshot of the interactive conflict solver. The solver's user interface consists of two main areas: the lateral view on the left and the vertical view on the right. The iCS allows ACTOs to input either a lateral resolution or a vertical one to resolve the presented conflict. In this work, we use the iCS for lateral resolution only.

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 (solid blue line) and the intruder (solid red 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 solid blue 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 Figure 3b. In Figure 3c, however, the location of the TCP does not result in 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' TCPs, 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.

B. Reinforcement Learning for Conflict Resolution

In this work, we formulate the conflict resolution problem as a decision-making problem that is solved using a reinforcement learning algorithm. Reinforcement learning has recently shown its great capability to successfully solve complex decision-making problems [21–23, 28–30]. 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 [31]. 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 [31, 32].

1. Deep Deterministic Policy Gradient algorithm

DDPG algorithm is a variant of the actor-critic model based on the Deterministic Policy Gradient (DPG) algorithm [32]. The unique feature of the DDPG algorithm is that it uses two neural networks to approximate the actor and the critic functions. We now describe the major components of the DDPG algorithm employed in this work, as being shown in Figure 4.

Learning Environment. In our design, the learning environment has two major roles. The environment presents conflict scenarios to the agent in the form of state vectors. Second, the environment evaluates the actions, or the conflict resolutions, suggested by the agent in response to the conflict scenarios. The evaluation of the agent's suggested resolution is sent to the agent in the form of a reward signal, which considers the similarity between the agent's resolutions and that provided by ATCOs. Before the training, a number of conflict scenarios and ATCO's resolutions for those scenarios are generated and collected as training data. The scenario state vector, the agent's action, and the reward mechanism are being elaborated below.

Scenario state vector. In reinforcement learning, any decision made by the agent is heavily influenced by the agent's perceived state of its environment, and the state representation determines how the agent apprehends the state. To ensure that actions taken by the agent always modify the separation status of the ownship, the scenarios representation must encapsulate the current separation status of the ownship aircraft. Therefore, it is reasonable, and also important, to include the CPA closure vectors $\overrightarrow{d_i}$ in the state vector, because these vectors carry the essential information on the separation statuses of the ownship with other aircraft. The separation status between the ownship aircraft and each of the other aircraft is encapsulated by 5 elements:

- x- and y- positions of the ownship at CPA (2 elements)
- CPA closure $\|\overrightarrow{d_i}\|$ (1 element)
- x- and y- directions of the CPA closure vector (2 elements)

In our experiments, we consider scenarios that have 5 aircraft in total; thus, the dimension of the scenario state vector is 20.

Agent's action. The action a taken by the agent is define as the location of the trajectory change point of the ownship aircraft. Specifically, $a = (x_{\text{TCP}}, y_{\text{TCP}})$, where x_{TCP} and y_{TCP} are x- and y- coordinates of the TCP $\mathbf{N}$ (see Figure 1).

Reward. In our problem, the reward mechanism is designed such that it gives heavy punishment to the agent for invalid resolutions, i.e. resolutions that fail to separate the ownship aircraft. For valid resolutions that successfully separate the aircraft, the agent is less penalized if the resolutions are closer to that provided by ATCOs. We formulate the reward signal as a penalty amount:

$$P = \lambda_1 \sum_{i=1}^{n} (e^{\min(1, \frac{d_i}{d_{\text{sep}}})-1} - 1) + \lambda_2 \|\mathbf{N} - \mathbf{N}_{\text{ATCO}}\| \tag{1}$$

where $\|\mathbf{N} - \mathbf{N}_{\text{ATCO}}\|$ is the distance between the agent's suggested TCP and that provided by ATCOs, and λ_1, λ_2 are penalty coefficients. In the above equation, the first term on the right represents the penalty for suggesting invalid resolutions, and the second term the penalty of suggesting TCPs that are distant from ATCOs' TCPs. One could observe that when there is no potential conflict, i.e. $d_i/d_{\text{sep}} \geq 1 \forall i$, the first term vanishes and the penalty P results from only the closure between the agent's TCPs and the ATCOs' TCPs. Otherwise, if there is at least one potential conflict, the first term is non-zero and the agent receives a punishment. Generally, λ_1 is significantly larger than λ_2; in other words, the punishment for invalid maneuver is heavier. This is to ensure the agent tries hard to avoid suggesting invalid maneuvers.

Here, we can see that the demonstrations from ATCOs are used as human feedback to guide the learning. Unlike inverse reinforcement learning, in which one tries to construct the reward function from human demonstrations [33], in this work we use ATCOs' demonstrations as a feedback signal that shapes the reward function and in turn influences the policy [34]. Such a technique is referred to as *reward shaping* and is a common way to integrate human feedback in reinforcement learning [35–37].

DDPG architecture. In the DDPG algorithm employed in our study, the core components include the actor and the critic blocks (see Figure 4). **The actor model**, $a = \mu(s)$, is a mapping from a state s to an action a. The input state s is in the form of a feature vector that represents the learning environment, as described earlier. Here, we approximate the actor $\mu(s)$ by a neural network, and the actor network is trained to predict an optimal action a^* that satisfies the current policy. **The critic model** assesses the quality of the action a (taken by the actor model) by calculating the action's value $Q(s, a)$, given a scenario s. In the DDPG algorithm, the value function $Q(s, a)$ is also approximated by a neural network (critic network).

During training, the critic network is being updated by minimizing the critic loss

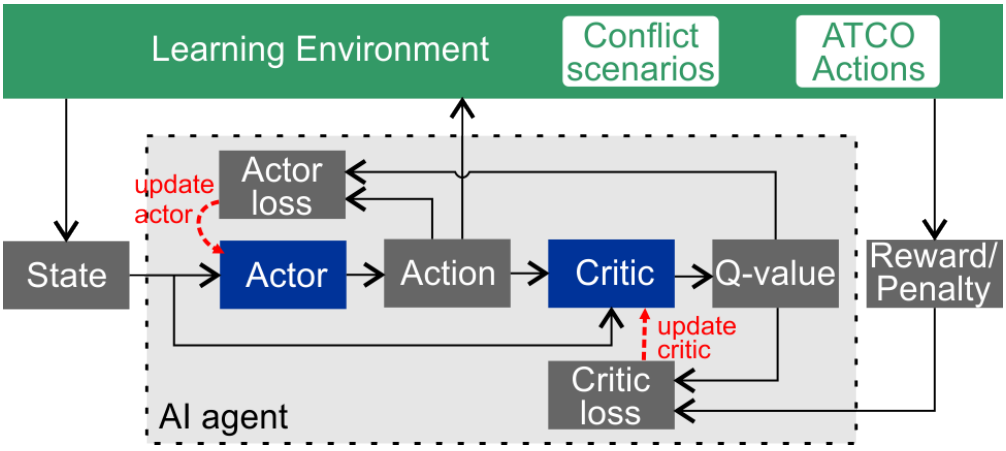

Fig. 4 Components of the reinforcement learning model for the conflict resolution problem

$$L = \frac{1}{K}\sum_{i=1}^{K}(y_i - Q(s_i, a_i|\theta^Q))^2. \tag{2}$$

The actor network, i.e. the policy, is being updated by minimizing the policy loss [32]

$$\nabla_{\theta^\mu}(J) \approx \frac{1}{K}\sum_{i=1}^{K}\nabla_{\theta^\mu}\mu(s_i|\theta^\mu)\nabla_\mu Q(s_i, \mu(s_i)|\theta^Q). \tag{3}$$

In the above equations, K is the batch size of the mini-batch training. In Equation 2, $Q(s_i, a_i|\theta^Q)$ takes a state s_i and an action a_i as input arguments ($1 \le i \le K$), and returns the approximated value, given the parameters θ^Q of the Q network. Thus, the critic loss L is the difference between the real reward y_i calculated by reward function (Equation 1) and the approximated value computed by the Q network. When training the policy network μ, the objective is to maximize the expected return $J = \mathbb{E}[Q(s, a)|_{a=\mu(s)}]$. The policy loss $\nabla_{\theta^\mu}(J)$ in Equation 3 is resulted from taking the derivative of the objective function J with respect to the policy parameters θ^μ [32].

C. Experiment setup

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 Figure 1). Assume that the ATCO resolves all the conflicts employing a consistent strategy, 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. We conduct the experiments using the parameters as shown in Table 1 below, which are determined by our experimental trials.

In each episode of the training process, the agent tries to resolve one conflict scenario by applying different TCPs.

Table 1 Parameters used in the experiment

Parameters	Values
Input layer size of actor network	20
Number of hidden layers in actor network	3
Number of units per hidden layer in actor network	300
Output layer size of actor network	2
Input layer size of critic network	22
Number of hidden layers in critic network	3
Number of units per hidden layer in actor network	300
Output layer size of critic network	1
Learning rate of actor network	10^{-4}
Learning rate of critic network	10^{-3}
First penalty coefficient, λ_1	10^3
Second penalty coefficient, λ_2	1.5
Batch size for mini-batch training, K	32

The episode is terminated once the agent has reached 20 trials or the penalty comes to below 100. In the training phase, we observe that on average, it takes approximately 45 seconds for 100 episodes to finish. In the testing phase, the model takes approximately 3.34 seconds to suggest TCPs for 100 conflict scenarios. The experimental computation was performed by a desktop PC with an Intel Core™ i7-7700HQ Processor @ 2.8GHz, 16 gigabytes of system memory. We believe that the model is scalable and extendable to deal with more complex scenarios without compromising much the performance, as the computational cost in this case linearly scales with the number of surrounding aircraft, and the assumption of having only one initial conflict pair is maintained.

III. 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. $\mathbf{Q}(s,a)$, 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 two thousand 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 6 shows the agent's resolution for an unseen conflict scenario after the model's convergence, as an example.

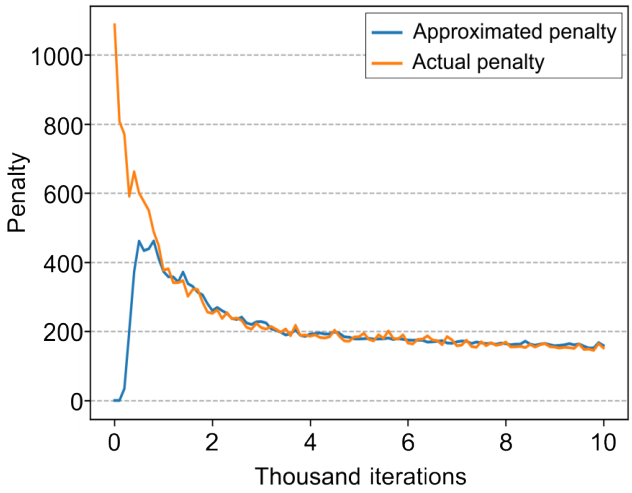

Fig. 5 Convergence of the learning model

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 Figure 6, 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.

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. In our setup, a penalty as low as 100 linearly translates to a distance of approximately 10 nautical miles between the agent's suggested TCPs and the TCPs provided by ATCOs. One could observe from Figure 7 that in some unseen scenarios, the agent can suggest TCPs that have very low penalties, which means the proposed TCPs are very close to the ATCO's choices. For suggested TCPs that have large penalties (i.e. larger than 100 in Figure 7), it does not mean that they are invalid because invalid TCPs would be penalized at least 1000 (see Equation 1). Here, suggested TCPs with larger penalties are less similar to human preferences, and we can expect that they are less likely to be accepted by the human.

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 in the dark blue region, showing that the agent can limit the penalties to very low values by suggesting resolutions that are close to the preference provided by a human.

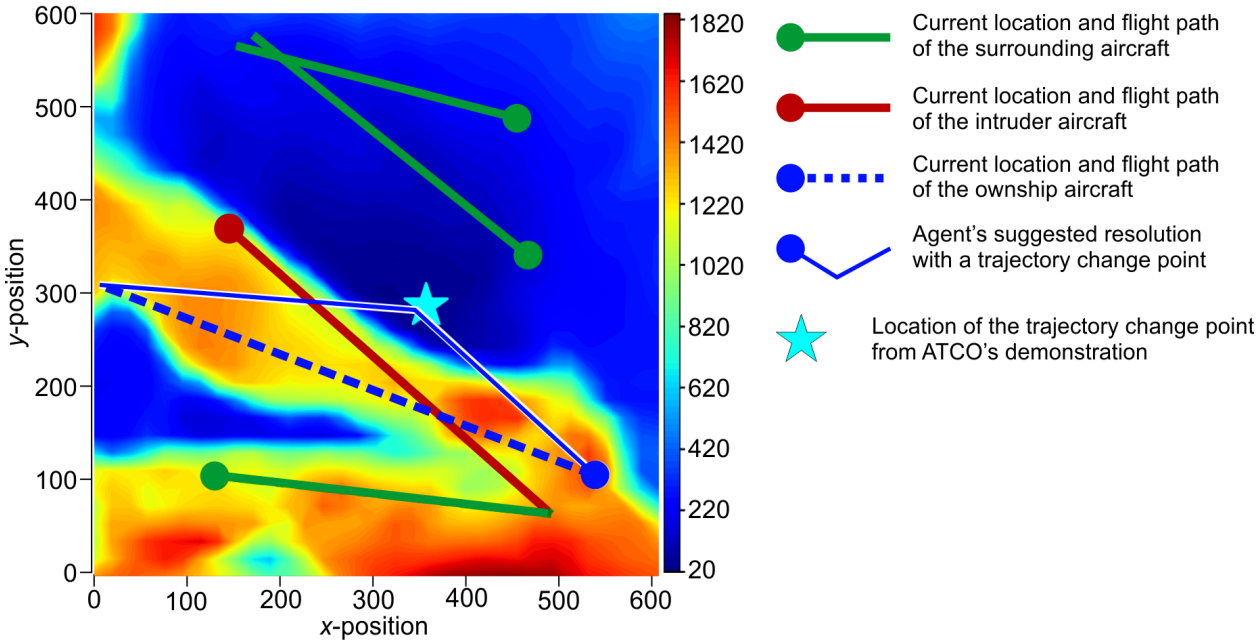

Fig. 6 An example of the agent's suggested resolution showing the similarity between agent and human resolutions. TCP in the darker blue regions is closer to ATCO's demonstration.

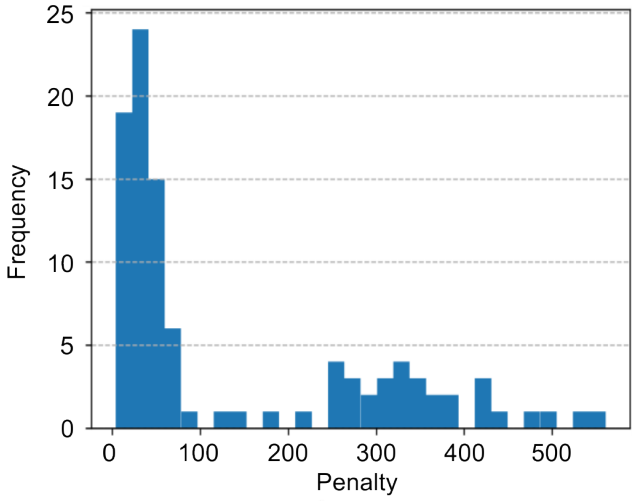

Fig. 7 Penalties distribution performed on test set after convergence

IV. Conclusion

We have proposed a framework to enable the direct involvement of air traffic controllers in the development of automated conflict resolution advisory tools. The advisory system is driven by an artificial intelligence agent capable of suggesting resolutions that incorporate the demonstrations by ATCOs. To acquire and characterize ATCOs' demonstrations, we have developed an interactive conflict solver that presents conflict scenarios to the ATCOs and record the ATCOs' actions during conflict resolving. The collected resolutions from ATCOs are employed as demonstrations

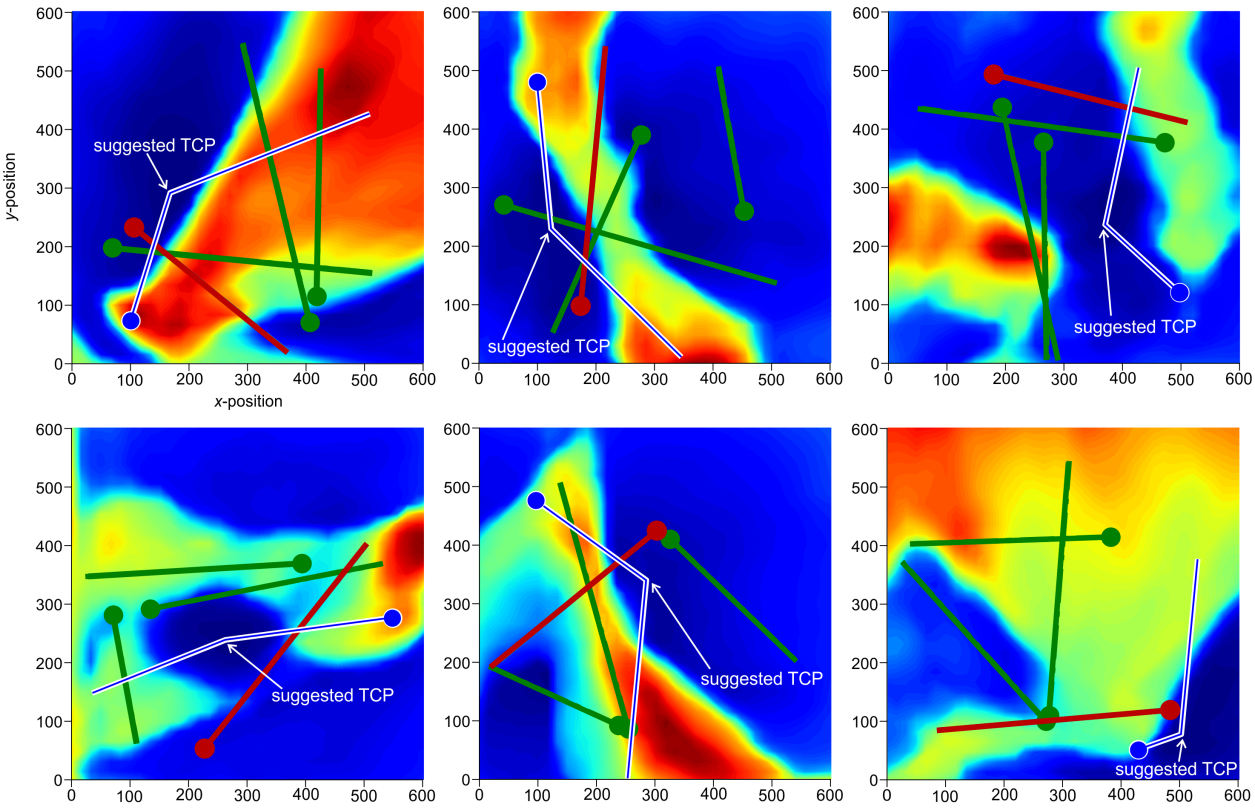

Fig. 8 The agent's suggested resolutions for 6 different unseen conflict scenarios. TCPs in the darker blue regions are closer to ATCO's demonstration.

to train the artificial intelligent agent using reinforcement learning. 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 experiences. We believe that the methodology proposed in this work is specifically useful in development automated conflict resolution advisory systems that require human-in-the-loop as feedback.

The results of this work could be further extended and improved, in terms of learning performance. An improvement in the representation of conflict scenarios could help the agent better apprehends its environment and makes a better decision. Also, an in-depth 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.

Acknowledgments

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

References

[1] IATA, "20 Year Passenger Forecast," https://www.iata.org/publications/store/Pages/20-year-passenger-forecast.aspx, 2018. [Accessed 27-December-2018].

[2] Kuchar, J. K., and Yang, L. C., "A review of conflict detection and resolution modeling methods," *IEEE Transactions on Intelligent Transportation Systems*, Vol. 1, No. 4, 2000, pp. 179–189. doi:https://10.1109/6979.898217.

[3] Erzberger, H., "Automated Conflict Resolution For Air Traffic Control," 2005.

[4] Hao, S., Cheng, S., and Zhang, Y., "A multi-aircraft conflict detection and resolution method for 4-dimensional trajectory-based operation," *Chinese Journal of Aeronautics*, Vol. 31, No. 7, 2018, pp. 1579–1593. doi:https://10.1016/j.cja.2018.04.017.

[5] Radanovic, M., Piera Eroles, M. A., Koca, T., and Ramos Gonzalez, J. J., "Surrounding traffic complexity analysis for efficient and stable conflict resolution," *Transportation Research Part C: Emerging Technologies*, Vol. 95, 2018, pp. 105–124. doi:https://10.1016/j.trc.2018.07.017.

[6] Yokoyama, N., *Decentralized Conflict Detection and Resolution Using Intent-Based Probabilistic Trajectory Prediction*, AIAA SciTech Forum, American Institute of Aeronautics and Astronautics, 2018, p. 1857. doi:https://10.2514/6.2018-1857.

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

[8] Allignol, C., Barnier, N., Durand, N., Gondran, A., and Wang, R., "Large Scale 3d En-Route Conflict Resolution," *12th USA/Europe Air Traffic Management Research and Development Seminar*, 2017.

[9] Liu, Z., Cai, K., Zhu, X., and Tang, Y., "Large scale aircraft conflict resolution based on location network," *2017 IEEE/AIAA 36th Digital Avionics Systems Conference (DASC)*, IEEE, 2017, pp. 1–8. doi:https://10.1109/DASC.2017.8102134.

[10] Stollenwerk, T., O'Gorman, B., Venturelli, D., Mandrà, S., Rodionova, O., Ng, H., Sridhar, B., Rieffel, E. G., and Biswas, R., "Quantum Annealing Applied to De-Conflicting Optimal Trajectories for Air Traffic Management," *IEEE Transactions on Intelligent Transportation Systems*, 2019, pp. 1–13. doi:10.1109/TITS.2019.2891235.

[11] Dhief, I., Dougui, N. H., Delahaye, D., and Hamdi, N., "Conflict resolution of North Atlantic air traffic with speed regulation," *Transportation Research Procedia*, Vol. 27, 2017, pp. 1242 – 1249. doi:10.1016/j.trpro.2017.12.155.

[12] Blom, H. A. P., and Bakker, G. J., "Safety Evaluation of Advanced Self-Separation Under Very High En Route Traffic Demand," *Journal of Aerospace Information Systems*, Vol. 12, No. 6, 2015, pp. 413–427. doi:https://10.2514/1.I010243.

[13] Ravizza, S., Chen, J., Atkin, J. A., Stewart, P., and Burke, E. K., "Aircraft taxi time prediction: comparisons and insights," *Applied Soft Computing*, Vol. 14, 2014, pp. 397–406. doi:https://10.1016/j.asoc.2013.10.004.

[14] Lee, H., Malik, W., and Jung, Y. C., "Taxi-out time prediction for departures at Charlotte airport using machine learning techniques," *16th AIAA Aviation Technology, Integration, and Operations Conference*, 2016, p. 3910. doi:https://10.2514/6.2016-3910.

[15] Ahmed, M., Alam, S., and Barlow, M., "A Cooperative Co-Evolutionary Optimisation Model for Best-Fit Aircraft Sequence and Feasible Runway Configuration in a Multi-Runway Airport," *Aerospace*, Vol. 5, No. 3, 2018, p. 85. doi:https://10.3390/aerospace5030085.

[16] Ayhan, S., and Samet, H., "Aircraft trajectory prediction made easy with predictive analytics," *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, 2016, pp. 21–30. doi:https://10.1145/2939672.2939694.

[17] Alligier, R., Gianazza, D., and Durand, N., "Machine learning and mass estimation methods for ground-based aircraft climb prediction," *IEEE Transactions on Intelligent Transportation Systems*, Vol. 16, No. 6, 2015, pp. 3138–3149. doi:https://10.1109/TITS.2015.2437452.

[18] Conde Rocha Murca, M., DeLaura, R., Hansman, R. J., Jordan, R., Reynolds, T., and Balakrishnan, H., "Trajectory clustering and classification for characterization of air traffic flows," *16th AIAA Aviation Technology, Integration, and Operations Conference*, 2016, p. 3760. doi:https://10.2514/6.2016-3760.

[19] Takeichi, N., Kaida, R., Shimomura, A., and Yamauchi, T., "Prediction of delay due to air traffic control by machine learning," *AIAA Modeling and Simulation Technologies Conference*, 2017, p. 1323. doi:https://10.2514/6.2017-1323.

[20] Choi, S., Kim, Y. J., Briceno, S., and Mavris, D., "Prediction of weather-induced airline delays based on machine learning algorithms," *Digital Avionics Systems Conference (DASC), 2016 IEEE/AIAA 35th*, IEEE, 2016, pp. 1–6. doi:https://10.1109/DASC.2016.7777956.

[21] Campbell, M., Hoane Jr, A. J., and Hsu, F.-h., "Deep blue," *Artificial intelligence*, Vol. 134, No. 1-2, 2002, pp. 57–83. doi:https://10.1016/S0004-3702(01)00129-1.

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

[23] Moravčík, M., Schmid, M., Burch, N., Lisỳ, V., Morrill, D., Bard, N., Davis, T., Waugh, K., Johanson, M., and Bowling, M., "Deepstack: Expert-level artificial intelligence in heads-up no-limit poker," *Science*, Vol. 356, No. 6337, 2017, pp. 508–513. doi:https://10.1126/science.aam6960.

[24] Brittain, M., and Wei, P., "Autonomous Air Traffic Controller: A Deep Multi-Agent Reinforcement Learning Approach," *arXiv preprint arXiv:0902.0885*, 2019.

[25] Mahboubi, Z., and Kochenderfer, M. J., "Autonomous air traffic control for non-towered airports," *Eleventh USA/Europe Air Traffic Management Research and Development Seminar (ATM2015) Aeronautics and Astronautics*, 2015.

[26] Westin, C., Hilburn, B., Borst, C., and Schaefer, D., "The effect of strategic conformance on acceptance of automated advice: concluding the MUFASA project," *Proceedings of the SESAR Innovation Days*, Vol. 3, 2013.

[27] Westin, C., Borst, C., and Hilburn, B., "Automation Transparency and Personalized Decision Support: Air Traffic Controller Interaction with a Resolution Advisory System," *IFAC-PapersOnLine*, Vol. 49, No. 19, 2016, pp. 201–206. doi:https://10.1016/j.ifacol.2016.10.520.

[28] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al., "Human-level control through deep reinforcement learning," *Nature*, Vol. 518, No. 7540, 2015, p. 529. doi:https://10.1038/nature14236.

[29] Schaeffer, J., "A gamut of games," *AI Magazine*, Vol. 22, No. 3, 2001, p. 29. doi:https://10.1609/aimag.v22i3.1570.

[30] 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, pp. 484–489. doi:https://10.1038/nature16961.

[31] Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D., "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.029711*, 2015.

[32] Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., and Riedmiller, M., "Deterministic policy gradient algorithms," *ICML*, 2014.

[33] Ng, A. Y., Russell, S. J., et al., "Algorithms for inverse reinforcement learning." *Icml*, Vol. 1, 2000, p. 2.

[34] Ng, A. Y., Harada, D., and Russell, S., "Policy invariance under reward transformations: Theory and application to reward shaping," *ICML*, Vol. 99, 1999, pp. 278–287.

[35] Tenorio-Gonzalez, A. C., Morales, E. F., and Villaseñor-Pineda, L., "Dynamic reward shaping: training a robot by voice," *Ibero-American conference on artificial intelligence*, Springer, 2010, pp. 483–492. doi:https://10.1007/978-3-642-16952-6_49.

[36] Pilarski, P. M., Dawson, M. R., Degris, T., Fahimi, F., Carey, J. P., and Sutton, R. S., "Online human training of a myoelectric prosthesis controller via actor-critic reinforcement learning," *2011 IEEE International Conference on Rehabilitation Robotics*, IEEE, 2011, pp. 1–7. doi:https://10.1109/ICORR.2011.5975338.

[37] Brys, T., Harutyunyan, A., Suay, H. B., Chernova, S., Taylor, M. E., and Nowé, A., "Reinforcement learning from demonstration through shaping," *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.