Transportation Research Part C 157 (2023) 104367

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

General real-time three-dimensional multi-aircraft conflict resolution method using multi-agent reinforcement learning

Yutong Chen [a,b,c], Yan Xu [a,*], Lei Yang [b,c], Minghua Hu [b,c]

[a] Cranfield University, Cranfield, MK43 0AL, United Kingdom
[b] Nanjing University of Aeronautics and Astronautics, Nanjing, 210000, China
[c] State Key Laboratory of Air Traffic Management System, Nanjing, 210000, China

ARTICLE INFO

Keywords:
Air traffic management
Three-dimensional multi-aircraft conflict resolution
Multi-agent reinforcement learning
Deep q-learning network
Generalisation
Uncertainty

ABSTRACT

Reinforcement learning (RL) techniques have been studied for solving the conflict resolution (CR) problem in air traffic management, leveraging their potential for computation and ability to handle uncertainty. However, challenges remain that impede the application of RL methods to CR in practice, including three-dimensional manoeuvres, generalisation, trajectory recovery, and success rate. This paper proposes a general multi-agent reinforcement learning approach for real-time three-dimensional multi-aircraft conflict resolution, in which agents share a neural network and are deployed on each aircraft to form a distributed decision-making system. To address the challenges, several technologies are introduced, including a partial observation model based on imminent threats for generalisation, a safety separation relaxation model for multiple flight levels for three-dimensional manoeuvres, an adaptive manoeuvre strategy for trajectory recovery, and a conflict buffer model for success rate. The Rainbow Deep Q-learning Network (DQN) is used to enhance the efficiency of the RL process. A simulation environment that considers flight uncertainty (resulting from mechanical and navigation errors and wind) is constructed to train and evaluate the proposed approach. The experimental results demonstrate that the proposed method can resolve conflicts in scenarios with much higher traffic density than in today's real-world situations.

1. Introduction

The aviation industry has experienced rapid growth, with air traffic showing a continuous upward trend. Despite the global impact of the 2019 epidemic on the air transport sector, the International Civil Aviation Organisation (ICAO) predicts an annual growth rate of up to 4.2% over the next 30 years (ICAO, 2022). However, the limited airspace resources and the continuous growth of air traffic pose a significant challenge to the development of air transport. Conflict resolution (CR) technology is a key air traffic management (ATM) technology that can improve air traffic flow and enable more flights to operate within limited airspace. In high-density scenarios, multi-aircraft conflict resolution (MACR) technology is critical for enhancing the efficiency and safety of air traffic operations.

Currently, air traffic conflict management primarily relies on well-trained air traffic controllers. However, there is a direct correlation between a controller's workload and the traffic volume (Djokic et al., 2010), which is constrained by the computing power of the human brain. To overcome this bottleneck, CR methods have been developed to assist controllers in completing CR

* Corresponding author.
E-mail addresses: yutong.chen@cranfield.ac.uk (Y. Chen), yanxu@cranfield.ac.uk (Y. Xu), laneyoung@nuaa.edu.cn (L. Yang), minghuahu@nuaa.edu.cn (M. Hu).

https://doi.org/10.1016/j.trc.2023.104367
Received 18 April 2023; Received in revised form 20 July 2023; Accepted 1 October 2023
Available online 10 October 2023
0968-090X/© 2023 The Author(s). Published by Elsevier Ltd. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

and reducing their workload. These methods typically employ geometric algorithms, optimal control theories, or heuristic algorithms (see this study (Ribeiro et al., 2020b) for specific methods in each category). However, there is still no widely-used CR method. This is because CR typically occurs in the tactical phase, which is subject to the dynamic environment (i.e., uncertainties). The models for these classical methods are generally based on deterministic models, and separate models need to be constructed and solved for each scenario in which a conflict is detected. This could pose a challenge if the environment changes while they are computing or sometime later (depending on the calculation speed and operating mechanism). Although some optimal control methods (e.g., model predictive control (Soler et al., 2016)) have the potential to account for uncertainty, they require additional operations (i.e., setting flight priorities (Chaloulos et al., 2010) or trajectory predictions (Yokoyama, 2018)) when dealing with multi-aircraft conflicts and the model is too complex (Pham et al., 2022) to meet the computational performance required for real-time scenarios. Therefore, there is an urgent need for a real-time MACR method that can effectively cope with the environment's dynamic.

In recent years, as research on reinforcement learning (RL) methods has advanced, the potential for practical application of RL-based CR methods has become increasingly apparent. One of the advantages of RL is its computational speed. Solving CR using RL methods involves two main steps. The first step is to train the neural network deployed on an agent in a designed RL environment to acquire a good policy (training process), while the second step is to deploy the agents with the trained neural network to a CR instance to solve it (solving process). Although the training process is typically time-consuming, the solving process is usually calculated quickly. Once the neural network has been trained, the solving process is fast and efficient, even in complex, high-density scenarios. RL methods transfer a significant amount of solution time to the training stage, allowing it to respond quickly to changes in the environment as they occur (i.e., update decisions in real-time). RL-based CR methods directly output actions by feeding observation matrices to the agents and calculating them through neural networks. The conflict detection process in this approach is implicit (black-box). Therefore, compared to model-based CR methods, RL-based CR methods circumvent the issue that trajectory prediction accuracy may significantly affect the effectiveness of conflict resolution. Furthermore, as a data-driven approach, RL methods hold the promise of learning ways to cope with the environment's dynamics by learning in many uncertain scenarios.

However, there are still many challenges in applying RL methods to CR:

1. *Three-dimensional manoeuvre*: Existing RL-based CR methods have only considered horizontal manoeuvres. However, in some high-density scenarios, only vertical manoeuvres can resolve the conflict, so RL methods should accommodate both horizontal and vertical manoeuvres. In this paper, models that consider only horizontal manoeuvres are referred to as two-dimensional (2D) models, while models that consider both horizontal and vertical manoeuvres are referred to as three-dimensional (3D) models.
2. *Generalisation*: Existing RL-based CR methods have often used simplified models in different aspects or personalised them according to the actual needs of their studies, leading to difficulties in compatibility with general scenarios. Therefore, a general RL framework for MACR is needed.
3. *Trajectory recovery*: Existing RL-based CR methods (especially those that perform multiple manoeuvres on the aircraft) usually only control the aircraft for CR and do not guarantee that it will return to its scheduled route or waypoint.
4. *Success rate*: CR methods based on multi-agent reinforcement learning (MARL) struggle to have high success rates of CR, especially in high-density scenarios, due to the inherent non-stationarity in MARL (i.e., it is difficult for an agent to predict the action of other agents) and the dynamic of the environment.

Although some of the existing methods can address a few of the challenges mentioned above, the more significant challenge is to design an RL-based CR method that can overcome all these difficulties to move from theory to practical application.

Furthermore, 2D and 3D scenarios differ in terms of position update model, conflict models, and operational rules, which makes the 2D model no longer applicable to 3D scenarios in essence. Therefore, extending existing 2D RL models to 3D presents the following difficulties:

(a) *Altitude maintenance*: The agent may need to acquire the capability to control the aircraft to remain at a specific flight level unless there is a need to change altitude to match the current airspace operation mode based on flight level. The climb or descent should be continuous, avoiding abrupt manoeuvres between adjacent levels.
(b) *3D conflict detection*: The agent may need to learn how to incorporate horizontal and vertical safety separation constraints to detect conflicts between aircraft in 3D space.
(c) *Neural network scale control*: The introduction of 3D space will lead to an increase in the dimensions of action and state, which may result in an explosion in neural network scale, leading to increased training difficulty.

This paper proposes a general MARL method for real-time three-dimensional multi-aircraft conflict resolution (3DMACR) to address the aforementioned challenges. The Rainbow Deep Q-learning Network (DQN) (Hessel et al., 2018), a state-of-the-art DQN method, is employed to improve the efficiency of RL. The agents share a neural network and are deployed on each aircraft to form a distributed decision-making system. The main contributions of this study are summarised as follows:

1. The method proposed facilitates a one-step completion of the necessary flight level change for aircraft. This is achieved through the reconstruction of the RL environment based on our 2D model (Chen et al., 2023), which ensures the aircraft's adaptation to the airspace operation mode based on the flight level (to cope with difficulty (a)), thus reducing the learning task of the agent.

2. A safety separation relaxation model has been designed to aid agents in detecting conflicts in three-dimensional space (to cope with difficulty (b)). With this model, agents can focus on conflict detection on three 2D planes instead of the more complex 3D planes. These 2D planes consist of the current flight level and its two adjacent levels above and below, significantly reducing the learning difficulty for the agent.
3. A modular observation matrix has been developed, comprising modules for current location, aircraft performance, flight target, and interest marks. Through experimental investigation, we have identified the optimal module composition for enhancing CR performance. This has enabled us to streamline the model by including only the essential modules (to cope with difficulty (c)), thereby reducing the scale of the neural network and making it easier to train.
4. In order to adapt to the operation of 3D environment, the adaptive manoeuvre strategy, the partial observation model based on the imminent threat and the conflict buffer model in our previous 2D model (Chen et al., 2023) are extended to 3D space.
5. The feasibility of the proposed MARL method in solving real-time 3DMACR problems is verified by a large number of high-traffic-density scenarios in a self-developed simulator containing a flight uncertainty model. Two key parameters in the model, i.e., detection range and the number of detection sectors, are analysed. It is demonstrated that the 3D model outperforms the 2D model in success rate and extra flight distance rate.

This paper is organised as follows. In Section 2, we provide a review of related work and introduce the motivation for this study. Section 3 defines and formulates the real-time 3DMACR problem. In Section 4, we introduce the general MARL model for real-time 3DMACR problems, the agent working mechanism and neural network structure, and the training algorithms for the neural network. In Section 5, we describe how the experiments are carried out, validate the proposed model's effectiveness and efficiency, analyse the model's key parameters, and compare the 3D model with the 2D model. Section 6 discusses the lessons learned and the limitations of this study. Finally, conclusions and future work are summarised in Section 7.

2. Literature review

In this section, we provide a review of the related work. Subsequently, we highlight the motivation for our study in addressing the gaps that exist between the current CR methods based on RL and the practical implementation of real-time 3DMACR.

2.1. Related work

2.1.1. Conventional 3D real-time conflict resolution methods

With advancements in communication, navigation, and monitoring equipment, real-time CR methods have evolved from 2D to 3D, allowing for higher CR performance due to the larger solution space in 3D space. Table 1 outlines the notable real-time 3D CR methods and their characteristics. Further details on these methods can be found in our previous paper (Chen et al., 2022). This paper will provide a comprehensive discussion on the challenges encountered by conventional real-time CR methods.

Sustainability refers to the ability of a CR method to handle conflicts that arise in the airspace, including secondary conflicts resulting from CR and conflicts arising from subsequent aircraft entering the airspace. Sustainable CR often involves a distributed decision-making mechanism. This is because centralised control models typically need to break down sub-problems for aircraft within a specific spatial–temporal range, and there may be conflicting aircraft in two adjacent sub-problems in the spatial–temporal dimension. Integrating uncertainty into conventional models usually significantly increases initial computation time, thus reducing their ability to respond quickly in real-time operations. Conventional methods listed in Table 1 are not capable of simultaneously accounting for trajectory recovery, uncertainty, and sustainability, which directly impacts their practical application potential.

Deconfliction manoeuvres usually involve adjusting heading, speed, and altitude, and the combined space of the three manoeuvres is typically greater than the space of a single manoeuvre. Real-time CR methods usually prioritise faster solution speed to respond quickly to dynamic environments. Conventional methods are usually based on optimisation models and have long calculation times that vary depending on the situation. Due to its computational mechanism, RL has clear advantages over conventional methods in terms of computation time, making it worth exploring its application value in real-time CR methods.

2.1.2. Conflict resolution methods based on reinforcement learning

In recent years, researchers have been exploring the potential of RL methods in CR due to their computational speed and ability to adapt to changing environmental dynamics. Table 2 provides a list of representative RL-based CR methods along with their features. A detailed analysis of these methods can be found in our previous paper (Chen et al., 2023), where we also discussed the contributions of Guo et al. (2021), Brittain and Wei (2022), Pham et al. (2019), Yilmaz et al. (2021) and Ribeiro et al. (2020a) in the field of RL-based CR methods. Additionally, Wang et al. have comprehensively reviewed the research on RL-based CR methods up to 2021 (Wang et al., 2022). This paper will discuss the specific challenges that may arise while extending the 2D RL-based methods listed in Table 2 to the 3D space.

The methods of Pham et al. (2022), Wang et al. (2019), and Lai et al. (2021) employ the approach of selecting one or multiple rerouting points to deconflict. To adapt to the current airspace operation mode based on flight level, a natural extension would be to modify the action from (x, y) to (x, y, k), where x and y denote the coordinates of the rerouting point, and k represents the flight level of the rerouting point, with options like 0, 1, or 2, indicating that the rerouting point is at the current flight level, one level below, or one level above, respectively. However, the action space of (x, y) is continuous, whereas that of k is discrete, which makes it challenging to unify them into a single action space.

Table 1
Features of real-time CR methods applied to 3D scenarios.

No.	Papers	Trajectory recovery	Uncertainty	Sustainability	Control	Manoeuvre	Computing time	Models
1	Šišlák et al. (2010)	✓	∘	✓	D	H/S/V	10 s/a	Agent
2	Cobano et al. (2013)	✓	∘	∘	C	H+S+V	47 s/5a	MINLP
3	Alonso-Ayuso et al. (2016)	∘	∘	∘	C	H+S+V	35 s/22a	SequentialMINLP
4	Alonso-Ayuso et al. (2017)	∘	∘	∘	C	H/S/V	2 s/4a	MINLP
5	Pritchett and Genton (2017)	∘	∘	∘	D	H/S/V	300 s/2a	Expertsystem
6	Wang et al. (2020)	✓	✓	∘	C	H/S/V	360 s/59a	ILP
7	Han et al. (2019)	✓	✓	∘	C	H+S+V	9 s/4a	SDP
8	Malaek and Golchoubian (2020)	✓	∘	∘	C	H/S/V	153 s/7a	Expertsystem
9	Chen et al. (2022)	✓	∘	✓	D	H+S+V	0.1 s/a	DP+INLP
10	This paper	✓	✓	✓	D	H+S+V	0.01 s/a	MARL

Check (✓) = Yes; Circle (∘) = No Control: D = Decentralised; C = Centralised Manoeuvre: H = Heading; S = Speed; V = Vertical; '/' = Or; '+' = And Computing time: 's/a' = second per aircraft Models: MINLP = Mixed integer nonlinear programming; ILP = Integer linear programming; SDP = Stochastic dynamic programming; DP = Dynamic programming; INLP = Integer nonlinear programming.

Table 2
Features of CR methods based on RL.

No.	Papers	MARL	MACR	General-isation	2D scenario	3D scenario	Uncertainty	Trajectory recovery	Shared policy	RL methods	Action space
1	Pham et al. (2022)	∘	∘	∘	✓	∘	✓	✓	N/A	DDPG	C
2	Wang et al. (2019)	∘	∘	∘	✓	∘	∘	✓	N/A	AC	C
3	Lai et al. (2021)	✓	✓	∘	✓	∘	∘	✓	∘	DDPG	C
4	Dalmau and Allard (2020)	✓	✓	✓	✓	∘	∘	∘	✓	PPO	D
5	Sui et al. (2022)	✓	✓	∘	✓	✓	∘	∘	✓	DQN	D
6	Brittain et al. (2021)	✓	✓	✓	∘	∘	✓	N/A	✓	PPO	D
7	Zhao and Liu (2021)	∘	✓	✓	✓	∘	∘	∘	✓	PPO	D
8	Chen et al. (2023)	✓	✓	✓	✓	∘	✓	✓	✓	DQN	D
9	This paper	✓	✓	✓	✓	✓	✓	✓	✓	DQN	D

Check (✓) = Yes; Circle (∘) = No RL methods: DDPG = Deep deterministic policy gradient; AC = Actor–critic; PPO = Proximal policy optimisation Action space: C = Continuous; D = Discrete.

The methods of Dalmau and Allard (2020) and Zhao and Liu (2021), and Chen et al. (our previous 2D model) (Chen et al., 2023) deconflict by making dynamic decisions based on time steps, with the manoeuvre being a combination of heading and speed. However, extending these methods to 3D scenarios presents the difficulties (a), (b), and (c) as mentioned above. Since the method of Zhao and Liu (2021) adopts image-based observation, it may also face the problem of solving multi-image cooperative observation.

The model of Brittain et al. (2021) is still a long way from being used in 3D scenarios, as it can only take one-dimensional actions at present. Although the method of Sui et al. (2022) can be used in 3D scenarios, its success rate is very low due to model defects, making it impractical (refer to the discussion in our previous paper (Chen et al., 2023) for more details).

2.2. Motivation

Considering that our previous 2D model had more comprehensive functions (refer to Table 2) and showed better performance in previous experiments (Chen et al., 2023), we attempt to develop a 3D model based on it. This study proposes an RL-based method for real-time 3DMACR to address the challenges and further explore the potential of RL methods applied to CR in practice.

In terms of 3D manoeuvres, while horizontal manoeuvres alone can handle some conflict scenarios, vertical manoeuvres have to be used for specific complex, high-density scenarios. Therefore, the proposed method considers horizontal and vertical manoeuvres,

including adjusting flight speed, heading, and altitude. The idea of extending our 2D model to a 3D model is referred to as our contributions 1, 2 and 3.

In terms of generalisation, RL-based CR methods are usually limited by their fixed observation matrix dimension. Because most existing RL-based CR methods use an intruder-centric observation approach, and the number of neurons in the input layer of a neural network is usually fixed, the agent can observe only a fixed number of intruders. When the number of intruders in the scenario changes, the trained agent may not be applicable. We propose a general MARL framework with an area-centric observation approach to tackle this issue. The partial observation model is based on imminent threats and observes only the intruder in each detection sector that poses the greatest threat to the ownship, such as the intruder closest to the ownship. The number of detection sectors is pre-defined and fixed, so the model is generalised for arbitrary scenarios. The trained policy is shared among all agents, contributing to the proposed method's generalisation.

In terms of trajectory recovery, aircraft are often required to pass through scheduled waypoints in practice, mainly as trajectory-based operation (TBO) is gradually being applied (ICAO, 2005). In the proposed method, the action space is discrete, and each action represents a flight intention instead of a specific manoeuvre. With the help of the adaptive manoeuvre strategy, the flight intention is calculated to a specific manoeuvre that meets the constraints of the aircraft's performance and ensures that the aircraft will reach its scheduled waypoint.

In terms of success rate, it is challenging for RL-based CR methods to reach 100% (Wang et al., 2022). This is most likely because the agent does not know what the conflict detection model (or minimum safety separation) is, and it is simply trained to act as appropriately as possible based on feedback from the RL environment. Moreover, for MARL-based CR methods, the inherent non-stationarity in MARL (i.e., it is difficult for an agent to predict the action of other agents) makes it difficult to cope. Unfortunately, the methods that can adequately deal with the non-stationarity are rarely reported (Papoudakis et al., 2019). Additionally, flight uncertainty (if considered) can also reduce the success rate of CR (Chen et al., 2023). The proposed method introduces a conflict buffer model to handle flight uncertainty and problems arising from the non-stationarity in MARL while it does not significantly degrade the performance of the proposed method (refer to our previous research (Chen et al., 2023) for more details on the conflict buffer model and the relevant experiments and discussions). The conflict buffer model can be used in 3D scenarios supported by a safety separation relaxation model for multiple flight levels.

3. Real-time three-dimensional multi-aircraft conflict resolution problem

In this section, we first define the real-time 3DMACR problem to further clarify this study's research scope. Next, we carefully formulate the problem to serve as the foundation of the MARL environment.

3.1. Problem definition

Fig. 1 depicts the diagram of the 3DMACR under study. The goal of the 3DMACR is to avoid the loss of safe separation by manoeuvring within the performance constraints of the aircraft during the tactical stage. The optimisation objective is to minimise the average extra flight distance. The operation context of the 3DMACR is a free route airspace (FRA) (Rezo and Steiner, 2020), where aircraft manoeuvres are not restricted by fixed routes (it differs from the traditional en-route airspace, i.e., systemised airspace), and the airspace has multiple flight levels. Aircraft operate autonomously based on airborne automation equipment, taking into account the differences in their performance, including speed ranges. The flight process also accounts for various sources of flight uncertainty, such as navigational or mechanical errors and weather. Moreover, the speed of problem-solving is designed to meet real-time operational requirements. To clarify the issue further, the following notes are provided:

- Assumptions:

    1. The constraint of consistent aircraft heading at the same altitude level (for example, the heading of aircraft in the odd-numbered flight level is 0 to 179 degrees) is not considered.
    2. Operational delays due to human factors (pilots and controllers) are not considered.
    3. Communication delays and failures are not considered.
    4. The kinematic state of an aircraft (e.g. speed and heading) changes instantaneously.
    5. Differences in aircraft performance at different flight levels are not considered.
    6. Aircraft must depart from a pre-defined start point and arrive at a pre-defined end point without specifying a flight level.
    7. Aircraft can get flight information about other aircraft in the airspace through airborne sensor (ICAO, 2016), such as position, heading and speed. Furthermore, we assume that intention (e.g., the location of end point) and performance (e.g., speed range) information can be shared between aircraft.
    8. Aircraft have already circumvented restricted areas (such as those caused by military activities or hazardous weather) through trajectory planning in the pre-tactical phase. Therefore, this paper does not consider restricted areas.

- Constraints:

    1. *Safety separation*: Both horizontal and vertical safety separation are considered.
    2. *Aircraft performance*: The constraints in aircraft performance (airspeed ranges and heading deflection rate) are considered.

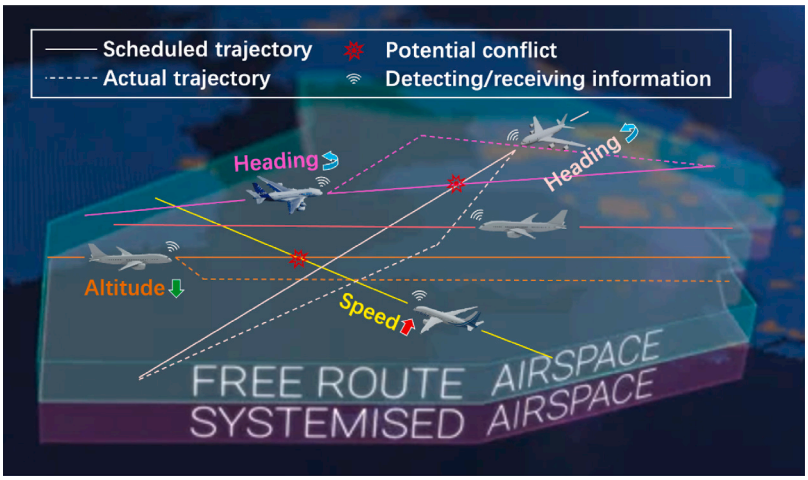


Fig. 1. Diagram of the studied 3DMACR. Aircraft's scheduled trajectories may have potential conflicts, and they need to perform manoeuvres (such as adjusting heading, speed, and altitude) based on the detected/received information to resolve these conflicts and obtain actual trajectories.

3. *Flight level stability*: Aircraft should try to remain at a flight level unless the aircraft is climbing or descending to another flight level.

It should be noted that while this paper does not consider communication delays and failures (assumption 3), we discuss and explain the potential of the proposed method to address them in Appendix B.2. Additionally, operational delays (assumption 2) can serve as a source of uncertainty considered in this paper.

3.2. Problem formulation

In the 3DMACR problem studied, all $N_{\mathrm{A/C}}$ aircraft are modelled as having the same mass in the 2D plane. Aircraft update their location information every Δt seconds. At each time step t, aircraft i ($i \in I$, where I is the set of aircraft) has access to the state $\mathbf{s}_t^i$. Based on this state, aircraft i selects an action a_t^i to guide itself toward the end point $\mathbf{p}_{\mathrm{End}}^i$ from its current position $\mathbf{p}_t^i$ while avoiding potential conflicts. For the sake of clarity, we omit the subscript i that indicates the aircraft number in the remainder of this paragraph. The action a_t is sampled from a policy π shared by all aircraft, as shown in the following equation:

$$a_t \sim \pi_\theta \left(a_t \left| \mathbf{s}_t \right.\right) \tag{1}$$

where θ denotes the neural network parameters. The action a_t is then translated into an aircraft manoeuvre $\mathbf{v}_t$ (velocity) via the RL environment:

$$\mathbf{v}_t = E\left(a_t\right) \tag{2}$$

where $E(\cdot)$ represents the RL environment. The flight management system on board performs aircraft manoeuvres autonomously based on the velocity $\mathbf{v}_t$. In consideration of flight uncertainty, the manoeuvre $\mathbf{v}_t$ is corrected to an actual velocity $\mathbf{u}_t$ based on the actual trajectory:

$$\mathbf{u}_t = U\left(\mathbf{v}_t\right) \tag{3}$$

where $U(\cdot)$ denotes the flight uncertainty. As we assume that the kinematic state of an aircraft changes instantaneously, the motion of the aircraft in each time step can be considered uniform linear motion. Therefore, the position of the aircraft at each time step can be represented as:

$$\mathbf{p}_{t+1} = \mathbf{p}_t + \mathbf{u}_t \Delta t \tag{4}$$

where $\mathbf{p}_t$ denotes the aircraft's position at time step t. Furthermore, the aircraft's position at any given moment can be represented as:

$$\mathbf{p}_{t+1}(x) = \mathbf{p}_t + x\mathbf{u}_t \Delta t, x \in (0, 1] \tag{5}$$

where $\mathbf{p}_t(x)$ represents the position of the aircraft at the $(t+x)\Delta t$-th second. Based on sequential decisions and sampling errors, the aircraft's actual trajectory l can be computed from its start point $\mathbf{p}_{\mathrm{Start}}$ to end point $\mathbf{p}_{\mathrm{End}}$ of the airspace:

$$l = \left\{ \overrightarrow{\mathbf{p}_t \mathbf{p}_{t+1}} \middle| \begin{array}{l} \mathbf{u}_t = U(\mathbf{v}_t), \mathbf{v}_t = E\left(a_t\right), a_t \sim \pi_\theta \left(a_t \left| \mathbf{s}_t \right.\right), \\ t = t_{\mathrm{Start}}, \ldots, t_{\mathrm{End}} - 1 \end{array} \right\} \tag{6}$$

where $\overrightarrow{\mathbf{p}_t\mathbf{p}_{t+1}}$ denotes the trajectory from $\mathbf{p}_t$ to $\mathbf{p}_{t+1}$. t_{Start} and t_{End}, respectively, denote the time steps when the aircraft starts and ends its flight (i.e., $\mathbf{p}_{\text{Start}} = \mathbf{p}_{t_{\text{Start}}}$ and $\mathbf{p}_{\text{End}} = \mathbf{p}_{t_{\text{End}}}$).

For each aircraft i, the horizontal and vertical safety separation constraints are represented separately as:

$$\left\|\mathbf{p}_t^i(x)\mathbf{p}_t^j(x)\right\|^{\text{Hori}} \geq D^{\text{Hori}}, j \in J_i, t = t_{\text{Start}}, \ldots, t_{\text{End}} - 1, x \in (0, 1] \tag{7}$$

$$\left\|\mathbf{p}_t^i(x)\mathbf{p}_t^j(x)\right\|^{\text{Vert}} \geq D^{\text{Vert}}, j \in J_i, t = t_{\text{Start}}, \ldots, t_{\text{End}} - 1, x \in (0, 1] \tag{8}$$

where $\|\cdot\cdot\|^{\text{Hori}}$ and $\|\cdot\cdot\|^{\text{Vert}}$ denote the horizontal and vertical distances between two points, respectively. D^{Hori} and D^{Vert} denote the horizontal and vertical minimum safety separations, respectively. J_i is the set of aircraft flying in the airspace of interest at the same time as aircraft i. Aircraft performance constraints are represented as:

$$V_{\min}^i \leq v_t^i \leq V_{\max}^i, t = t_{\text{Start}}, \ldots, t_{\text{End}} - 1 \tag{9}$$

$$\left\langle \mathbf{v}_t^i, \mathbf{v}_{t+1}^i \right\rangle \leq \omega_{\max}^i \Delta t, t = t_{\text{Start}}, \ldots, t_{\text{End}} - 1 \tag{10}$$

In Eq. (9), v_t^i denotes the horizontal speed of aircraft i at time step t, while $V_{\min}^i$ and $V_{\max}^i$ denote aircraft i's minimum and maximum horizontal speed, respectively. In Eq. (10), $\left\langle \mathbf{v}_t^i, \mathbf{v}_{t+1}^i \right\rangle$ denotes the angle between $\mathbf{v}_t^i$ and $\mathbf{v}_{t+1}^i$ in the horizontal dimension (i.e., the yaw angle of aircraft i at time step t), while $\omega_{\max}^i$ denotes the maximum heading deflection rate of aircraft i. Note that these two performance constraints are only considered in the horizontal dimension. Flight level stability constraint is represented as:

$$\mathbf{p}_t^i \in F_k, t = t_{\text{Start}}, \ldots, t_{\text{End}} - 1, k \in K \tag{11}$$

where F_k denotes the flight level k, and K is the set of flight levels of interest. It is important to note that the aircraft is assumed to perform uniform linear motion during each time step. Therefore, if the aircraft's position at the start or end of a time step is at a flight level, it is ensured that the aircraft is either cruising at a certain flight level or flying to another altitude level.

To summarise the formulation above, we define $\mathbb{L}$ as the set of trajectories for all aircraft:

$$\mathbb{L} = \left\{ l_i, i \in I \left| \begin{array}{l} \forall j \in J_i, t = t_{\text{Start}}, \ldots, t_{\text{End}} - 1, \\ \forall x \in (0, 1], k \in K : \\ \left(\begin{array}{l} \left\|\mathbf{p}_t^i(x)\mathbf{p}_t^j(x)\right\|^{\text{Hori}} \geq D^{\text{Hori}} \\ \vee \left\|\mathbf{p}_t^i(x)\mathbf{p}_t^j(x)\right\|^{\text{Vert}} \geq D^{\text{Vert}} \end{array} \right) \\ \wedge V_{\min}^i \leq v_t^i \leq V_{\max}^i \\ \wedge \left\langle \mathbf{v}_t^i, \mathbf{v}_{t+1}^i \right\rangle \leq \omega_{\max}^i \Delta t \\ \wedge \mathbf{p}_t^i \in F_k \end{array} \right\} \tag{12}$$

In order to find an optimal policy that minimises the expected value of the average flight distance of all aircraft in the same scenario, we define the objective model as:

$$\arg\min_{\pi_\theta} \mathbb{E}\left[\frac{1}{N} \sum_{i \in I} \|l_i\| \,|\pi_\theta \right] \tag{13}$$

where $\|l_i\|$ denotes the distance of trajectory l_i.

4. General multi-agent reinforcement learning

In this section, we first present the top-level framework of the proposed MARL-based 3DMACR method. Then, we introduce the key ingredients of the RL environment: action, state, and reward. Then, we discuss the agent working mechanism, highlighting the neural network structure. Finally, we introduce the training algorithm.

4.1. Top-level framework

The top-level framework of the proposed MARL-based 3DMACR method is shown in Fig. 2. In this method, an agent is deployed on each aircraft (i.e., the agent can be regarded as an aircraft). Agents construct a distributed decision-making system and share the same neural network for the model's generalisation. Agents operate in the 'self-interested' mode. It should be noted that there is no need for communication between the agents. Communication here refers specifically to agents informing other agents of their chosen actions. Information about other aircraft required by the agents, such as current speed, heading, position, and target points, can be obtained through onboard sensing devices or air-ground information-sharing platforms (refer to assumption 7). At each step, agents observe the state from the RL environment using a partial observation model based on imminent threats (see Section 4.3.1). The agent's state is represented in a modular observation matrix (see Section 4.3.2). The agent selects an action from the action space based on flight intention by the neural network (see Section 4.2.1). The flight intention is calculated to a manoeuvre using an

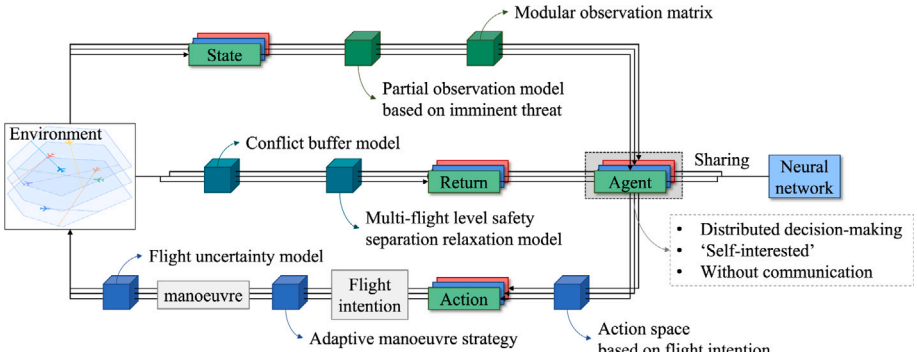


Fig. 2. Top-level framework of the MARL-based 3DMACR method.

adaptive manoeuvre strategy (see Section 4.2.2). The manoeuvre is transformed into the actual trajectory by the flight uncertainty model to update the environment (see Section 4.2.3). Supported by the safety separation relaxation model for multiple flight levels (see Section 4.4.1) and the conflict buffer model (see Section 4.4.2), the updated environment provides feedback to the agent through the designed reward function (see Section 4.4.3). The received reward updates the neural network to make the agent's decision-making ability close to the optimal policy step by step (see Sections 4.5 and 4.6).

4.2. Action

4.2.1. Action space based on flight intention

To simplify the agent's learning task, the action space **a** is discretised and consists of flight intentions, rather than specific manoeuvres. As this study takes into account the speed differences of the aircraft and multiple flight levels, the action a_t at time step t comprises three sub-actions: speed intention a_t^{S}, heading intention a_t^{H} and altitude intention a_t^{A} (the aircraft's number i is omitted for clarity). Thus, the action can be represented as a three-tuple:

$$a_t = \left(a_t^{\mathrm{S}}, a_t^{\mathrm{H}}, a_t^{\mathrm{A}}\right) \tag{14}$$

In this method, the agent has three choices for each sub-intention. They are defined as follows:

- **Speed intention**: fly at (1) high, (2) normal or (3) high speed.
- **Heading intention**: flying to the end point by (1) yawing to the left, (2) aiming for the end point, or (3) yawing to the right.
- **Altitude intention**: (1) climbing upwards to the next upper flight level, (2) maintaining at the current flight level, or (3) descending downwards to the next lower flight level

Based on the definition of flight intention, the three sub-intentions, respectively, are represented as:

$$a_t^{\mathrm{S}} = \begin{cases} 1 & \text{Low speed} \\ 2 & \text{Normal speed} \\ 3 & \text{High speed} \end{cases} \tag{15}$$

$$a_t^{\mathrm{H}} = \begin{cases} 1 & \text{Yaw left} \\ 2 & \text{Aiming} \\ 3 & \text{Yaw right} \end{cases} \tag{16}$$

$$a_t^{\mathrm{A}} = \begin{cases} 1 & \text{Climbing} \\ 2 & \text{Maintaining} \\ 3 & \text{Descending} \end{cases} \tag{17}$$

Hence, by combining the three sub-intentions, the agent has 27 actions (flight intentions) to be selected at time step t (i.e., $a_t \in [1, 2, \ldots, 27]$). Each action number represents a specific flight intention, and their correspondence is given by:

$$a_t = a_t^{\mathrm{S}} + 3 \times \left(a_t^{\mathrm{H}} - 1\right) + 9 \times \left(a_t^{\mathrm{A}} - 1\right) \tag{18}$$

In Eq. (18), since $a_t^{\mathrm{S}}, a_t^{\mathrm{H}}, a_t^{\mathrm{A}} \in [1, 2, 3]$, a_t and $\left(a_t^{\mathrm{S}}, a_t^{\mathrm{H}}, a_t^{\mathrm{A}}\right)$ have a unique correspondence. For instance, when a_t is 12, $\left(a_t^{\mathrm{S}}, a_t^{\mathrm{H}}, a_t^{\mathrm{A}}\right)$ must be $(3, 1, 2)$. Thus, $a_t = 12$ corresponds to the flight intention of 'high speed – yaw left – maintaining'.

4.2.2. Adaptive manoeuvre strategy

The adaptive manoeuvre strategy converts the selected action a_t into a specific manoeuvre. The manoeuvre consists of speed, heading and altitude manoeuvres corresponding to the heading, speed and altitude intentions, respectively. Note that the speed and heading manoeuvres are discussed in the horizontal dimension. Therefore, the velocity $\mathbf{v}_t$ is decomposed as:

$$\mathbf{v}_t = \mathbf{v}_t^{\mathrm{H}} + \mathbf{v}_t^{\mathrm{V}} \tag{19}$$

where $\mathbf{v}_t^{\mathrm{H}}$ and $\mathbf{v}_t^{\mathrm{V}}$ are the horizontal and vertical components of $\mathbf{v}_t$, respectively.

In accordance with Eq. (15), the speed intention comprises of low, normal and high speeds. The exact values of the three speeds depend on several factors such as aircraft performance, airline preference, and aircraft load. This study assumes that three speeds are assigned to each aircraft as follows:

$$v_t^{\mathrm{H}} = \left\|\mathbf{v}_t^{\mathrm{H}}\right\| = \begin{cases} V^{\mathrm{Low}} & a_t^{\mathrm{S}} = 1 \\ V^{\mathrm{Normal}} & a_t^{\mathrm{S}} = 2 \\ V^{\mathrm{High}} & a_t^{\mathrm{S}} = 3 \end{cases} \tag{20}$$

where V^{Low}, V^{Normal} and V^{High} denote the low, normal and high speeds in the horizontal dimension, respectively. The normal speed is the average of low and high speeds and can be expressed as:

$$V^{\mathrm{Normal}} = \frac{V^{\mathrm{Low}} + V^{\mathrm{High}}}{2} \tag{21}$$

Fig. 3 shows the diagram of the heading manoeuvre. In the proposed method, two headings are defined for an aircraft:

- **Flight heading** refers to the direction in which the aircraft is currently flying, represented by $\mathbf{v}_t^{\mathrm{H}}$.
- **Reference heading** refers to the direction pointed by the aircraft towards its end point, represented by $\overrightarrow{\mathbf{p}_t\mathbf{p}_{\mathrm{End}}}^{\mathrm{Hori}}$, which is the horizontal component of $\overrightarrow{\mathbf{p}_t\mathbf{p}_{\mathrm{End}}}$.

To guide the aircraft to its end point more easily, the heading intention is based on the reference (refer to Eq. (16)). At time step t, if the heading intention is 'aiming', the heading manoeuvre (i.e., the direction of velocity $\mathbf{v}_t^{\mathrm{H}}$) is the same as the aircraft's reference heading. If the heading intention is 'yaw left' or 'yaw right', the heading manoeuvre is to deflect α_t to the left (anticlockwise) or the right (clockwise, that is, $-\alpha_t$) based on the reference heading, respectively. Therefore, the aircraft's turning angle at time step t is denoted as $\left\langle \mathbf{v}_{t-1}^{\mathrm{H}}, \mathbf{v}_t^{\mathrm{H}} \right\rangle$, which is the heading change from $\mathbf{v}_{t-1}^{\mathrm{H}}$ to $\mathbf{v}_t^{\mathrm{H}}$. $\left\langle \mathbf{v}_{t-1}^{\mathrm{H}}, \mathbf{v}_t^{\mathrm{H}} \right\rangle$ is positive if $\mathbf{v}_t^{\mathrm{H}}$ is anticlockwise to $\mathbf{v}_{t-1}^{\mathrm{H}}$; otherwise, $\left\langle \mathbf{v}_{t-1}^{\mathrm{H}}, \mathbf{v}_t^{\mathrm{H}} \right\rangle$ is negative. We use a heading manoeuvre supervisor to ensure that the heading change is smooth, which means that $\left|a_t^{\mathrm{H}} - a_{t-1}^{\mathrm{H}}\right| \le 1$. To effectively resolve conflicts, we assume that when the aircraft selects to yaw, it always yaws to the maximum angle allowed. The maximum yaw angle $\alpha_{\max}$ is determined based on the aircraft performance and flight distance at each time step. To ensure that $\left\langle \mathbf{v}_{t-1}^{\mathrm{H}}, \mathbf{v}_t^{\mathrm{H}} \right\rangle \le \alpha_{\max}$ in any case, we only need to consider the worst-case scenarios, and we propose an adaptive yaw angle (as shown in Fig. 4). For instance, if $a_{t-1}^{\mathrm{H}} = 2$, $a_t^{\mathrm{H}} = 1$ and $a_{t+1}^{\mathrm{H}} = 2$, the absolute value of $\left\langle \mathbf{v}_t^{\mathrm{H}}, \mathbf{v}_{t+1}^{\mathrm{H}} \right\rangle$ is the greatest. Let $\left\langle \mathbf{v}_t^{\mathrm{H}}, \mathbf{v}_{t+1}^{\mathrm{H}} \right\rangle = \alpha_{\max}$ and sine theorems can calculate the value of α_t:

$$\alpha_t = \alpha_{\max} - \arcsin\left(\frac{\left\|\mathbf{p}_t\mathbf{p}_{t+1}\right\|^{\mathrm{Hori}} \times \sin\alpha_{\max}}{\left\|\mathbf{p}_t\mathbf{p}_{\mathrm{End}}\right\|^{\mathrm{Hori}}}\right) \tag{22}$$

where $\left\|\mathbf{p}_t\mathbf{p}_{t+1}\right\|^{\mathrm{Hori}}$ can be represented as:

$$\left\|\mathbf{p}_t\mathbf{p}_{t+1}\right\|^{\mathrm{Hori}} = \begin{cases} V^{\mathrm{Low}} \times \Delta t & a_t^{\mathrm{S}} = 1 \\ V^{\mathrm{Normal}} \times \Delta t & a_t^{\mathrm{S}} = 2 \\ V^{\mathrm{High}} \times \Delta t & a_t^{\mathrm{S}} = 3 \end{cases} \tag{23}$$

Note that although the yaw angle decreases as the aircraft gets closer to the exit point, the adaptive yaw angle does not significantly reduce the aircraft's manoeuvrability in general (refer to Appendix A.1 for more analysis).

In this method, the horizontal and altitude manoeuvres are calculated independently. With respect to the altitude manoeuvre, we will set the length of time step Δt to an appropriate value so that the aircraft can climb upwards to the next upper flight level or descend downwards to the next lower one in a single time step. Additionally, as a safety separation relaxation model for multiple flight levels is applied in the proposed method (as discussed in Section 4.4.1), the difference in climb/descent rates between aircraft does not affect the conflict detection calculation. Therefore, the rate of altitude change v_t^{V} can be simplified as follows:

$$v_t^{\mathrm{V}} = \left\|\mathbf{v}_t^{\mathrm{V}}\right\| = \begin{cases} \frac{S^{\mathrm{FL}}}{\Delta t} & a_t^{\mathrm{A}} = 1 \\ 0 & a_t^{\mathrm{A}} = 2 \\ -\frac{S^{\mathrm{FL}}}{\Delta t} & a_t^{\mathrm{A}} = 3 \end{cases} \tag{24}$$

where S^{FL} denotes the vertical distance between two adjacent flight levels, which is typically 1000 ft.

An altitude manoeuvre supervisor is deployed to prevent aircraft from climbing or descending out of the altitude range of interest. For instance, if FL360 is the highest flight level in a scenario, and an aircraft's current flight level is FL360, but its altitude intention

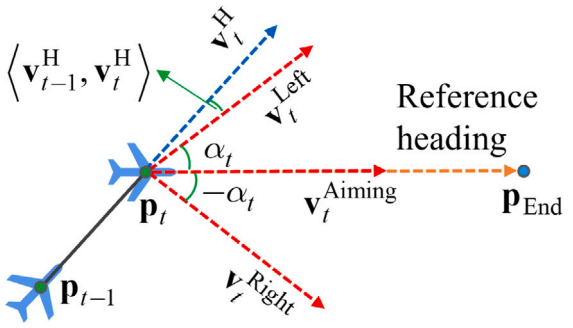


Fig. 3. Heading manoeuvre based on the agent's action. For example, if the agent intends to yaw to the left, its target heading $\mathbf{v}_t^{H}$ would be $\mathbf{v}_t^{Left}$. Suppose the heading at the last time step is $\mathbf{v}_{t-1}^{H}$. The turning angle for the current time step can be calculated as $\langle \mathbf{v}_{t-1}^{H}, \mathbf{v}_t^{H} \rangle$.

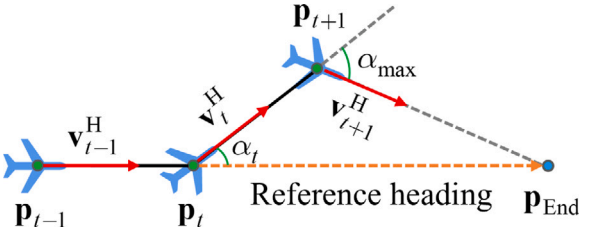


Fig. 4. Adaptive yaw angle. The yaw angle α_t is adaptive to guarantee that the subsequent yaw angle meets the aircraft performance constraints.

is to climb to the next higher flight level ($a_t^{A} = 1$), the altitude intention will be automatically changed to maintain the current flight level ($a_t^{A} = 2$).

4.2.3. *Flight uncertainty model*

In this study, we have taken into account flight uncertainty to verify the effectiveness of the proposed method in coping with it. However, developing a completely realistic flight uncertainty model is outside the scope of this study. Therefore, we have constructed a simplified flight uncertainty model based on state-of-the-art research on MACR with flight uncertainty (Wang et al., 2020). To account for flight uncertainty, we have made the following assumptions:

1. Mechanical errors may result in uncertainty in speed.
2. Navigation errors may result in uncertainty in heading.
3. To account for the effect of wind on flight, the actual velocity of the aircraft is simplified as the vector sum of its velocity and the wind velocity.
4. Flight uncertainty is considered in the horizontal dimension, but not in the vertical dimension.

Fig. 5 illustrates the symbol schematic for the flight uncertainty model. At time step t, the aircraft has a speed error v_t^{E} and a heading error α_t^{E}, which follow standard normal distributions:

$$v_t^{E} \sim N(0,1), v_t^{E} \in \left[-V_t^{E}, V_t^{E}\right] \tag{25}$$

$$\alpha_t^{E} \sim N(0,1), \alpha_t^{E} \in \left[-A_t^{E}, A_t^{E}\right] \tag{26}$$

$N(0,1)$ represents the standard normal distribution. The values of V_t^{E} and A_t^{E} are determined based on the degree of speed and heading changes, respectively:

$$V_t^{E} = \left|v_t - v_{t-1}\right| \times \varepsilon^{S} + E^{S} \tag{27}$$

$$A_t^{E} = \left|\langle \mathbf{v}_t, \mathbf{v}_{t-1} \rangle\right| \times \varepsilon^{H} + E^{H} \tag{28}$$

ε^{S} and ε^{H} represent the error factors for speed and heading, respectively. E^{S} and E^{H} are the basic error ranges for speed and heading, respectively. These constants will be provided in Section 5.1. As shown in Eqs. (27) and (28), the error's range for speed and heading increases as the change in the respective variable increases. Even if there is no change, an error is still present. The manoeuvre $\mathbf{v}_t^{H}$ is transformed into the error-based velocity $\mathbf{v}_t^{E}$ by applying the speed error v_t^{E} and heading error α_t^{E}. In this study,

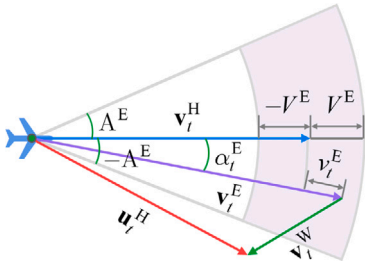


Fig. 5. Symbol schematic for the flight uncertainty model. The aircraft's horizontal velocity $\mathbf{v}_t^{\mathrm{H}}$ (blue arrow), error-based velocity $\mathbf{v}_t^{\mathrm{E}}$ (purple arrow), wind velocity $\mathbf{v}_t^{\mathrm{W}}$ (green arrow) and actual horizontal velocity $\mathbf{u}_t^{\mathrm{H}}$ (red arrow) are highlighted. The light purple area represents the range of error-based velocity $\mathbf{v}_t^{\mathrm{E}}$.

a scenario is provided with a base wind $\mathbf{v}^{\mathrm{BW}}$, which consists of a base wind direction α^{BW} and a base wind speed v^{BW}. The wind speed v_t^{W} is represented as:

$$v_t^{\mathrm{W}} = v^{\mathrm{BW}} + v_t^{\mathrm{C}} \tag{29}$$

v_t^{C} is the change of wind speed at time step t:

$$v_t^{\mathrm{C}} \sim N(0,1), v_t^{\mathrm{C}} \in \left[-V^{\mathrm{WC}}, V^{\mathrm{WC}}\right] \tag{30}$$

where $\left[-V^{\mathrm{WC}}, V^{\mathrm{WC}}\right]$ represents the range of wind speed change based on the base wind speed V^{BW}. The actual horizontal velocity of the aircraft $\mathbf{u}_t^{\mathrm{H}}$ is then represented as:

$$\mathbf{u}_t^{\mathrm{H}} = \mathbf{v}_t^{\mathrm{E}} + \mathbf{v}_t^{\mathrm{W}} \tag{31}$$

Since flight uncertainty in the vertical dimension is not taken into account, the aircraft's actual velocity is represented as:

$$\mathbf{u}_t = \mathbf{u}_t^{\mathrm{H}} + \mathbf{v}_t^{\mathrm{V}} \tag{32}$$

4.3. State

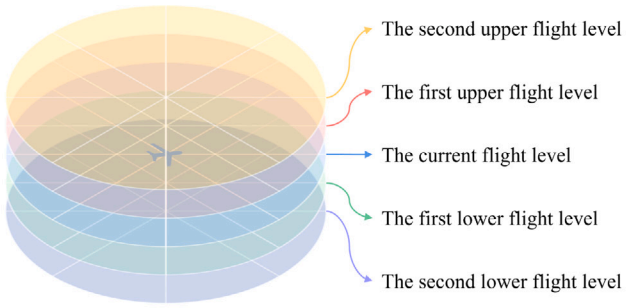
4.3.1. Partial observation model based on imminent threats

In RL methods, the approaches to obtaining the agent's state are typically categorised as global or local observations. In the context of this study's problem definition (see Section 3.1) and in the real world, the aircraft can obtain a global observation comprising all aircraft state information within the detection area. Despite providing less information than global observation, local observation offers two significant advantages over global observation in the 3DMACR problem.

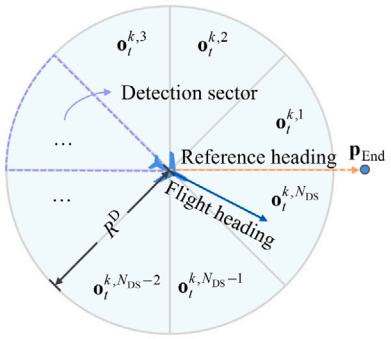
1. Aircraft are generally more affected by nearby intruders. If the aircraft only partially observes within an appropriate range, agents can focus on critical information, even though there is no guarantee of the global optimal solution. As such, learning efficiency in solving local CR will be significantly improved.
2. More importantly, intruder-based global observations are typically incompatible with fixed-dimensional observation matrices because the number of observed pieces of information varies with the number of intruders. Area-based partial observation effectively solves the dimensionality problem of the observation matrix because the number of observation areas can be preset to a fixed value.

Given that this study aims to explore a general MARL method, we favour local observation methods.

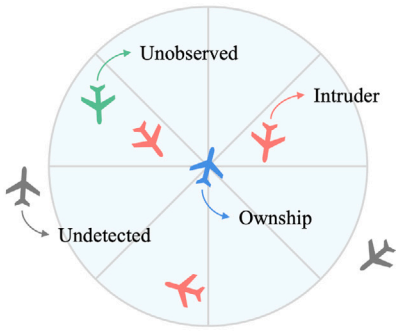
Taking inspiration from autonomous collision avoidance robots that use laser scanning for partial observation (Fan et al., 2020), we proposed a partial observation approach based on imminent threats in detection sectors (as shown in Fig. 6). To effectively use the partial observation method to solve MACR problems in three dimensions, an aircraft needs to observe five flight levels centred on its current flight level, which ensures that potential conflicts can be detected in a timely manner. The five flight levels (as shown in Fig. 6(a)), namely the second upper flight level, the first upper flight level, the current flight level, the first lower flight level and the second lower flight level, are represented as k^{U2}, k^{U1}, k^{C}, k^{L1} and k^{L2}, respectively. The detection area is a circle centred on the aircraft at each flight level detected (refer to Fig. 6(b)). The radius of the circle is denoted by R^{D} (i.e., the detection range), and it is divided into N_{DS} detection sectors arranged counter-clockwise from the reference heading. The agent obtains a corresponding sub-observation $\mathbf{o}_t^{k,s}$ from each detection sector, where k and s are flight level and detection sector numbers, respectively. The aircraft



(a) detection areas for multi-flight level



(b) detection sectors and reference heading



(c) aircraft classification

Fig. 6. Partial observation based on imminent threats for multi-flight level.

number i is omitted for better readability. Thus, the agent's observation at time step t is represented as:

$$\mathbf{o}_t = \begin{bmatrix} \mathbf{o}_t^{k^{\text{U2}},1} & \cdots & \mathbf{o}_t^{k^{\text{U2}},N_{\text{DS}}} \\ \mathbf{o}_t^{k^{\text{U1}},1} & \cdots & \mathbf{o}_t^{k^{\text{U1}},N_{\text{DS}}} \\ \mathbf{o}_t^{k^{\text{C}},1} & \cdots & \mathbf{o}_t^{k^{\text{C}},N_{\text{DS}}} \\ \mathbf{o}_t^{k^{\text{L1}},1} & \cdots & \mathbf{o}_t^{k^{\text{L1}},N_{\text{DS}}} \\ \mathbf{o}_t^{k^{\text{L2}},1} & \cdots & \mathbf{o}_t^{k^{\text{L2}},N_{\text{DS}}} \end{bmatrix} \tag{33}$$

where observation $\mathbf{o}_t$ consists of $5N_{\text{DS}}$ sub-observations based on detection sectors. Note that since the proposed method uses partial observation, the agent's input is the partial observation $\mathbf{o}_t$ rather than the global state $\mathbf{s}_t$. Aircraft are classified into four categories based on their detection status, as shown in Fig. 6(c):

- **The ownship** refers to the aircraft conducting the detection.
- **The undetected** refers to the outside the ownship's detection area.
- **The intruder** refers to the closest aircraft to the ownship within each detection sector and is the only aircraft observed in a given detection sector.
- **The unobserved** refers to the aircraft in the detection area other than intruders.

4.3.2. Modular observation matrix

The agent learns information about intruders in the corresponding detection sector through the sub-observation matrix $\mathbf{o}_t^{k,s}$, which consists of three modules of intruder information:

$$\mathbf{o}_t^{k,s} = \left[\left(\mathbf{o}^{\text{L}}\right)_t^{k,s}, \left(\mathbf{o}^{\text{P}}\right)_t^{k,s}, \left(\mathbf{o}^{\text{T}}\right)_t^{k,s}, \left(\mathbf{o}^{\text{I}}\right)_t^{k,s} \right] \tag{34}$$

where $\left(\mathbf{o}^{\text{L}}\right)_t^{k,s}$, $\left(\mathbf{o}^{\text{P}}\right)_t^{k,s}$, $\left(\mathbf{o}^{\text{T}}\right)_t^{k,s}$ and $\left(\mathbf{o}^{\text{I}}\right)_t^{k,s}$ represent the location, performance, target, and interest modules, respectively, and serve the following purposes

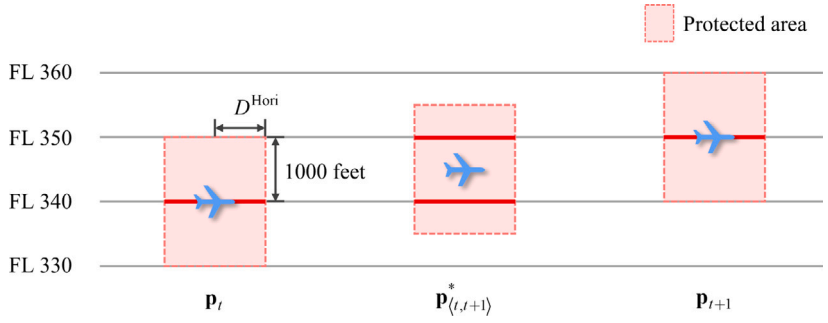


Fig. 7. Diagram of the multi-flight level safety separation relaxation model (2D side view). The protected areas are highlighted in red, but the dotted lines are not included. The thick red solid lines represent the flight levels and the corresponding ranges occupied by the aircraft.

1. The location module reveals the intruder's current position.
2. The performance module displays the intruder's low, normal, and high speeds.
3. The target module indicates the location of the intruder's target point.
4. The interest module marks whether the detection sector is of interest.

For more information on the design of the modular observation matrix, please refer to Appendix A.2.

Of the four modules of the sub-observation matrix $\mathbf{o}_t^{k,s}$, the location module $\left(\mathbf{o}^{\text{L}}\right)_t^{k,s}$ and interest module $\left(\mathbf{o}^{\text{I}}\right)_t^{k,s}$ are essential for the agent to understand the current situation. The performance module $\left(\mathbf{o}^{\text{P}}\right)_t^{k,s}$ and the target module $\left(\mathbf{o}^{\text{T}}\right)_t^{k,s}$ provide more detailed information to the agent. However, since the agents in the proposed method make distributed decisions, even if an agent has all the information except the intruder's manoeuvre at the next step, accurately predicting the intruder's future flight trajectory is difficult. Since a conflict buffer model is applied in the proposed method, an agent may be able to resolve conflicts appropriately by relying only on the intruder's location information. Although the performance module $\left(\mathbf{o}^{\text{P}}\right)_t^{k,s}$ and the target module $\left(\mathbf{o}^{\text{T}}\right)_t^{k,s}$ provide the agent with more information about the scenario, for RL methods, more information means a larger observation matrix dimension and a higher learning difficulty. It is difficult to determine whether the benefits of observing the intruder's aircraft performance and flight target outweigh the drawbacks of doing so. Therefore, experimental results are necessary to determine whether including the performance module $\left(\mathbf{o}^{\text{P}}\right)_t^{k,s}$ and the target module $\left(\mathbf{o}^{\text{T}}\right)_t^{k,s}$ in the sub-observation matrix $\mathbf{o}_t^{k,s}$ benefits the method's performance (refer to Section 5.4.1).

4.4. Reward

4.4.1. Safety separation relaxation model for multiple flight levels

As discussed in Section 4.2.1, the agent is required to select an action at each time step. However, in real-world scenarios, the rate of climb/descent can vary significantly between different aircraft. Training agents to accurately predict an aircraft's altitude by learning its rate of climb/descent in an already complex environment can substantially increase the difficulty of solving 3DMACR problems. To address this challenge, we propose a safety separation relaxation model for multiple flight levels, which simplifies the learning task for predicting aircraft altitudes.

The diagram of the multi-flight level safety separation relaxation model is shown in Fig. 7. Since the vertical safety separation typically refers to the vertical separation between flight levels, which is usually 1000 ft, the following assumptions are made:

1. If an aircraft is flying at a flight level (i.e., its altitude flight intention is to maintain at the current flight level) at a given time step, it is assumed to occupy only the current flight level with a circle of radius D^{Hori}.
2. If an aircraft is flying between two flight levels (i.e., its altitude flight intention is to climb or descend to the next flight level) at a given time step, it is assumed to occupy both the current and next flight levels with a circle of radius D^{Hori}.

Based on the multi-flight level safety separation relaxation model, we use $\left(K^{\text{Occ}}\right)_t^i$ to represent the set of flight levels occupied by aircraft i at time step t, and it can be expressed as follow:

$$\left(K^{\text{Occ}}\right)_t^i = \begin{cases} \{k, k^{\text{U}}\} & \text{Climbing from flight level } k \\ \{k\} & \text{Maintaining at flight level } k \\ \{k^{\text{L}}, k\} & \text{Descending from flight level } k \end{cases} \tag{35}$$

where k^{U} and k^{L} denote the next upper and lower flight levels for flight level k, respectively. With this definition, we can express the 3D safety separation constraint for aircraft i at time step t as:

$$\begin{aligned} &\left(\left\|\mathbf{p}_t^i(x)\,\mathbf{p}_t^j(x)\right\|^{\text{Hori}} \geq D^{\text{Hori}}\right) \vee \\ &\left(\left(K^{\text{Occ}}\right)_t^i \bigcap \left(K^{\text{Occ}}\right)_t^j = \varnothing\right), x \in (0,1], j \in J_i \end{aligned} \tag{36}$$

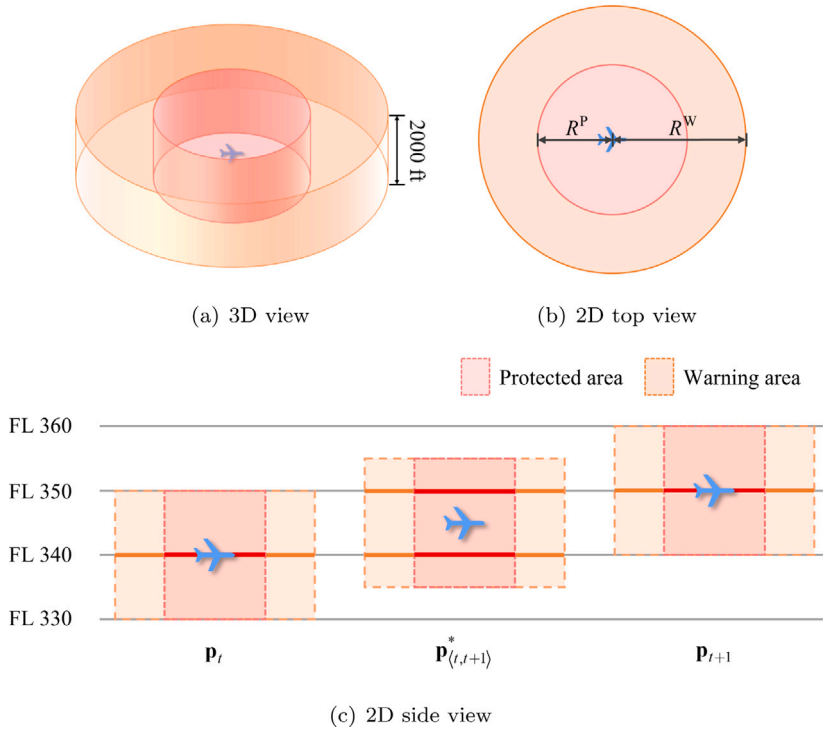


Fig. 8. Diagram of the conflict buffer model.

4.4.2. Conflict buffer model

Since non-stationarity in MARL systems is challenging to avoid (Papoudakis et al., 2019), and flight uncertainty is considered in this study, the proposed method is expected to be more robust in practice. Therefore, we propose a conflict buffer model, where each aircraft is assigned a protected area and a warning area. Fig. 8 illustrates the conflict buffer model, where R^W and R^P denote the radii of the warning and protected areas, respectively, and $R^P = D^{Hori}$.

In RL methods, intruders inside the protected area always trigger a major penalty for the ownship. In our method, the proposed conflict buffer model allows the ownship to receive a minor penalty if an intruder is detected within the warning area. The agent's behaviour tends to be more conservative to avoid penalties, and it strives to keep aircraft out of the warning area as much as possible. If some intruders accidentally enter the warning area due to flight uncertainty or the non-stationarity of MARL, the agent can respond promptly to prevent potential conflicts. Thus, the conflict buffer model is expected to buffer aircraft that are about to breach safe separation and mitigate the problem arising from the non-stationarity of MARL.

We use the marks $\left(m^P\right)_t^i$ and $\left(m^W\right)_t^i$ to indicate the presence of intruders in aircraft i's protected and warning areas at time step t, respectively:

$$\left(m^P\right)_t^i = \begin{cases} & \exists j \in J_i, x \in (0,1]: \\ 1 & \left(\left\|\mathbf{p}_t^i(x)\,\mathbf{p}_t^j(x)\right\|^{Hori} \le R^P\right) \wedge \\ & \left(\left(K^{Occ}\right)_t^i \bigcap \left(K^{Occ}\right)_t^j \neq \varnothing\right) \\ 0 & \text{Otherwise} \end{cases} \tag{37}$$

$$\left(m^W\right)_t^i = \begin{cases} & \exists j \in J_i, x \in (0,1]: \\ 1 & \left(\left\|\mathbf{p}_t^i(x)\,\mathbf{p}_t^j(x)\right\|^{Hori} \le R^W\right) \wedge \\ & \left(\left(K^{Occ}\right)_t^i \bigcap \left(K^{Occ}\right)_t^j \neq \varnothing\right) \\ 0 & \text{Otherwise} \end{cases} \tag{38}$$

4.4.3. Reward function design

As discussed in Section 3.1, the optimisation objective of the proposed method is to minimise the average extra flight distance while ensuring that the trajectory complies with the 3D safety separation and aircraft performance constraints. It is important to note that RL models have no objective functions and constraints like in optimisation models. Instead, a similar purpose is achieved

by assigning significantly higher weight coefficients to the constraints (indicators that need to be satisfied) in the reward function compared to the optimisation objectives (indicators that are expected to achieve preferences as much as possible). To guide the aircraft towards this objective, a reward function has been designed:

$$r_t^i = \left(r^{\mathrm{S}}\right)_t^i + \left(r^{\mathrm{E}}\right)_t^i \tag{39}$$

r_t^i denotes the reward of agent i at time step t. The value of the reward r_t^i depends on the safety and economy of the agent's action a_t^i. $\left(r^{\mathrm{S}}\right)_t^i$ and $\left(r^{\mathrm{E}}\right)_t^i$ denote the safety and economy rewards, respectively.

The safety reward $\left(r^{\mathrm{S}}\right)_t^i$ is calculated based on protected and warning areas:

$$\left(r^{\mathrm{S}}\right)_t^i = \left(r_{\mathrm{P}}^{\mathrm{S}}\right)_t^i + \left(r_{\mathrm{W}}^{\mathrm{S}}\right)_t^i \tag{40}$$

$\left(r_{\mathrm{P}}^{\mathrm{S}}\right)_t^i$ and $\left(r_{\mathrm{W}}^{\mathrm{S}}\right)_t^i$ quantify the degree to which intruders have penetrated the protected and warning areas, respectively:

$$\left(r_{\mathrm{P}}^{\mathrm{S}}\right)_t^i = \begin{cases} -\eta_{\mathrm{P}}^{\mathrm{S}}\left(\left(p_{\mathrm{P}}^{\mathrm{S}}\right)_t^i + 1\right)^{\mu_{\mathrm{P}}^{\mathrm{S}}} & \left(m^{\mathrm{P}}\right)_t^i = 1 \\ 0 & \text{Otherwise} \end{cases} \tag{41}$$

$$\left(r_{\mathrm{W}}^{\mathrm{S}}\right)_t^i = \begin{cases} -\eta_{\mathrm{W}}^{\mathrm{S}}\left(\left(p_{\mathrm{W}}^{\mathrm{S}}\right)_t^i + 1\right)^{\mu_{\mathrm{W}}^{\mathrm{S}}} & \left(m^{\mathrm{W}}\right)_t^i = 1 \wedge \left(m^{\mathrm{P}}\right)_t^i = 0 \\ 0 & \text{Otherwise} \end{cases} \tag{42}$$

$\left(p_{\mathrm{P}}^{\mathrm{S}}\right)_t^i$ and $\left(p_{\mathrm{W}}^{\mathrm{S}}\right)_t^i$ denote the penetration indices for protected and warning areas, respectively:

$$\left(p_{\mathrm{P}}^{\mathrm{S}}\right)_t^i = \frac{R^{\mathrm{P}} - \left(d_{\min}^{\mathrm{Hori}}\right)_t^i}{R^{\mathrm{P}}} \tag{43}$$

$$\left(p_{\mathrm{W}}^{\mathrm{S}}\right)_t^i = \frac{R^{\mathrm{W}} - R^{\mathrm{P}} - \left(d_{\min}^{\mathrm{Hori}}\right)_t^i}{R^{\mathrm{W}} - R^{\mathrm{P}}} \tag{44}$$

$\left(d_{\min}^{\mathrm{Hori}}\right)_t^i$ denotes the horizontal distance from the ownship to the nearest intruder occupying a flight level that overlaps with the ownship:

$$\left(d_{\min}^{\mathrm{Hori}}\right)_t^i = \min\left\{ \left\| \mathbf{p}_t^i(x)\, \mathbf{p}_t^j(x) \right\|^{\mathrm{Hori}} \middle| \begin{array}{l} x \in (0,1], j \in J_i, \\ \left(K^{\mathrm{Occ}}\right)_t^i \bigcap \left(K^{\mathrm{Occ}}\right)_t^j \neq \varnothing \end{array} \right\} \tag{45}$$

We refer to $\left(d_{\min}^{\mathrm{Hori}}\right)_t^i$ as the minimum approach distance of intruders for clarity. The return factors for $\left(r_{\mathrm{P}}^{\mathrm{S}}\right)_t^i$ and $\left(r_{\mathrm{W}}^{\mathrm{S}}\right)_t^i$ are denoted by $\eta_{\mathrm{P}}^{\mathrm{S}}$ and $\eta_{\mathrm{W}}^{\mathrm{S}}$, respectively, and they are positive numbers. The expansion factors for $\left(r_{\mathrm{P}}^{\mathrm{S}}\right)_t^i$ and $\left(r_{\mathrm{W}}^{\mathrm{S}}\right)_t^i$ are denoted by $\mu_{\mathrm{P}}^{\mathrm{S}}$ and $\mu_{\mathrm{W}}^{\mathrm{S}}$, respectively, and they are positive numbers greater than one. Therefore, $\left(r_{\mathrm{P}}^{\mathrm{S}}\right)_t^i$ and $\left(r_{\mathrm{W}}^{\mathrm{S}}\right)_t^i$ are negative numbers or equal to zero (i.e., negative feedback or penalties). To ensure that the agent receives a greater penalty when an intruder is in the protected area than when an intruder is only in the warning area, $\eta_{\mathrm{P}}^{\mathrm{S}}$ is greater than $\eta_{\mathrm{W}}^{\mathrm{S}}$. Using Eqs. (43) and (44), we can calculate that $\left(p_{\mathrm{P}}^{\mathrm{S}}\right)_t^i, \left(p_{\mathrm{W}}^{\mathrm{S}}\right)_t^i \in [0,1)$. Therefore, it should be ensured that $\eta_{\mathrm{P}}^{\mathrm{S}} \geq 2^{\mu_{\mathrm{W}}^{\mathrm{S}}} \eta_{\mathrm{W}}^{\mathrm{S}}$. The safety reward $\left(r^{\mathrm{S}}\right)_t^i$ is a segmented function with the minimum approach distance of intruders and can be integrally represented as (refer to Appendix A.3 for visualisation of the safety reward function):

$$\left(r^{\mathrm{S}}\right)_t^i = \left(r_{\mathrm{P}}^{\mathrm{S}}\right)_t^i + \left(r_{\mathrm{W}}^{\mathrm{S}}\right)_t^i = \begin{cases} -\eta_{\mathrm{P}}^{\mathrm{S}}\left(\frac{R^{\mathrm{P}} - \left(d_{\min}^{\mathrm{Hori}}\right)_t^i}{R^{\mathrm{P}}} + 1\right)^{\mu_{\mathrm{P}}^{\mathrm{S}}} & \left(d_{\min}^{\mathrm{Hori}}\right)_t^i \in \left[0, R^{\mathrm{P}}\right) \\ -\eta_{\mathrm{W}}^{\mathrm{S}}\left(\frac{R^{\mathrm{W}} - R^{\mathrm{P}} - \left(d_{\min}^{\mathrm{Hori}}\right)_t^i}{R^{\mathrm{W}} - R^{\mathrm{P}}} + 1\right)^{\mu_{\mathrm{W}}^{\mathrm{S}}} & \left(d_{\min}^{\mathrm{Hori}}\right)_t^i \in \left[R^{\mathrm{P}}, R^{\mathrm{W}}\right) \\ 0 & \text{Otherwise} \end{cases} \tag{46}$$

The economy reward $\left(r^{\mathrm{E}}\right)_t^i$ is computed based on speed, heading and altitude intentions:

$$\left(r^{\mathrm{E}}\right)_t^i = \left(r_{\mathrm{S}}^{\mathrm{E}}\right)_t^i + \left(r_{\mathrm{H}}^{\mathrm{E}}\right)_t^i + \left(r_{\mathrm{A}}^{\mathrm{E}}\right)_t^i \tag{47}$$

$\left(r_{\mathrm{S}}^{\mathrm{E}}\right)_t^i$, $\left(r_{\mathrm{H}}^{\mathrm{E}}\right)_t^i$ and $\left(r_{\mathrm{A}}^{\mathrm{E}}\right)_t^i$ denote the rewards for speed economy, heading economy and altitude economy, respectively. The reward for speed economy is represented as:

$$\left(r_{\mathrm{S}}^{\mathrm{E}}\right)_t^i = \begin{cases} -\eta_{\mathrm{SL}}^{\mathrm{E}} & a_t^{\mathrm{S}} = 1 \\ 0 & a_t^{\mathrm{S}} = 2 \\ -\eta_{\mathrm{SH}}^{\mathrm{E}} & a_t^{\mathrm{S}} = 3 \end{cases} \tag{48}$$

$\eta_{\mathrm{SL}}^{\mathrm{E}}$ and $\eta_{\mathrm{SH}}^{\mathrm{E}}$ denote the return factors for $\left(r_{\mathrm{S}}^{\mathrm{E}}\right)_t^i$ when the agent's speed intention is low speed and high speed, respectively, and they are positive numbers. In the 3DMACR problem, we assume that the agent's most desired speed is the normal speed, which is represented by $a_t^{\mathrm{S}} = 2$ (as described in Eq. (15)). Therefore, if the agent's speed intention is not normal speed, it will receive a penalty, as shown in Eq. (48). The reward for the heading economy is represented as:

$$\left(r_{\mathrm{H}}^{\mathrm{E}}\right)_t^i = \begin{cases} -\eta_{\mathrm{H}}^{\mathrm{E}} & a_t^{\mathrm{H}} \in \{1,3\} \\ 0 & a_t^{\mathrm{H}} = 2 \end{cases} \tag{49}$$

$\eta_{\mathrm{H}}^{\mathrm{E}}$ denotes the return factor for $\left(r_{\mathrm{H}}^{\mathrm{E}}\right)_t^i$, which is a positive number. When the agent's heading intention is 'aiming' (i.e., $a_t^{\mathrm{H}} = 2$, refer to Eq. (16)), the aircraft will be closer to its end point at the end of a time step, promoting the shorter extra flight distance. Due to the proposed adaptive manoeuvre strategy (refer to Section 4.2.2), yaw left and yaw left may result in different extra flight distances for an aircraft in different scenarios. However, since the primary purpose of reward $\left(r_{\mathrm{H}}^{\mathrm{E}}\right)_t^i$ is to encourage the aircraft to fly straight to its end point, this model ignores the difference to simplify the problem. Thus, as represented by Eq. (49), if the agent's heading intention is not 'aiming', the agent will receive a penalty. The reward for altitude economy is represented as:

$$\left(r_{\mathrm{A}}^{\mathrm{E}}\right)_t^i = \begin{cases} -\eta_{\mathrm{AC}}^{\mathrm{E}} & a_t^{\mathrm{A}} = 1 \\ 0 & a_t^{\mathrm{A}} = 2 \\ -\eta_{\mathrm{AD}}^{\mathrm{E}} & a_t^{\mathrm{A}} = 3 \end{cases} \tag{50}$$

$\eta_{\mathrm{AC}}^{\mathrm{E}}$ and $\eta_{\mathrm{AD}}^{\mathrm{E}}$ denote the return factors for $\left(r_{\mathrm{A}}^{\mathrm{E}}\right)_t^i$ when the agent's altitude intention is 'climbing' and 'descending', respectively, and they are positive numbers. In terms of fuel consumption, changing flight levels can have varying effects depending on factors such as aircraft performance, current flight level, and climb/descent rate. However, since this study is primarily concerned with maintaining the orderliness and stability of the airspace, we assume that aircraft should minimise changes in altitude as much as possible. Thus, as shown in Eq. (50), any deviation from maintaining altitude will result in a penalty for the agent.

4.5. Agent working mechanism and neural network structure

The workflow of a single agent is illustrated in Fig. 9, where the architecture of the neural network is highlighted. To begin with, the agent detects intruders and uses the partial observation model based on imminent threats to construct the modular observation matrix. Each detection sector's sub-observation $\mathbf{o}_t^{k,s}$ is composed of four modules: $\left(\mathbf{o}^{\mathrm{L}}\right)_t^{k,s}$, $\left(\mathbf{o}^{\mathrm{P}}\right)_t^{k,s}$, $\left(\mathbf{o}^{\mathrm{T}}\right)_t^{k,s}$ and $\left(\mathbf{o}^{\mathrm{I}}\right)_t^{k,s}$. Next, the observation $\mathbf{o}_t$ is flattened and inputted to the neural network's input layer. The number of neurons in the input layer equals the number of elements in the observation $\mathbf{o}_t$, which is $40N_{\mathrm{DS}}$. The neural network has two hidden layers. The first hidden layer, denoted as hidden layer 1, comprises $50N_{\mathrm{DS}}$ neurons, while the second consists of hidden layers 2 A and 2 V, representing the advantage function and value function, respectively, for the Duelling DQN algorithm (see Section 4.6). Hidden layer 2 V consists of a single neuron, whereas hidden layer 2 A and the output layer have 27 neurons each, which is the same number as the agent's optional actions. The neuron's value in the output layer is obtained by summing the corresponding neuron's values in hidden layers 2 A and 2 V. Therefore, hidden layers 2 A and 2 V are not fully connected to the output layer. However, the input layer and hidden layer 1 are fully connected layers. The action corresponding to the neuron with the maximum value in the output layer is selected as the agent's selected action a_t. The action a_t represents a flight intention, which can be decomposed into three sub-intentions, namely, a_t^{S}, a_t^{H} and a_t^{A}. The flight intention is transformed into velocity $\mathbf{v}_t$ using the adaptive manoeuvre strategy. The agent's actual velocity $\mathbf{u}_t$ for the current time step is then calculated based on the flight uncertainty and weather. The trajectory is calculated based on actual velocity $\mathbf{u}_t$ using Eq. (4) to update the environment. After that, the agent receives the reward r_i from the environment, which consists of the safety reward $\left(r^{\mathrm{S}}\right)_t$ and the economy reward $\left(r^{\mathrm{E}}\right)_t$. The safety reward $\left(r^{\mathrm{S}}\right)_t$ consists of $\left(r_{\mathrm{P}}^{\mathrm{S}}\right)_t$ and $\left(r_{\mathrm{W}}^{\mathrm{S}}\right)_t$, which are calculated based on the environment. The economy reward $\left(r^{\mathrm{E}}\right)_t$ consists of $\left(r_{\mathrm{S}}^{\mathrm{E}}\right)_t$, $\left(r_{\mathrm{H}}^{\mathrm{E}}\right)_t$ and $\left(r_{\mathrm{A}}^{\mathrm{E}}\right)_t$, which are calculated based on a_t^{S}, a_t^{H} and a_t^{A}, respectively. Finally, the neural network is updated by the reward r_t.

4.6. Training algorithm

Since the agent's action space only contains a limited number of actions (27, as discussed in Section 4.2.1), we chose to use DQN as our RL algorithm, which is highly efficient for discrete actions (Mnih et al., 2015). Over the years, various DQN performance improvement techniques have been proposed. Inspired by a recently proposed advanced DQN method, Rainbow DQN (Hessel et al., 2018), we adaptively employ and combine the Double-DQN (Van Hasselt et al., 2016), Duelling DQN (Wang et al., 2016), Prioritised Replay Buffer (Schaul et al., 2015) and Multi-Step Learning (Hernandez-Garcia and Sutton, 2019) to enhance the performance of DQN (refer to Appendix A.4 for the training algorithm's pseudo-code and description).

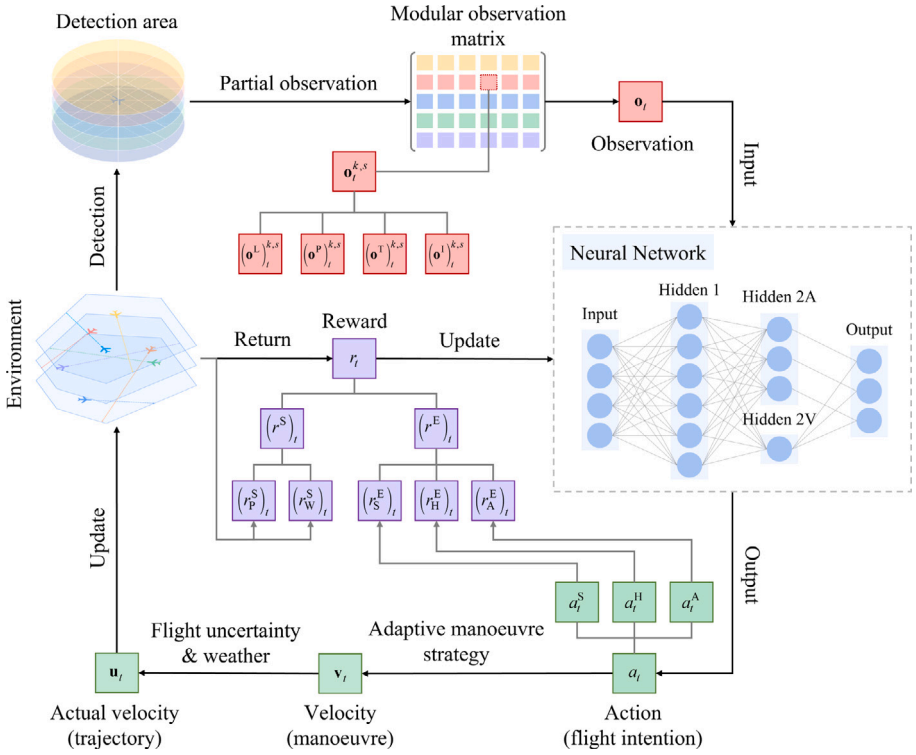


Fig. 9. Workflow of a single agent, which is part of Fig. 2. The neural network architecture is highlighted. The aircraft number i is omitted for clarity.

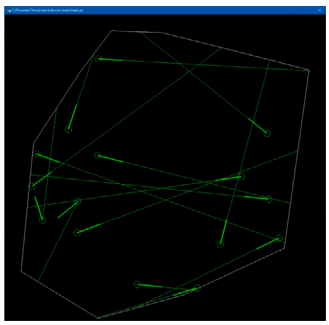


Fig. 10. Simulator's visualisation for a flight level.

5. Simulation experiments

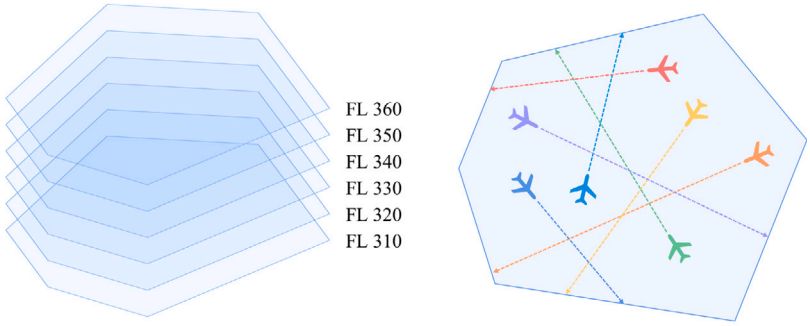
In this section, we present our experimental setup, including the training and test scenarios and the corresponding parameter settings. We then describe three sets of experiments and analyse their results, with the following purposes:

1. Section 5.4: Verify the effectiveness of the training, test the performance of the trained models, and determine which modules of the observation matrix are most suitable for the proposed method.
2. Section 5.5: Explore the impact of detection range and the number of detection sectors on the proposed method.
3. Section 5.6: Compare the performance differences between the proposed 3D model and the corresponding 2D model.

We conducted the simulation on an improved simulator based on Eurocontrol'sopensourcesimulator. Fig. 10 shows an example of the simulator's visualisation for a flight level.

5.1. Training scenarios

In this study, simulation data was used instead of real data for training scenarios due to two main reasons.



(a) airspace with multi-flight level (b) example of a flight level

Fig. 11. Generation of training scenarios.

Table 3
Differences between training scenarios and test scenarios.

Differences	Training scenarios	Test scenarios
Number of aircraft in a scenario $N_{A/C}$	$N_{A/C} = 60$	$N_{A/C} \in [36, 42, 48, 54, 60, 66, 72]$
Number of aircraft at a flight level $N^k_{A/C}$	$N^k_{A/C} \in [0, N_{A/C}]$	$N^k_{A/C} \in \left[\frac{N_{A/C}}{N_{FL}} - 2, \frac{N_{A/C}}{N_{FL}} + 2\right]$

1. In order to achieve a high-performance model, ultra-high traffic density scenarios are required for training, which is rare in the real world.
2. The proposed method targets the 3DMACR problems in free-route airspace, which is not commonly used in most airspace in the real world that still relies on fixed routes. Therefore, scenarios based on real data may not be general enough.

For each training scenario, a random airspace construction with six flight levels and sixty aircraft was generated. The algorithms used to generate the training scenarios and an explanation of the process can be found in Appendix B.1. An example of the generated training scenarios can be seen in Fig. 11.

5.2. Parameter setting

5.2.1. Simulator parameters

The parameters of the simulator were carefully set to replicate the real-world scenario as closely as possible based on the experimental purpose and conditions of this study. Readers are directed to Appendix B.2 for detailed simulator parameter settings and notes.

5.2.2. Environment parameters

The environment parameters serve the purpose of enabling the agent to learn a policy that satisfies the optimisation objectives. In practical applications, the return factors (namely, η^S_P, η^S_W, η^E_{SL}, η^E_{SH}, η^E_H, η^E_{AC} and η^E_{AD}) and the expansion factors (namely, μ^S_P and μ^S_W) can be adjusted as per the safety and economic efficiency preferences of the ATM system. Readers are directed to Appendix B.3 for detailed environment parameter settings and notes.

5.2.3. Hyper-parameters

Hyper-parameters are utilised to enhance the efficacy and efficiency of the training process. The neural network was constructed using the Pytorch library in the Python environment and was trained on a computer with an i5-8300H CPU. The network parameters were updated by the Adam optimisation algorithm (Kingma and Ba, 2014). Readers are directed to Appendix B.4 for detailed hyper parameter settings and notes.

5.3. Test scenarios

This study uses test scenarios to test the performance of the trained model. Test scenarios are generated in a similar way to training scenarios. The difference is the number of aircraft generated (as shown in Table 3).

Regarding the number of aircraft in a scenario, during training, we fixed $N_{A/C}$ to 60, as discussed in Section 5.2.1. However, during testing, we varied the number of aircraft to assess the performance of the trained model in different traffic density situations. Based on air traffic predictions (Ribeiro et al., 2020b) and flight distributions at each flight level (EUROCONTROL, 2018), the traffic density at the busiest flight level in the upper airspace is expected to reach 4.80 aircraft per 2500π square nautical miles by 2025 (see Appendix B.5 for more detail). To test the CR performance of the proposed method in high-density airspace, we generated

scenarios with seven different numbers of aircraft: 36, 42, 48, 54, 60, 66 and 72 (i.e., 6, 7, 8, 9, 10, 11 and 12 aircraft per flight level), which exceed the predicted traffic density in 2025. It should be noted that the number of aircraft was selected based on our experimental goals and does not reflect any theoretical limits or air traffic regulations.

In terms of the number of aircraft at a flight level ($N^k_{\mathrm{A/C}}$ denotes the number of aircraft at flight level k), during training, $N^k_{\mathrm{A/C}}$ was between zero and $N_{\mathrm{A/C}}$ (i.e., sixty), which means that the number of aircraft at a flight level was random and without limitation. This approach was adopted to ensure that agents are exposed to more (extreme) situations during the training process. On the other hand, during testing, we limited the number of aircraft generated at each flight level to a range that is more likely to occur in real-world scenarios (i.e., $N^k_{\mathrm{A/C}} \in \left[\frac{N_{\mathrm{A/C}}}{N_{\mathrm{FL}}} - 2, \frac{N_{\mathrm{A/C}}}{N_{\mathrm{FL}}} + 2\right]$). For instance, if the number of aircraft in a scenario is 54, the number of aircraft at a flight level is restricted to between 7 and 11.

Based on the aforementioned settings, we generated seven sets of test scenarios, each consisting of 100 random scenarios, with the numbers of aircraft being 36, 42, 48, 54, 60, 66 and 72, separately (i.e., their average numbers of aircraft per flight level were 6, 7, 8, 9, 10, 11 and 12, separately). In total, the test dataset comprised 700 scenarios.

5.4. Training and test

To validate the efficacy of our designed models, we trained them using the training scenarios outlined in Section 5.1 and evaluated their performances using the test scenarios outlined in Section 5.3. In this study, we developed four models with different modular observation matrices. As discussed in Section 4.3.2, the location and interest modules are essential to the modular observation matrix in our RL model. However, the role of the performance and target modules in the model needs further clarification. To address this, we created four observation matrices with varying inclusion of the performance and target modules and compared their performances. The sub-observation matrices for these four models (namely, model LI, LIP, LIT, and LIPT, respectively) are defined below:

$$(\mathbf{o}_t^{k,s})_{\mathrm{LI}} = \left[(\mathbf{o}^{\mathrm{L}})_t^{k,s}, (\mathbf{o}^{\mathrm{I}})_t^{k,s}\right] \tag{51}$$

$$(\mathbf{o}_t^{k,s})_{\mathrm{LIP}} = \left[(\mathbf{o}^{\mathrm{L}})_t^{k,s}, (\mathbf{o}^{\mathrm{P}})_t^{k,s}, (\mathbf{o}^{\mathrm{I}})_t^{k,s}\right] \tag{52}$$

$$(\mathbf{o}_t^{k,s})_{\mathrm{LIT}} = \left[(\mathbf{o}^{\mathrm{L}})_t^{k,s}, (\mathbf{o}^{\mathrm{T}})_t^{k,s}, (\mathbf{o}^{\mathrm{I}})_t^{k,s}\right] \tag{53}$$

$$(\mathbf{o}_t^{k,s})_{\mathrm{LIPT}} = \left[(\mathbf{o}^{\mathrm{L}})_t^{k,s}, (\mathbf{o}^{\mathrm{P}})_t^{k,s}, (\mathbf{o}^{\mathrm{T}})_t^{k,s}, (\mathbf{o}^{\mathrm{I}})_t^{k,s}\right] \tag{54}$$

5.4.1. Training curves

The training curves are presented in Fig. 12, which illustrate the performance of each model during training. Each model underwent a million episodes of training, and the trained model's overall performance was assessed based on the average reward. The average reward is defined as the average reward of all actions taken by all aircraft in each episode, and is calculated using the following formula:

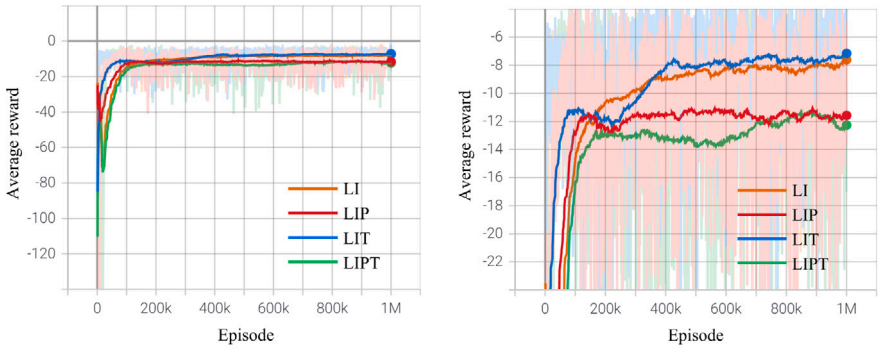
$$r_j^{\mathrm{Ave}} = \frac{\sum_i \sum_{t \in T_i^j} r_t^i}{\sum_i \left(N_{\mathrm{A}}\right)_i^j} \tag{55}$$

where r_j^{Ave} denotes the average reward of episode j, T_i^j denotes the set of time steps of aircraft i in episode j, and $\left(N_{\mathrm{A}}\right)_i^j$ denotes the number of actions performed by aircraft i in episode j.

It can be observed that the curves of all four models have converged within one million episodes, indicating that the training of the four models was effective. Based on the average rewards obtained during training, the overall performance order of the four models is LIT > LI > LIP > LIPT. It should be noted that the overall performance is based on multiple optimisation objectives, including safety and economy (refer to Eq. (39)), and the model's decisions during training are exploratory in nature (i.e., ϵ-greedy, refer to line 14 in Algorithm 1, Appendix A.4). Therefore, the training curve may not accurately reflect the model's performance in each specific indicator. To obtain the key performance indicators, the models will be tested using the test scenarios (refer to Section 5.4.2).

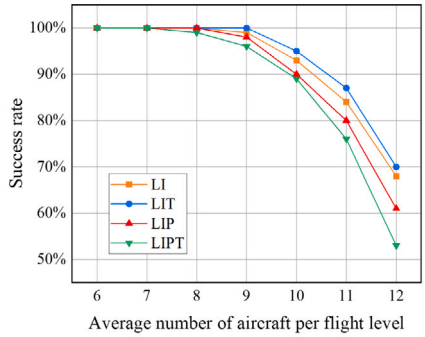
5.4.2. Performance test

The four trained models, LI, LIT, LIP and LIPT, were subjected to testing using the test dataset, and their key performance indicators were measured. The results of the performance test, including the success rate, extra flight distance rate, and number of flight level changes, are presented in Fig. 13. The success rate indicates the effectiveness of the model in CR, where a higher success rate corresponds to a more effective model. The extra flight distance rate measures the efficiency (economy) of the model in CR, with a lower rate indicating a more efficient (economical) model. The number of flight level changes represents the impact of the model's CR on the stability of the traffic system to some extent. In practice, frequent changes in flight level during the cruise phase are discouraged, and a model that produces fewer flight level changes is less likely to destabilise the traffic system. The performance test results were evaluated based on the traffic density, which is the average number of aircraft per flight level, to demonstrate (a) the trend of performance indicators with traffic density and (b) the differences in the performance of the four models in scenarios

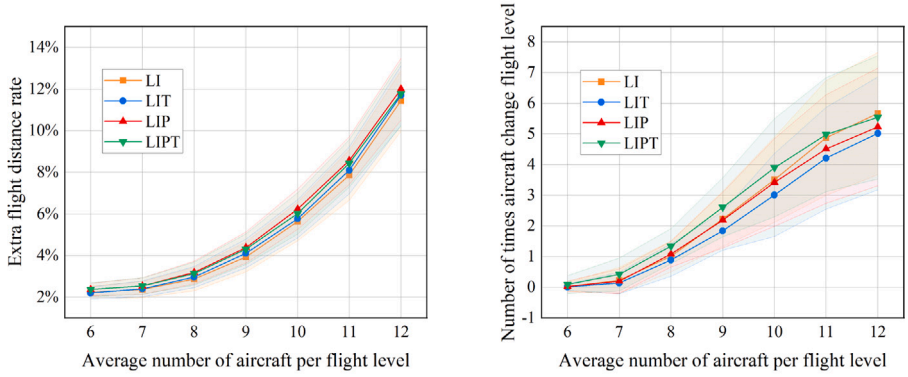


(a) training curves (b) training curves (focusing on comparison)

Fig. 12. Training results. The figure was generated using the Python toolkit for machine learning data visualisation, TensorboardX. The light curves represent the actual result data, while the dark curves are fitted based on the former, with a 95% smoothing coefficient to improve legibility. Sub-figure (b) is a zoomed-in version of sub-figure (a), focusing on the comparison of the four models. It shows the average reward in the range of −22 to −6.



(a) success rate



(b) extra flight distance rate (c) number of flight level changes

Fig. 13. Performance test results for four models with different modular observation matrix. The coloured area represents the standard deviation of the corresponding index.

with the same traffic density. It is worth noting that the calculation time for the proposed method is not reported in this article. However, since the agents operate in a distributed manner, the computation time for each decision of each agent is less than 0.01 s, which is deemed sufficient to meet the real-time operation requirements.

The success rate is calculated as the ratio of the number of conflict-free scenarios to the total number of scenarios (100 scenarios with the same traffic density) in which the agents perform CR. As shown in Fig. 13(a), the success rate decreased exponentially with increasing traffic density once the traffic density reached a certain value, which is considered to be the model's maximum capacity

for CR (i.e., the maximum traffic density in the experiment where the conflict resolution success rate is 100%). The maximum capacities for CR of models LI, LIT, LIP and LIPT are 8, 9, 8 and 7 aircraft per flight level, respectively, i.e., LIT > LI = LIP > LIPT. It should be noted that, in the experimental scenarios, the area of each flight level is 2500π square nautical miles (as mentioned in Section 5.2.1 and Appendix B.2), and it is projected that by the year 2025, in the real world, the busiest flight level with the same airspace area will have an estimated 4.80 aircraft (as mentioned in Section 5.3 and Appendix B.5).

The extra flight distance rate is calculated as the sum of all aircraft's total extra flight distance in a scenario, divided by the total minimum flight distance. The minimum flight distance is the straight-line distance from the aircraft's start point to the end point. The extra flight distance means the difference between the aircraft's actual and minimum flight distances. Note that the flight distances are only calculated for horizontal flight distances. Fig. 13(b) shows the average extra flight distance rate for the 100 scenarios with the same traffic density. The results indicate an exponential increase in the extra flight distance rate with traffic density. This trend may be due to the increasing need for aircraft to manoeuvre to avoid conflicts as traffic density rises. Generally, the extra flight distance rates for the four models follow this order: LI < LIT < LIPT < LIP.

The number of flight level changes refers to the total number of times that all aircraft in a given scenario change flight level. In Fig. 13(c), this number is the average of 100 scenarios with the same traffic density. As shown in Fig. 13(c), the number of flight level changes increased as traffic density increased, but the rate of increase initially accelerated and then decelerated. This pattern may arise because, as traffic density rises from a low level, aircraft must change flight level more often to explore a wider solution space; at high traffic densities, however, there may not be sufficient safe separation for aircraft to change flight level, given the large number of aircraft at each level. The number of flight level changes for the four models, from lowest to highest, is rough: LIT < LIP < LI < LIPT.

After conducting the performance test, it was found that model LIT had a better success rate and fewer flight level changes than other models, while model LI had the highest extra flight distance rate. Since success rate is generally considered the most crucial performance indicator in CR methods, this paper considers that model LIT has greater practical potential. This result is consistent with Brittain's study, which found that knowing intruders' flight intentions can improve CR success rate (Brittain and Wei, 2022). The under-performance of models LIP and LIPT may be due to the insignificant effect of aircraft performance differences on CR (the conflict buffer model can counteract the uncertainty associated with such differences), and the observation of aircraft performance increases the size of the neural network and thus the learning difficulty. In the following experiments, the model LIT will represent the method proposed in this study. Although the performance of the proposed method is obtained in the experimental scenarios, it can, to some extent, reflect its potential for achieving high performance in real-world scenarios. Additionally, since the proposed method exhibits satisfactory performance across all experimental scenarios within the model's maximum capacity, it can be considered a general approach applicable to any scenario defined in the experiments.

5.4.3. Illustrative example

To help readers better understand the characteristics of aircraft manoeuvres during CR using the proposed method (i.e., model LIT), we present a case study involving 60 aircraft. The scenario used is one of the test cases in the dataset (see Section 5.3) and covers an airspace area of 7856.3 square nautical miles. The trained model LIT was deployed and simulated in the scenario, using a base wind direction α^{BW} of 2.804 radians and a base wind speed v^{BW} of 13.48 metres per second.

Fig. 14 displays the trajectories of the scenario before and after CR using the proposed method from a global view. The trajectories are represented by six different coloured curves, which indicate their scheduled flight levels. Note that the colour of an aircraft's trajectory remains the same even if it changes flight level. A red mark on the trajectory indicates a flight conflict that occurred when the aircraft was in that position, while orange denotes that the aircraft was in the range of warning. *Schedule* and *Solution* depict the trajectories before and after CR, respectively. *Schedule* shows the trajectories of aircraft flying at their normal speeds. These trajectories do not consider uncertainty factors such as wind and flight uncertainties. On the other hand, the trajectories in *Solution* these uncertainties, which is why an aircraft's trajectory may not be straight even if it attempted to fly in a straight line. In contrast, all trajectories in *Schedule* are straight.

Fig. 15 displays the trajectories of the scenario before and after conflict resolution from each flight-level view. The same labels used in Fig. 14 are employed, and the flight segments of aircraft flying at their high and low speeds are marked with two specific colours, respectively. The points where agents made decisions are marked on the trajectory, denoting decision points. The length of the time step Δt was set to 30 s, which is the time interval between two adjacent decision points in the same trajectory. Due to the safety separation relaxation model for multiple flight levels introduced in the proposed method, if an aircraft changes flight level at a decision point, the trajectory for 30 s after this decision point is still plotted in the original flight level to indicate the occupation of this flight level. FLX denotes flight level X. For example, FL5 in Fig. 15 corresponds to the flight level where the flight level is 5 in Fig. 14. In *Solution*, if an aircraft flew at the normal speed, maintaining its scheduled flight level, it is plotted by the colour representing its scheduled flight level. For manoeuvres, the numbers of 'low speed', 'high speed', 'yaw left', 'yaw right', 'climbing' and 'descending' are 38, 41, 33, 85, 1 and 2, respectively.

In this scenario, the proposed method effectively resolved all potential conflicts and eliminated most warnings. While CR was the main objective of the proposed method, the elimination of warnings has both advantages and disadvantages. While it can provide an effective buffer to reduce the likelihood of conflict, it may also lead to unnecessary manoeuvres. In this case, some aircraft only received warnings without potential conflict, but they still manoeuvred to eliminate the warning. This finding is consistent with our previous research, which showed that the range of the warning area is positively correlated with extra flight distance. Warnings that were not eliminated may be due to uncertainty, which we have found to be negatively correlated with the success of CR. In practice, it is recommended to set appropriate warning ranges based on the intensity of uncertainty, which can reduce the extra flight distance while ensuring the success of CR. For more information about the study of the warning range and the intensity of uncertainty, we refer the reader to our previous research (Chen et al., 2023).

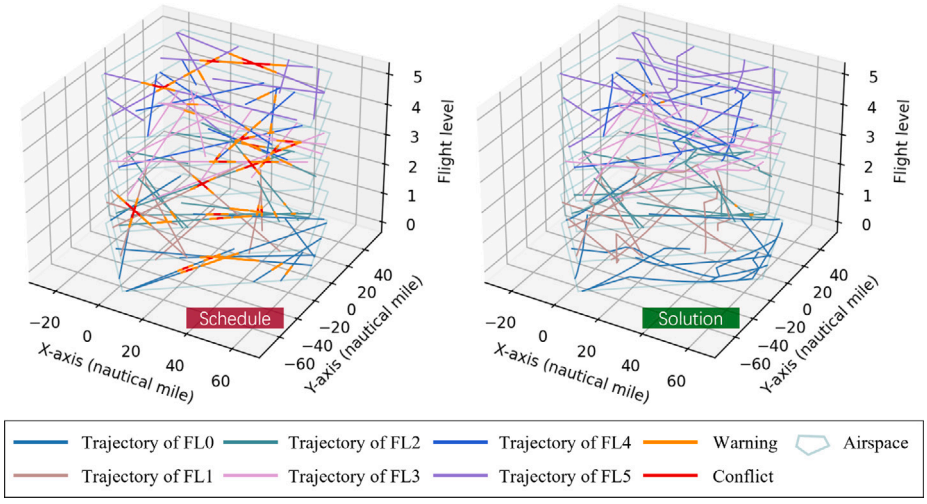


Fig. 14. Comparison of trajectories before and after conflict resolution (global view).

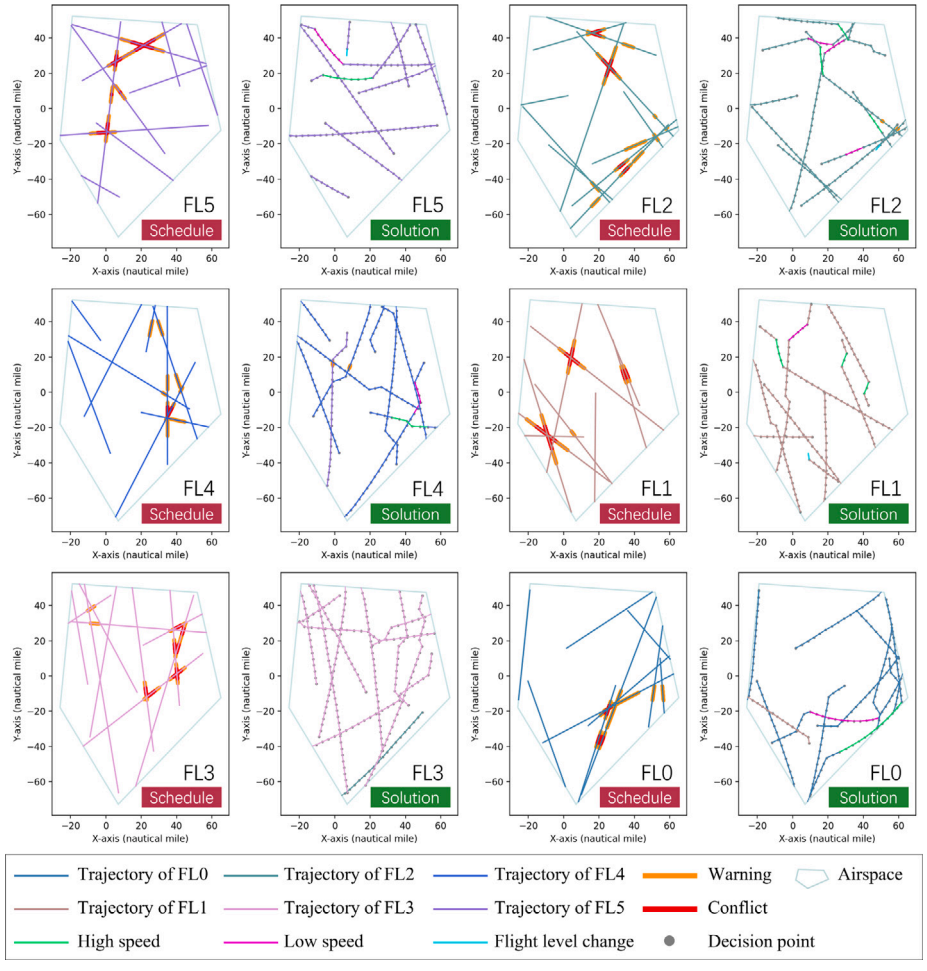
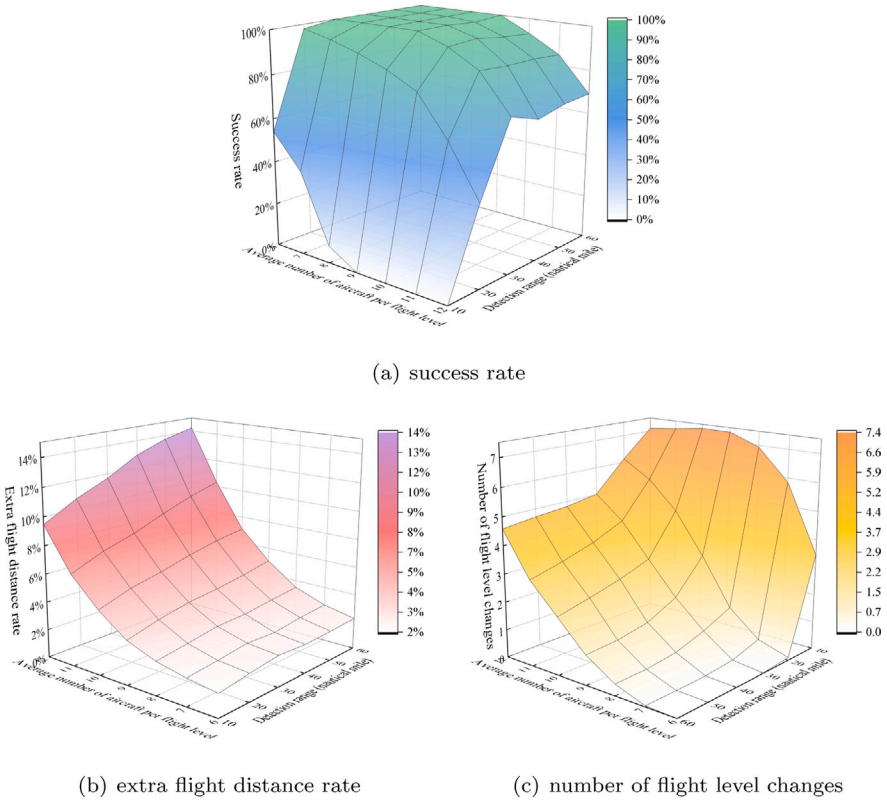


Fig. 15. Comparison of trajectories before and after conflict resolution (flight-level view). This figure is generated using the data from the same case as Fig. 14.



(a) success rate

(b) extra flight distance rate

(c) number of flight level changes

Fig. 16. Sensitivity analysis for detection range. Note that the orientation of the bottom axes (representing the average number of aircraft per flight level and/or detection range) in the three sub-figures may differ to facilitate the observation of trend changes.

5.5. Sensitivity analysis

The proposed model's detection range and the number of detection sectors are crucial parameters. A larger detection range enables the agent to observe more information, detecting intruders earlier, but may also increase the burden of analysing a larger dataset. The number of detection sectors affects the agent's ability to observe the intruder completely under the partial observation model based on imminent threats (refer to Section 4.3.1). However, it also increases the agent's learning burden. To better understand the impact of these parameters, we conducted a sensitivity analysis on the detection range and the number of detection sectors.

5.5.1. Detection range

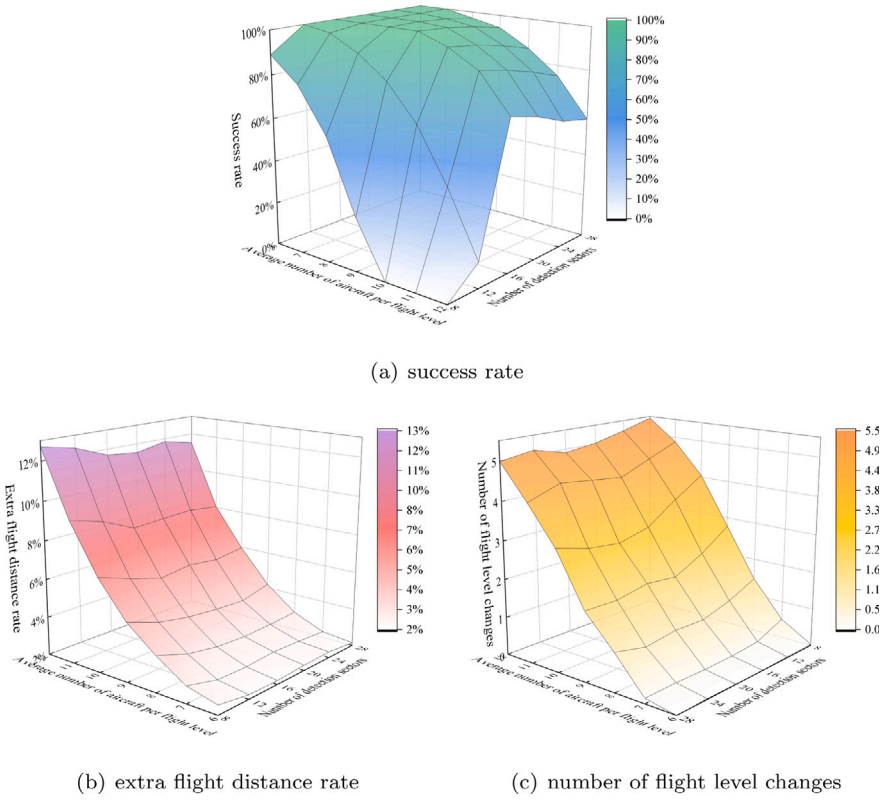
The proposed method's performance was evaluated under different detection ranges, which were set to 10, 20, 30, 40, 50, and 60 nautical miles, while the other parameters of the proposed model (model LIP) were kept constant. The six models were trained and tested using the method described in Section 5.4. The evaluation was based on the success rate, extra flight distance rate, and the number of flight level changes, which are defined in Section 5.4.2. The results of the performance test are presented in Fig. 16.

A general conclusion from this sensitivity analysis is that a detection range that allows the agent to detect the intruder in time for effective CR is sufficient. While a larger detection range can provide more information to the agent, it does not significantly improve the success rate of CR and may even slightly reduce it while also potentially resulting in extra flight distance. Readers are directed to Appendix C.1 for more discussions on the sensitivity analysis for the detection range.

5.5.2. Number of detection sectors

The proposed model (model LIP) was tested with varying numbers of detection sectors. The numbers of detection sectors tested were 8, 12, 16, 20, 24 and 28, while the other parameters of the model remained constant. The six models were trained and tested using the same method as introduced in Section 5.4. The performance test results in success rate, extra flight distance rate and the number of flight level changes are shown in Fig. 17, with the definition of each indicator being the same as in Section 5.4.2.

A general conclusion from the this sensitivity analysis is that having too few detection sectors may result in a significant reduction in the success rate of CR due to unobserved aircraft, accompanied by an increase in the extra distance flown and an increase in the number of changes in altitude. An excessive number of detection sectors may make it more difficult for the agent to learn due to a



(a) success rate

(b) extra flight distance rate

(c) number of flight level changes

Fig. 17. Sensitivity analysis for number of detection sectors. Note that the orientation of the bottom axes (representing the average number of aircraft per flight level and/or detection range) in the three sub-figures may differ to facilitate the observation of trend changes.

large number of neurons, potentially reducing the success rate of CR. Therefore, it is recommended to set the number of detection sectors to an appropriate value. Readers are directed to Appendix C.2 for more discussions on the sensitivity analysis for the number of detection sectors.

5.6. Comparison of 2D and 3D models

Most current CR models neglect vertical manoeuvres due to the potential operational instability they may cause in airspace traffic (Ribeiro et al., 2020b). However, the performance difference between 2D and 3D models for MARL-based CR methods is not yet fully understood. To investigate this, we modified the proposed 3D model to a 2D model and conducted comparative experiments (refer to Appendix C.3 for more details on the 2D model).

The parameters of the 2D model were set using the same approach as that of the 3D model (when applicable). The 2D model was trained and tested using the same method outlined in Section 5.4 for comparison with the 3D model (Model LIT). It should be noted that the results of the experiments in this section can also be used as a reference for comparing the proposed method with state-of-the-art conflict RL-based resolution methods for the following reasons:

1. Our developed 2D model has achieved competitive results in terms of success rates and extra flight distance than state-of-the-art RL-based CR models in previous studies (Chen et al., 2023);
2. Neural network parameters for other state-of-the-art RL-based CR models are not publicly available;
3. The current research on RL methods in CR lacks a baseline (Wang et al., 2022);
4. 3D RL-based CR methods (especially those that consider flight uncertainty and wind) are rarely reported.

It should be noted that, in the comparison experiments between our 2D model and state-of-the-art RL-based CR models, we tested our 2D models in publicly available scenarios and compared our results with the publicly available results of other researchers. However, this comparison approach would not be applicable to this study as the scenario requires 3D modelling.

5.6.1. Training efficiency

Fig. 18 displays the training curve of the 2D model. The 3D model needed approximately one million episodes to converge (refer to Fig. 12), whereas the 2D model converged in ten thousand episodes (1% of the 3D model). The training time for the 3D model

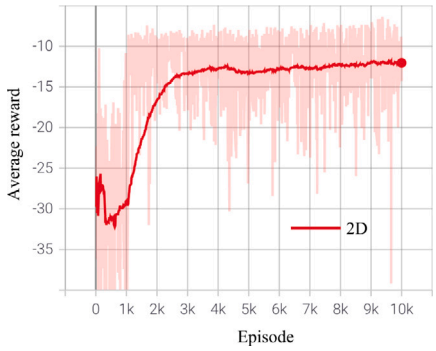


Fig. 18. Training results of the 2D model. The figure was generated using the Python toolkit for machine learning data visualisation, TensorboardX. The light curves represent the actual result data, while the dark curves are fitted based on the former, with a 95% smoothing coefficient to improve legibility.

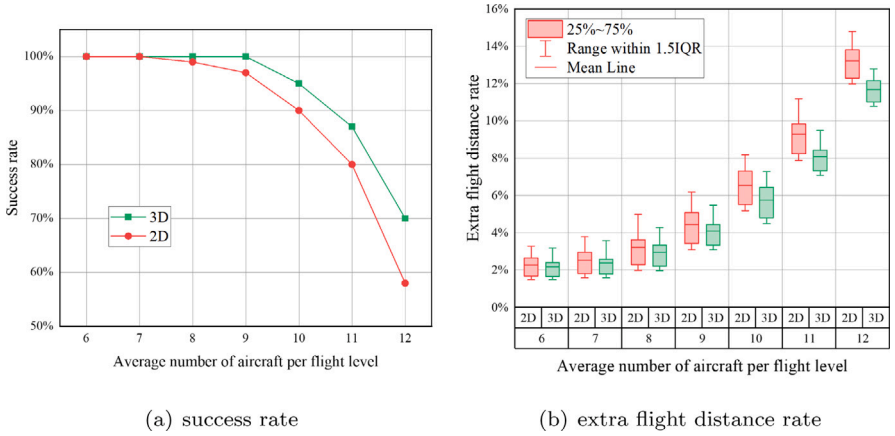


Fig. 19. Performance test results for 2D and 3D models.

was around 354.5 h, while the 2D model's training time was about 2.7 h (less than 1% of the 3D model). Since the 2D model only observes the current flight level of the aircraft, its neural network has approximately one-fifth the number of neurons of the 3D model (refer to Eq. (33), Equation C.2 and Fig. 9), further reducing the training time of the 2D model. Although training time is not the most critical parameter for an RL method, we still disclose this parameter for the reader's information. Although training time is not the most important parameter for an RL method, we still disclose this parameter for the reader's information. We consider that if we enhance the model further by adding more elements to approximate reality, the gap between the training time of the 2D and 3D models will increase.

5.6.2. Optimisation performance

Fig. 19 shows the performance test results in success rate and extra flight distance rate. The definition of each indicator is the same as in Section 5.4.2.

In terms of the success rate of CR, the 2D and 3D models had the same trends with traffic density, and the 3D model outperformed the 2D model (as shown in Fig. 19(a)). The maximum capacity for CR of the 2D model is seven aircraft per flight level, while that of the 3D model is nine. For most CR methods, 3D models have a larger solution space than 2D models (if applicable). This result may indicate that, for the proposed RL-based CR method, the 3D model can effectively use its larger solution space to obtain greater CR capacity than the 2D model.

In terms of the extra flight distance rate, the 2D and 3D models had the same trends with traffic density, and the 3D model also outperformed the 2D model (as shown in Fig. 19(b)). The difference between the two models in terms of extra flight distance rate increased with increasing traffic density. In some scenarios, the 3D model can resolve conflicts more easily by changing the flight level of the aircraft, whereas the 2D model may need to resolve conflicts by making the aircraft perform a wider range of horizontal manoeuvres, which usually results in more extra flying distance.

6. Discussions

In this section, we discuss the lessons from this study, including the trade-offs between information and learning difficulty and the practicality of 2D and 3D models. Then, we point out the limitations of this study and suggest directions for future improvement.

Table 4
Effect of inappropriate parameter settings on model performance.

Parameters	Variation	Information	Number of neurons	The effect of excess on performance			Major difficulties
				Success rate	Extra flight distance rate	Number of flight changes	
Number of modules of observation	+	+	+	−	≈	+	Learning
	−	−	−	−	≈	+	Predicting intruders
Detection range	+	+	=	−	+	−	Unnecessary information
	−	−	=	−	−	+	Detecting conflicts
Number of detection sectors	+	+	+	−	≈	≈	Learning
	−	−	−	−	+	+	Detecting conflicts

Symbols: increase (+), decrease (−), unchanged (=), fluctuating (≈).

6.1. Trade-offs: information and learning difficulty

In algorithms, trade-offs often exist between various factors, such as between solution accuracy and computation time. For RL methods, the trade-off of concern in this study is between the amount of information and the learning difficulty. While more information usually means higher solution accuracy for optimisation problems, in RL-based methods, it could lead to larger neural networks, potentially increasing the learning difficulty for agents. Our study of the proposed method identified similar situations, and we concluded that there is a trade-off between information and learning difficulty.

The amount of information obtained by the agent in the proposed method mainly depends on three parameters: the number of modules of observation (i.e., sub-observation matrix $\mathbf{o}_t^{k,s}$), the detection range and the number of detection sectors as discussed in Section 5. We found that appropriate values for these parameters maximise the proposed model's performance, and their inappropriate settings could have a significant impact on the model's performance. Table 4 shows the effect of inappropriate parameter settings on model performance, based on the experimental results in Sections 5.4.2 and 5.5.

Increasing these parameters enables the agent to obtain more information, but it also increases the number of neurons and may lead to greater learning difficulty and reduced model performance if they are oversized. While increasing the detection range does not change the number of neurons, its success rate still decreases when it exceeds a certain value. Therefore, too much unnecessary information and too large a neural network may increase the agent's learning difficulty, ultimately leading to reduced model performance.

Conversely, if these parameters are too small, the agent may not obtain enough information for effective and efficient CR. Reducing the number of observation modules may make it difficult for agents to predict intruders' actions, leading to reduced model performance. Similarly, reducing the detection range and the number of detection sectors may result in the agent not detecting potential conflicts in time and consequently lead to reduced model performance.

Therefore, we recommend a careful trade-off between information and learning difficulty when studying reinforcement-based learning approaches to CR. It is crucial to avoid the agent not obtaining enough information for effective and efficient CR while also avoiding unnecessary information and too large neural networks that may increase the learning difficulty and reduce model performance.

6.2. Practicality: 2D or 3D models?

The test results have demonstrated that the 3D model outperforms the 2D model in terms of the effectiveness (success rate) and efficiency (extra flight distance rate) of CR. Despite the fact that the training time for the 3D model is more than 100 times that of the 2D model, it does not affect the model's performance in actual CR due to the nature of RL methods. Thus, theoretically, for our RL-based CR methods, 3D models have greater potential for development than 2D models to cope with more complex scenarios in the future.

However, we also believe that the 2D model is worth further exploration due to its advantages. Although the 2D model is not as effective at resolving conflicts as the 3D model, it can handle scenarios with traffic densities that far exceed those of the real world today. This study has found that overly large neural networks may be detrimental to the agent's learning of CR due to many parameters. Therefore, smaller neural networks for 2D models offer greater potential for further model expansion to consider more real-world factors.

The observation that 3D models generally have better CR performance than 2D models due to more solution space is not limited to models based on RL but also those based on other technologies (e.g., geometric algorithms, optimal control theories, and heuristic algorithms). For 3D models, we need to investigate further the effect of vertical manoeuvres on traffic stability and the airspace user's flight altitude preference to more fully assess their practicality (which is beyond the scope of this study).

6.3. Limitations

To make the simulation environment as close to the real world as possible, this study has considered performance differences between aircraft, aircraft performance constraints, flight uncertainties, and simplified wind fields. However, despite all these efforts, the simulator still has limitations compared to the real world. Therefore, the conclusions of this paper are only valid for the simulation environment described. In future work, further consideration could be given to the following:

1. dynamic wind fields based on realistic meteorological data,
2. aerodynamically based kinematic models, and
3. more realistic communication, navigation and surveillance environments.

As an early study exploring the use of RL methods for 3D CR, this study focuses on verifying the feasibility of the proposed method. The key performance indicators in the experiments of this study focus on the direct results of the optimisation (e.g., the success rate for effectiveness, the extra flight distance rate for efficiency, and the number of flight level changes for the stability of the traffic system). Some of the key performance indicators that provide a more intuitive picture of the operational characteristics and mechanisms of the proposed method are to be further explored in future research, such as:

1. the smoothness of flight trajectory (which is related to fuel economy and passenger comfort),
2. consistency of manoeuvre strategy (which is used to show whether the action of agents is consistent in similar scenarios and can be used to guide the study of geometry-based CR methods), and
3. the Domino Effect of manoeuvre (i.e., whether CR manoeuvres will result in more or worse secondary conflict, which can more accurately reflect the impact of the proposed method on the stability of the air traffic system).

7. Conclusions

This paper presents a general MARL method for real-time 3DMACR. We evaluated the performance of our method through a series of experiments and made the following key findings:

1. The proposed model can be applied to arbitrary real-time 3DMACR problems. In the experimental scenario containing uncertainty models and ideal assumptions constructed in this study, our method can handle a higher traffic density than is currently seen in real-world scenarios.
2. It is important to strike a reasonable balance between the amount of information the agent observes and the learning difficulty. The agent should acquire enough information for effective conflict detection and resolution while avoiding over-acquisition of information that increases unnecessary data and neurons, thus making learning more difficult.
3. The proposed 3D model outperforms the 2D model in CR success rate and extra flight distance rate. Although the training time required by the 3D model is more than one hundred times that of the 2D model, the training process is separate from the solving process, which hardly reduces the practical application potential of the 3D model.

This study serves as the first step towards reducing the gap between the theory and practice of MARL methods used for real-time 3DMACR. We recognise that practical scenarios are far more complex than our experiments. Thus, we will continue to enhance the simulation environment to make it closer to the real world and evaluate the proposed model's performance more comprehensively (i.e., operational and communication delays). Systematic exploration of other high-performance reinforcement learning algorithms compatible with the proposed model is expected to improve the performance of the proposed method further. We will also attempt to integrate aerodynamics-based motion models into the RL method or make the planned trajectory compatible with aerodynamics-based control methods. In the future, we hope to extend our method to terminal area airspace for continuous climb/descent operations (CCO/CDO) or urban airspace for unmanned traffic management (UTM).

CRediT authorship contribution statement

Yutong Chen: Methodology, Validation, Writing – original draft. **Yan Xu:** Methodology, Supervision, Writing – review & editing. **Lei Yang:** Supervision, Writing – review & editing. **Minghua Hu:** Supervision, Writing – review & editing.

Acknowledgements

We would like to express our gratitude to EUROCONTROL for supporting this study and providing the simulator prototype. The proposed method's prototype was ranked second-best in the 2022 EUROCONTROL Innovation Master Class Competition. The training method used was inspired by a SESAR exploratory research project we participated in: Artificial Intelligence Solutions to METEO-Based DCB Imbalances for Network Operations Planning (ISOBAR). The opinions expressed in this paper reflect the authors' views only. Under no circumstances shall the SESAR Joint Undertaking or EUROCONTROL be responsible for any use that may be made of the information contained in this paper. We also would like to thank the National Key R&D Program of China (No. 2022YFB4300905) and the National Natural Science Foundation of China (No. 61903187).

Appendix A. Supplementary explanations, discussions, and nomenclature

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.trc.2023.104367.

References

Alonso-Ayuso, A., Escudero, L.F., Martín-Campo, F.J., 2016. Multiobjective optimization for aircraft conflict resolution. A metaheuristic approach. European J. Oper. Res. 248 (2), 691–702.

Alonso-Ayuso, A., Escudero, L.F., Martin-Campo, F.J., Mladenović, N., 2017. On the aircraft conflict resolution problem: A VNS approach in a multiobjective framework. Electron. Notes Discrete Math. 58, 151–158.

Brittain, M., Wei, P., 2022. Scalable autonomous separation assurance with heterogeneous multi-agent reinforcement learning. IEEE Trans. Autom. Sci. Eng. 19 (4), 2837–2848.

Brittain, M.W., Yang, X., Wei, P., 2021. Autonomous separation assurance with deep multi-agent reinforcement learning. J. Aerosp. Inf. Syst. 18 (12), 890–905.

Chaloulos, G., Hokayem, P., Lygeros, J., 2010. Distributed hierarchical MPC for conflict resolution in air traffic control. In: Proceedings of the 2010 American Control Conference. IEEE, pp. 3945–3950.

Chen, Y., Hu, M., Yang, L., 2022. Autonomous planning of optimal four-dimensional trajectory for real-time en-route airspace operation with solution space visualisation. Transp. Res. C 140, 103701.

Chen, Y., Hu, M., Yang, L., Xu, Y., Xie, H., 2023. General multi-agent reinforcement learning integrating adaptive manoeuvre strategy for real-time multi-aircraft conflict resolution. Transp. Res. C 151, 104125. http://dx.doi.org/10.1016/j.trc.2023.104125, URL https://www.sciencedirect.com/science/article/pii/S0968090X23001146.

Cobano, J.A., Alejo, D., Heredia, G., Ollero, A., 2013. 4D trajectory planning in ATM with an anytime stochastic approach. In: Proceedings of the 3rd International Conference on Application and Theory of Automation in Command and Control Systems. pp. 1–8.

Dalmau, R., Allard, E., 2020. Air Traffic Control using message passing neural networks and multi-agent reinforcement learning. In: Proceedings of the 10th SESAR Innovation Days. pp. 1–8.

Djokic, J., Lorenz, B., Fricke, H., 2010. Air traffic control complexity as workload driver. Transp. Res. C 18 (6), 930–936.

EUROCONTROL, 2018. Performance Review Report An Assessment of Air Traffic Management in Europe during the Calendar Year 2018. Tech. rep., EUROCONTROL, Brussels, Belgium.

Fan, T., Long, P., Liu, W., Pan, J., 2020. Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios. Int. J. Robot. Res. 39 (7), 856–892.

Guo, W., Brittain, M., Wei, P., 2021. Safety enhancement for deep reinforcement learning in autonomous separation assurance. arXiv preprint arXiv:2105.02331.

Han, Y., Zhang, J., Huang, X., 2019. Method for optimal conflict-free aircraft trajectory generation. Aircr. Eng. Aerosp. Technol. 92 (2), 156–162.

Hernandez-Garcia, J.F., Sutton, R.S., 2019. Understanding multi-step deep reinforcement learning: a systematic study of the DQN target. arXiv preprint arXiv:1901.07510.

Hessel, M., Modayil, J., Van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M., Silver, D., 2018. Rainbow: Combining improvements in deep reinforcement learning. In: Thirty-Second AAAI Conference on Artificial Intelligence. pp. 1–8.

ICAO, 2005. Global Air Traffic Management Operational Concept. Montréal, https://www.icao.int/Meetings/anconf12/Document%20Archive/9854_cons_en[1].pdf.

ICAO, 2016. Procedures for Navigation Services - Air Traffic Management (16th Edition of DOC 4444). Tech. rep., ICAO.

ICAO, Post-COVID-19 Forecasts Scenarios, https://www.icao.int/sustainability/Pages/Post-Covid-Forecasts-Scenarios.aspx.

Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Lai, J., Cai, K., Liu, Z., Yang, Y., 2021. A multi-agent reinforcement learning approach for conflict resolution in dense traffic scenarios. In: 2021 IEEE/AIAA 40th Digital Avionics Systems Conference (DASC). IEEE, pp. 1–9.

Malaek, S.M.-B., Golchoubian, M., 2020. Enhanced conflict resolution maneuvers for dense airspaces. IEEE Trans. Aerosp. Electron. Syst. 56 (5), 3409–3420.

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., 2015. Human-level control through deep reinforcement learning. nature 518 (7540), 529–533.

Papoudakis, G., Christianos, F., Rahman, A., Albrecht, S.V., 2019. Dealing with non-stationarity in multi-agent deep reinforcement learning. arXiv preprint arXiv:1906.04737.

Pham, D.-T., Tran, P.N., Alam, S., Duong, V., Delahaye, D., 2022. Deep reinforcement learning based path stretch vector resolution in dense traffic with uncertainties. Transp. Res. C 135, 103463.

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. In: 2019 IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF). IEEE, pp. 1–6.

Pritchett, A.R., Genton, A., 2017. Negotiated decentralized aircraft conflict resolution. IEEE Trans. Intell. Transp. Syst. 19 (1), 81–91.

Rezo, Z., Steiner, S., 2020. South east common sky initiative free route airspace-implementation aftermath. Transp. Res. Procedia 45, 676–683.

Ribeiro, M., Ellerbroek, J., Hoekstra, J., 2020a. Improvement of conflict detection and resolution at high densities through reinforcement learning. In: ICRAT 2020: International Conference on Research in Air Transportation. pp. 1–4.

Ribeiro, M., Ellerbroek, J., Hoekstra, J., 2020b. Review of conflict resolution methods for manned and unmanned aviation. Aerospace 7 (6), 79.

Schaul, T., Quan, J., Antonoglou, I., Silver, D., 2015. Prioritized experience replay. arXiv preprint arXiv:1511.05952.

Šišlák, D., Volf, P., Pěchouček, M., 2010. Agent-based cooperative decentralized airplane-collision avoidance. IEEE Trans. Intell. Transp. Syst. 12 (1), 36–46.

Soler, M., Kamgarpour, M., Lloret, J., Lygeros, J., 2016. A hybrid optimal control approach to fuel-efficient aircraft conflict avoidance. IEEE Trans. Intell. Transp. Syst. 17 (7), 1826–1838.

Sui, D., Xu, W., Zhang, K., 2022. Study on the resolution of multi-aircraft flight conflicts based on an IDQN. Chin. J. Aeronaut. 35 (2), 195–213.

Van Hasselt, H., Guez, A., Silver, D., 2016. Deep reinforcement learning with double q-learning. In: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 30. pp. 1–7.

Wang, R., Alligier, R., Allignol, C., Barnier, N., Durand, N., Gondran, A., 2020. Cooperation of combinatorial solvers for en-route conflict resolution. Transp. Res. C 114, 36–58.

Wang, Z., Li, H., Wang, J., Shen, F., 2019. Deep reinforcement learning based conflict detection and resolution in air traffic control. IET Intell. Transp. Syst. 13 (6), 1041–1047.

Wang, Z., Pan, W., Li, H., Wang, X., Zuo, Q., 2022. Review of deep reinforcement learning approaches for conflict resolution in air traffic control. Aerospace 9 (6), 294.

Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M., Freitas, N., 2016. Dueling network architectures for deep reinforcement learning. In: International Conference on Machine Learning. PMLR, pp. 1995–2003.

Yilmaz, E., Sanni, O., Kotwicz Herniczek, M., German, B., 2021. Deep reinforcement learning approach to air traffic optimization using the MuZero algorithm. In: AIAA AVIATION 2021 Forum. p. 2377.

Yokoyama, N., 2018. Decentralized conflict detection and resolution using intent-based probabilistic trajectory prediction. In: 2018 AIAA Guidance, Navigation, and Control Conference. p. 1857.

Zhao, P., Liu, Y., 2021. Physics informed deep reinforcement learning for aircraft conflict resolution. IEEE Trans. Intell. Transp. Syst. 23 (7), 8288–8301.



https://dspace.org/

General real-time three-dimensional multi-aircraft conflict resolution method using multi-agent reinforcement learning

Chen, Yutong; Xu, Yan; Yang, Lei; Hu, Minghua

2023-12-01
Elsevier
https://doi.org/10.1016/j.trc.2023.104367
Chen Y, Xu Y, Yang L, Hu M. (2023) General real-time three-dimensional multi-aircraft conflict resolution method using multi-agent reinforcement learning. Transportation Research Part C: Emerging Technologies, Volume 157, December 2023, Article number 104367

Downloaded from a DSpace repository. https://dspace.org/