



HAL
open science

Machine learning-based flight trajectories prediction and air traffic conflict resolution advisory

Duc-Think Pham

► **To cite this version:**

Duc-Think Pham. Machine learning-based flight trajectories prediction and air traffic conflict resolution advisory. Machine Learning [cs.LG]. Université Paris sciences et lettres, 2019. English. NNT : 2019PSLEP027 . tel-02870575

HAL Id: tel-02870575

<https://theses.hal.science/tel-02870575v1>

Submitted on 16 Jun 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE DE DOCTORAT

de l'Université de recherche Paris Sciences et Lettres
PSL Research University

Préparée à l'École Pratique des Hautes Études

Prédiction de Trajectoire et Avis de Résolution de Conflits de Trafic
Aérien basée sur l'Apprentissage Automatique

École doctorale de l'EPHE - ED 472

Spécialité INFORMATIQUE, MATHÉMATIQUE ET APPLICATIONS

Salle Gaston Paris D064-EPHE
Sorbonne, 17 rue de la Sorbonne,
75005 Paris

Soutenue par **Duc-Thinh PHAM**
le 21/06/2019

Dirigée par
Marc BUI
Vu DUONG



École Pratique
des Hautes Études



COMPOSITION DU JURY :

M. Charles Tijus
Université Paris 8
Président du jury

M. Sameer Alam
Nanyang Technological University, Singapore
Rapporteur

M. Soufian Ben Amor
Université de Versailles
Saint-Quentin-en-Yvelines
Rapporteur

M. Marc Bui
École Pratique des Hautes Études
Directeur de thèse

M. Vu Duong
Nanyang Technological University, Singapore
Co-Directeur de thèse

Machine Learning -based Flight Trajectories Prediction and Air Traffic Conflict Resolution Advisory

Duc-Thinh, PHAM

Supervisors: Prof. Marc BUI

Prof. Vu DUONG

École Pratique des Hautes Études
Paris Sciences & Lettres – PSL Research University Paris

This dissertation is submitted for the degree of
Doctor of Philosophy
of
Paris Sciences & Lettres – PSL Research University Paris

June 2019

Acknowledgements

First of all, I sincerely thank my supervisors, Professor Marc Bui and Professor Vu Duong, for the continuous support of my Ph.D study and research, for their patience, motivation, enthusiasm, and immense knowledge. They have given me the opportunities and conditions to follow and finish this PhD with their tremendous kindness.

Lastly, I would like to thank my family. Many thanks to my wife who has supported me all the time and stayed with me whenever I felt happy, sad, pleasant, and frustrated. I could not have finished this long journey without you. This PhD thesis is dedicated to my parents who raised me with unconditional love.

Abstract

The increasing in traffic demand has strained air traffic control system and controllers which lead to the need of novel and efficient conflict detection and resolution advisory. In the scope of this thesis, we concentrate on studying challenges in conflict detection and resolution by using machine learning approaches. We have attempted to learn and predict controller behaviors from data using Random Forest. We also propose a novel approach for probabilistic conflict detection by using Heteroscedastic Gaussian Process as predictive models and Bayesian Optimization for probabilistic conflict detection algorithm. Finally, we propose an artificial intelligent agent that is capable of resolving conflicts, in the presence of traffic and uncertainty. The conflict resolution task is formulated as a decision-making problem in large and complex action space, which is applicable for employing reinforcement learning algorithm. Our work includes the development of a learning environment, scenario state representation, reward function, and learning algorithm. Machine learning methods have showed their advantages and potential in conflict detection and resolution related challenges. However, more studies would be conducted to improve their performances such as airspace network representation, multi-agent reinforcement learning or controller's strategy reconstruction from data.

Table of contents

List of figures	ix
List of tables	xi
1 Introduction	1
1.1 Motivation	1
1.2 Related Works	4
1.2.1 Controller Strategy in Traffic Management	4
1.2.2 Data-driven Approach for Probabilistic Conflict Detection	5
1.2.3 Machine Learning Approach for Conflict Resolution	7
1.3 Contribution of this thesis	8
1.4 Organization of the thesis	9
2 Learning ATCo Actions from ADS-B Data	11
2.1 Introduction	12
2.2 Approach	12
2.3 Trajectory Data	13
2.3.1 Data-set	13
2.3.2 Selected Sector	15
2.3.3 Data Cleaning and Filtering	15
2.4 Extraction of ATCO Action	16
2.4.1 Extracting Flight Change/ Action Values	16
2.4.2 Encoding ATCO Actions from Action Values	18
2.5 Predictive Models	20
2.5.1 Random Forest Method	20
2.5.2 Building Predictive Models	22
2.6 Experiments and Results	23
2.7 Discussion	25
2.7.1 Extracted Actions as Macro Actions	25
2.7.2 Grouping Trajectory to form Flight Routes	26
2.8 Conclusions	27

3	Data-driven approaches for Trajectory Prediction and Conflict Detection	29
3.1	Introduction	30
3.2	Approach	31
3.3	Predictive model for flight trajectory with learned uncertainty	32
3.3.1	Input independent noise - homoscedastic noise	32
3.3.2	Input dependent noise - heteroscedastic noise	33
3.3.3	Predictive model for individual airplane	33
3.4	Probabilistic Conflict Detection using Bayesian Optimization	34
3.4.1	Bayesian Optimization	34
3.4.2	Probabilistic Closest Point of Approach - PCPA	35
3.5	Experiment Setup	37
3.5.1	Trajectory data for learning and evaluation	37
3.5.2	Exp 1:Evaluating GP-Generative model	38
3.5.3	Exp 2:Bayesian optimization vs Monte Carlo	39
3.6	Result and Discussion	39
3.6.1	Result for Exp 1:	39
3.6.2	Result for Exp 2:	42
3.7	Conclusion	44
4	Artificial Intelligent Agent for Conflict Resolution	45
4.1	Deep Reinforcement Learning in Conflict Resolution	45
4.2	Learning Environment	48
4.2.1	Conflict scenarios	48
4.2.2	Ownship's Maneuver	50
4.2.3	Environmental uncertainty	50
4.2.4	Scenario representation	51
4.2.5	Maneuver evaluation	52
4.3	AI Agent and Learning Mechanism	53
4.3.1	Agent's Action for Deep Reinforcement Learning	53
4.3.2	Deep Deterministic Policy Gradient (DDPG)	55
4.3.3	Learning Mechanism	56
4.4	Experiment Setup	59
4.5	Results and Discussion	61
4.6	Case Study: AI Agent with Human References	65
4.7	Conclusion	68
5	Conclusion and Future Work	71
5.1	Conclusion	71
5.2	Future research directions	73
	References	75

List of figures

1.1	Information Flow between D-Side and R-Side air traffic controllers for traffic flow planning and separation assurance in a sector.	3
2.1	The Learning Process using Random Forrest.	13
2.2	Demonstration of Sector 2E	15
2.3	Distribution of Entry and Exit points in Sector 2E in term of number of flights.	16
2.4	Distribution of Vertical Actions: Climb, Maintain Level and Descend in Sector 2E in term of number of flights.	17
2.5	Visualization for trends in ground speed and Altitude from data. Each line in the figure belongs to a different flight in data.	18
2.6	Definitions of ATCo Actions from Entry and Exit points.	18
2.7	Flow chart of Encoding Actions	20
2.8	(a), (c), (e) are distributions of computed trajectories' changes (actions) from ADS-B data. (b), (d) and (f) are distributions of encoded actions.	21
2.9	Illustration for parameter tuning process for regression models. Estimators are [50:300, step 50] and Max_Depth are [4:20, step 1], the experiment index is generated in the same order (ascending)	23
2.10	Example for Flight Direction: Black dots are entry points and Red dots are exit points.	24
2.11	Illustration for parameter tuning process for classification models. Estimators are [50:300, step 50] and Max_Depth are [4:20, step 1], the experiment index is generated in the same order (ascending)	25
2.12	Clustering deviations of heading changes overtime as lateral actions	26
2.13	Clustering vertical profile overtime as vertical actions	26
2.14	Examples of grouping trajectories based on entry and exit points	27
3.1	Illustrating approach for probabilistic conflict detection using Bayesian Optimization.	31
3.2	Example of Learning ADS-B Trajectory's Variances	39
3.3	Learning ADS-B Trajectory Variation	40
3.4	Comparing Kullback-Leibler divergences of heteroscedastic and homoscedastic models	41

3.5	Prediction results from Generative Model with ADS-B data	42
3.6	Prediction results from Generative Model for Toy sample	42
4.1	Interaction between the learning environment and the AI agent	47
4.2	a) Example of a conflict scenario involving a two-aircraft conflict in the presence of four surrounding aircraft. $\mathbf{A}_0\mathbf{B}_0$ is the ownship and $\mathbf{A}_1\mathbf{B}_1$ is the intruder. b) The conflict occurs where \mathbf{PQ} is the closest distance between two aircraft and smaller than minimum separation	49
4.3	a) An example of maneuver. The ownship makes a heading change α° at point \mathbf{M} at $t = t_1$, continues in the new heading \mathbf{MN} during t_2 seconds, and heading back towards original end point at return point \mathbf{N} . A maneuver is fully defined by a set of three parameters (t_1, α, t_2) . b) The maneuver implemented in the traffic scenario.	50
4.4	Environmental uncertainty and its impact on the agent's action	51
4.5	Model for learning conflict resolution.	55
4.6	Examples of a searching episode to suggest resolution	56
4.7	Generated scenarios are classified into 24 groups, based on their time to CPA (t_{CPA} , across the rows) and conflict angle (ϕ , across the columns).	60
4.8	Convergence of learning model at different configurations of uncertainty level (σ) and number of aircraft (n)	61
4.9	Effects of the uncertainty on the performance stability after convergence.a) Variances of achieved scores. b) Variances being normalized with variances at no uncertainty.	62
4.10	Average reward and successful rate achieved by the agent after convergence.	63
4.11	Examples of predicted resolutions in scenarios with different number of flights	63
4.12	The approximation of Q_value to real Reward when Number of Flights (NoF) from 2 to 14 and Uncertainty (U) from 0% to 5%	64
4.13	Convergence of the learning model	66
4.14	An example of the agent's suggested resolution showing the similarity between agent's and human resolutions	66
4.15	Penalties distribution performed on test set after convergence	67
4.16	The agent's suggested resolutions for different conflict scenarios	68

List of tables

2.1	4D Trajectory Data Features	14
2.2	One row sample of 4D Trajectory Data	14
2.3	Flight information (features) at Entry point and actions, actions' values . . .	22
2.4	Experiment result for predicting 3 group of actions	24
2.5	Predictive accuracy for different models	24
2.6	Feature Importance in 3-Actions Random Forest Model	25
3.1	List of Way-points	38
3.2	List of sample routes	38
3.3	Performances of BO and MC for detecting potential conflicts	43
4.1	Parameters for training the AI agent	60

Summary

ICAO's latest long-term air traffic forecasts predict a 250% increase in the number of air passengers and departures by 2040. Although there are 42,000 airports in the world, traffic is not evenly distributed and demand is concentrated on a small number of them. For example, the top 30 airports are required to serve about one-third of all passengers, while the busiest airport can serve more than 880,000 air operations and 104,000 passengers annually. As a result, the air transport system with limited capacity has already reached its limits. The increase in traffic demand has put the stress on this system and has led to significant congestion, flight delays or pollution. The consequence can be observed in air traffic control (ATC) which is in charge of the safety and efficiency of flights. The major resources of ATC such as airspace or controllers are limited. Thus, to deal with new traffic demand, it requires efficient advisory tools to increase controllers' productivity and reduce their workload. Traditional researches in ATC mainly focus on proposing advisory tool using multi-objective optimization problems that optimize objective values while maintaining the safety and security of the system. These objectives are difficult to achieve in practice due to the challenges posed by the presence of uncertainties, human factors or incomplete information. One potential solution is to take advantage of the increasingly available operational data for learning and training new models, also called data-driven models. During the past decade, more and more operational data is available for researchers and new breakthroughs in machine learning and statistical learning models also allow simpler and more realistic models.

In ATC, conflict detection and resolution (CD&R) plays an important role by maintaining safety and efficiency of flight traffic. This is also the fundamental task of controllers which contribute significantly on their workload. A good tool like conflict detection and resolution advisory tool can significantly support controllers in handling traffic and potential conflict. However, one of the biggest issues of advisory tools is the trust of controllers for automation. In conflict resolution, each controller has his own way of handling traffic or resolving conflict. Thus controller has the tendency to trust and accept resolution which is similar to his thinking. It poses an interesting research question, which we have partially investigated: whether we learn controller' strategy in conflict resolution from data. This question is difficult to completely answer and it also requires sufficient understanding on applications of machine learning and data mining in CD&R. Thus, in this thesis, there are three sub-problems which we have targeted and investigated: (1) how we can predict controller actions, (2) whether

machine learning can support conflict detection and (3) how we can apply machine learning in conflict resolution.

Related to the first question, we have applied data mining and machine learning techniques for extracting and predicting controller's actions. From literature, learning strategy and behavior from data has been studied in other domains like strategic game. Since behavior and decision of controllers are also encapsulated in operational data like flight trajectory, we can investigate the data to extract controllers' actions. However, reconstructing human strategies from recorded data is difficult, especially when there are no secondary sources of information on strategic and tactical actions, complete environment information and human thinking process, etc. Previous researches on air traffic controllers' strategies from data have provided interesting findings, but they were too generic in nature and did not predict controller's actions in given traffic scenario. To study the action and strategy of controllers, we perform data mining on ADS-B trajectory data. We are also faced the challenge where flight trajectories in ADS-B data are affected by several factors other than controller's actions. It is almost impossible to observe and reconstruct them accurately. Our approach is introducing a simpler definition of actions, called macro-actions which can be observed and extracted from the relationship between the sector entry and sector exit information of flight. Then Random Forest models are developed for learning and predicting those macro-actions given entry information of flight. Using this approach, we can observe and validate the repeated patterns in deciding sector exit point and time of controllers. It can be useful for trajectory prediction and planning flight handover between sectors. In additionally, the repeated patterns can form common flight routes with variations of trajectories within each group.

The second and third questions are related as they work on two important components of CD&R: conflict detection and conflict resolution. The selection of conflict detection algorithm can affect the approach and performance of conflict resolution because it is considered as one of the evaluating criteria for conflict resolution algorithm. Thus, the basic requirements for conflict detection are low computational cost and high accuracy. However, making assumption on uncertainty distribution of aircraft positions and applying grid-search on time dimension for probabilistic conflict detection are the common limitations of studies for this topic. Due to the increase in available trajectory data (i.e., ADS-B), these uncertainties can be learned and approximated directly from the data using Heteroscedastic Gaussian Process instead of pre-defined distributions. In addition, as we build predictive models for aircraft position using data, Bayesian optimization algorithm is applied for probabilistic conflict detection. This approach can easily detect how severe the conflict situation is by quickly and flexibly estimating the time step with highest conflict probability between two flights.

The goal of the third question is developing an Artificial Intelligent agent which is capable of resolving conflicts in the presence of traffic and uncertainty for en-route airspace. Inspired from practical behavior of controllers, the AI agent only resolves conflict with lateral deviations. Recently, the combination of depth learning and reinforcement learning, known

as deep reinforcement learning, has increased the potential of automation for many decision-making problems that were previously difficult to solve due to their high dimensional state and spaces for action. Inspired from Deep Deterministic Policy Gradient Algorithms, we develop a deep reinforcement learning model that is a variant of actor-critic model. In which, the action policy function is approached by a neural network (actor model), while the estimator of the reward function is formed with the second (critic model). This approach is suitable for conflict resolution since it can work with incomplete knowledge to effectively resolve a conflict. It is also able to self-evolve when exposed to unseen scenarios. Finally, it can use controller historical conflict resolution data to build models.

As the results, we have shown that patterns can be extracted and controller's actions can be predicted from data using machine learning models. The controller's actions in conflict resolution also can be mimicked and reproduced by carefully constructing data-driven objective function. In detail, we have formulated the conflict resolution in the presence of traffic and uncertainty as a reinforcement learning problem. For this approach, we have proposed and developed a set of important components, such as learning environment, scenario state representation, reward function, and learning algorithm. Our developed AI agent has the great capability to suggest high quality conflict resolution, with a successful rate of over 81% in the presence of dense traffic and strong environmental disturbance. Machine learning approaches like Heteroscedastic Gaussian Process also show good results in modeling aircraft trajectories and Bayesian Optimization also shows its potential in probabilistic conflict detection. In summary, machine learning and data-driven have shown their potentials as novel approaches for air traffic control or particularly conflict detection and resolution. However, several researches should be conducted to increase the performance of machine learning in this topic, such as airspace/scenario representation, multi-agent reinforcement learning, controller's strategy reconstruction, etc.

Résumé

Les dernières prévisions à long terme du trafic aérien de l'OACI prévoient une augmentation de 250% du nombre de passagers aériens et de départs d'ici 2040. Bien qu'il y ait 42 000 aéroports dans le monde, le trafic n'est pas réparti également et la demande est concentrée sur un petit nombre d'entre eux. Par exemple, les 30 principaux aéroports doivent desservir environ un tiers de tous les passagers, tandis que l'aéroport le plus achalandé peut desservir plus de 880 000 opérations aériennes et 104 000 passagers annuellement. En conséquence, le système de transport aérien à capacité limitée a déjà atteint ses limites. L'augmentation de la demande de trafic a mis ce système à rude épreuve et a entraîné d'importants encombrements, retards de vols ou pollution. La conséquence peut être observée dans le contrôle du trafic aérien (ATC) qui est chargé de la sécurité et de l'efficacité des vols. Les principales ressources de l'ATC, comme l'espace aérien ou les contrôleurs, sont limitées. Ainsi, pour répondre à la nouvelle demande de trafic, il faut des outils de consultation efficaces pour accroître la productivité des contrôleurs et réduire leur charge de travail. Les recherches traditionnelles en ATC se concentrent principalement sur la proposition d'outils de conseil utilisant des problèmes d'optimisation multi-objectifs qui optimisent les valeurs objectives tout en maintenant la sécurité et la sûreté du système. Ces objectifs sont difficiles à atteindre dans la pratique en raison des défis posés par la présence d'incertitudes, de facteurs humains ou d'informations incomplètes. L'une des solutions possibles consiste à tirer parti des données opérationnelles de plus en plus disponibles pour l'apprentissage et la formation de nouveaux modèles, également appelés modèles axés sur les données. Au cours de la dernière décennie, de plus en plus de données opérationnelles sont disponibles pour les chercheurs et de nouvelles percées dans l'apprentissage machine et les modèles d'apprentissage statistique permettent également des modèles plus simples et plus réalistes.

Au sein de l'ATC, la détection et la résolution des conflits (CD&R) jouent un rôle important en maintenant la sécurité et l'efficacité du trafic aérien. C'est également la tâche fondamentale des contrôleurs qui contribuent de manière significative à leur charge de travail. Un bon outil comme la détection et la résolution des conflits peut aider les contrôleurs à gérer le trafic et les conflits potentiels. Cependant, l'un des plus grands problèmes des outils de conseil est la confiance des contrôleurs dans l'automatisation. Dans la résolution des conflits, chaque contrôleur a sa propre façon de gérer le trafic ou de résoudre les conflits. Ainsi, le contrôleur a tendance à faire confiance et à accepter une résolution qui est similaire à sa pensée. Il pose une question de recherche intéressante, que nous avons partiellement

étudiée : si nous apprenons la stratégie des contrôleurs dans la résolution des conflits à partir des données. Il est difficile de répondre complètement à cette question et elle exige également une compréhension suffisante des applications de l'apprentissage machine et de l'exploration de données sur CD&R. Ainsi, dans cette thèse, il y a trois sous-problèmes que nous avons ciblés et étudiés : (1) comment nous pouvons prédire les actions des contrôleurs, (2) si l'apprentissage machine peut soutenir la détection des conflits et (3) comment nous pouvons appliquer l'apprentissage machine à la résolution des conflits.

En ce qui concerne la première question, nous avons appliqué des techniques d'exploration de données et d'apprentissage machine pour extraire et prédire les actions du contrôleur. A partir de la littérature, la stratégie d'apprentissage et le comportement des données ont été étudiés dans d'autres domaines comme le jeu stratégique. Puisque le comportement et la décision des contrôleurs sont également encapsulés dans des données opérationnelles comme la trajectoire de vol, nous pouvons étudier les données pour extraire les actions des contrôleurs. Cependant, il est difficile de reconstruire des stratégies humaines à partir de données enregistrées, surtout lorsqu'il n'existe pas de sources secondaires d'information sur les actions stratégiques et tactiques, d'information environnementale complète et de processus de réflexion humaine, etc. Des recherches antérieures sur les stratégies des contrôleurs de la circulation aérienne à partir de données ont donné des résultats intéressants, mais elles étaient de nature trop générique et ne permettaient pas de prédire les actions du contrôleur dans un scénario de trafic donné. Pour étudier l'action et la stratégie des contrôleurs, nous effectuons un data mining sur des données de trajectoire ADS-B. Nous sommes également confrontés au défi où les trajectoires de vol dans les données ADS-B sont affectées par plusieurs facteurs autres que les actions des contrôleurs. Il est presque impossible de les observer et de les reconstruire avec précision. Notre approche consiste à introduire une définition plus simple des actions, appelées macro-actions, qui peuvent être observées et extraites de la relation entre les informations d'entrée et de sortie du secteur du vol. Ensuite, des modèles forestiers aléatoires sont développés pour apprendre et prédire ces macro-actions en fonction de l'information d'entrée de vol. En utilisant cette approche, nous pouvons observer et valider les schémas répétés pour décider du point de sortie du secteur et de l'heure de sortie des contrôleurs. Il peut être utile pour la prévision de trajectoire et la planification du transfert de vol entre secteurs. De plus, les modèles répétés peuvent former des routes de vol communes avec des variations de trajectoires à l'intérieur de chaque groupe.

Les deuxième et troisième questions sont liées car elles portent sur deux éléments importants du CD&R : la détection et la résolution des conflits. Le choix de l'algorithme de détection des conflits peut influencer sur l'approche et la performance de la résolution des conflits, car il est considéré comme l'un des critères d'évaluation de l'algorithme de résolution des conflits. Ainsi, les exigences de base pour la détection des conflits sont un faible coût de calcul et une grande précision. Cependant, l'hypothèse sur la distribution de l'incertitude de la position des aéronefs et l'application de la recherche par grille à la

dimension temporelle pour la détection probabiliste des conflits sont les limites communes aux études sur ce sujet. En raison de l'augmentation des données de trajectoire disponibles (c.-à-d. ADS-B), ces incertitudes peuvent être apprises et approximées directement à partir des données en utilisant le processus gaussien hétéroscédastique au lieu de distributions prédéfinies. De plus, alors que nous construisons des modèles prédictifs de la position de l'avion en utilisant des données, l'algorithme bayésien d'optimisation est appliqué pour la détection probabiliste des conflits. Cette approche permet de détecter facilement la gravité de la situation conflictuelle en estimant rapidement et de manière flexible le pas de temps avec la plus forte probabilité de conflit entre deux vols.

L'objectif de la troisième question est de développer un agent intelligent artificiel capable de résoudre les conflits en présence de trafic et d'incertitude dans l'espace aérien en route. Inspiré du comportement pratique des contrôleurs, l'agent AI ne résout que les conflits avec les déviations latérales. Récemment, la combinaison de l'apprentissage en profondeur et de l'apprentissage du renforcement, connu sous le nom d'apprentissage du renforcement profond, a augmenté le potentiel d'automatisation pour de nombreux problèmes de prise de décision qui étaient auparavant difficiles à résoudre en raison de leur état dimensionnel élevé et des espaces d'action. Inspiré de Deep Deterministic Policy Gradient Algorithms, nous développons un modèle d'apprentissage de renforcement profond qui est une variante du modèle acteur-critique. Dans lequel, la fonction de politique d'action est abordée par un réseau neuronal (modèle acteur), tandis que l'estimateur de la fonction de récompense est formé avec le second (modèle critique). Cette approche convient à la résolution de conflits puisqu'elle peut fonctionner avec des connaissances incomplètes pour résoudre efficacement un conflit. Il est également capable de s'auto-évoluer lorsqu'il est exposé à des scénarios invisibles. Enfin, il peut utiliser les données historiques de résolution des conflits du contrôleur pour construire des modèles.

Comme résultats, nous avons montré qu'il est possible d'extraire des modèles et de prédire les actions du contrôleur à partir de données en utilisant des modèles d'apprentissage machine. Les actions du responsable du traitement en matière de résolution de conflits peuvent également être imitées et reproduites en construisant soigneusement une fonction objective axée sur les données. En détail, nous avons formulé la résolution des conflits en présence de trafic et d'incertitude comme un problème d'apprentissage de renforcement. Pour cette approche, nous avons proposé et développé un ensemble d'éléments importants, tels que l'environnement d'apprentissage, la représentation des états des scénarios, la fonction de récompense et l'algorithme d'apprentissage. Notre agent AI développé a la grande capacité de suggérer une résolution de conflit de haute qualité, avec un taux de réussite de plus de 81% en présence d'un trafic dense et d'une forte perturbation environnementale. Les approches d'apprentissage machine telles que le processus gaussien hétéroscédastique donnent également de bons résultats dans la modélisation des trajectoires des avions et l'optimisation bayésienne montre également son potentiel dans la détection probabiliste des conflits. En résumé, l'apprentissage machine et l'apprentissage guidé par les données ont

montré leur potentiel en tant que nouvelles approches pour le contrôle du trafic aérien ou en particulier la détection et la résolution des conflits. Cependant, plusieurs recherches devraient être menées pour augmenter la performance de l'apprentissage machine dans ce domaine, comme la représentation de l'espace aérien/scénario, l'apprentissage du renforcement multi-agent, la reconstruction de la stratégie du contrôleur, etc.

Chapter 1

Introduction

1.1 Motivation

Air transportation plays an important role in current society by making the world connected. According to ICAO annual global statistics 2017 [36], the total number of air passengers approximate 4.1 billion, increasing 7.2% compared to previous year, while the number of departures has reached 36.7 million with the 3.1% growth. In which, Asia Pacific still has the biggest contribution with 34% of world traffic and also the largest growth of 10.7% in 2017. Besides North America only grew 4.1% in 2017, world traffic in the other regions also increase from 6.5% to 8.6%. The latest ICAO long-term air traffic forecasts expect the growth of 250% in airline passengers and number of departures by 2040. Even though there are 42 thousand airports worldwide, the traffic is not equally distributed and the demand concentrates on a small number of them. For example, top 30 airports must serve around one third of passengers, while the busiest airport may serve more than 880,000 aircraft operations and 104,000 passengers annually. Because of this fact, the air transport system which is capacity-limited has already reached its limit. The increasing in traffic demand has further strained this system and led to significant congestion, flight delays or pollution [21].

With the expected growth of air transport demand in next decades, there is a need for new analysis techniques and operational strategies for air transportation management (ATM). Traditional researches in ATM yield a range of multi-objective optimization problem which optimize the objective values while maintaining the safety and security of the system. Those objectives are difficult to achieve in practice due to the challenges posed by the presence of uncertainties, human factors or incomplete information. A potential solution is taking advantages of the increasingly available operational data for learning and training new models, called data-driven models. Studying data-driven approaches in ATM has been reported in literature for long time. However, for the last decade, more operational data can be accessed and study as well as new breakthroughs in machine learning and statistical

learning, data-driven approaches can build simpler and more realistic models without the need of too many assumptions on data and operation.

ATM is a complex system with multiple components. From the definition of Eurocontrol, it primarily consists of three main distinct components: Air Traffic Control, Air Traffic Flow Management and Aeronautical Information Services. In the scope of this thesis, we only concentrate on studying challenges in Air Traffic Control. As mentioned in [35], the primary purpose of Air Traffic Control (ATC) worldwide is to prevent collisions, organize and expedite the flow of air traffic, and provide information and other support for pilots. In regions where the Air Traffic Management (ATM) system is well-developed, three types of control facilities play a critical role during the successive phase of a typical flight: (1) airport traffic control tower (aerodrome control), (2) the terminal airspace control center (approach control), and the (3) en route control center (area control) [56]. En-route airspace is one of the most congested airspace as it is mainly used in the cruise phase of a flight. It is divided both vertically and horizontally according to local air structure and traffic flows into smaller area called “sectors”, and a sector is generally considered as fundamental “unit” of airspace volume from the ATM point of view. The en-route sector is usually managed by a team of two air traffic controllers: planning controller (D-side) and executive controller (R-side) [53].

Both D-Side and R-Side air traffic controllers (ATCOs) are responsible for airspace monitoring, conflict detection and resolution, along with managing route/altitude modification requests from the aircraft. The difference between the two roles lies in the strategic and tactical level of intervention. The D-side controller is primarily responsible for processing flight-plan information to plan, coordinate and organize the flow of air-traffic entering in to a sector. The D-side controller uses the flight-plan information and employs Medium Term Conflict Detection (MTCD) tool [2] to predict aircraft trajectories in a 20 minute look-ahead time window. D-side controller employs a variety of strategies/actions i.e. combination of altitude, speed, heading change, hold maneuvers etc. to maintain an orderly flow of the incoming traffic in a sector, such that it minimizes crossings events which may lead to loss of separation. This ensures, at a tactical level, a minimum intervention is required from R-Side controller while managing the air-traffic in a given sector. The R-Side controller uses Short Term Conflict Alert (STCA) tool [70] to predict any loss of separation in a 4 to 8 minutes look-ahead time window. R-side controller is mainly concerned with tactical interventions to maintain safe separation between flights.

Fig. 1.1 illustrates the information flow between R-side controller and D-side controller and how the two ATCOs work collaboratively to achieve an orderly flow of air traffic in a sector. D-side controller receives flight plan information of the flight before it enters the sector (transfer of communication), at this point the aircraft is in contact with both, the previous sector D-side controller as well as next sector D-side controller. The D-side controller then negotiates with the aircraft regarding entry flight level, entry speed and entry

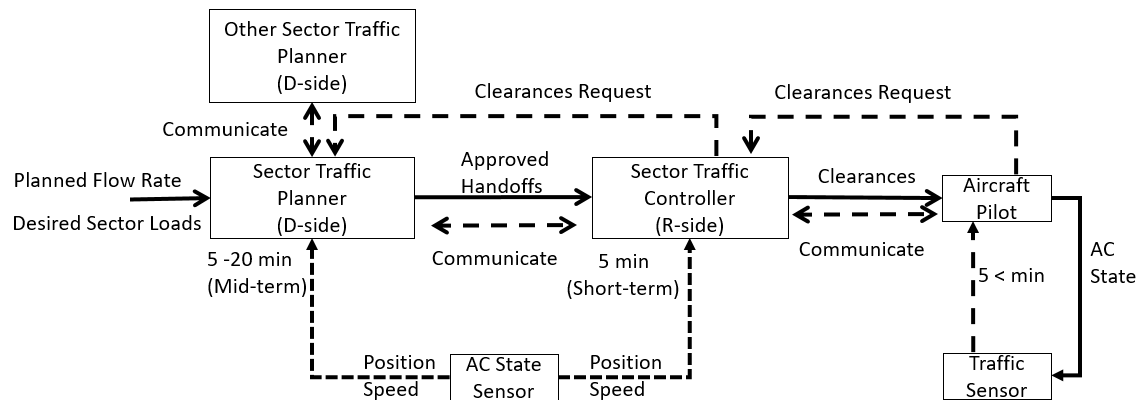


Fig. 1.1 Information Flow between D-Side and R-Side air traffic controllers for traffic flow planning and separation assurance in a sector.

way-point depending upon the strategic situation in his/her sector at a certain look ahead time. The primary objective of this planning is to maintain an orderly flow of traffic and to minimize crossings which may lead to a loss of separation (LOS) scenario for an R-Side controller to intervene. Once the aircraft enters the sector boundaries (transfer of control), the D-side hand-off the aircraft to R-side controller who then provide ATC services via radio communication. In some circumstances (e.g. bad weather) the aircraft may need to be handed off differently than the letter of agreement, in this case, the D-side controller must coordinate with the other sector controller to ask for approval for another route which is not specified in the letter of agreement before the aircraft cross the boundary.

As ATC is becoming increasingly complex and dynamic, the role of ATCOs in an ATC system is getting more and more challenging [74]. Within the safety critical domain of ATC, workload remains dominant consideration in improving ATC system performance. Since the main responsibility of the D-side controller is to manage and organize traffic flow such that the tactical flight interventions from R-side controller is minimized, it is desirable to automate D-side controller tasks such that its task load is reduced. A possible way is to develop a mechanism which can learn D-side controller's traffic management strategies and have the ability to predict such actions for a given traffic scenario. There are two big challenges for this approach are understanding controller strategy from data and proposing novel learning model for this task.

I believe that, in ATM and ATC, the controller is always the center of the system. Thus, instead of aiming for full automation, I focus on developing a hybrid human-machine system. This thesis has introduced an approach to develop personalized automation or assistance tool for controllers which mimic their action in resolving flight conflict. The tool will suggest the resolution to the controller and receive feedback from his decision to update the model. As a result, it is customized for each controller and evolve with him over time. The machine always recommends resolution similar to his preference and the trust of the controller on the

machine will be built over the process. Then, the controller can let his assistance tool handle daily traffic when he only needs to focus on abnormal scenarios or to manage traffic at a higher level. With good support from the machine, the controller can handle more traffic with less workload while maintaining safety.

In this thesis, we want to explore the applications of data-driven and machine learning techniques for conflict detection and resolution problem. The work includes multiple steps from mining data, developing learning models for conflict detection and conflict resolution. Firstly, We have proposed a method to extract ATCO actions from ADS-B data. The Random Forest predictive models are used to validate extracted patterns which imply the relation between entry and exit points for the given sector. From obtained knowledge, we process the ADS-B data by clustering trajectories to form flight routes which are used for trajectory prediction and other applications. Secondly, we propose a machine learning approach for probabilistic conflict detection. We apply Heteroscedastic Gaussian Process in modeling trajectories. The proposed method is used for trajectory prediction considering uncertainty of positions. Later, Bayesian Optimization method is applied for detect potential conflict quickly and flexibly, comparing to classical Monte Carlo method. Finally, we have formulated the problem of conflict resolution in the presence of traffic and uncertainty as a Reinforcement Learning problem. Important components of the algorithm, such as learning environment, scenario state representation, reward function, and learning algorithm, have been discussed in great details. We also introduce a framework for the assessment of reinforcement learning method applied to conflict resolution problem.

1.2 Related Works

1.2.1 Controller Strategy in Traffic Management

The quest of understanding and learning human's strategies in games like Chess, Backgammon, Game of Go etc., and predicting the next move of an opponent is well known in literature [22], [13], [64], [25]. Machine learning methods such as deep neural networks, tree-search methods and Bayesian reinforcement learning have recently been quite successful in learning game strategies and outperforming world champions [71], [19], [69], [10].

However, a major assumption in such machine learning algorithms is that the training and future data must be in the same feature space and have the same distribution [77]. In air traffic domain, the feature space (airspace structure including its airways and way-points) and the data distribution (aircraft trajectory points) varies a lot. As every airspace is unique, every air traffic scenario is also different. Further, the flight maneuvers or conflict resolutions rely almost exclusively on the judgment of air traffic controllers [57]. In handling the flight traffic, specially, resolving conflict, each ATCo has his own way to perform his task. The strategy can be the result of training process, their experiences as well as their own

personality [23]. Even controllers who follow the same training and work at the same place may have different ways in resolving conflict while still maintaining the safety and efficiency of airspace. Previous research into identifying air traffic controller's strategies from traffic data have found some interesting insights, but they were generic in nature and lacks any predictability of ATCo's actions given a traffic scenario. For example, Authors of [59] find out that in the presence of conflict between few aircraft, the velocity variation strategy seems to cost more (in terms of time of flight) than the heading angle deviation strategy. In [2], authors developed an evolutionary computation framework to identify flight maneuvers that may expose a traffic scenario to loss of separation, but falls short of generalizing it to a range of traffic scenarios. In [28], authors predicted ATCo workload from past sector merge and split actions but could not generalize the learning to new sectors due to over-fitting of the training data. In [78], authors proposed use of game theory for conflict resolution in en-route airspace. Apart from en-route airspace, machine learning methods have also been applied in terminal airspace. For example, in [42] a simulator was designed which can simulates control of air traffic, landing clearance and departure by using Back Propagation Network based on various controlling parameters, albeit for single runway only.

Furthermore, multiple researches are performed to prove the ability of machine learning in modeling behavior from recorded data, especially in video game with certain number of replays [17, 27, 46, 51, 83, 86]. Inspired from those work, the appearance of strategy in data, called *patterns*, can be observed and extracted from operational data. However, building learning models for human's strategies with recorded data is challenging since there are not separately information about the strategic, tactic actions, actual obtained environment information or human thinking process, etc. The work in [32] reviews data-driven approaches for modeling players in video game which extracts a game player's traits and tendencies which include player's behaviors, dispositions and aims [8, 84]. The models are derived either fully or semi-automatically [84, 33] from significant quantity of extracting information.

For studying ATCo's actions and strategy, we will study ADS-B trajectory data which encapsulates actions. However, we also face the challenge where ADS-B trajectory data is affected by several factors besides ATCo's actions. Observing and reconstructing exactly them are almost impossible. Our approach is mining patterns from data and proposing a simpler definition of actions called macro action which can be helpful for further studies in conflict detection and resolution.

1.2.2 Data-driven Approach for Probabilistic Conflict Detection

The conflict detection and resolution problem (CDR) is usually considered as one problem but it also can be divided into two sub-problems: Conflict Detection (CD) and Conflict Resolution (CR). They are related but can be approached and solved separately. A conflict

between two aircraft is defined as the violation of vertical and lateral separate conditions ($s_v \leq 1000 ft$ and $s_h \leq 5NM$) [56]. The conflict is detected based on predicted trajectories of aircraft in the presence of uncertainties such as weather, wind or noise of aircraft location. Furthermore, the new concepts like free flight from NextGen can also introduce new challenges by introducing the uncertainty in pilots' intent in which the pilot can choose the routes between two consequent way-points based on their references, weather and traffic. They increase the difficulty for predicting flight trajectories and then detect potential conflicts.

Several scientific approaches for conflict detection have been investigated. Some work focus on solving the conflict detection with given nominal routes and flight plans. Matsuno et.al. [52] proposes approach for detecting potential conflict at merging point of air traffic network. While, Hao et.al. [31] solves CD problem for free flight by modeling movement of aircraft as random walk with unknown intent, then conflict probability is computed at each moment of discretised time. In [43], authors group conflict detection algorithm into three main categories: deterministic, worst case and probabilistic. In which, probabilistic approaches are considered as the potential direction and more practical for CD in incorporating different type of uncertainties in prediction. In probabilistic setting, some studies consider empirical distribution model of future aircraft positions [43, 58, 81] or model the dynamic of aircraft [34, 65] with modeled uncertainty. Two main categories in probabilistic conflict detection are analytic [58] and Monte Carlo [11, 80]. In analytic approaches, they require an actual closed form solution which is often not possible to achieve because of the complex of real world scenario. Several constraints should be put on aircraft dynamic, uncertainty, etc. to compute conflict probability. Monte Carlo is more practical approach because it can incorporate a more complicated state-space model with flexible uncertainty models (e.g. non-Gaussian, multi-modal). Especially, when additional information such as weather or pilots' intent are available, new uncertainties can be included in the approach. However, the biggest disadvantage of Monte Carlo approach is its computation speed. When working with uncertainty, we must simulate multiple different potential trajectories for each aircraft. Additionally, conflicts are rare events in the space of all possible trajectories. In worst case or probability, identifying and simulating such event is expensive. Yang et. al. [80] proposes an algorithm to speed up Monte Carlo simulation by simplifying aircraft dynamic model. Authors in [5, 6, 14, 49] also try to detect probabilistic conflict at discretized time by computing the probability at each moment with assumptions about uncertainty and aircraft dynamic. Those approaches share common limitations such as their assumption about uncertainty distribution and high computational cost for estimating conflict probability. Besides, in study of conflict resolution, computational cost of conflict detection is also a bottle neck and challenge for evaluating a large amount of candidate maneuvers.

Recently, data-driven and machine learning approaches with real-time data gain more attentions from ATM community. There are also studies applying machine learning in trajectory prediction [18, 7, 4]. These kinds of approaches can reduce the amount of

assumptions, especially weather uncertainty and aircraft dynamic. The aircraft position can be predicted by learning models and there will be no explicit formula for conflict probability. That leads to the requirement for new studies in conflict detection using Monte Carlo methods.

1.2.3 Machine Learning Approach for Conflict Resolution

There are several mathematical models for conflict resolution which have been reported in the literature. For a comprehensive review see Yang et al. [41]. Some recent works look into enhancing the capability of such automated conflict solvers. For instances, Yang et al. [82] used probability reach sets to represent aircraft locations, and aircraft deconfliction is performed by separating these reach sets using second-order cone programming with aircraft dynamics considered. However, this approach does not perform well in handling large number of aircraft with uncertainty. In the recent research, Hao et al. [30] employed aircraft reachable space, where conflict resolution scheme accounts for the intent of the aircraft via aircraft's space-time Prism. Yet, the execution time for this method scales up significantly with the number of aircraft involved, especially when a fine grid is applied. Model predictive control (MPC) is also a promising approach for conflict resolution. Yokohama [85] applied MPC to perform trajectory prediction and conflict resolution simultaneously, in which the aircraft separation condition is implicitly imposed during trajectory prediction. However, the mathematical model is highly complex and the resolution quality depends on the quality (noise free) of available historical data. MPC was also employed in the work by Jikov et al. [39], in which the authors proposed multiple models for conflict resolution considering the minimization of the cost due to the maneuver, using efficient algorithm. In another approach, advanced surrounding traffic analysis was proposed as the basis for conflict resolution decision [66]. The analysis of surrounding traffic includes the concept of aerial ecosystem and traffic complexity evaluation for the determination of resolution, in which the domino effect, i.e. the number of aircraft causally involved in the separation service, is considered. Large scale conflict resolution models were also proposed by Allignol et al. [6] and Liu et al. [50]. While the work in [50] uses aircraft location network and limits its resolution's maneuver to velocity adjustment only, the model provided in [6] provides a for 3D conflict resolution with limited uncertainty.

From our observation of the literature, mathematical models for conflict resolution have several common limitations. First, complete knowledge of the mapping from conflict scenarios to maneuvers is required; this makes mathematical models highly complex and results in poor quality resolutions in the presence of high uncertainty, as the full knowledge about the environmental uncertainty could never be obtained. Second, the input scenarios must be well standardized for the mathematical models to work properly, and the models do not self-evolve when dealing with unseen and non-standard scenarios. In this thesis,

we attempt to overcome these drawbacks by considering machine learning approach for conflict resolution, as learning method does not require prior knowledge of how to efficiently resolve a conflict, and learning algorithm is able to self-evolve when being exposed to unseen scenarios.

1.3 Contribution of this thesis

This thesis is an initial attempt to investigate machine learning approaches for conflict detection and resolution (CDR). It is complicated and requires investigating from multiple perspectives.

First of all, we have **proposed a simple method to extract ATCO actions from ADS-B data**. Different from classical approaches for trajectory prediction which always assume no intervention from controllers. The Random Forest predictive models are used to validate extracted patterns which imply the relation between entry and exit points for the given sector. This can be extended by extracting or defining more complex actions using ours as marco/proxy actions. This study is helpful for understanding and investigating the nature and patterns in a given sector. The results can be used to explore Air Traffic Controller's strategies in conflict resolution. Moreover, it also can be considered as the preprocessing step to support building dataset for training and evaluating AI-Agent in conflict detection and resolution (CD&R). From obtained knowledge, we process the ADS-B data by clustering trajectories to form flight routes based on grouping their entry and exit points. Those flight routes can be used for trajectory prediction and other applications.

Secondly, we have **proposed machine learning approaches for probabilistic conflict detection in both trajectory prediction and probabilistic closest point of approach estimation**. We have confirmed the advantages of Heteroscedastic Gaussian Process in modeling trajectories with inhomogeneous variance of aircraft position over time. The proposed method is used for trajectory prediction considering uncertainty of positions. These trained predictive models are necessary for Bayesian Optimization approach to detect potential conflict quickly and flexibly, comparing to classical Monte Carlo method. The algorithm can work with continuous time it means it can locate the conflict position without the impact of fine level of discretized time. Especially, when the cost of computing conflict probability is expensive, the contribution of this method is more significant. Thus, this approach can be used to speed up the conflict detection in flight simulators in the presence of uncertainty.

Finally, we have **formulated conflict resolution problem in the presence of traffic and uncertainty as a reinforcement learning problem**. Important components of the algorithm, such as learning environment, scenario state representation, reward function, and learning algorithm, have been discussed in great details. We have also laid out the evaluation of model's performance, which could be considered as a framework for the assessment of reinforcement learning method applied to conflict resolution problem.

1.4 Organization of the thesis

Each chapter in this thesis describes a sub problem. The organization is as follows. Chapter 2 performs two main data mining tasks to explore ADS-B and understand controller's strategy or behavior. The first task is performing action extraction from raw data. We proposed simple and novel definitions for controller's actions which can be observed and extracted as existing patterns. Secondly, we build machine learning models to predict the controller's actions based on flight information at entry points. The Random Forest models are used as learning model because of their Interpretability. Model performance are evaluated and reported in term of *accuracy* for classification and *R_square* (R^2) for regression. The data-driven algorithms for trajectory prediction and probabilistic conflict detection are described in Chapter 3. There are two types for Gaussian Process (GP) will be investigated which are homoscedastic and heteroscedastic noise models. In our case, the variations in longitude, latitude or altitude of aircraft positions will vary along time dimension. The heteroscedastic GP is expected to capture the variance of real data. We develop the predictive models for flights based on learned uncertainty and current flight information (conditional points). Secondly, we apply Bayesian optimization algorithm for probabilistic conflict detection problem. Our approach's performance is compared to Monte Carlo method in term of detected values and computational cost. In Chapter 4, we propose an artificial intelligent agent that is capable of resolving conflicts, in the presence of traffic and given uncertainties in conflict resolution maneuvers, without the need of prior knowledge about a set of rules mapping from conflict scenarios to expected actions. The conflict resolution task is formulated as a decision-making problem in large and complex action space, which is applicable for employing reinforcement learning algorithm. Our work includes the development of a learning environment, scenario state representation, reward function, and learning algorithm. Chapter 5 concludes with a summary and extensions for future research.

Chapter 2

Learning ATCo Actions from ADS-B Data

En-route airspace is one of the most congested airspace, as it is mainly used in the cruise phase of the flight. The en-route sector is usually managed by a team of two air traffic controllers: planning controller (D-side) and executive controller (R-side). D-side controller is responsible for processing flight-plan information to plan and organize the flow of traffic entering the sector. R-side controller deals with ensuring safety of flights in their sector. A better understanding and predictability of D-side controller actions, for a given traffic scenario, may help in automating some of its tasks and hence reduce workload. In this chapter, we want to study how ATCo performs strategy in maintaining safety of traffic in their sectors. We propose a learning model to predict D-side controller actions. It is modeled as a supervised learning problem where the target variables are D-side controller actions and the explanatory variables are the aircraft 4D trajectory features. The model is trained on one month of ADS-B data over an en-route sector, and its generalization performance was assessed, using cross-fold validation, on the same sectors. Results indicate that the model for vertical maneuver actions provides highest prediction accuracy (99.7%). Besides, model for speed change and heading change action provides predictability accuracy of 88.7% and 72.4% respectively. The model to predict the set of all the actions (altitude, speed and heading change) for each flight achieves an accuracy of 0.68 implying for 68% of flights, D-Side Controller's can be predicted for all the actions from trajectory information at sector entry position. Experiment results have confirmed the existence of patterns in data and also highlight the strong relation between the exit and entry points which can be used for trajectory prediction.

2.1 Introduction

In the air traffic management (ATM) system, Air traffic control (ATC) plays a crucial role as it is responsible for maintaining flights safety and efficiency. To improve the current system of air traffic control, new capacity of airspace will be necessarily introduced with the development of assistant tools for the air traffic controller (ATCo), especially in giving conflict resolution advisories. Previous studies focus more on using mathematical models to optimize the maneuvers to resolve conflict. They made use of rules, constraints and objectives in this field, but not controller's behaviors or preferences. In handling the flight traffic, specially, resolving conflict, each ATCo has his own way to perform his task. Learning and understanding what controller's strategy is and how it looks like has strong impact on building learning model architecture, mining and learning from available data. The appearance of strategy in data is called "patterns" which can be observed and extracted from operational data.

Recorded Automatic Dependent Surveillance—Broadcast (ADS-B) data open new opportunities for research in air traffic control. Useful information can be extracted and combined with traditional data for more understanding about flight operation. The first question we aim at is how ATCo performs strategy in maintaining safety of traffic in their sectors. This question can be answered indirectly by investigating patterns of traffic from ADS-B data. We propose using tree-based methods for regression and classification [29] to explore these patterns. A motivation of using tree-based methods is that they closely mirror human decision-making than other classification approaches [24].

2.2 Approach

We propose a learning process demonstrated in Fig. 2.1, which contains pre-processing data, extracting ATCO actions and building Learning Models. The 4D trajectory change points for individual flights are constructed directly from ADS-B raw data and the spatial information about sector is collected and processed from Aeronautical Information Publication (AIP). Then pre-processing techniques are applied to clean data and remove trajectory points which are outside of the selected sector.

However, when considering controller's actions, there are the limitations in what could be observed and extracted from ADS-B data (GPS positions every 30 seconds). Without ground truth about controllers' actions, it would be impossible to achieve extracted set of actions from data. Thus, in the scope of this thesis, we define more simple actions which can be successfully extracted from data but still reflect patterns of traffic in given sector. The detail will be discussed in following sections. The illustration of general process is presented in Fig. 2.1.

Two points from each flight (entry and exit points in term of *timestamp*) were used to identify the new flight trajectory upon entering the sector, which reflect the main course of the trajectory inside sector. Actions or interventions are extracted by comparing the above results with flight information (speed, altitude, course) at the entering point. Those can be seen as the changes in *Speed*, *Heading*, *Altitude* which need to be applied for each flight to reach the exit point at the given 4D position (latitude, longitude, altitude, time). Up to this point, two sets are generated: action values (continuous) and actions ($[-1,0,1]$) which are related to ground speed rate, vertical speed and heading for each flight.

Finally, using the information at entry points as the input and the mentioned two sets as targets, we build two groups of Random Forest Models: Regression and Classification. Output of those models are the changes or applied actions for each given flight at its entry point.

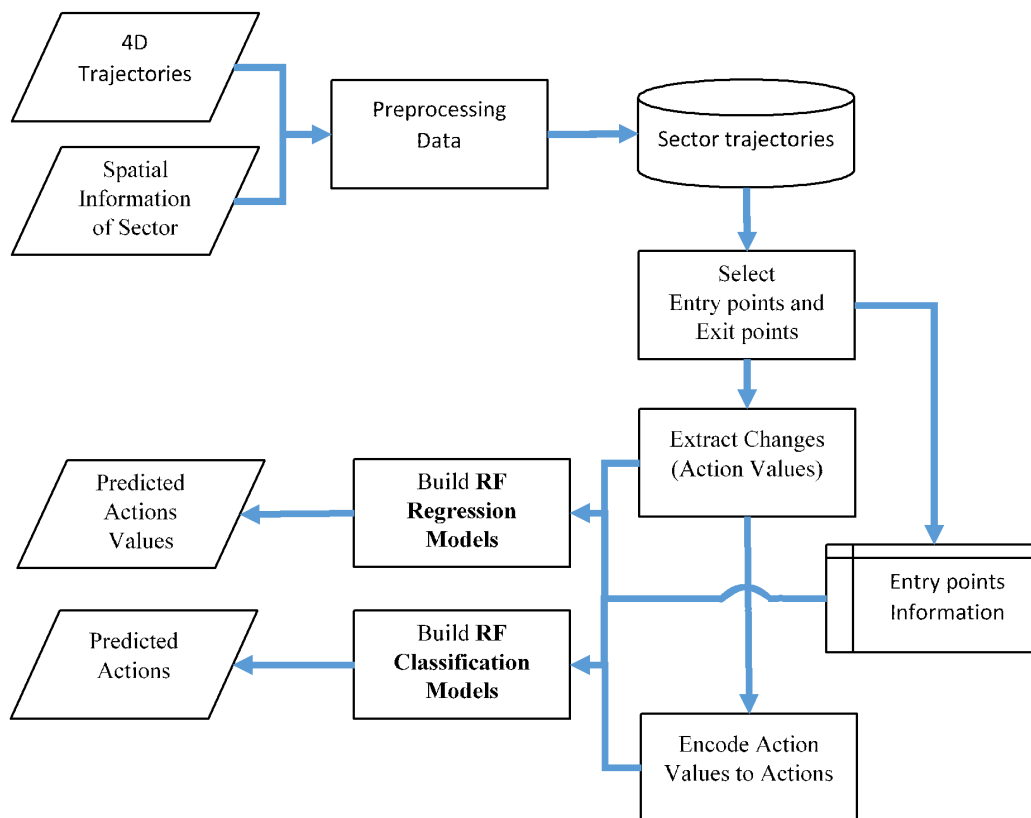


Fig. 2.1 The Learning Process using Random Forrestr.

2.3 Trajectory Data

2.3.1 Data-set

ADS-B is a surveillance technology using satellite navigation to locate aircraft's position and broadcast this positions to other aircraft and ground antennas. An aircraft will estimate its

positions and periodically broadcasts that information to air traffic control ground stations or other aircraft. This technique makes aircraft become visible which support both situational awareness and self-separation.

In this work, the ADS-B data is obtained from FlightAware. It is collected for South-East Asian region for one month (December 2016). Each sample of data contains features as shown in Table 2.1, and an example of one row sample of 4D trajectory data is shown in Table 2.2.

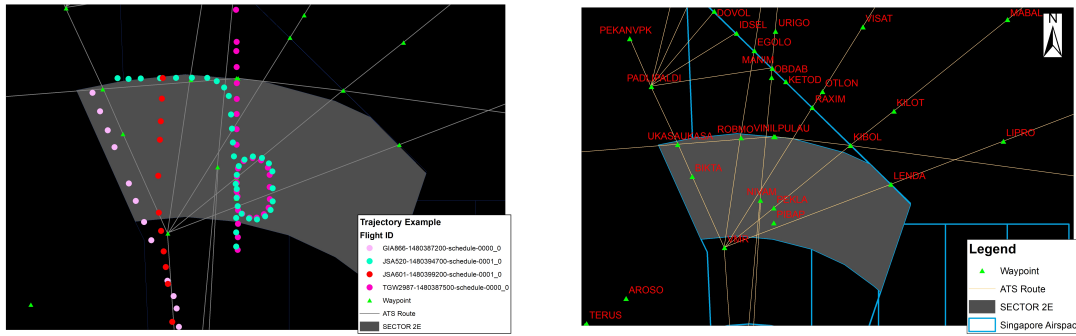
Feature	Description
Position	Latitude (decimal degrees),Longitude(decimal degrees), Altitude (ft).
Ground Speed	Horizontal speed relative to the ground (knots)
Rate of Climb	Altitude change (feet per minute)
Course	Aircraft heading relative to North (decimal degrees).
Flight ID	Unique serial number represents each flight.
Time	Time(UTC) that data been recorded.

Table 2.1 4D Trajectory Data Features

Flight ID	CDG4963-1482966600-schedule-0000
Time (UTC)	12/31/2016 00:58:14
Latitude (dec. deg)	34.29153
Longitude (dec. deg)	108.5708
Groundspeed (kts)	182
Altitude (ft)	3900
Rate (fps)	-514
Course (deg)	18

Table 2.2 One row sample of 4D Trajectory Data

Each group of records represents trajectory of a flight, carrying status of the flight spatially throughout time. Fig. 2.2a illustrates trajectories of 4 different flights passing through the sector. Sample points with same color belong to same flight, time interval between each point is about 15 to 30 seconds.



(a) An example of four different flight Trajectories, with Speed, Altitude, heading change and Hold maneuver in the sample data.

(b) The layout of Sector 2E, with its way point and ATS routes, used in this study.

Fig. 2.2 Demonstration of Sector 2E

2.3.2 Selected Sector

For this research, we have identified Sector 2E, an en-route sector within Kuala Lumpur FIR, managed by Singapore ACC, for providing air traffic service from FL120 to FL360 inclusive. Fig. 2.2b depicts the spatial characteristics of the selected sector. It takes about 5 minutes in average for a typical flight to cross sector. The sector contains 8 waypoints and is crossed by 8 ATS Route. There is one crossing in the sector and one converge point at the south of the sector(at waypoint **VMR**). The spatial characteristics and the airspace structure of the sector are simple, therefore the scenario of the sector can be easily identified.

2.3.3 Data Cleaning and Filtering

The original ADS-B data-set is a large data-set with noises and missing data points. Moreover, with the given spatial information of Sector 2E, only a subset of trajectories should be considered and investigated. Thus, some pre-processing steps need to be applied:

1. At first we apply a 2D spatial-filtering to filter out all trajectories which do not pass through the sector. We found that there were 12,141 flights that passes though Sector 2E in Dec 2016 data.
2. A second 2D spatial-filtering is applied to filter out trajectory points outside the sector. It is separated from Step (1), since the criteria for filtering can be changed in future for different information extraction.
3. To deal with missing data points, we remove all the flight trajectories which have less than three data points in the sector. After this step, the working data-set remains 9,082 flights.

2.4 Extraction of ATCO Action

In practical scenario, pilots communicate with D-Side controller while entering the sector and with R-side controller once inside the sector. An aircraft trajectory points bears signature of both R-Side and D-Side Controller actions. However, D-Side controller actions can be identified in the trajectory data by observing trajectory prior to entering a sector.

To better understand the relationship between controller actions and aircraft trajectory data derived from ADS-B, we first visualized the 4D data with GIS. The observing airspace is visualized discretely by grids with 3 Nautical miles in length and width, action in the same grid will be summed up and every grid will be classified into 5 classes using Jenks Natural Breaks Classification method [38], a data clustering method designed to reduce the variance within classes and maximize the variance between classes. The color of the grid from yellow to red means the higher frequency of certain feature appear in the position, and the first class was set not to visualize. Fig. 2.3 shows the spatial density of entering point and exit point of the aircraft of Sector 2E while Fig. 2.4 shows spatial distribution of ATCO actions in the sector. These figures indicate that there are patterns in ATCO actions.

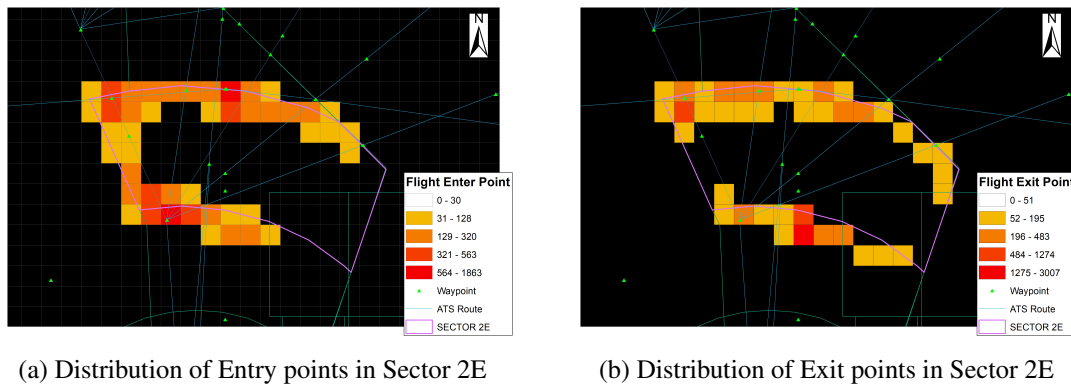


Fig. 2.3 Distribution of Entry and Exit points in Sector 2E in term of number of flights.

2.4.1 Extracting Flight Change/ Action Values

Observing from flight trajectories Fig. 2.2a, there are multiple changes in trajectories of aircraft flying over the sector. However, the flight usually enters the sector at specific region and would be directed to follow the designed ATS routes and waypoints which means all the changes should be applied for aircraft to reach an specific region to exit the sector. Fig. 2.3 shows the distribution of entry and exit points in the sector and also highlights that idea. Thus, to simplify the extraction of ATCO actions, we can only consider the changes between entry and exit points. This approach can capture the major changes of the flight in the sector. Three values will be extracted from those pairs of points (illustrated in Fig. 2.6):

- Ground Speed Rate: while cruising inside Sector 2E, flight speed usually varies. However, because of the nature of this sector, three common and simple trends can

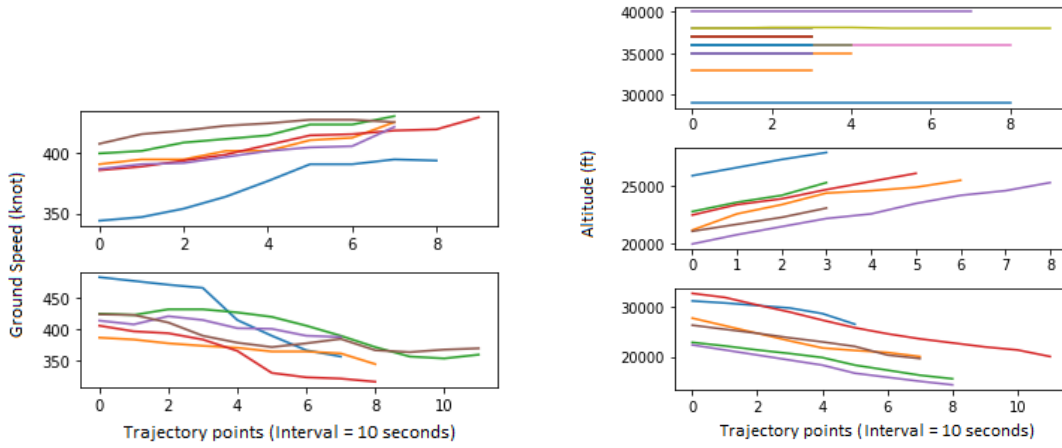


Fig. 2.4 Distribution of Vertical Actions: Climb, Maintain Level and Descend in Sector 2E in term of number of flights.

be observed from data: maintain speed, increasing (speed up) and decreasing (slow down), see in Fig. 2.5a. It indicates that the rate of ground speed change is quite stable and can be used as an action of flight. From that observation, the rate of change is extracted and considered for next learning steps. In detail, it is computed based on estimating the required rate for a flight with a given speed at entry point to travel from entry to exit points within given travel time.

- **Vertical Speed:** the actions related to vertical speed. Similar to Ground Speed Rate, we can observe some common trends in altitude changes from data (Fig. 2.5b). The vertical speed is used as the vertical actions and computed simply based the ratio of difference in altitude between entry and exit points and the travel duration.
- **Delta Course:** it is the difference between course at the entry point and "course in sector". Since course of the flight varies throughout the sector and course at exit point also doesn't reflect the travel direction, we simplify the definition for "course in sector" as the direction from entry and exit points which is the expected direction for flight to travel through our sector. We use delta course instead of "course in sector" because it reflects the turning actions of flights after entering sector.

Our extracted actions can be considered as macro actions. Based on predicting those actions, we can estimate the exit time and position for each flight given entry information.



(a) Example of increasing and decreasing trends in ground speed from real data.

(b) Example of trends in Altitude from real data.

Fig. 2.5 Visualization for trends in ground speed and Altitude from data. Each line in the figure belongs to a different flight in data.

This level of detail can improve the accuracy of prediction models and be useful to explore patterns from data. The more detail actions will be discussed in Section 2.7.

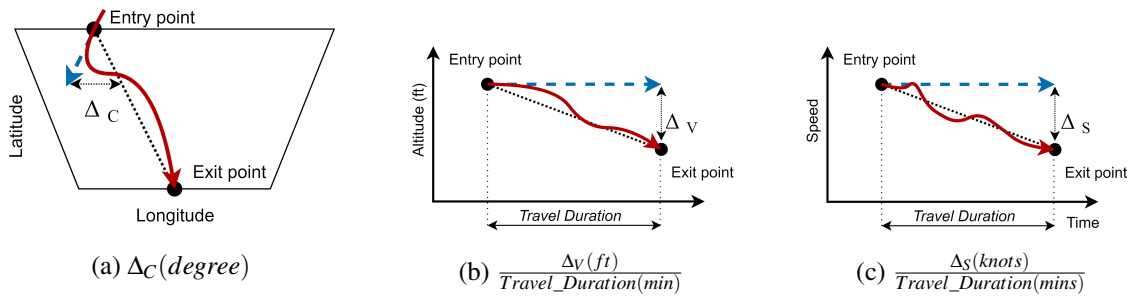


Fig. 2.6 Definitions of ATCo Actions from Entry and Exit points.

The detail of action extraction from entry and exit points is described in below algorithm (Algorithm 1).

2.4.2 Encoding ATCO Actions from Action Values

As the requirement for supervised learning, we need a set of actions as labels for building classification model. Thus, for each flight, the set of actions should be converted from extracted action values. Fig. 2.7 displays idea on how the labels are encoded from values. There are three type of actions which related to ground speed rate, vertical speed and delta course. Each of them is encoded into 3 actions: -1, 0, 1 based on the *thresholds* selected as follow:

- Speed Actions: If the absolute change of speed between exit and entry points is less than 10 knots, we consider it as maintaining speed. Besides the expected travel time of the sector is 5 mins. From both of that, $threshold_{sr} = \pm 0.017(m/s^2)$ is selected.

Algorithm 1: Action Extraction from Entry and Exit

Input: Entry and Exit Points
Output: GSpeed Rate, VSpeed, Delta Course

- 1 $Travel_Dist = Euclidean_distance(Entry, Exit)$
- 2 $Travel_Time = Exit.Time - Enter.Time$
- 3 $Max_GSpeed = 2 * \frac{Travel_Dist}{Travel_Time} - Entry.Speed$
- 4 $GSpeed_Rate = \frac{Max_GSpeed - Enter.Speed}{Travel_Time}$
- 5 $VSpeed = \frac{Exit.Alt - Entry.Alt}{Travel_Time}$
- 6 $\theta = atan2(\frac{Exit.Lat - Entry.Lat}{Exit.Lng - Entry.Lng}) \in (-\pi, \pi]$
- 7 **if** $\theta \geq 0$ **then**
- 8 | $Course = \theta$
- 9 **else**
- 10 | $Course = 2\pi + \theta$
- 11 $Delta_Course = Course - Entry.Course$
- 12 **return** $GSpeed_Rate, VSpeed, Delta_Course$

- Vertical Actions: If the absolute change in altitude between two points is less than 100 feet, we consider it as maintaining level. Besides the expected travel time of the sector is 5 mins. Thus, $threshold_vs = \pm 20(\text{feet}/\text{minute})$ is selected. As showed in Fig. 2.4, the *climb* and *descend* actions are mainly distributed corresponding to South and North of the sector which will have a strong influence in building predictive model given entry information of a flight.
- Course Actions: If the absolute of delta course is less than 3 degrees, we consider it as maintaining course. $threshold_dc = \pm 3(\text{degree})$ is selected for Course Actions.

Fig. 2.8 illustrates the distribution of all extracted actions. The distribution of Speed Actions and Course Actions have bell shapes (in (a) and (c)). From (c) and (d), we conclude that Course Action has balanced distribution. However, the mean of Ground Speed Rate is positive, therefor there are more *speed up* actions than others in Lateral Actions. It is confirmed by (b): around 86% of Lateral Actions in this sector are *speed up*. However, since every action is equivalently considered, we do not solve unbalanced problem in learning model. In (e), the changes in vertical speed can be seen as two separated normal distributions. Then, there are only two major actions: *climb* and *descend* corresponding to two distributions. Maintain Level is kept but there are limited samples for this action, as see in (f).

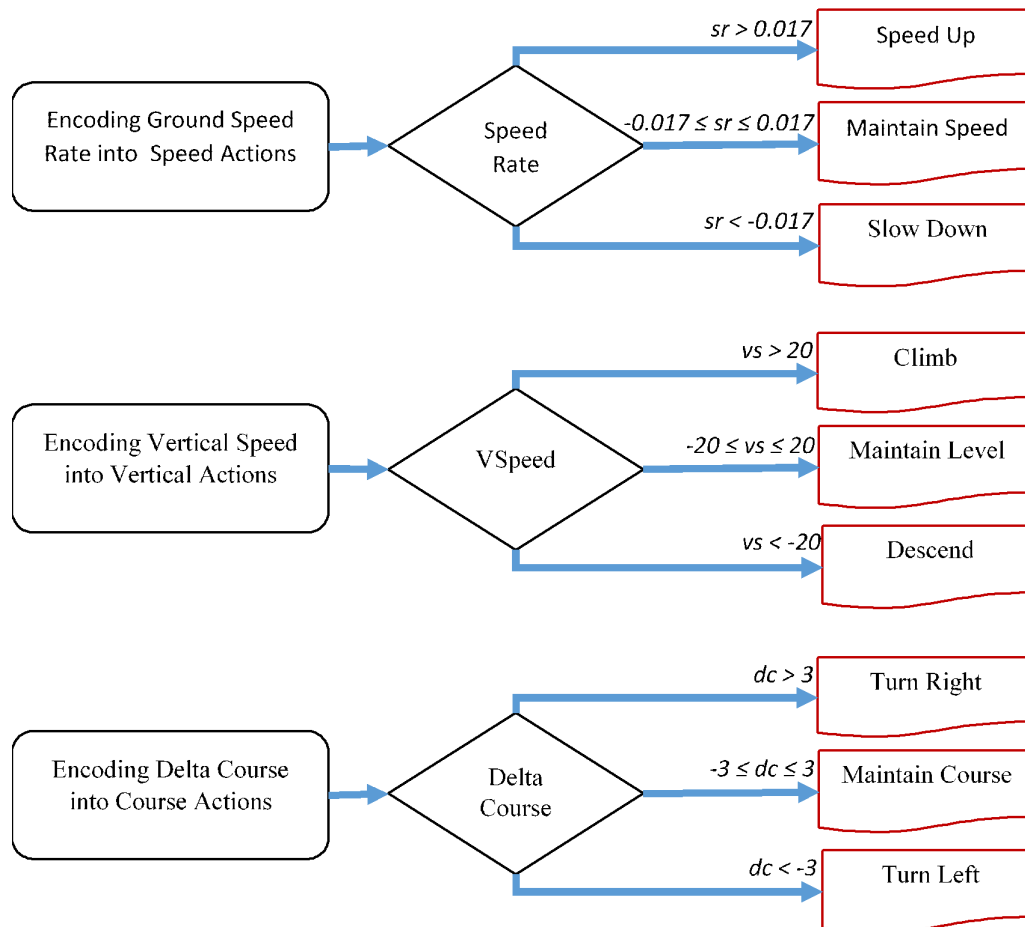


Fig. 2.7 Flow chart of Encoding Actions

2.5 Predictive Models

In this section, we will describe our approach for predicting the next actions of flight after entering the sector. The training data includes features of flight at entry point and the extracted actions from real data as the targets. Table 2.3 mentions list of features and all targets. We propose Random Forest Method [24] for building predictive models for ATCO actions.

2.5.1 Random Forest Method

Random Forest Method is an ensemble learning method for classification and regression. It constructs multiple decision trees with different subsets of features and samples for each tree. The tree learns different knowledge and then votes for final prediction. Random Forest is used in different domains and predictive problems as it provides high accuracy with simple implementation. It is highly robust since it can deal with outliers/noises without skewing the prediction results and avoids over-fitting due to the diversity of trees. One of the key advantages of Random Forest method that suits our problem is its capability to handle

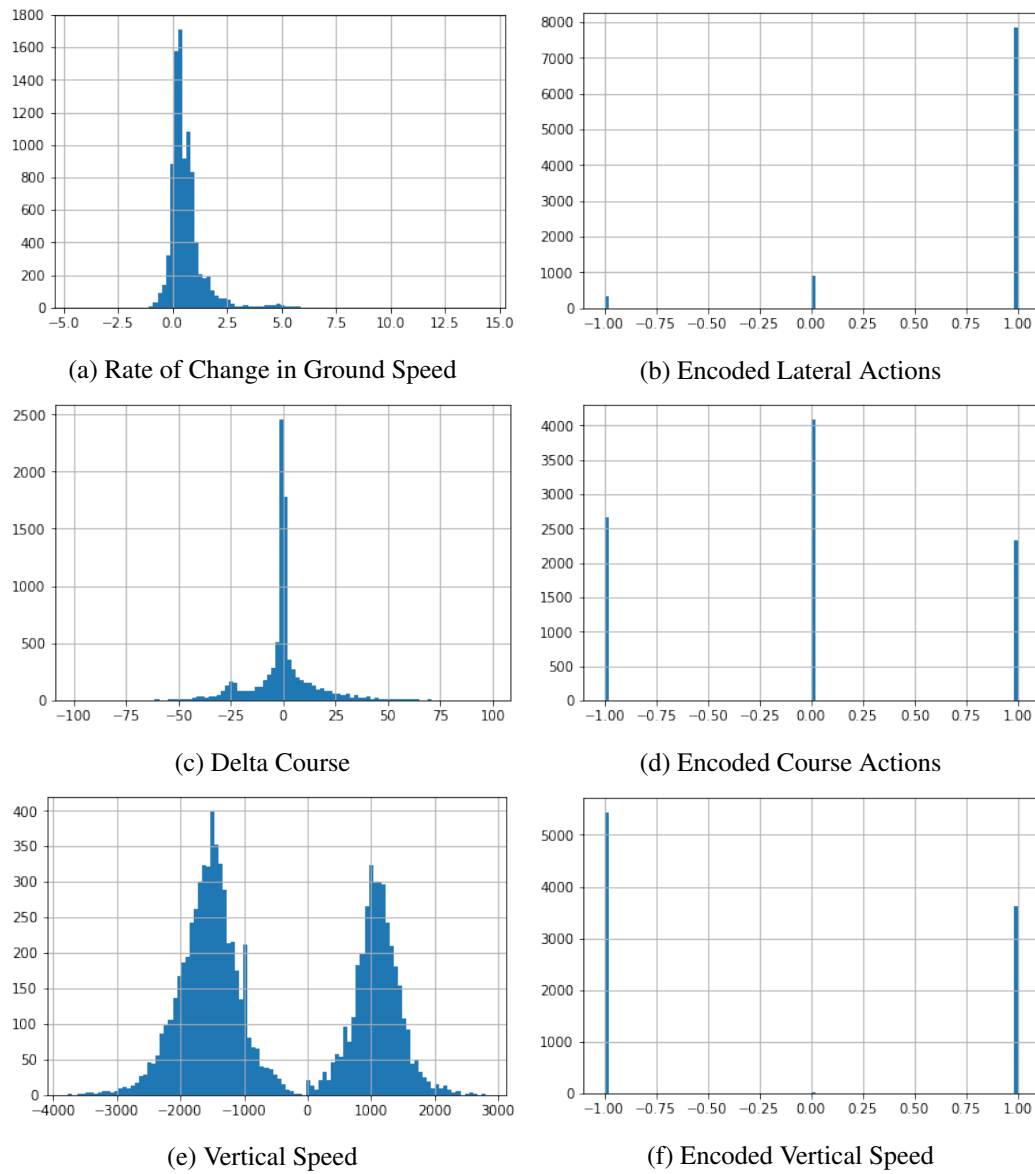


Fig. 2.8 (a), (c), (e) are distributions of computed trajectories' changes (actions) from ADS-B data. (b), (d) and (f) are distributions of encoded actions.

Features	Action Values	Actions
Longitude	Δ Ground Speed	(1)Speed Up
Latitude	Δ Vertical Speed	(0) Maintain Speed
Altitude	Δ Course	(-1) Slow Down
Ground Speed		(1) Cimb
Vertical Speed		(0) Maintain Level
Course		(-1) Descent
		(1) Turn Right
		(0) Maintain Course
		(-1) Turn Left

Table 2.3 Flight information (features) at Entry point and actions, actions' values

unbalanced data-sets and able to work with different types of features and range of feature values

Further, the 4D trajectories are derived from ADS-B data usually contains noisy data points and the input features has different meaning and scale. Moreover, interpret-ability of the model is also considered for understanding the important factors for predicting actions, thus Random Forest is found suitable for this purpose.

2.5.2 Building Predictive Models

We use Random Forest to build two groups of models:

1. Random Forest Regression Models:

- Model to predict Rate of Change in Ground Speed.
- Model to predict Vertical Speed.
- Model to predict Delta Course.

We use R^2 score as performance metric for this group of models.

2. Random Forrest Classification Models:

- Model to predict Speed Actions.
- Model to predict Vertical Actions.
- Model to predict Course Actions.
- Model to predict all 3-Actions.

We use *Accuracy* as performance metric for this group of models.

2.6 Experiments and Results

For both groups of models, we apply the same experiment setup:

- Using flight information at the entry point as the input for predictive model and the targets are extracted actions (mentioned in Table 2.3).
- Parameter tuning: number of estimators runs from 50 to 300 while max_depth of forest varies from 4 to 20.
- 10-fold cross-validation is used for evaluating for each set of parameters. The performance is averaged to select best set of parameters.

Experimenting result for parameter tuning of regression models is shown in Fig. 2.9 that depicts that maximum number of tree = 300 and maximum of tree depth = 20 are sufficient for tuning parameters. The result indicates that the number of estimators (≥ 50) does not affect performance of models, instead, max_depth plays a more important role. $Max_Depth \geq 8$ makes the model stable and the best value is $Max_Depth = 10$ for all three models. Thus, all Random Forest Regression Models are used with [$Estimator = 50$, $Max_Depth = 10$].

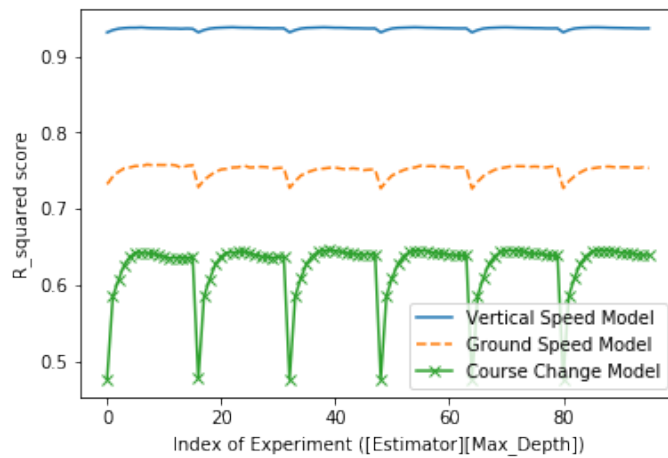


Fig. 2.9 Illustration for parameter tuning process for regression models. Estimators are [50:300, step 50] and Max_Depth are [4:20, step 1], the experiment index is generated in the same order (ascending)

Table 2.4 presents the regression performance of Random Forest for each kind of actions. Note that R^2 score is best at 1 and worst at 0. All three models perform well to predict next action values for each given flight with the worst R^2 score is 0.636 while predicting Delta Course. Delta Course is hard to predict due to the nature of this sector. As we can observe from Fig. 2.10. The group of black dots in the bottom of the figure are closed to each other while their corresponding red dots are spatially diverse and that also the nature of the airspace of this sector (Fig. 2.2b).

Predictive Action Value	R^2 score
Δ Ground Speed	0.752
Δ Vertical Speed	0.936
Δ Course	0.636

Table 2.4 Experiment result for predicting 3 group of actions

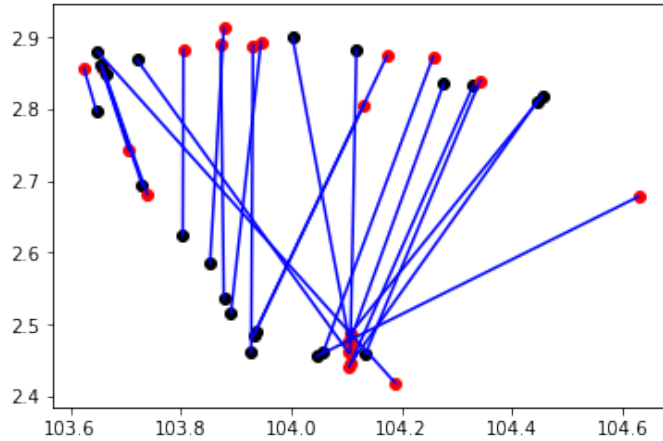


Fig. 2.10 Example for Flight Direction: Black dots are entry points and Red dots are exit points.

The parameter tuning for classification models (detailed in Fig. 2.11) has the similar characteristics as regression models. Thus, the selected parameter set is [$Estimator = 50$, $Max_Depth = 10$] for all classification models.

Table 2.5 presents the performances of classification models. Four models are used to predict flight actions. Results indicate that the model for vertical maneuver actions provides highest prediction accuracy (99.7%). Besides, model for speed change and heading change action provides predictability accuracy of 88.7% and 72.4% respectively. The model to predict the set of all the actions (altitude, speed and heading change) for each flight achieves an accuracy of 0.68 implying for 68% of flights, D-Side Controller's can be predicted for all the actions from trajectory information at sector entry position.

Predictive Action	Accuracy
Vertical Change [-1, 0, 1]	0.997
Speed Change [-1, 0, 1]	0.887
Course Change [-1, 0, 1]	0.724
All 3-actions	0.680

Table 2.5 Predictive accuracy for different models

As the benefit of Random Forest Method, we also have the feature-importance information for each model. An example can be seen in Table 2.6. The information on important features of each model can provide explanation on the percentage contribution of each

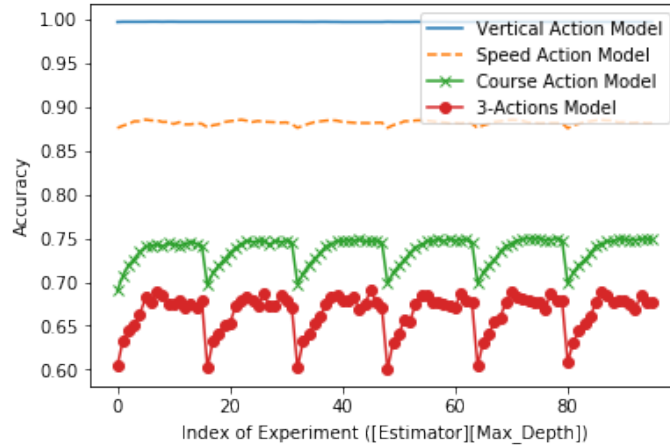


Fig. 2.11 Illustration for parameter tuning process for classification models. Estimators are [50:300, step 50] and Max_Depth are [4:20, step 1], the experiment index is generated in the same order (ascending)

feature on predictability. This information can be used to improve further exploration in learning controller actions.

Features	% contribution
VSpeed	14.1
Course	19.5
Latitude	5.9
Longitude	3.1
Altitude	35.4
Speed	22.1

Table 2.6 Feature Importance in 3-Actions Random Forest Model

2.7 Discussion

2.7.1 Extracted Actions as Macro Actions

The performance of predictive models confirms the relation between entry and exit points. However, the extracted actions or pattern is simple and abstract comparing to practical controllers' actions and behaviors. Thus, we can treat those defined action as the macro actions which is usually applied in studying strategy game [47]. Later, for each macro action, we can break it down into sequence of actions or commands. In our case, those actions can extracted from trajectory data by clustering similar patterns of deviations with some preprocessing and standardizing steps. Fig. 2.12 and 2.13 show some examples of potential patterns which can be extracted from data for heading change and vertical profile. However, in practical applications, we can also define manually the sequence of micro actions for each predicted macro action.

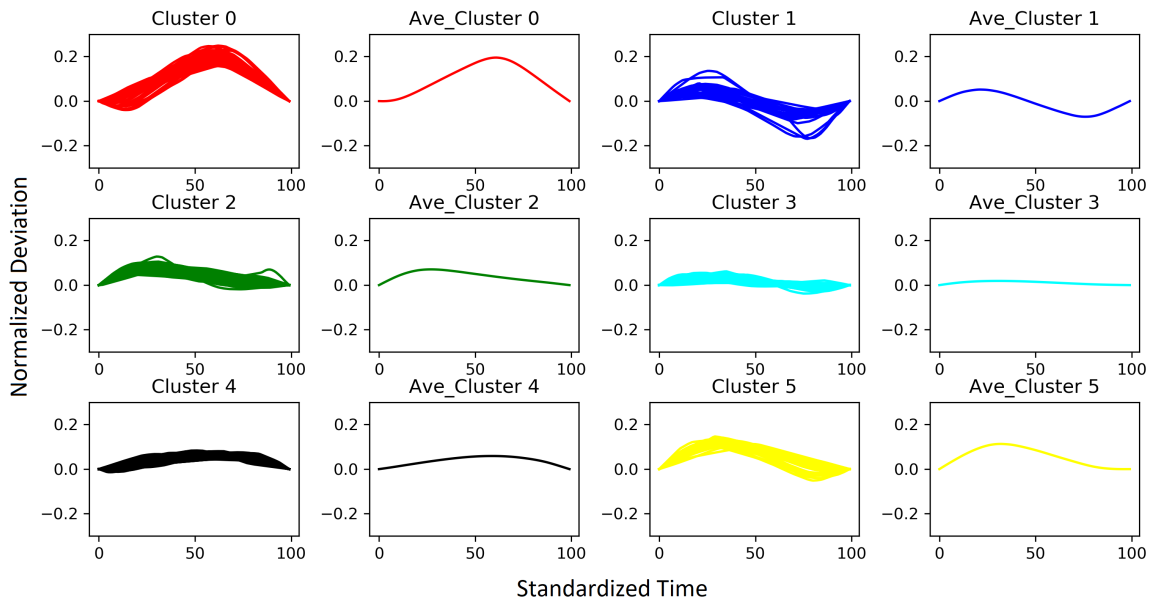


Fig. 2.12 Clustering deviations of heading changes overtime as lateral actions

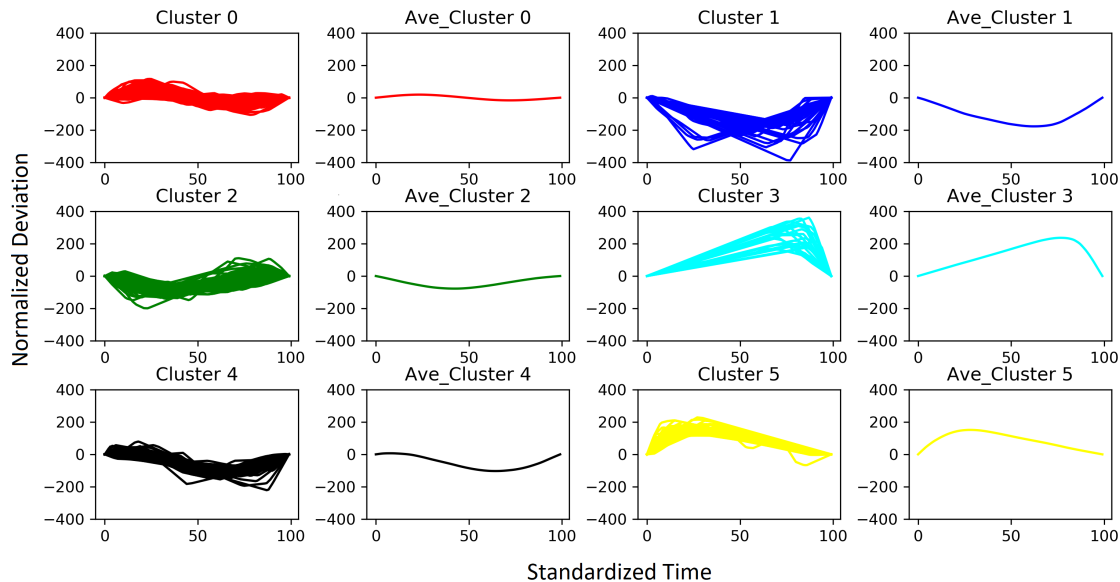


Fig. 2.13 Clustering vertical profile overtime as vertical actions

2.7.2 Grouping Trajectory to form Flight Routes

There are different ways to group or cluster trajectories in sectors since entry and exit points are distributed all over the boundary. From our experiment, given similar entry information, we can achieve similar 4D exit points. The different movements between entry and exit points are detail commands of controllers or requests of pilots in response to traffic or weather. Thus, for en-route sectors, we propose grouping flight based on their entry and exit information to form flight routes, Fig. 2.14. The deviation of each group can be considered as the uncertainty for aircraft position. The result of this step will be used in Chapter 3 for learning and evaluating generative models.

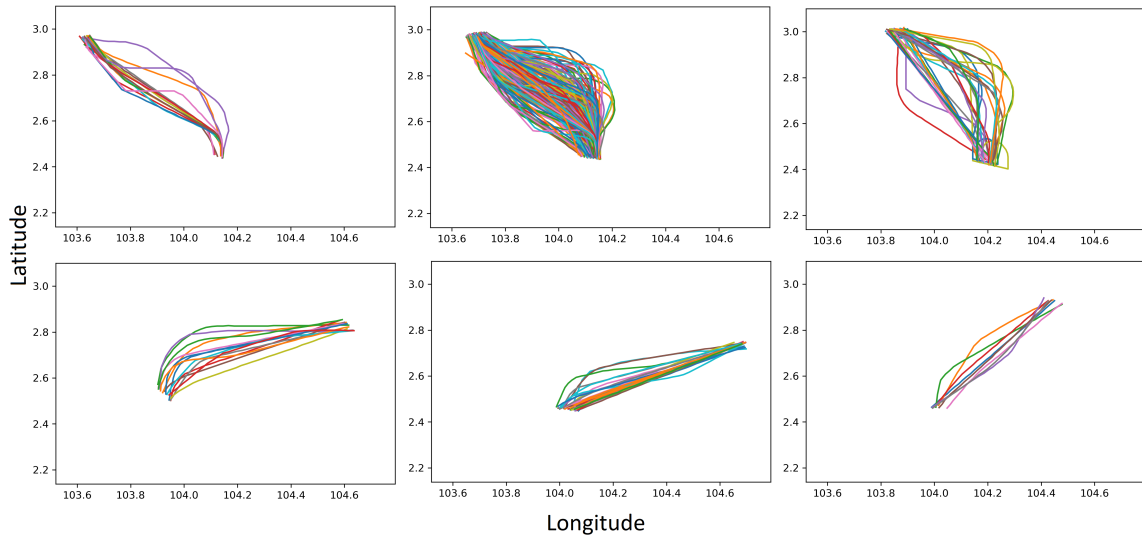


Fig. 2.14 Examples of grouping trajectories based on entry and exit points

2.8 Conclusions

In this chapter we have looked into learning and predicting the D-side controller's action for a given traffic scenario in a sector using a tree-based regression and classification method known as Random Forest. This learning problem was modeled as a classification problem where the target variable is D-side controller actions and the explanatory variables are the aircraft 4D trajectory features prior to entering a sector. The air traffic trajectories constructed through ADS-B data are analyzed spatial-temporal with sector data to establish that patterns in D-side controllers exists. Two group of models were developed, one to predict the actions and the other to predict the associate action value. We used flight information at the sector entry point as the input for predictive model and the targets are extracted actions. The model for vertical action provided the highest accuracy with 99.7% whereas, model for speed change and heading change action provides predictability accuracy of 88.7% and 72.4% respectively. This was attributed to highly complex sector entry and exit point configuration that makes learning challenging. The model that predicts the set of all 3 kinds of actions (multiple-output) for each flight, achieved an accuracy of 0.68. This means that, for 68% flights, ATCO actions can be predicted on all actions by using flight information at sector entry position. The lower predictability can be due to over-fitting of the training data for controller's actions, leading to poor generalization performance. The noise and low quality of ADS-B data can be another area of improvement as the model is as good as the data used to train it. Besides, as mentioned, this chapter only focus on predicting simple actions from individual flight entry information. Those actions can be treated as macro actions which are able to be further explored and detailed in practical applications.

This chapter alone is helpful for understanding and investigating the nature and patterns in a given sector. Besides, results from this chapter can be used to explore Air Traffic

Controller's strategies in conflict resolution. However, it also can be considered as the preprocessing step to support building dataset for training and evaluating AI-Agent in conflict detection and resolution (CD&R) which will be discussed in following chapters.

Publication related to this chapter is:

- "A Machine Learning Approach on Past ADS-B Data to Predict Planning Controller's Actions". *8th International Conference on Research in Air Transportation (ICRAT '18)*. 2018. Barcelona, Spain [60].

Chapter 3

Data-driven approaches for Trajectory Prediction and Conflict Detection

Conflict detection is one of the fundamental elements in air traffic control. Its output is the essential input for conflict resolution algorithm that is mandatory for maintaining safety and efficiency for flights. Even though there are tools to support air traffic controller in detect potential conflict, their quality and accuracy are still limited since the difficulty in encapsulating uncertainty in predicting flight trajectory. To tackle that challenge, several studies focus on applying probabilistic approaches by modeling aircraft dynamic and uncertainty. They share common limitations such as their assumption about distribution of uncertainty and high computational cost for locating and computing the probability of conflict. In this chapter, we propose a machine learning approach to model the uncertainty of flight trajectory combining with the technique for quickly locating position with highest conflict probability. Firstly, we use heteroscedastic Gaussian Process to capture the complex patterns and their uncertainty directly from trajectory data. Those models are then tuning with reference route of aircraft to predict aircraft position in future with uncertainty. For detecting the time stamp with highest conflict probability, Bayesian Optimization algorithm is applied. Our proposed approach for predictive models can capture the heteroscedastic noise from data. This method can be used for modeling trajectories in any phase with high performance. Besides, Bayesian Optimization method also shows its potential for probabilistic conflict detection in term of computational cost and flexibility. There are two studies can be extended from this work is generative models with multi-output and Bayesian Optimization method for multiple conflicts. The result of this study can be applied in flight traffic simulator or learning environment for conflict resolution.

3.1 Introduction

There are three groups of conflict detection algorithm reported in [43]: deterministic, worst case and probabilistic. In which, probabilistic approaches gain more attention because it is more practical by considering uncertainty in detection. Some studies focus on empirical distributions of future aircraft positions [43, 58, 81] or model the dynamic of aircraft [34, 65] with modeled uncertainty for conflict detection. Given defined uncertainty models, Monte Carlo is usually used to simulate conflict event [80, 6]. Those approaches share common limitations such as their assumptions about uncertainty distribution and high computational cost for estimating conflict probability. Because of increasing of available trajectory data (i.e. ADS-B), those uncertainties can be learned and approximated directly from data instead of predefined distributions, formula or dynamic models of aircraft. Furthermore, since we can build generative models for flight position prediction using data, new approaches like Bayesian Optimization can be investigated for probabilistic conflict detection.

In this chapter, we use Gaussian Process (GP) to model the uncertainty of flight trajectory combining with Bayesian Optimization (BayesOpt) for quickly locating position with highest conflict probability. Applying Gaussian process for learning trajectory prediction is inspired from [20]. Firstly, we use Gaussian Process as a generative model to capture the complex trajectory pattern and its uncertainty directly from trajectories. The generative model must consider two challenges: multi-output and multi-variance of data. Our problem is multi-output since the position of aircraft is 3D (Longitude, Latitude and Altitude). To handle it, we train three GPs, one for each dimension. Thus, our generative model is, in fact, a set of three GP models. There are two types for GP will be investigated which are homoscedastic and heteroscedastic noise models. Homoscedastic GP can be considered classical method where the variance of data is consistence. In our case, the variations in longitude, latitude or altitude of aircraft positions will vary along time dimension. The heteroscedastic GP [44] is expected to capture the variance of real data. After we obtain generative model, current flight information (conditional points) such as current position and reference route is provided to the model for training trajectory prediction model with uncertainty. The uncertainty is obtained by conditioning data variation on conditional points using a second Gaussian Process. Secondly, we apply Bayesian optimization algorithm for probabilistic conflict detection problem. We aim to obtain similar result as Closest-Point-of-Approach (CPA) but in probabilistic manner by identifying the time at which two aircraft have highest probability of conflict, called Probabilistic Closest-Point-of-Approach (P-CPA). Since conflict is a rare event so that the cost of estimating conflict probability will be expensive. Our proposal method is faster and more flexible in finding P-CPA comparing to grid-search approach.

3.2 Approach

The proposed approach (shown in Fig. 3.1) includes two parts which need to be discussed: Predictive model and Bayesian Optimization model.

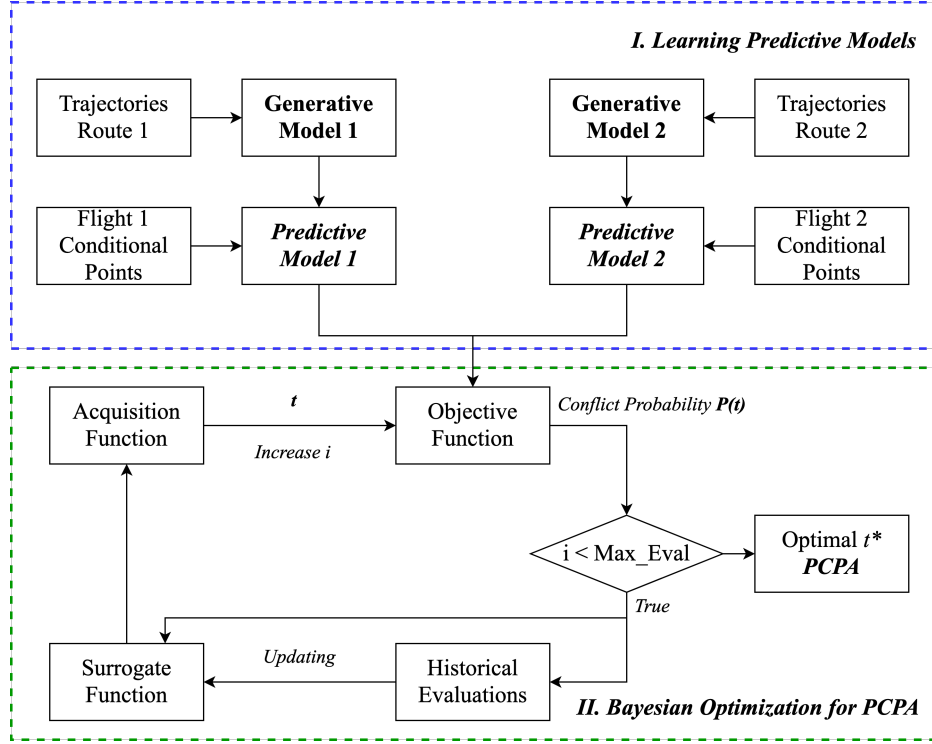


Fig. 3.1 Illustrating approach for probabilistic conflict detection using Bayesian Optimization.

First of all, trajectories which follow similar route will be grouped or clustered based on their entry and exit positions in the sector or airspace. Each group of trajectories will represent a flight route with variation where all flights sharing begin and end points but different in between due to uncertainty from environment. Gaussian Process models are used to learn the variance of trajectories data. As mentioned, the trained generative model is a set of three Gaussian Processes. Then to perform the prediction for incoming flights, some flight reference positions will be provided: recently positions and future reference positions as the conditional points. Then, based on given conditional points, we filter our dataset to achieve a subset of trajectories which pass through all conditional points. Finally, we train our GP predictive models using filtered trajectories and learned flight position uncertainty.

For conflict detection, We only consider the conflict of two aircraft, thus two predictive models will be trained and used for estimating conflict probability. We construct this conflict detection problem as an optimization problem where the objective function is the conflict probability and our task is finding time stamp t to maximize it. Bayesian Optimization algorithm contains four main components which will be discussed in section 3.4. In general, given a time stamp t , we can estimate conflict probability $P(t)$ using Monte Carlo (sampling)

method. Then the result will be stored to memory and also used to update the probabilistic model, called surrogate function which is much more easier to investigate than original objective function. Acquisition function will use updated surrogate function to make the prediction on the next value of t for evaluation. The searching process will stop when the iteration (i) reaches the maximum number of evaluations (Max_Eval).

3.3 Predictive model for flight trajectory with learned uncertainty

Recently, some machine learning models has been used for trajectory prediction such as hidden Markov model or random forest. However, when considering predictive model with uncertainty, Gaussian process is a promising approach because it could model the dispersion of a trajectory. Because we are working with three dimensional positions over time for trajectories, three Gaussian processes are trained with time as input and latitude, longitude and altitude as output. Gaussian process is a collection of random variables, any finite number of which have (consistent) joint Gaussian distributions [73]. A Gaussian process $f \sim \mathcal{G}(m(t), k(t, t'))$ is fully specified by its mean function $m(t)$ and covariance function $k(t, t')$ which need to be symmetric and satisfy Mercer's condition.

3.3.1 Input independent noise - homoscedastic noise

We consider a data of single trajectory with K realizations. For each realization i we will have N_i input vectors t paired with outputs y . The training data $D_N = \{(t_j, y_j)\}_{j=1..N}$ with $N = \sum_{i=1}^K N_i$.

Gaussian process regression is a machine learning technique for inferring likely values of y for an input t as $y(t) = f(t) + \varepsilon$ where f is a Gaussian process and $\varepsilon \sim N(0, \sigma^2)$ is a input noise, where σ is a positive constant.

For the generative trajectory model, we train three one-dimensional models with t is the time and y is either longitude, latitude or altitude of the aircraft at time t . In this manner, we could specify the noise variance for each dimension independently. However, one drawback of this approach is that it does not model the correlation among the noises of the three dimensions.

3.3.2 Input dependent noise - heteroscedastic noise

Instead of inferring y for an input t as $y(t) = f(t) + \varepsilon$ where f is a Gaussian process and $\varepsilon \sim N(0, \sigma^2)$ with σ is a positive constant, we consider the model

$$\begin{aligned} y(t) &= f(t) + \varepsilon(t), \\ \varepsilon(t) &\sim N(0, \sigma(t)), \end{aligned}$$

because it can capture the different noise levels in trajectories and we use kernel regression to fit the noise-level function $\sigma(t)$.

3.3.3 Predictive model for individual airplane

After fitting a GP model with data and obtained mean function m^* and kernel k^* , we use this model to sample K next locations of the aircraft $\hat{D}_K = (t_i, \hat{y}_i)_{i=1}^K = (T^K, \hat{y}^K)$ after observed its M previous locations $D_M = (t_i, y_i)_{i=1}^M = (T^M, y^M)$. This could be done by using the posterior distribution

$$\begin{aligned} \hat{y}^K | T^K, T^M, y^M &\sim \mathcal{N}(k^*(T^K, T^M)(k^*(T^M, T^M) + \sigma(T^K)I)^{-1}y^M, \\ &k^*(T^K, T^K) + \sigma(T^K)I - k^*(T^K, T^M)(k^*(T^M, T^M) + \sigma(T^K)I)^{-1}k^*(T^K, T^M)). \end{aligned} \quad (3.1)$$

Equation (3.1) could also be used to sample trajectories with given conditional points by setting $D_M = (T^M, y^M)$ as the set of conditional points. Each of those points is either the current points, the reference points for flight routes, or the target points. Instead of providing the model with the current positions only, we consider scenarios where flights have information of the reference routes from flights plan or from air traffic controllers. This setting can be used in conflict resolution where each maneuver can be considered as a reference flight route or set of conditional points and input into our model.

Algorithm 2: Algorithm for Predictive Model

Input: $Traj_{GID}, N_T, NP_{MAX}, N_{CP}, \Delta$

Result: GP_{Tf}

```

1 if  $Traj_{GID} \geq N_T$  then
2    $Traj_{samples} \subseteq Traj_{GID}$  and  $|Traj_{samples}| \leq NP_{MAX}$ 
3    $GP_{Ts} = \text{Train\_Gaussian\_Process}(Traj_{samples})$ 
4    $CP(X, Y, Z, Time) = \text{Initializing\_conditional\_points}(N_{CP})$ 
5    $V = \text{Estimating\_Variance}(GP_{Ts}, CP.Time)$ 
6    $Normalized\_V = V / \min(V)$ 
7    $Radius_{Filtered} = (Normalized\_V[i] * \Delta[i])_{i=1}^{N_{CP}}$ 
8    $Traj_{Filtered} = \text{Filter\_Conditional\_Trajectories}(Traj_{GID}, CP, Radius_{Filtered})$ 
9    $GP_{Tf} = \text{Train\_Gaussian\_Process}(Traj_{Filtered}, \text{kernels} = GP_{Ts}.kernels)$ 
10 end

```

Algorithm 2 for training predictive model requires processed trajectories for each flight route (clustered trajectories), minimum number of trajectories (N_T), maximum number of data point for subset of trajectories (NP_{MAX}), number of conditional points (N_{CP}) and initial radius value of each dimension for filtering trajectories with conditional points (Δ). In which, N_T is used to guarantee the sufficient of input data for training, however, due to the limitation of computational power, NP_{MAX} can be adjusted to generate smaller subset of data for training the three Gaussian processes (high complexity $O(n^3)$). Moreover, Δ is important for building training data for predictive model. It will be multiplied with normalized learned variance (*Normalized_V*) to compute filtering region (within radius *Radius_Filtered*) for each conditional point. Only trajectories which pass through all filtering regions of conditional points remain in the new data set which then is used to train predictive models. Those input parameters can be modified depending on particular application. We use GaussianProcessRegression from SKlearn for learning. In case of heteroscedastic noise, a heteroscedastic kernel [37] is used.

3.4 Probabilistic Conflict Detection using Bayesian Optimization

3.4.1 Bayesian Optimization

Bayesian Optimization is a class of machine-learning-based optimization methods that focuses on solving the problem

$$\max_{t \in A} F(t),$$

where A is the feasible set and F is a "black-box" objective function or costly to be evaluated.

In contrast to GS, BayesOpt algorithm keeps track of the past evaluation results to update a probabilistic model which maps the given parameters to probability of a score s from objective function $F(t)$. The probabilistic model $P(s|t)$ is called "surrogate" function in which t is a input parameter, and s is a predicted score. Five main steps in Bayesian optimization:

1. Building surrogate probability model of objective function: The surrogate function is the probability representation of the objective function. There are several different forms of the surrogate function including: Gaussian processes, Random Forest regression, and Tree-structured Parzen Estimator [9] which is used in this study. Instead of directly representating $P(s|t)$, the Tree-structured Parzen Estimator (TPE) applies Bayes rule to get surrogate probability $P(s|t)$:

$$P(s|t) = \frac{P(t|s)P(s)}{P(t)}$$

Where $P(t|s)$ is the probability of the parameters x given the score y of the objective function and is defined as:

$$P(t|s) = \begin{cases} l(t) & \text{if } s < s^* \\ g(t) & \text{if } s \geq s^* \end{cases}$$

where s^* is the threshold value of the objective function, $l(t)$ and $g(t)$ are two Gaussian Mixture Models (GMM) which are fitted using evaluated values associated with objective function values.

2. From surrogate function, find the next parameters to be evaluated using Acquisition Function. Given the surrogate function, the acquisition function $A_{EI}(t)$ is the criteria to find the next parameters. This study uses the most common choice of criteria, which is the expected improvement:

$$A_{EI}(t) = EI_{s^*}(t) = \int_{-\infty}^{\infty} s^*(s^* - s)P(s|t)ds$$

where t is the proposed parameter, s is the actual value of objective function using parameter t . The aim of this method is to maximize the Expected Improvement with respect to parameter t or finding the best parameters t under the surrogate function $P(s|t)$. If $EI_{s^*}(t) > 0$, the parameters x are expected to yield a better result than the threshold value.

3. Apply these parameters to the original objective function.
4. Update the surrogate model incorporating the new results.
5. Repeat steps 2-4 until maximum iterations or time is reached.

3.4.2 Probabilistic Closest Point of Approach - PCPA

We model the uncertainty in two trajectories by two set of Gaussian processes f^1 and f^2 which are fitted to data, with the format $f(t) = (GP_{longitude}(t), GP_{latitude}(t), GP_{altitude}(t))$. Our purpose is to find the time in which two aircraft have highest probability of conflict:

$$\bar{t} = \operatorname{argmax}_{t \in T} P(t), \quad (3.2)$$

where $P(t) = \mathbb{P}(\|f_{1,2}^1(t) - f_{1,2}^2(t)\| \leq s_h, |f_3^1(t) - f_3^2(t)| \leq s_v)$, with $f_{1,2}(t) = (GP_{longitude}(t), GP_{latitude}(t))$, and $f_3(t) = GP_{altitude}(t)$. In this paper we set $s_h = 5NM$ and $s_v = 1000ft$.

Taking the advantage of Gaussian process, we can approximately compute $P(t)$ by Monte Carlo simulation:

$$P(t) \approx \frac{1}{N_s} \sum_{i=1}^{N_s} I_{A_i(t)},$$

where N_s is the number of MC runs, $A_i(t)$ is the event $\{\|f_{1,2}^{1,i}(t) - f_{1,2}^{2,i}(t)\| < s_h, |f_3^{1,i}(t) - f_3^{2,i}(t)| < s_v\}$, $\{f_{1,2}^{1,i}(t)\}_{i=1,\dots,N_s} \stackrel{\text{i.i.d}}{\sim} \mathbb{P}_{f_{1,2}^1(t)}$, $\{f_{1,2}^{2,i}(t)\}_{i=1,\dots,N_s} \stackrel{\text{i.i.d}}{\sim} \mathbb{P}_{f_{1,2}^2(t)}$ and $\{f_3^{1,i}(t)\}_{i=1,\dots,N_s} \stackrel{\text{i.i.d}}{\sim} \mathbb{P}_{f_3^1(t)}$, $\{f_3^{2,i}(t)\}_{i=1,\dots,N_s} \stackrel{\text{i.i.d}}{\sim} \mathbb{P}_{f_3^2(t)}$.

Because conflict of two trajectories is a rare event, we usually need a large number N_s of simulation runs to estimate the conflict probability at each time t . Also, we need to find the maximum value of $P(t)$ with respect to t (3.2); therefore, the computational load is very high. To deal with this, we apply BayesOpt technique by choosing Tree Parzen Estimators (TPE) for surrogate probability model, and use Expected Improvement for selection function.

The estimations of the conflict possibility of two flights (with uncertainty) are described in algorithm 3 using Grid-Search (GS) and in algorithm 4 using Bayesian optimization (BayesOpt). In both algorithms, at a given timestamp t , we perform the sampling with size N_s for both flights: $[P_{1..N_s}^1]$, $[P_{1..N_s}^2]$. Then, for each pair, the distance between two aircraft $Dist(P_k^1, P_k^2)$ is compared with the minimum separation allowed (constraints). The conflict probability is computed directly from number of pairs which violating the constraints over N_s . The output of those algorithms are the maximum conflict probability (*CProb*), the time to probabilistic closest point of approach (*Time - to - PCPA*), and the computational cost (*Cost*). The difference between these two algorithms is how they find the value of t for evaluation. GS algorithm evaluates all values of t in a given range. The time interval (R) is given; therefore, only a predefined set of values t is evaluated. In the other hand, BayesOpt determines a sequence of values t depending on historical evaluations and the value of t could be continuous. By employing BayesOpt, the sample size is reduced and consequently lower computational cost is required.

Algorithm 3: Grid-Search Method

Input: N , R , GP1_2, GP2_2, Duration, Threshold

Result: CProb, Time-to-PCPA, Cost

```

1 Start_time = time.time()
2 Prob_T = []
3 for t ← 0 to Duration by R do
4   Traj_1 = GP1_2.sample_y(t, N_s)
5   Traj_2 = GP2_2.sample_y(t, N_s)
6   Dist = Distance(Traj_1, Traj_2)
7   Prob_t = Conflict_Prob(Dist, Threshold)
8   Prob_T.append(Prob_t)
9 end
10 CProb = max(Prob_T)
11 Time-to-PCPA = argmax(Prob_T)
12 Cost = time.time() - Start_time

```

The *hyperopt* Python package is used for BayesOpt algorithm. Since we want to avoid any assumption about the search space for t , uniform distribution is used to sample t . It is

worthy to note that the used package tries to minimize our defined objective function while our objective is to maximize conflict probability. Thus, in the objective function, the returned probability is multiplied with negative one. In BayesOpt, there are two terminating conditions: the convergence of maximum objective values or maximum number of evaluations (*max_evals*) is reached. In this study, we use *max_evals* as the terminating criterion.

Algorithm 4: Bayesian Optimization Method

Input: N, GP1_2, GP2_2, Duration, Threshold

Result: CProb, Time-to-PCPA, Cost

```

1 def objective(t):
2     Traj_1 = GP1_2.sample_y(t, N)
3     Traj_2 = GP2_2.sample_y(t, N)
4     Dist = Distance(Traj_1, Traj_2)
5     Prob_t = Conflict_Prob(Dist, Threshold)
6     return -Prob_t
7 end
8 space = hp.uniform('t', 0, Duration)
9 algo = tpe.suggest #Tree Parzen Estimators
10 trials = Trials()
11 Start_time = time.time()
12 Time-to-PCPA = hp.fmin(objective, space, algo, trials, max_evals)
13 CProb = get_max_prob(trials, Time-to-PCPA)
14 Cost = time.time() - Start_time

```

3.5 Experiment Setup

3.5.1 Trajectory data for learning and evaluation

In this study, we will assess the performance of generative model using Gaussian process with ADS-B data. However, simple toy sample data is also designed and used for conflict detection since it is easier to manipulate for trajectory manipulating and conflict generating.

Toy Sample Data

Toy sample is designed to evaluate the performance of Bayesian Optimization and highlight its advantages comparing to Monte Carlo method. We consider crossing scenario with given way-points (Table 3.1) and flight routes (Table 3.2) of airspace. Each flight route includes three way-points and sharing the middle way-point (crossing point). A sample dataset is generated from this airspace network by simulating flight trajectories with noise. To simplify this toy samples, we setup simple assumptions:

- Applying white noise with different deviations (Lateral noise = [$\sigma_{L1} = 1(NM)$, $\sigma_{L2} = 3(NM)$, $\sigma_{L3} = 2(NM)$] and Vertical noise = [$\sigma_{V1} = 0.1(FL)$, $\sigma_{V2} = 0.5(FL)$, $\sigma_{V3} = 0.5(FL)$]) at given way-points to generate reference points for each flight route.
- Interpolating sample points between reference points to complete trajectories.

Table 3.1 List of Way-points

Way-point ID	X(nm)	Y(nm)	Z(FL)
1	30	10	400
2	55	50	400
3	110	100	400
4	10	90	400
5	100	50	400

ADS-B Data:Extracting Flight Patterns given Entry and Exit Points

The raw ADS-B trajectories are pre-processed for removing noises, and interpolating. Then, to apply Gaussian Process as generative model, we need to perform clustering over trajectories and only select groups which have sufficient data samples. The trajectories are clustered base on their entering and exiting points on the given sector, because, from our observations, there is pattern of trajectories given those two points. The selected groups are used as test cases to evaluate our proposed approach.

3.5.2 Exp 1:Evaluating GP-Generative model

In this study, the main purpose of the generative model is learning the uncertainty from data and then use it as the input for probabilistic conflict detection algorithm (e.g. Bayesian Optimization, Monte Carlo, etc). Training online prediction model includes two steps:

- Training Gaussian Processes with trajectories In this step, we train one Gaussian Process for each data dimension (3 in total). Besides, two different models of Gaussian Process are evaluated: homoscedastic and heteroscedastic models.
- The prediction Gaussian Process models use the trained kernel of previous models as initial kernels, and conditional points as input samples. The Algorithm 2 is used with parameters: [$N_T = 10, NP_{MAX} = 500, N_{CP} = 4, \Delta = [1NM, 1NM, 0.5FL]$]. For toy

Table 3.2 List of sample routes

Default Route ID	List of Way-points	Arriving Time Stamp(s)
1	[1,2,3]	[0, 510, 1200]
2	[4,2,5]	[0, 600, 120]

sample, we use the given way-points in routes as the conditional points for training the prediction model. The same process is also applied for real ADS-B data to illustrate the performance of Gaussian Processes as predictive model.

3.5.3 Exp 2: Bayesian optimization vs Monte Carlo

The experiments for BayesOpt and Monte Carlo (grid-search) are performed for conflict detection. Both algorithms are evaluated with sampling size \mathbf{N} from $[500K, 1000K, 2000K]$ and Monte Carlo also use time interval \mathbf{R} between data points or fine-level of time from $[1s, 5s, 10s, 30s, 60s]$. Three values of stopping condition Max_Eval are considered and reported $[50, 100, 200]$.

3.6 Result and Discussion

3.6.1 Result for Exp 1:

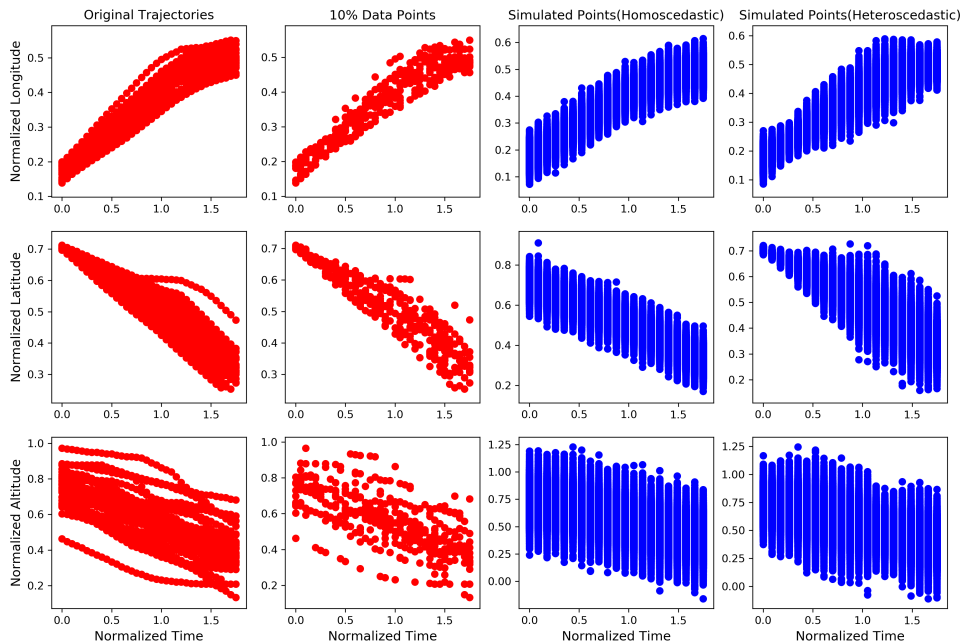


Fig. 3.2 Example of Learning ADS-B Trajectory's Variances

Fig. 3.2 illustrates the results of generative models for three dimensions (Longitude, Latitude and Altitude) for one trajectory route (group). From left two right, the raw data represents the distribution of values over time. Following algorithm 2 we will perform sampling step (i.e. keeping 10% of data ≈ 500 data points). This can speed up the training

of Gaussian Process and also similar to the case where the number of data points is limited for less traffic routes. Two columns with blue scatter plots on the right hand side show the learning results. As we can see, the Gaussian Process with homoscedatic kernel will provide a consistence in variance over time for three dimensions while ones with heteroscedastic kernel replicate the distribution of raw data very well. Additionally, we can observe the bigger dispersion in Latitude dimension than the others which also reflect in the differences between the shapes of simulated data for both types of models. The simulated Latitude values of heteroscedastic model has very small variance at begin with significant increase over time.

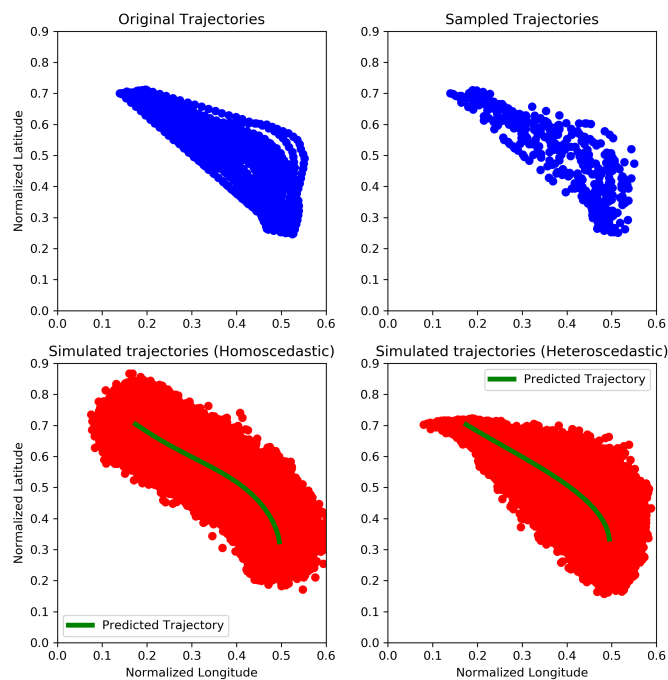


Fig. 3.3 Learning ADS-B Trajectory Variation

Using three models to handle multi-output may lead to issue with correlations between different dimensions. However, Fig. 3.3 shows our prediction result for trajectory in 2 dimensions (Longitude and Latitude). In which, the mean predicted trajectories (green lines) of both models are smooth and fit to the trajectory data. The results show that combining predictive values from our trained models still can model the distribution of multi-dimensional trajectory data. Moreover, model with heteroscedastic kernel provide a good result in capturing the uncertainty from trajectory data comparing to homoscedatic model. The comparison of two types of models using Kullback-Leibler (KL) divergences is represented in Fig. 3.4. From left to right, we can observe the results for different dimensions (Longitude, Latitude and Altitude). The performance of two Altitude predictive models are similar even though the median value of heteroscedastic model is slightly smaller. A possible explanation is that the variance of Altitude dimension in raw data is consistence over time which will eliminates benefits of our proposed approach. Our proposed approach

outperforms classical Gaussian Process in modeling both Longitude and Latitude, especially a significant improvement in performance of model for Latitude. We can conclude that heteroscedastic model can better model trajectory data comparing to classical GP. Thus, we only consider heteroscedastic noise models for predictive models.

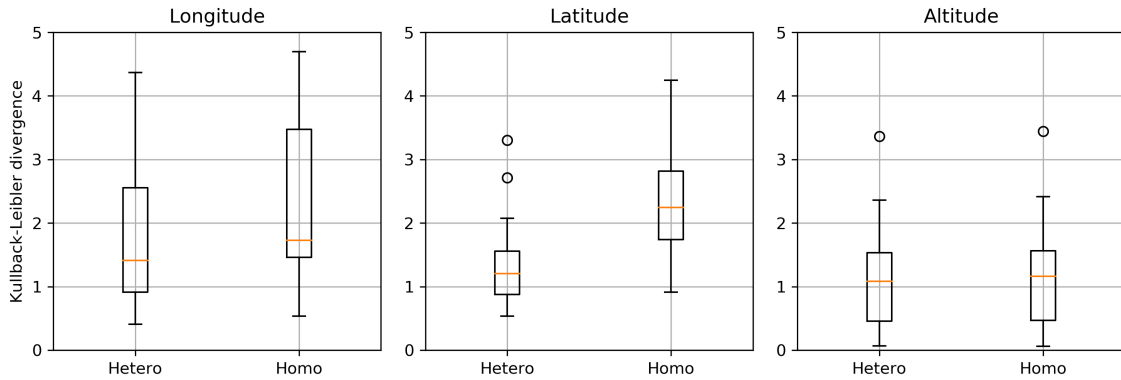


Fig. 3.4 Comparing Kullback-Leibler divergences of heteroscedastic and homoscedastic models

The results for predictive models for ADS-B data is illustrated in Fig. 3.5. From the group of ADS-B data, we can input a list of $N_{CP} = 4$ conditional points for prediction. Here, we instead of handcrafting those points, we just select them from a random trajectory of the group. The second columns of figures show the results of filtering process. Because the filtered radius will be proportional to the variance of the group at given time stamps, the filtered trajectories can also have big dispersion by keeping trajectory quite far from conditional points. The results for predictive models which can be observed in third column of Fig. 3.5 are interesting. The mean predicted trajectory can be used to anticipate the flight trajectory which flow the reference route (through conditional points) since it is fitted well with the shape of the trajectories in data. Another interesting observation is the learned variance or uncertainty of the model. The sampling data from the model have shown its ability in capture not only the dispersion of filtered trajectories but also the full data.

Another example result for prediction can be observed in Fig.3.6 for toy sample. The sample learning process is applied and at the end, we can obtain two predictive models for two routes (called G1, G2). This is a crossing scenario where the conflict happens because of the uncertainty on the position of the aircraft at given time stamp t . They are well-trained models for predicting aircraft position with uncertainty from given data. Then those models are used in following experiments for evaluating BayesOpt as probabilistic conflict detection algorithm.

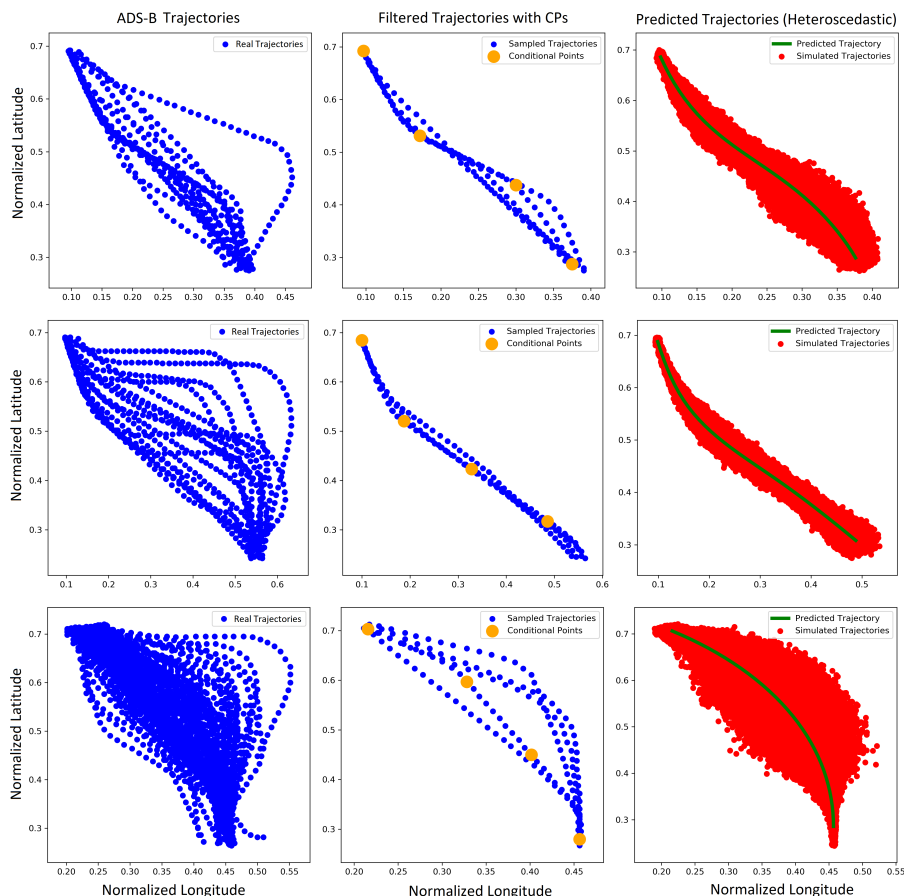


Fig. 3.5 Prediction results from Generative Model with ADS-B data

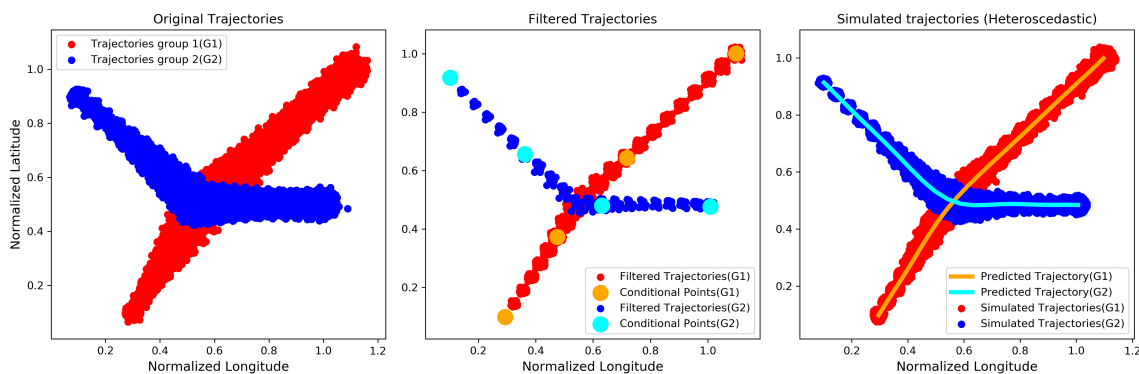


Fig. 3.6 Prediction results from Generative Model for Toy sample

3.6.2 Result for Exp 2:

Monte Carlo method is evaluated with 15 different configurations (3 for \mathbf{N} and 5 for \mathbf{R}). BayesOpt's performance is reported for $Max_Eval \in [50, 100, 200]$. The results of all experiments (Monte Carlo and BayesOpt) are reported in Table 3.3. The first observation is the similar in performance of Bayesian Optimization and Monte Carlo with $\mathbf{R} \leq 10$. They provide similar result in term of Maximum Conflict Probability (P-CPA) or estimating Time to Conflict (T-PCPA). While with $R \geq 30$ s the performance of Monte Carlo is significantly

reduced since it can't locate the right position of conflict due to the limitation of fine level. Besides, we can see the standard deviation for estimating conflict probability is quite small ($std < 0.53 \times 10^{-3}$) with 95% confidence level. Even though the found P-CPAs are similar, when the number of sampling(N) is increased, we can observe the decrease of the standard deviation by half from $0.53 \times 10^{-3} \rightarrow 0.26 \times 10^{-3}$.

Table 3.3 Performances of BO and MC for detecting potential conflicts

N	R(s)	T-PCPA(s)	P-CPA($\times 10^{-2}$)	C_Cost(s)
500K	BO-50	553	3.707 ± 0.052	8.06
	BO-100	554	3.758 ± 0.053	16.11
	BO-200	558	3.759 ± 0.053	32.07
	1	556	3.835 ± 0.053	180.63
	5	555	3.766 ± 0.053	35.98
	10	560	3.782 ± 0.053	17.78
	30	570	3.283 ± 0.049	5.90
	60	540	2.720 ± 0.045	3.03
1000K	BO-50	556	3.805 ± 0.037	14.20
	BO-100	556	3.805 ± 0.037	28.39
	BO-200	556	3.805 ± 0.037	56.50
	1	557	3.800 ± 0.037	332.84
	5	560	3.764 ± 0.037	67.07
	10	560	3.768 ± 0.037	34.57
	30	570	3.268 ± 0.035	11.19
	60	540	2.652 ± 0.031	5.44
2000K	BO-50	550	3.600 ± 0.026	28.14
	BO-100	560	3.788 ± 0.026	56.29
	BO-200	560	3.788 ± 0.026	112.01
	1	559	3.798 ± 0.026	657.58
	5	560	3.776 ± 0.026	132.71
	10	560	3.810 ± 0.027	64.91
	30	570	3.295 ± 0.025	23.07
	60	540	2.669 ± 0.022	11.07

More importantly, the computational cost also provide interesting observation. The costs increase linearly for all algorithms when we increase the number of samples (N). Besides, when R increases, there are more values of t which should be evaluated thus, the cost will also increase. For the given conflict scenario, with 20-minutes trajectories, the cost for BayesOpt with 100 evaluations (BO-100) is close to costs of MC-10 (120 evaluations). However, BayesOpt can achieve the result up to smaller time unit (1s) similar to MC-1 (in case of $N = 1000K$) but much faster in term of speed.

3.7 Conclusion

The experiment results have shown the potential of our approach for trajectory prediction and conflict detection. Heteroscedasticity Gaussian Process Regression can learn the uncertainty from data and combine it into the aircraft position prediction. Although some recent approaches try to increase the confidence of prediction by using more data and advanced machine learning technique, the uncertainty from data is still available because of human intervention and incomplete information. Our approach can model trajectory data with heteroscedasticity noise which can be also applied in abnormal detection for unstable approaches. However, as discussed, this approach can not handle multi-output to model the relation between three dimensions. The results in experiment 2 have highlighted the benefits of Bayesian Optimization in combining with generative model over classical Monte Carlo. The algorithm can work with continuous time it means it can locate the conflict position without the impact of fine level of discretized time. Especially, when the cost of computing conflict probability is expensive, the contribution of this method is more significant.

This approach can speed up the conflict detection in flight simulators in the presence of uncertainty. However, this study is still in early state in which it has not been evaluated for multi-conflicts yet. Thus we do not apply it in building our simulation environment for conflict resolution in next chapter. As future work, we want to solve the multi-output problem which we believe will increase the performance of predictive model and speed up the training. Neural Network with Dropout as Bayesian approximation [26] is an interesting extension for our predictive model. Besides, the BayesOpt approach for probabilistic conflict detection can be generalized to work for the whole sector with multiple flights and conflicts. The results of those studies can be combined to develop a new simulation environment for conflict resolution in future.

Publication related to this chapter is:

- "Conflict Prediction Using Generative Model from ADS-B Data". *2019 Integrated Communications Navigation and Surveillance (ICNS)*. IEEE, 2019. Virginia, USA [61].

Chapter 4

Artificial Intelligent Agent for Conflict Resolution

With the continuous growth in the air transportation demand, air traffic controllers will have to handle increased traffic and consequently more potential conflicts, and this gives rise to the need of conflict resolution advisory tools that can perform well in high density traffic scenario given a noisy environment. Unlike model-based approaches, learning-based ones can take advantages of historical traffic data and flexibly encapsulate the environmental uncertainty in performing conflict resolution. In this study, we propose an artificial intelligent agent that is capable of resolving conflicts, in the presence of traffic and given uncertainties in conflict resolution maneuvers, without the need of prior knowledge about a set of rules mapping from conflict scenarios to expected actions. The conflict resolution task is formulated as a decision-making problem in large and complex action space, which is applicable for employing reinforcement learning algorithm. Our work includes the development of a learning environment, scenario state representation, reward function, and learning algorithm. As the result, the proposed method, inspired from Deep Q-learning and Deep Deterministic Policy Gradient algorithms, is able to resolve conflicts, with a success rate of over 81%, in the presence of traffic and varying degrees of uncertainties.

4.1 Deep Reinforcement Learning in Conflict Resolution

Machine learning methods have emerged as promising method for solving air traffic management problems such as Taxi-out time prediction [67, 45], aircraft sequencing [1], trajectory prediction [7], aircraft performance parameter predicting [3], air traffic flow extraction [16], flight delay prediction [75, 15]. Deep learning models, like Long Short-Term Memory (LSTM), are also investigated in [40] for air traffic delay prediction tasks.

For decision-making problems like conflict resolution, their large and continuous state and action spaces are a challenge for machine learning methods. However, Reinforcement

Learning (RL) can be considered as one of the promising approaches for their success in building playing engine for classic board game with expert level [68] such as in Backgammon, Checker and Scrabble. Moreover, advanced machine learning algorithms, like deep learning, have demonstrated breakthroughs such as Deep Blue [12] for Chess, Poker-CNN [79] and DeepStack [55] for Poker.

Recently, the combination of deep learning and reinforcement learning which is called deep reinforcement learning (DRL) has increased the potential of automation for many decision-making problems that were previously intractable because of their high-dimensional state and action spaces. In 2015, Mnih et. al. [54] introduced Deep Q-Network model which could learn to play a range of Atari 2600 video games at a superhuman level, directly from raw image pixels. Secondly, AlphaGo [71], that defeated a human world champion in Go used neural networks that were trained using supervised and RL, in combination with a traditional heuristic search algorithm. Differentiating from those works dealing with discrete action space, [72, 48] has tackled the problem of continuous action space by introducing a general-purpose continuous DRL framework, the actor-critic Deterministic Policy Gradient Algorithms. The action policy function is approximated by a neural network while the reward function estimator is trained with the second one. These approaches can be potential approaches for conflict resolutions since they can work with incomplete knowledge for efficiently resolving a conflict. They are be able to self-evolve when being exposed to unseen scenarios. Finally, they can take advantages of historical conflict resolution data from ATCO in building models.

In this chapter, we describe an AI agent that is capable of resolving conflicts in the presence of uncertainty. The AI agent resolve conflict with only lateral deviation. A reinforcement learning (RL) algorithm is developed as learning model which can handle the large and continuous state and action spaces of conflict scenario. Similarly to air traffic controllers (ATCOs), the AI agent can also learn and form its strategy when dealing with conflict scenarios and have the ability to self-evolve via trial-and-error.

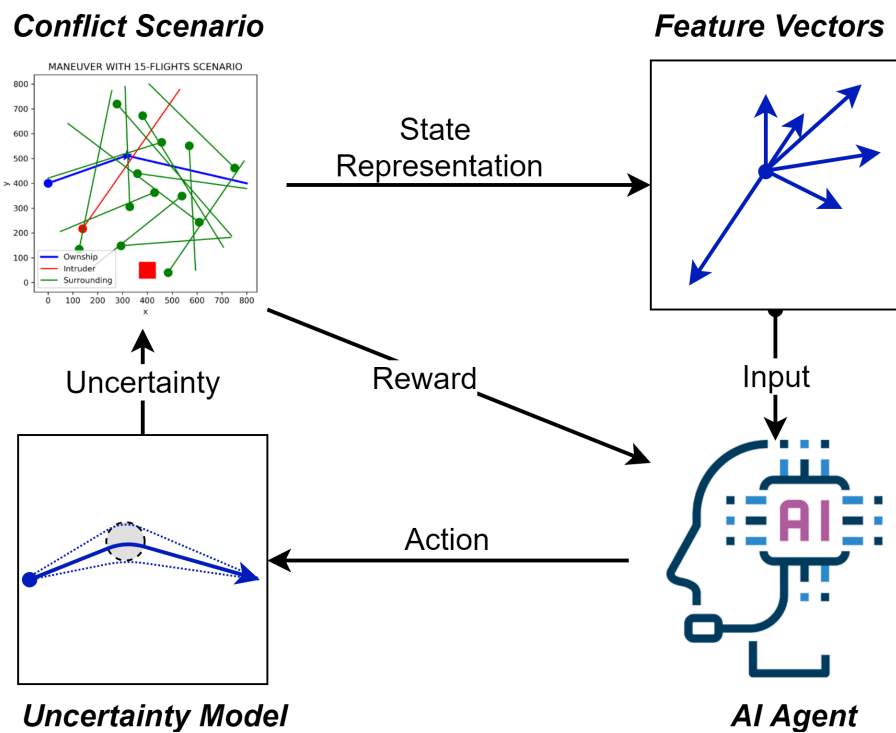


Fig. 4.1 Interaction between the learning environment and the AI agent

The process of training the AI agent for conflict resolution is illustrated in Fig 4.1. Conflict scenarios involving multiple aircraft are generated and presented to the AI agent by the learning environment. The agent, which is powered by RL algorithm, learns to resolve these conflicts by applying a number of maneuvers given the environmental uncertainty. The agent receives a reward for every maneuver it has tried as performance feedback, and the value of the reward depends on the quality of the maneuvers: positive rewards for maneuvers that successfully resolve the conflicts and negative rewards, or penalties, for maneuvers that are unable to safely separate the conflicting aircraft. The learning objective is to maximize the reward and the AI agent is assumed to have been trained when it consistently gains (converges) high rewards for resolving unseen conflict scenarios.

To achieve this we formulate the conflict resolution problem as a decision-making problem which is suitable for reinforcement learning algorithm. To accomplish this, we give special considerations to the following sub-tasks.

1. Our problem is designed as space-based searching action where agent will perform list of actions at a given time; Unlike the time-based continuous control problem reported in [48].
2. A learning environment is developed for flight conflict detection and resolution that possesses the following characteristics.

- We design the reward function to consider not only the conflict status of the scenario but also the quality (e.g. computing the magnitude of deviation, maneuverability, total travel time, etc) of the maneuvers.
 - Maneuver's uncertainty is encapsulated in learning model with four different levels.
 - A novel scenario representation (state vector) is proposed which contains information of conflict scenario such as conflict status, optimal status and uncertainty level. This state vector must be carefully designed in order to guarantee the convergence of the training.
3. The learning model is designed to handle multi-dimensional actions with different physical scales and units (e.g. time and distance).

In the following sections, we describe these subtasks one by one.

4.2 Learning Environment

In the RL for conflict resolution (Fig. 4.1), the main roles of the learning environment are (1) to present its state to the agent in a form that provides sufficient information to support the agent's decision-making, (2) to receive and evaluate the AI agent's action, and (3) to give feedback to the agent as a reward. To obtain such environment, we develop scenario generator that generates conflict scenarios and represents them in a form perceivable to the agent. We propose the definition of agent's action and the mapping from actions to the maneuvers. Each maneuver can be scored using our proposed reward function. Finally, we also consider the environmental uncertainty that occurs during the implementation of the agent's actions in order to assess its learning performance.

4.2.1 Conflict scenarios

A conflict scenario is considered as a traffic scenario that occurs within a circular area of interest (airspace) of radius r , in which there is one pair of potential conflict between an ownship and an intruder aircraft, in the presence of surrounding traffic. An example of a conflict scenario considered in this study is shown in Fig. 4.2a, and the conflict pair between the ownship and the intruder in this scenario is separately plotted in Fig. 4.2b for clear presentation. In the scope of this work, we only consider the conflict between ownship and another aircraft in the given airspace. Here, for convenience and without loss of generality, we generate conflict scenarios such that the ownships always point along the horizontal direction and the traveling distance of the ownship in the given airspace is equal to the diameter of the airspace's boundary. Any direction of the ownship could be achieved

by performing a simple rotational transformation of the interested airspace, and different travelling distances of the ownship could be considered by simply setting the size of the interested area.

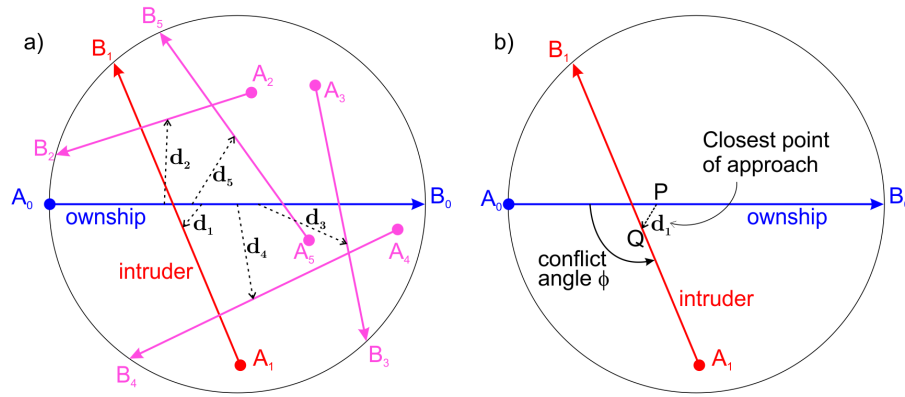


Fig. 4.2 a) Example of a conflict scenario involving a two-aircraft conflict in the presence of four surrounding aircraft. $\mathbf{A}_0\mathbf{B}_0$ is the ownship and $\mathbf{A}_1\mathbf{B}_1$ is the intruder. b) The conflict occurs where \mathbf{PQ} is the closest distance between two aircraft and smaller than minimum separation

Let n be the number of aircraft in the given airspace when a potential conflict is being considered, \mathbf{A}_i denote the locations of the aircraft at the moment the conflict scenario presented to the agent ($t_0 = 0$), and \mathbf{B}_i the locations where the aircraft exit the given airspace ($0 \leq i < n$), see Fig. 4.2a. Consequently, $\mathbf{A}_i\mathbf{B}_i$ represent the aircraft's trajectories and $\overrightarrow{A_i B_i}$ are the initial headings of the aircraft. If the aircraft continue their journeys with this original flights plan, the ownship (following route $\mathbf{A}_0\mathbf{B}_0$) and the intruder (following route $\mathbf{A}_1\mathbf{B}_1$) are converging; they will simultaneously reach \mathbf{P} and \mathbf{Q} (Fig. 4.2b). Since the scenario is generated such that the distance \mathbf{PQ} is less than d_{sep} , which is the safe separation to maintain or minimum separation standard, the two aircraft are losing their safe separation if none of them takes any maneuver. Here, \mathbf{PQ} is called the closest point of approach (CPA) between the ownship and the intruder, also denoted by the CPA closure vector \vec{d}_1 from \mathbf{P} to \mathbf{Q} . Similarly, the CPAs between the ownship and the surrounding aircraft are denoted by \vec{d}_i where $2 \leq i < n$. Note that at the beginning, $\|\vec{d}_1\| < d_{sep}$ while $\|\vec{d}_i\| \geq d_{sep}$, $2 \leq i < n$; this imposes the single initial conflict condition to the generated scenarios, which is the interest of this work.

We now briefly describe the computation of CPA between the ownship and the intruder, and the same procedure is applied to find CPA between the ownship and the surrounding aircraft. Assume that all aircraft are cruising at the same speed of v_c . At $t_0 = 0$, the ownship is at \mathbf{A}_0 and the intruder \mathbf{A}_1 . The velocities of the ownship and the intruder are $\vec{u} = v_c(\overrightarrow{A_0 B_0}/\|\overrightarrow{A_0 B_0}\|)$ and $\vec{v} = v_c(\overrightarrow{A_1 B_1}/\|\overrightarrow{A_1 B_1}\|)$, respectively. At a time $t > 0$, the locations of the ownship and the intruder are respectively given by $\vec{P}(t) = \vec{A}_0 + \vec{u}t$ and $\vec{Q}(t) = \vec{A}_1 + \vec{v}t$, and distance between them renders as $d_1(t) \equiv \|\vec{d}_1\| = \|\vec{W}_0 + (\vec{u} - \vec{v})t\|$ where $\vec{W}_0 =$

$\overrightarrow{A_0A_1}$. Minimizing $d_1(t)$ yields the time to CPA as $t_{CPA} = -\overrightarrow{W_0} \cdot (\vec{u} - \vec{v}) / \|\vec{u} - \vec{v}\|^2$, and the closure at CPA as $d_{1(CPA)} = d_1(t_{CPA})$.

4.2.2 Ownship's Maneuver

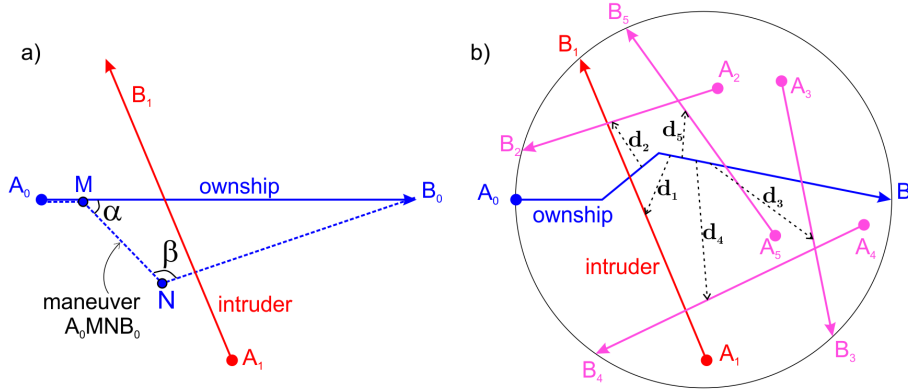


Fig. 4.3 a) An example of maneuver. The ownship makes a heading change α° at point **M** at $t = t_1$, continues in the new heading **MN** during t_2 seconds, and heading back towards original end point at return point **N**. A maneuver is fully defined by a set of three parameters (t_1, α, t_2) . b) The maneuver implemented in the traffic scenario.

A maneuver, e.g. maneuver A_0MNB_0 in Fig. 4.3a, is defined as a series of actions performed by the ownship: deviate from original path at time t_1 seconds and at location **M** (measuring from $t_0 = 0$ at A_0) by changing the heading by an angle α , and then keep heading along vector \overrightarrow{MN} in t_2 seconds before heading back towards **B**₀ at return point **N**. Thus, a maneuver is fully defined by a set of three parameters (t_1, α, t_2) . In addition, a valid maneuver is defined as the maneuver that satisfies $t_1 < t_{CPA}$, $\|\alpha\| < 90$ degrees, and t_2 takes a value such that the return point **N** located within the interested area. In this thesis, we assume that any applied maneuver modifies the path of the ownship while leaves the intruder's path unchanged.

Fig. 4.3b demonstrates an example of a maneuver being implemented in a scenario. An employed maneuver changes the scenario from the current state into the next one by updating the CPA closure vectors \vec{d}_i . The quality of a maneuver, therefore, is essentially determined by these CPA closure vectors, which reflect the aircraft's separation status in the scenario. We shall elaborate the evaluation of the agent's actions and the resultant maneuvers in the definition of reward function later in this section.

4.2.3 Environmental uncertainty

The working environment in air traffic control have high degrees of uncertainties. The controllers have to deal with unknowns originated from, for example, inaccurate trajectory prediction, navigation instruments errors, weather, and other unexpected events in the

airspace. Therefore, any conflict resolution tool for ATCO must perform effectively in the presence of uncertainty. In this work, we consider environmental uncertainty as a parameter that affects the accurate/precise implementation of the agent's conflict resolution actions.

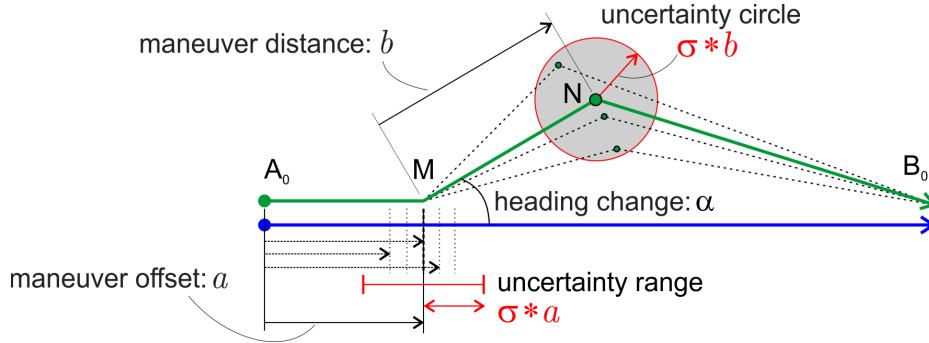


Fig. 4.4 Environmental uncertainty and its impact on the agent's action

Consider a situation in which the agent suggests a maneuver determined by (t_1, α, t_2) , e.g. the maneuver in green shown in Fig. 4.4, where a is the expected maneuver offset, α the expected heading change, and b the expected maneuver distance. The ownship is expected to change its heading at the expected heading change point M and turn back at the expected return point N . Due to environmental uncertainties, however, the actual maneuver is slightly deviated from the expected one. In particular, the actual heading change point M' is determined by the actual maneuver offset $a' = a + \mathcal{N}(0, \sigma * a)$, where $\mathcal{N}(0, \sigma * a)$ is a normal distribution noise with zero mean and variance $\sigma * a$. Similarly, the actual return point N' is computed as $(x', y') = (x, y) + \mathcal{N}(0, \sigma * b)$, where (x, y) are the coordinates of N and (x', y') of N' . Here, the variances of the noise distributions affecting the heading change point and the return point are controlled by $\sigma * a$ and $\sigma * b$, where σ is the parameter whose values represent the uncertainty level. This model implies that a less deviated and immediately implemented maneuver suffers less from the environmental uncertainties, while a maneuver with large deviation and further in time suffers more.

4.2.4 Scenario representation

In RL, it is never too much to emphasize the importance of the environment's state representation, as every decision made by the agent is heavily influenced by the agent's perceived state of its environment, and the state representation determines how the agent apprehends the state. In the given problem, to ensure that actions taken by the agent always modify the separating conditions of the ownship, scenario representation must encapsulate the ownship's current separation status. Therefore, it is reasonable, and also important, to include the CPA closure vectors \vec{d}_i in the state vector, because these vectors carry the essential information on the separation statuses of the ownship with other aircraft in the environment. With this in mind, we design the one-dimensional state vector s to represent a scenario as follows.

1. The first element is σ indicating the current environmental uncertainty level.
2. Separations between the ownship and each other aircraft are encapsulated by every 5 next elements:
 - x - and y - positions of the ownship at CPA (2 elements)
 - CPA closure $\|\vec{d}_i\|$ (1 element)
 - x - and y - directions of the CPA closure vector (2 elements)
3. Directional guidance vector (the last 2 elements). This vector is chosen to be $\vec{\mathbf{NO}}_{\text{CPA}}$, where \mathbf{N} is the return point and \mathbf{O}_{CPA} the location at CPA of the ownship against the intruder at the beginning. We shall discuss this in the definition of reward function below.

Note that the total length of the state vector depends on the maximum number of aircraft being considered.

4.2.5 Maneuver evaluation

Maneuver score as reward value

We design the reward mechanism to give merit to any maneuver suggested by the agent that successfully separates the aircraft and to punish one that fails to improve the separation status. The environment evaluates the reward based on the resultant state of the scenario upon implementation of the suggested maneuver. As the ultimate aim is to separate the aircraft, more positive rewards are given to maneuvers that improve the separation status, while maneuvers that worsen the situation are punished by negative reward. Furthermore, for a valid maneuver that successfully resolves the conflict, the quality of the maneuver is also evaluated, such as deviation from the original trajectory and maneuverability of the resolution.

Let $R(a, s')$ being the reward function that takes an action a together with its resultant state vector s' as two input arguments and returns the reward value. Also, we denote $d_{\min} = \arg \min_i \|\vec{d}_i\|$, ($1 \leq i < n$), as the minimum value among all the separation distances of the ownship against other aircraft. Then, the reward function is defined as

$$R(a, s') = \begin{cases} e^{d_{\min}^{-1}} - 1, & \text{if } d_{\min} < d_{\text{sep}} \\ (1 - \frac{\Delta D}{\Delta D_{\max}}) * 100, & \text{otherwise} \end{cases} \quad (4.1)$$

where ΔD denotes the deviation of the maneuver from the original ownship's trajectory, and ΔD_{\max} the maximum deviation that could occurs. In the definition of the reward function, Eq. 4.1 punishes invalid maneuvers that cause $d_{\min} < d_{\text{sep}}$ and therefore fail

to separate the ownship from other aircraft. On the other hand, Eq. 4.2 calculates the rewards for valid maneuvers, which successfully separate the ownship and eliminate all potential conflicts, by evaluating the deviations of the maneuvers from the ownship's original trajectory. Less deviated maneuvers receive higher rewards, on a score scale of maximum 100. The maneuver's deviation is defined as

$$\Delta D = w_1 * dist(\mathbf{M}, \mathbf{O}_{CPA}) + w_2 * dist(\mathbf{N}, \mathbf{O}_{CPA}), \quad (4.3)$$

where $dist()$ yields the distance between two points. Here, Eq. 4.2 and Eq. 4.3 imply that less deviated maneuvers shorten the distances \mathbf{MO}_{CPA} and \mathbf{NO}_{CPA} . This reflects the design of the reward mechanism to maintain the positions of the action points (\mathbf{M} and \mathbf{N}) within the neighborhood of the initial conflict location, which could help preventing the maneuver from causing secondary conflicts with surrounding aircraft. This also justifies the inclusion of $\vec{\mathbf{NO}}_{CPA}$ in the state representation as mentioned in Section 4.2.4.

4.3 AI Agent and Learning Mechanism

In our problem, the ultimate goal is to train the AI agent such that given a conflict scenario, it could resolve the conflict and earn a possibly highest reward after a finite number of iterations, as quickly as possible. Instead of using classical optimization approaches, here, we adapt the Deep Deterministic Policy Gradient (DDPG) algorithm [48] for our learning model. We propose this approach because it does not require prior knowledge of how to efficiently resolve a conflict, and learning algorithm is able to self-evolve when being exposed to unseen scenarios. In this section, we briefly describe our AI agent, show the characteristics of the proposed DDPG algorithm that make it appropriate for training the agent, and discuss the training process as well as some implementation considerations.

4.3.1 Agent's Action for Deep Reinforcement Learning

When resolving a conflict, the agent could suggest a possible maneuver by computing the set of three parameters (t_1, α, t_2) that fully defines the maneuver, as mentioned in Section 4.2.2. We could see from Fig. 4.3 that any value of (t_1, α, t_2) is equivalent to a choice of (t, x, y) , where $t = t_1$ is the heading change time, x and y are the coordinates of the return point \mathbf{N} , relatively to the center of the interested area. Moreover, the possible valid choices of (x, y) highly depend on t ; therefore, it is rational to treat the agent's action as a two-stage decision-making process. In the first stage, the agent determines the heading change time t that results in the heading change point \mathbf{M} . In the second stage, it decides the coordinates (x, y) of the return point \mathbf{N} , being aware of the updated aircraft's locations at time t . This treatment of the agent's action is beneficial in two ways. First, as t and (x, y) are different

in nature, the two-stage process allows us to handle them independently. Second, such approach avoids the computing of the original parameters (t_1, α, t_2) using the same model, which could be problematic because they might be very different in scale.

Thus, reward for a conflict scenario can be defined as:

$$V(s) = R(t|s) + R((x,y)|t,s) \quad (4.4)$$

In which $R(t = t_i|s)$ is the immediate reward for selecting $t = t_i$ as time duration and $R((x,y)|t = t_i, s)$ is the reward for selecting the return position (x,y) given previous decision $t = t_i$ and conflict scenario s . Therefore, the optimal reward for a given scenario is:

$$\begin{aligned} V^*(s) &= \max_{t,(x,y)} (R(t|s) + R((x,y)|t,s)) \\ &= \max_t [R(t|s) + \max_{(x,y)} R((x,y)|t,s)] \end{aligned} \quad (4.5)$$

We apply the principle of dynamic programming to obtain the last component in Eq. 4.5. The Eq. 4.5 shows how we convert this problem from finding a 3-dimensional maneuver into finding the time duration t . For each value of t , we always compute the maximum reward over a set of possible return positions (x,y) . Then finding optimal value for given scenario is equivalent to search value of t^* to maximize the reward. Additionally, the ultimate goal is to recommend the best maneuver $m^* = (t^*, (x,y)^*)$ for a given conflict scenario. Besides value of t^* , we also need to get the return point $(x,y)^*$ which provides the optimal value for $R^*((x,y)|t,s)$. An actor-critic algorithm (i.e. DDPG) is a good candidate for modeling the second part, $R((x,y)|t,s)$, since it can provide both optimal values (Q^* _value) and optimal maneuver $((x,y)^*)$ at the same time. The propose approach is presented in Fig. 4.5 in detail. For a given scenario s' , and a time duration t ($<$ Time to CPA), a time-shifted scenario s is computed. This conflict scenario s is the input for an actor-critic model which provides optimal maneuver $m'^* = v(x,y)^*$ and optimal value $Q^*(s, m') = Q^*(s, (x,y)|s', t)$. To simplify the problem, t is a set of discrete values (t_0, t_1, \dots, t_N) which can be looped over to find the optimal value. Moreover, in air traffic control domain, the deterministic characteristic of decision is important for any model i.e. given a conflict scenario, we always receive the same recommended maneuver (without re-training the model). Finally, the possible space for return point is large and continuous in nature which is a challenge for several RL models. They are the reasons on how we select DDPG algorithm [48] as our proposed method.

The next challenge for applying DDPG is to define the *action* for AI agent. The *action* for an AI agent can be defined in several ways, which shape the learning mechanism and affect agent strategy. For instance, the agent's action can be the same as the maneuver $a \equiv m' \equiv (x,y)$ or similar to the searching step in which agent searches around in multi-steps for the good return point position, etc. In this study, we apply the second method. The agent's action is a moving step (dx, dy) , and the agent performs a sequence of actions

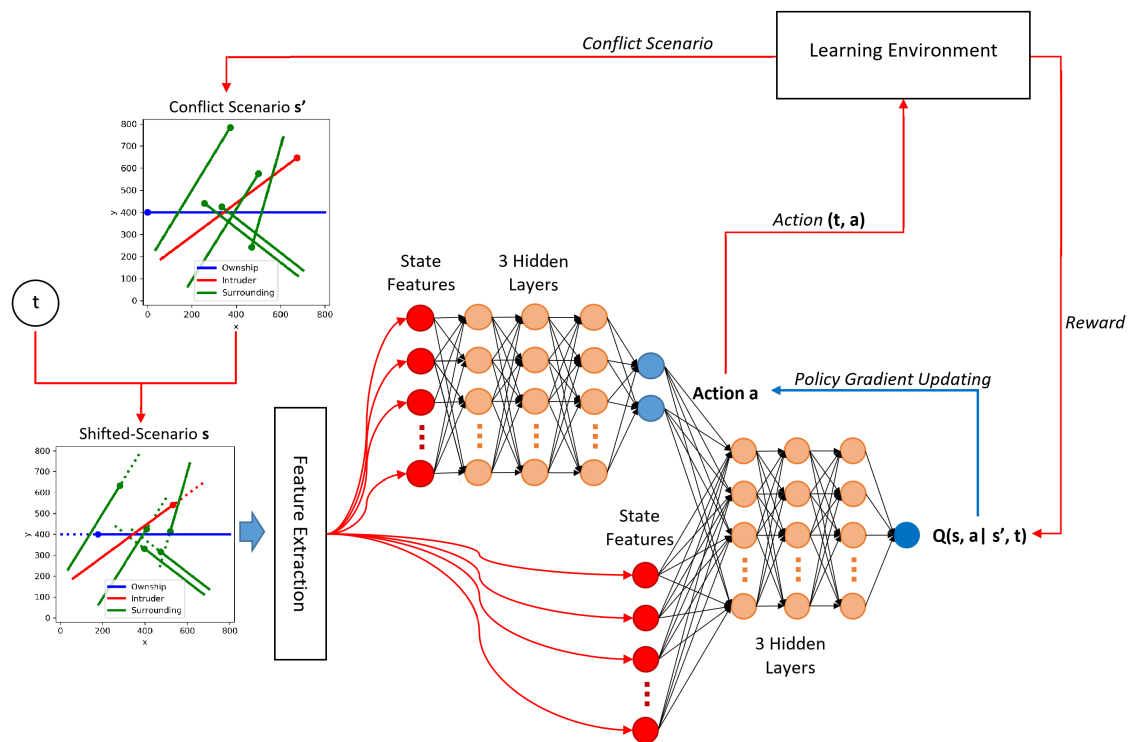


Fig. 4.5 Model for learning conflict resolution.

$[(dx_0, dy_0), (dx_1, dy_1), \dots, (dx_k, dy_k)]$ ($dx, dy \leq \text{Radius}(rd)$) to optimize the location (x, y) of the return point \mathbf{N} . Let $(x, y)^* \equiv (x^*, y^*)$ denote the optimal location of \mathbf{N} , where $x^* = x_0 + \sum_{i=0}^k (dx_i)$ and $y^* = y_0 + \sum_{i=0}^k (dy_i)$; in which the values of the $\text{Radius}(rd)$ is provided in Section 4.4. The value of each action and the number of actions are controlled by the learning algorithm with the learning mechanism for AI agent is described in detail in Section 4.3.3.

4.3.2 Deep Deterministic Policy Gradient (DDPG)

DDPG is a variant of actor-critic model based on Deterministic Policy Gradient (DPG) algorithm [72]. One of its main contributions is introduction of a neural network as actor model to deal with continuous action space. The DDPG algorithm has two models:

1. **Actor Model:** This is a neural network for learning the mapping from state to action, $\mu(s)$. Given a state feature vector, described in 4.2.4, actor model will predict an optimal action a^* under current policy. In this case, given the state vector, actor model will predict the moving action $a_i^* = (dx_i, dy_i)^* = \mu(s_i)$ to update the current maneuver under current policy.
2. **Critic Model:** This is neural network to evaluate the quality of action given conflict scenario. It receives scenario s_i and action a_i as inputs and estimates the expected value/reward $Q(s_i, a_i)$.

As mentioned in Section 4.3.1, the main learning algorithm is DDPG but we also introduce a searching step like in Deep Q-Learning [54] to identify the heading change time t . Our proposed 2-stages action DDPG is described in Algorithm 5. Our implementation considers following enhancements in DDPG:

1. **Replay Memory:** to store pass experiences for batch training which can solve the problem about dependence of samples, predicted maneuvers in our cases. The training process begins only when the Replay Memory has been filled with a minimum number of samples. The memory's capacity (i.e. the maximum number of samples in the Replay Memory) is fixed, and this helps to eliminate out-of-date samples in the training process.
2. **Batch Normalization:** to deal with multiple units and ranges in input scenario.
3. **Soft target update** is used to increase the stability of learning. Line 16 in Algorithm 5 shows how to apply soft target. The learning rate is control by parameter τ
4. **Action in RL** is considered to balance between exploration and exploitation. DDPG allows to separate exploration from the algorithm by introducing a new noise policy $\mu_{\mathcal{N}} = \mathcal{N} + \mu$ as exploring policy where exploring noise \mathcal{N} is a random process. Ornstein-Uhlenbeck Noise (OU noise) [48] is implemented as our exploring noise.

4.3.3 Learning Mechanism

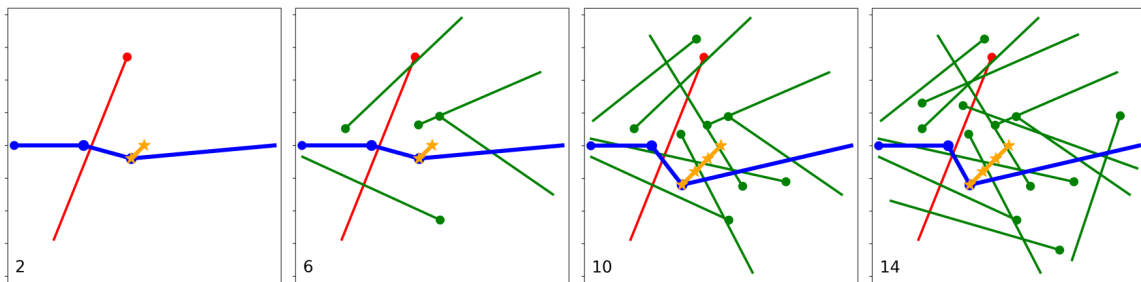


Fig. 4.6 Examples of a searching episode to suggest resolution

The interaction between AI Agent and Learning Environment is the core mechanism for training and testing for RL as in Fig. 4.5. Since the conflict resolution is a continuous control problem, thus the episode should be different from the classical time-based episode. The episode is designed as a searching process to locate an “acceptable resolution” (examples can be observed in Fig. 4.6). At each step, the agent will predict the best action (dx, dy) to modify the current resolution and send it to environment. Learning environment will update the current maneuver $(x', y') = (x + dx, y + dy)$, evaluate it and send feedback back to agent. The process is repeated until receiving “acceptable resolution” or the number of searching

Algorithm 5: DDPG Algorithm for 2-stages action

-
- 1: Randomly initialize weight θ^Q for Critic Net $Q(s, a|\theta^Q)$
 - 2: Randomly initialize weight θ^μ for Actor Net $\mu(s|\theta^\mu)$
 - 3: Initialize target networks Q' and μ' by $\theta^{Q'} \leftarrow \theta^Q, \theta^{\mu'} \leftarrow \theta^\mu$.
 - 4: Initialize replay buffer R
 - 5: **for** episode = 1, M **do**
 - 6: Initialize a random process \mathcal{N} for action exploration.
 - 7: Receive scenario s' from Environment
 - 8: Computing scenario s_1 by shifting all flights in s' a duration $heading_change_Time t_0 = random(0, Max_T)$
 - 9: **for** t = 1, Max_steps **do**
 - 10: Select action $a_t = \mu(s_t|\theta^\mu) + \mathcal{N}_t$ according to policy μ .
 - 11: Execute action a_t , observe reward r_t and new state s_{t+1}
 - 12: Store transition s_t, a_t, r_t, s_{t+1} in R
 - 13: Sample a random K experiences s_i, a_i, r_i, s_{i+1} from R
 - 14: Set $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$
 - 15: Update critic by minimizing the loss:

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

- 16: Update actor policy using sampled policy gradient:

$$\nabla_{\theta^\mu}(J) \approx \frac{1}{K} \sum_i \nabla_{\mu} Q(s_i, \mu(s_i)) \nabla_{\theta^\mu} \mu(s_i)$$

- 17: Update the target networks:

$$\theta^{Q'} \leftarrow \tau \theta^Q + (1 - \tau) \theta^{Q'}$$

$$\theta^{\mu'} \leftarrow \tau \theta^\mu + (1 - \tau) \theta^{\mu'}$$

- 18: **end for**

- 19: **end for**
-

steps is reached. “Acceptable resolution” can be flexibly defined by controlling the threshold for minimum reward of acceptable resolution. Without satisfying those stopping conditions, the steps are considered as intermediate steps and their rewards are constant number $x (= -0.1)$ as a small penalty.

Training Process

Algorithm 5 and Fig. 4.5 can be used to describe training and testing process for AI Agent in detail. The main purposes of training phase include generating learning samples, training actor and critic target network using DDPG algorithm. While in testing phase, given an unseen scenario, only actor target network is used to predict "optimal" maneuver. The step by step algorithm for training is as follows.

1. Learning Environment generates random scenario s' .
2. The conflict scenario s' is shifted with random heading change, time t_0 to obtain shifted conflict scenario s_0 .
3. Feature extraction algorithm is applied on Shifted conflict scenario s_t (t is initialized by 0 for each new scenario) to obtain state vector which is the input for DDPG algorithm.
4. Given the state vector, current exploration actor policy (current actor model $\mu(s_t|\theta^\mu)$ plus exploration noise \mathcal{N}_t) will provide a candidate action $a_t \equiv (dx, dy)$.
5. The action is sent to learning environment to compute the real reward r . If the conditions for stopping episode are reached, the uncertainty model will create noise-action by adding random noise to proposed action and then compute the reward r_t for it. Else, environment just return immediate constant reward $r_t = 0.1$ and updated scenario s_{t+1} after applying action to modify the current maneuver.
6. The sample tuples (s_t, a_t, r_t, s_{t+1}) is stored in replay buffer for later use in training model.
7. When the replay buffer has stored enough samples (\geq minimum start size), a batch of samples is sampled randomly from replay buffer for training.
8. The critic model is updated by minimizing the defined loss function which is similar to training supervise learning model.
9. The policy gradient is computed from the gradient of critic model and applied to update actor model at each step.
10. Finally, the target networks are updated in soft updating manner.
11. If end of episode is reached, the searching step will be stopped and go back to step 1. Else, increase $t = t + 1$ and go back to step 3.

Evaluating Process

The testing phase or predicting phase is relative simple since we only need to obtained the final recommended maneuver for given conflict scenario. However, in practical use, the experiences generated in this phase can also be stored in replay buffer for tuning the model via batch training. This setting can help the model tuning to be faster and keep the model up-to-date with new incoming data. The step-by-step for maneuver prediction is described as follows.

1. Given unseen scenario s' (i.e. from learning environment):

2. The heading change time t is looped over the range $[T_{min}, T_{max}]$ with step-value Δt seconds. The conflict scenario s' is shifted with each given heading change time t to obtain shifted conflict scenario s .
3. Feature extraction algorithm is applied on shifted conflict scenario s_i to obtain state vector which is the input for actor model.
4. Given state vector, actor target model suggests action $a_i^* \equiv (dx_i, dy_i)^*$.
5. The action is sent to the environment. If end of episode is reached, go to next step, else compute next state s_{i+1} , $i = i + 1$ and return to step 3.
6. Given state vector and "optimal" action, critic target model will provide the Q-value $Q^*(s, a^*|t)$.
7. Compute $MAX_Q = \max_t Q^*(s, a|t)$ and store corresponding maneuver $m = (t, (x, y)^*)$.
8. Finally, after checking with all values of heading change time t , the "optimal" maneuver for given conflict scenario s' is obtained $(t^*, (x, y)^*)$

4.4 Experiment Setup

In our experiments, conflict scenarios are randomly generated in an interested area of radius $r = 50$ nm. For the initial conflict, $d_{1(CPA)} < d_{sep}$ where $d_{sep} = 5$ nm, and $240 \leq t_{CPA} \leq 480$ seconds, given that the common speed of aircraft $v_c = 400$ knots (nm/hr). This configuration implies that the potential loss of separation between two aircraft is foreseen 4-8 minutes. We consider the maximum number of aircraft in the airspace $n_{max} = 15$; therefore, the state vector has fixed size of 73 (see Section 4.2.4 for state representation). In the event the number of aircraft is less than n_{max} , the elements representing the absent aircraft are replaced by that of the intruder. During maneuvers implementation, we consider 4 levels of environmental uncertainty, $\sigma = \{0, 2\%, 5\%, 10\%\}$.

Fig. 4.7 shows examples of 24 groups of initial conflicts generated in our experiments, consisting of 4 groups of t_{CPA} (time to CPA) and 6 groups of conflict angle ϕ (see Fig. 4.2 for the definition of conflict angle). This classification allows us to assess the model's performance in different classes of initial conflicts. Note that surrounding traffic are not shown in Fig. 4.7.

Parameters used for training the agent are shown in Table 1. The values of these hyper parameters are chosen from practise. For example, γ ($[0,1]$) is the discount factor to weight the importance for future rewards. If $\gamma \leftarrow 0$, the agent will focus only on immediate rewards. Else, if $\gamma \leftarrow 1$, the future rewards have greater weight in the model. In our case, final step of episode is the required maneuvers which is the main source of reward, thus in this study $\gamma = 1$.

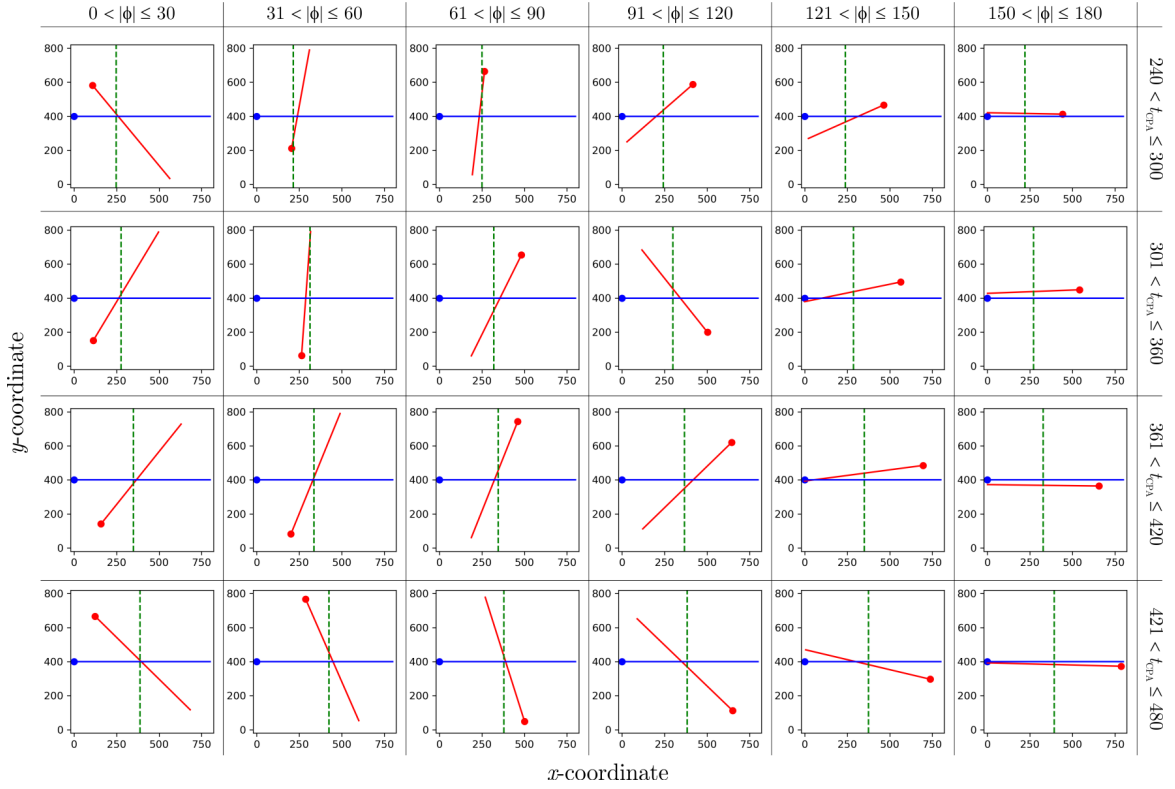


Fig. 4.7 Generated scenarios are classified into 24 groups, based on their time to CPA (t_{CPA} , across the rows) and conflict angle (ϕ , across the columns).

Table 4.1 Parameters for training the AI agent

Parameter	Meaning	Value
lr_{actor}	Control learning rate of actor model	10^{-4}
lr_{critic}	Control learning rate of critic model	10^{-3}
$batch_size$	Size of training batch	64
γ	Discount factor for future rewards	1
τ	Control rate of updating target networks for both models	10^{-3}
$\mu_e, \theta_e, \sigma_e$	Parameter set for exploration noise	0, 0.1, 0.5
$thres_{maneuver}$	minimum acceptable reward	40
Max_steps	Maximum number of searching steps	10
$Radius(rd)$	Upper bound of agent's action	5NM
Δt	Time step for heading change time	30 seconds

We train the RL Model by interacting with our environment and learn from those experiences. At each 500 iterations, learning environment will generate 4800 random scenarios (200 scenarios from each group) for evaluating current model. Each conflicted scenario then is assigned a randomly number of aircraft (2 to 15) and uncertainty level (4 levels). This setup tries to limit the computational cost for intermediate evaluation.

4.5 Results and Discussion

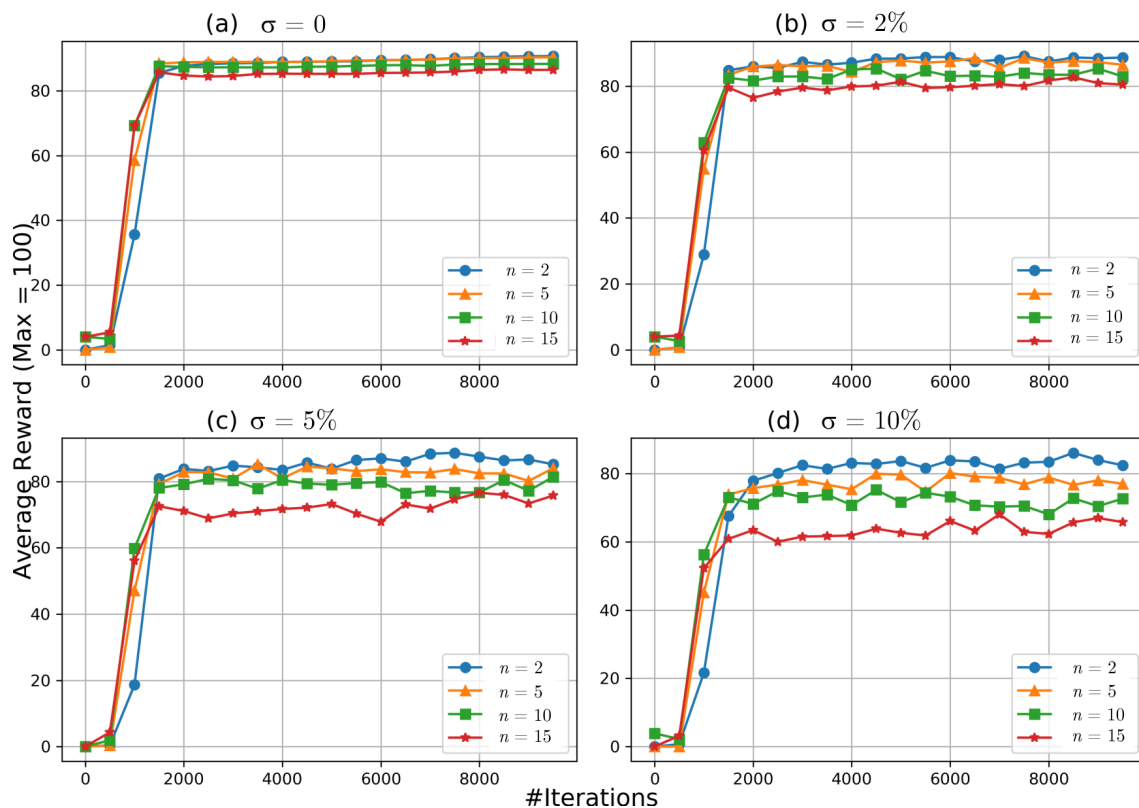


Fig. 4.8 Convergence of learning model at different configurations of uncertainty level (σ) and number of aircraft (n)

We first assess the model's performance by the average score (on the scale of 100) that the agent earns for solving 4800 unseen scenarios of a common test set. Fig. 4.8 shows the test results (or model's convergence) as the model is evolving during training at different levels of environmental uncertainty (subplots (a) for no uncertainty, (b) for 2%, (c) for 5%, and (d) for 10%). Each data point represents the average score performed on the common test set after every 500 training iterations. Each curve reports the result for a different number of aircraft n (blue curve for $n = 2$, orange for 5, green for 10 and red for 15).

A common trend is observed from Fig. 4.8 that for all configurations, the training curves admit three distinguished phases, approximately: the warming up phase in the first 500 iterations, the evolving phase in the next 1500 interactions, and the converging phase after about 2000 iterations. The observed trend in the performance curves suggests that our current setting is able to guarantee the model's convergence at different uncertainty conditions and numbers of aircraft.

On the other hand, the different impacts of environmental uncertainty σ and total number of aircraft n on the learning performance are also observed. One could observe from Fig. 4.8 that the environmental uncertainty has stronger impact on the convergent speed than the number of aircraft does. Higher levels of the environmental uncertainty more delay

the model's convergence. In particular, at low levels of uncertainty, i.e. $\sigma = \{0, 2\%\}$, convergence occurs after about 1500 iterations, while at higher levels, i.e. $\sigma = \{5\%, 10\%\}$, the model only starts to converge when the training has reached about 2000 iterations.

The environmental uncertainty not only reduces the convergent speed, but it also affects the stability of the convergent score. Fig. 4.9a plots the variances of the achieved scores at different levels of uncertainty. We can see that variances increases with the uncertainty level, suggesting that the model's performance is less stable at higher uncertainty. Fig. 4.9a also shows how the convergent stability is reduced by increasing the number of aircraft. At this point, it is meaningful to plot the variances being normalized with that at no uncertainty, as shown in Fig. 4.9b; this allows us to observe the increasing rate of variances when the environment becomes more uncertain. For instance, at the number of aircraft $n = 2$, introducing 2% uncertainty causes the score variance increased by about 2.8 times, 5% uncertainty increases the variance by 4 times, and 10% uncertainty results in 5 times increase. The fact that the growth rate of the variances (with increasing uncertainty) is lower at higher number of aircraft, as seen from 4.9, suggests that the model could control the total score variance caused by the increasing in both uncertainty and the number of aircraft.

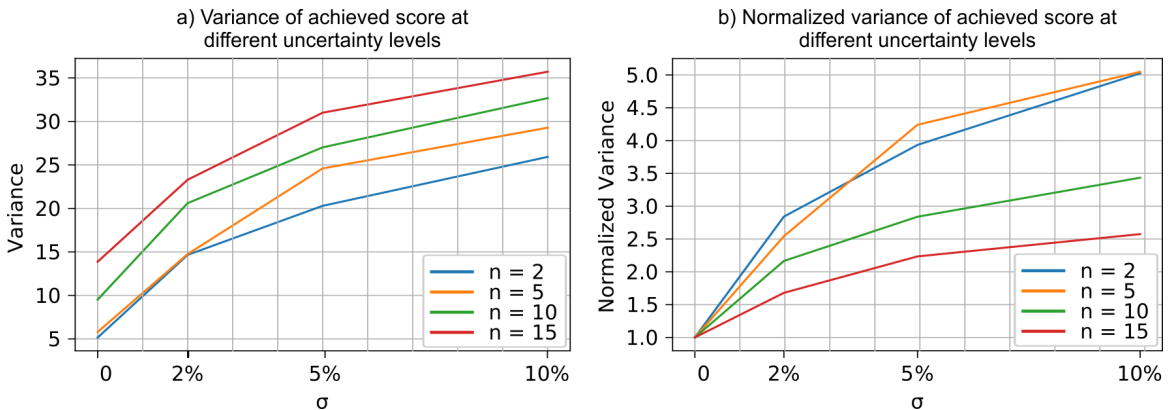


Fig. 4.9 Effects of the uncertainty on the performance stability after convergence. a) Variances of achieved scores. b) Variances being normalized with variances at no uncertainty.

Fig. 4.10 provides a closer look to the performance of the model after convergence, indicated by the average reward and the successful rate. We define the successful rate as the percentage of the conflicts being successfully resolved by the agent with a reward higher than an acceptable threshold (40 out of 100 in our setting). From Fig. 4.10, we observe linear relations between the model's performance indicators (i.e. score and successful rate) and the number of aircraft involved. Increasing the number of aircraft cause the performance to drop, and the environmental uncertainty even worsens this drop. For instance, by increasing the number of aircraft from 2 to 15, the successful rate drops by 1% (from $\approx 99\%$ to $\approx 98\%$) at no uncertainty, and by 10% (from $\approx 91\%$ to $\approx 81\%$) at 10% uncertainty. Similar trend is also observed in the change of reward with the increasing number of aircraft. The results reported in Fig. 4.10 indicate the significant impact of the number of aircraft on the performance

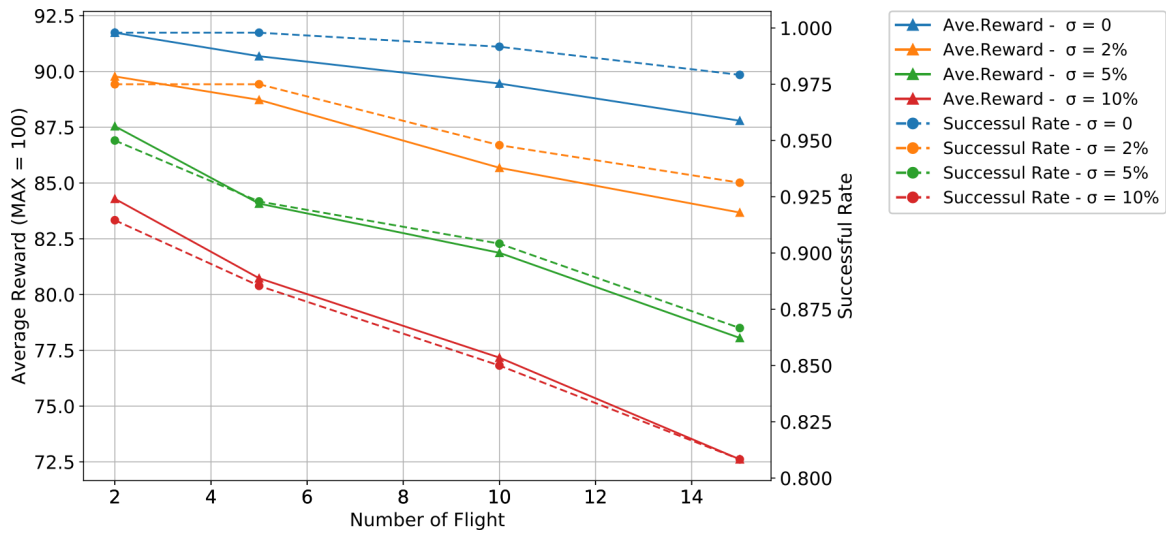


Fig. 4.10 Average reward and successful rate achieved by the agent after convergence.

of the agent; nevertheless, the successful rate of about 81% achieved by our agent at high uncertainty ($\sigma = 10\%$) and dense surrounding traffic ($n = 15$) proves the high performance of the learning model.

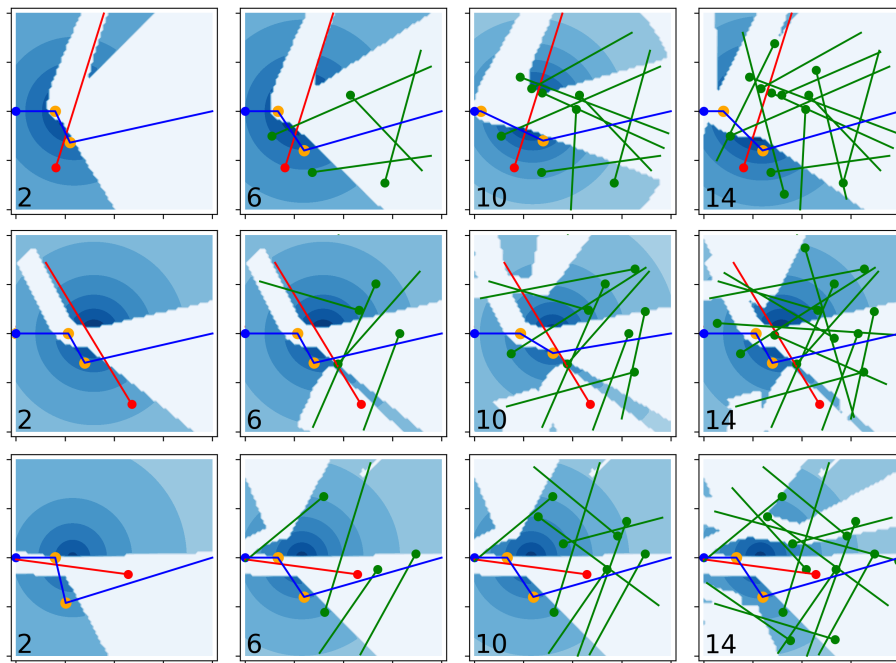


Fig. 4.11 Examples of predicted resolutions in scenarios with different number of flights

Fig. 4.11 presents the maneuvers suggested by the agent for resolving 3 conflicts (each in a row) at different numbers of surrounding aircraft. The blue gradients in the backgrounds represent the feasible regions of the maneuvers' return points (at pre-determined heading change points). Maneuvers with return points located in the darker regions receive higher rewards. As the number of aircraft increases, the feasible region fragments into more disconnected smaller regions. The fragmentation of the feasible region causes a drop

in model's performance (Fig. 4.10), especially under high uncertainty, because strong environmental disturbance more possibly shifts the agent's suggested return point from a feasible region to a impractical one.

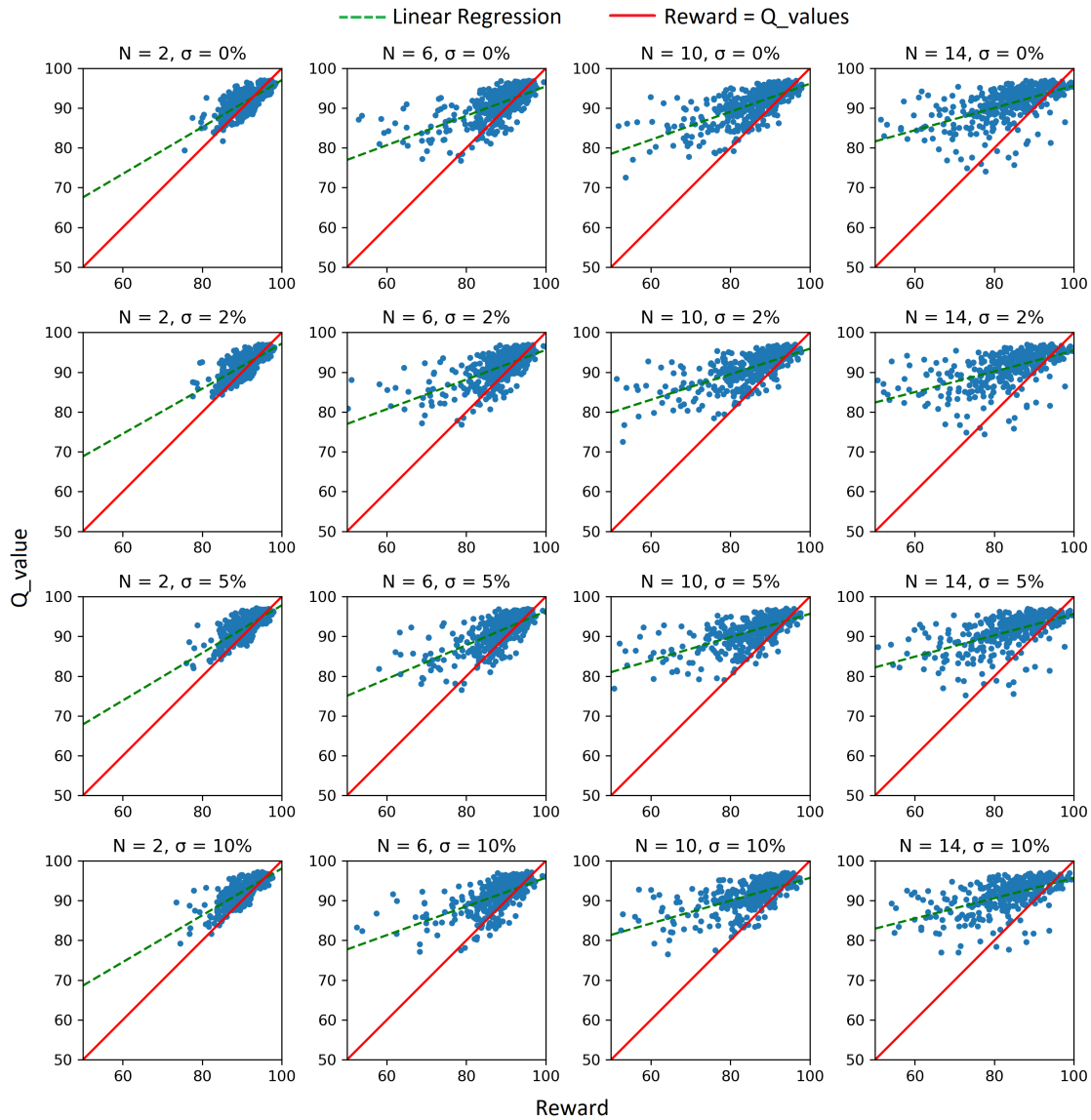


Fig. 4.12 The approximation of Q_value to real Reward when Number of Flights (NoF) from 2 to 14 and Uncertainty (U) from 0% to 5%

Another interesting observation from Fig. 4.11 is the constructed strategy of AI Agent to resolve conflict scenario. The searching steps is always going down, thus, it is equivalent the flight will turn right at the heading change point. This strategy is simple and although it provides high quality solution, from Fig. 4.11, we notice that it is not globally optimal but just locally optimal. This phenomena can be a results of several reasons but one of the main reason is the equalization of turning left and turning right as resolutions for conflict scenario as well as the design of reward itself.

Finally, in Fig. 4.12, we perform the assessment on how our model can learn and approximate real reward from environment given a conflict scenario. The red line is "the optimal line" where the approximation (Q_value) is exactly the same as read reward. The approximated values is higher than real rewards in most of the case. We apply simple linear regression to capture the linear relation between Q_values and Rewards. The green dash line shows the tendency of over-estimated reward of the proposed model. It is not affected much by the uncertainty but the number of flights in scenario. It reflects the difficulty of high traffic scenario in learning approximator because of its diversity. However, the regression lines also report the linear relation between real rewards and Q values. This relation is important for training the actor model because actor model needs the relative scores between different $Q(s,a)$ pairs for choosing best action rather than their exact values. This means that given current level of approximation, the performance of actor won't be affected.

4.6 Case Study: AI Agent with Human References

500 conflict scenarios to an ATCO and collects his resolutions for these scenarios. Each scenario has one potential conflict between the ownship and the intruder aircraft, and there are three other surrounding aircraft (see 4.2). Assume that the ATCO resolves all the conflicts employing a consistent strategy; therefore, the resolutions provided encapsulate the ATCO's preference in resolving conflicts. Among the 500 pairs of conflicts and captured resolutions, we randomly choose 400 pairs to form the training set of data, and the remaining 100 pairs being the testing set used for model evaluation.

The most important indicator of the model's performance is the similarity between the agent's and the ATCO's resolutions. Since this similarity is assessed by the reward mechanism, the approximation of the reward (or the equivalent penalty) signal, i.e. $V(s,a)$, is essential to the model's performance. Fig. 4.13 presents the convergence of the approximated penalty and the actual penalty as the training is progressing. Note that the use of penalty instead of reward for model assessment does not alter the model's qualities in any manner. Fig. 4.13 shows two qualities of the model when it is converging: (1) the approximated reward signal (the blue curve), i.e. the output of the critic network in Section 4.5, converges to the actual one (the orange curve), and (2) they both become stable. We could see that the model well converges after circa 2,000 iterations. At the beginning of the training, the actual penalty starts from a very large value ($> 1,000$) because at that moment the agent is still in the exploration phase. The approximated penalty starts from a very small value because of the initialization of the critic network.

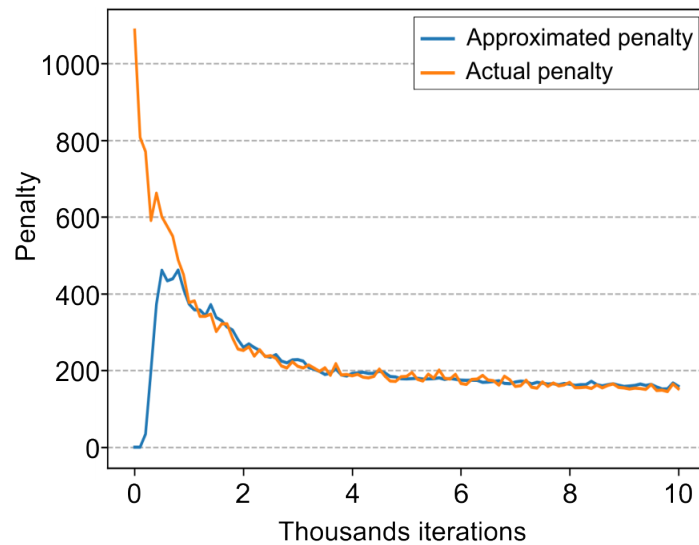


Fig. 4.13 Convergence of the learning model

Fig. 4.14 shows the agent's resolution for a unseen conflict scenario after model's convergence, as an example. The dashed white line is the originally planned trajectory of the ownship, and the solid thick white curve is the resolution suggested by the agent. The trajectory change point (TCP) of the agent's resolution is very close to that provided by the ATCO, which is located by the star marker. In Fig. 4.14, the heat map represents values of the penalty, where lower penalties (i.e. more desirable TCP) are located in the darker blue regions and high penalties (i.e. low quality TCP) are at the darker red locations.

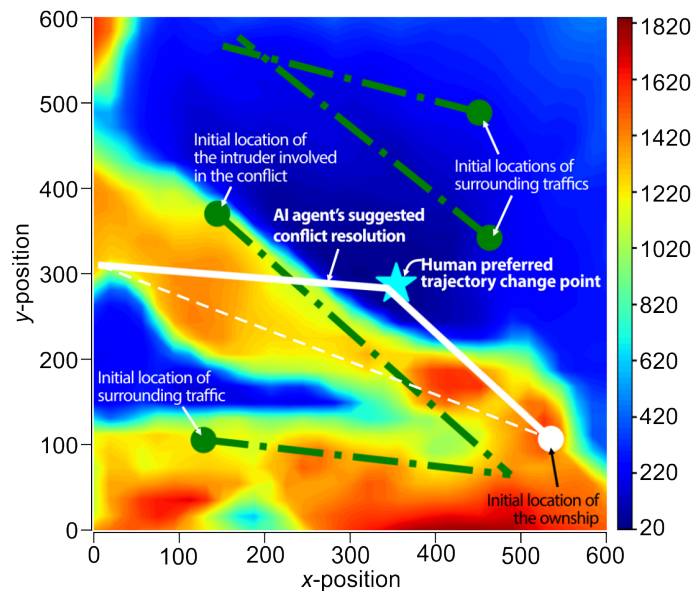


Fig. 4.14 An example of the agent's suggested resolution showing the similarity between agent's and human resolutions

After the model has converged, we allow the agent to resolve 100 unseen conflict scenarios in the test set, and the distribution of the penalties given to the agent is shown

in 4.15, where more than 65% of the resolutions suggested by the agent received very low penalties (i.e. lower than 100). The majority of suggested resolutions receive low penalties indicates the high capability of the agent of suggesting resolutions that capture the ATCO's preference.

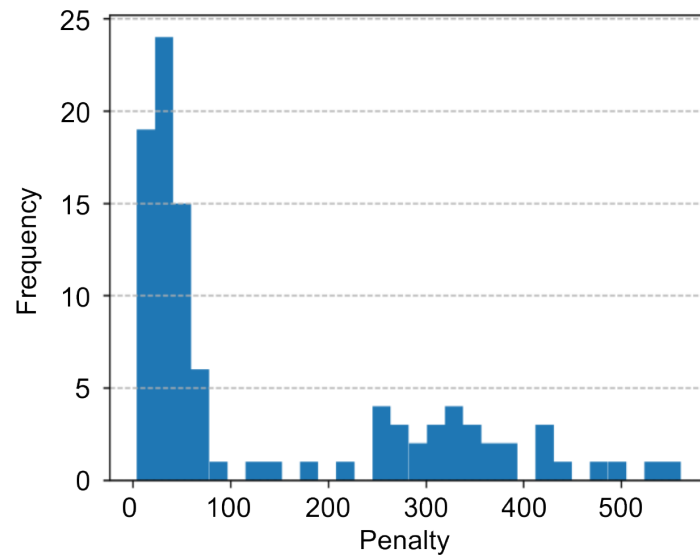


Fig. 4.15 Penalties distribution performed on test set after convergence

In 4.16, we demonstrate the agent's suggested resolutions for six unseen conflict scenarios. It could be observed that in all scenarios, the trajectory change point is always located at the dark blue region, showing that the agent is able to limit the penalties to very low values by suggesting resolutions that are close to the preference provided by human.

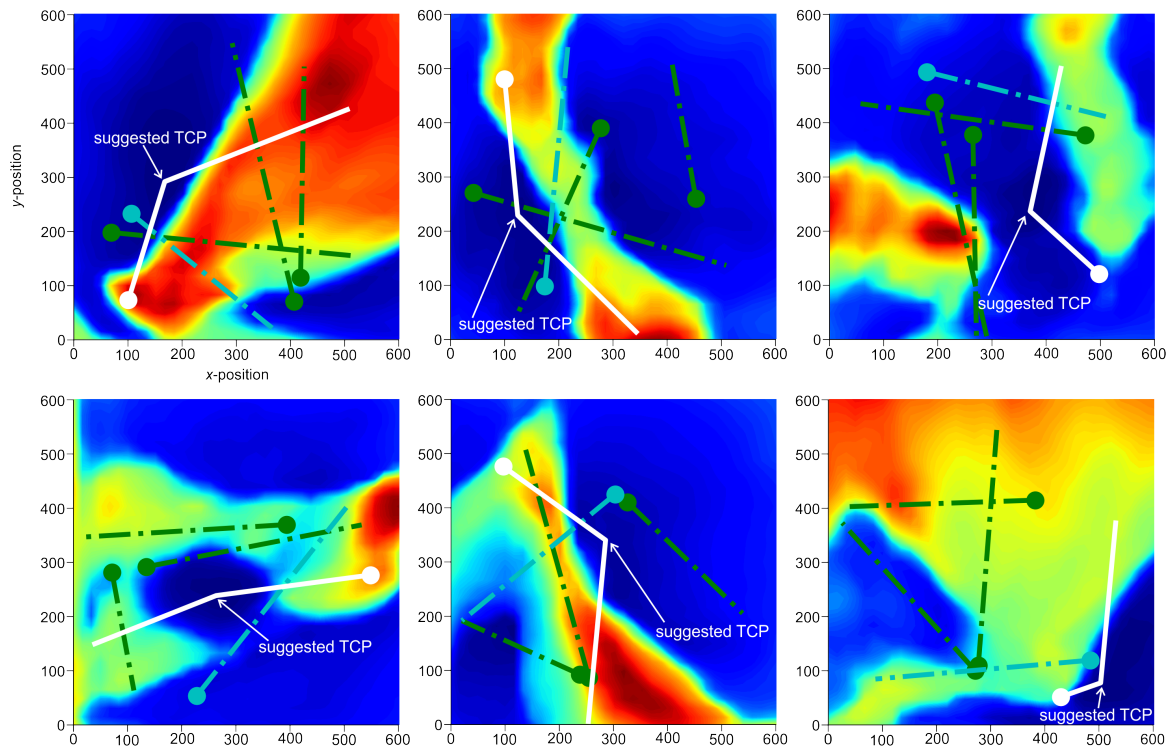


Fig. 4.16 The agent's suggested resolutions for different conflict scenarios

4.7 Conclusion

In this chapter, we have formulated the problem of conflict resolution in the presence of surrounding traffic and uncertainty as a reinforcement learning problem. Important components of the reinforcement learning algorithm for conflict resolution, such as learning environment, scenario state representation, reward function, and learning algorithm, have been discussed in great details. We have also laid out the evaluation of model's performance, which could be considered as a framework for the assessment of reinforcement learning method applied to conflict resolution problem.

Our findings show that the combination of Deep Q-learning and Deep Deterministic Policy Gradient algorithms gives the AI agent the great capability to suggest high quality conflict resolution, with a successful rate of over 81% in the presence of dense surrounding traffic and strong environmental disturbance. Here, it should be highlighted that the agent achieved this high successful rate without the need of prior knowledge about a set of rules mapping from conflict scenarios to expected actions.

To improve system performance, possible future considerations include but not limited to (1) the enhancement of the scenarios state representation to help the agent to better "perceive" its learning environment and (2) the extension of the current work to multi-agent system for cooperative conflict resolutions.

Publications related to this chapter are:

1. "Reinforcement Learning for Two-Aircraft Conflict Resolution in the Presence of Uncertainty". *The 2019 RIVF International Conference on Computing & Communication Technologies-Research, Innovation, and Vision for Future (RIVF)*. IEEE, 2019. Da Nang, Viet Nam [63].
2. "An Intelligent Interactive Conflict Solver Incorporating Air Traffic Controllers' Preferences Using Reinforcement Learning". *2019 Integrated Communications Navigation and Surveillance (ICNS)*. IEEE, 2019. Virginia, USA [76].
3. "A Machine Learning Approach for Conflict Resolution with Uncertainties in Dense Traffic Scenarios", *ATM R&D Seminars 2019*. 2019. Vienna, Austria [62].

Chapter 5

Conclusion and Future Work

5.1 Conclusion

This thesis concentrates on investigating machine learning approaches for conflict detection and resolution which are divided into sub questions.

First of all, we have looked into learning and predicting the D-side controller's action for a given traffic scenario in a sector using a tree-based methods known as Random Forest. This approach use controller's actions as target variables and aircraft 4D trajectory features prior to entering a sector as explanatory variables. We have processed and analyzed the air traffic trajectories to establish that patterns in D-side controllers exists. Two groups of models were developed, one to predict the actions and the other to predict the associate action values. The model for vertical action provided the highest accuracy with 99.7% whereas, model for speed change and heading change action provides predictability accuracy of 88.7% and 72.4% respectively. This was attributed to highly complex sector entry and exit point configuration that makes learning challenging. Even though it is not used in practice, we also build predictive model for set of all 3 kinds of actions (multiple-output) for each flight, and achieve an accuracy of 0.68. The lower predictability can be due to over-fitting of the training data for controller's actions, leading to poor generalization performance. This exploration process only focuses on predicting simple actions from individual flight entry information. It can be extended by extracting or defining more complex actions using ours as marco/proxy actions. Moreover, this task is helpful for understanding and investigating the nature and patterns in a given sector as well as ATCO's strategies in conflict resolution. However, it also can be considered as the preprocessing step to support building dataset for training and evaluating AI-Agent in conflict detection and resolution (CD&R). From obtained knowledge, we process the ADS-B data by clustering trajectories to form flight routes based on grouping their entry and exit points. Those flight routes can be used for trajectory prediction and other applications.

Secondly, we have proposed an novel approach for probabilistic conflict detection in both modeling uncertainty and detecting highest potential conflict. We have proved the advantages of Heteroscedastic Gaussian Process in modeling trajectories. Because from our observation, trajectory data has inhomogeneous variance which can be approximated better by Heteroscedastic model. The proposed method is used for trajectory prediction considering uncertainty of positions. Given those predictive models, we propose using Bayesian Optimization approach for conflict detection by estimating the time stamp corresponding to highest conflict probability. Our method can detect potential conflict quickly and flexibly comparing to classical Monte Carlo method. Since it can work with continuous time while Monte Carlo is constrained by predefined discretized time step. Especially, when the cost of computing conflict probability is expensive, the contribution of this method is more significant. Thus, this approach can be used to speed up the conflict detection in flight simulators in the presence of uncertainty.

Finally, we have formulated the problem of conflict resolution in the presence of traffic and uncertainty as a reinforcement learning problem. For this approach, we have proposed and developed a set of important components, such as learning environment, scenario state representation, reward function, and learning algorithm. In our work, we also propose a framework for assessment of reinforcement learning method for conflict resolution. Our developed AI agent has the great capability to suggest high quality conflict resolution, with a successful rate of over 81% in the presence of dense traffic and strong environmental disturbance. Here, it should be highlighted that the agent achieved this high successful rate without the need of prior knowledge about a set of rules mapping from conflict scenarios to expected actions. Although, this approach (81% successful rate) is still far from operational application, it can be developed and tailored for training new controller purposes. By carefully designing objective function, this approach can be adapted and modified for different applications. The discussed case study is a successful case where controller's preferences can be learned and imitated.

Publications related to this thesis are:

1. "A Machine Learning Approach on Past ADS-B Data to Predict Planning Controller's Actions". *8th International Conference on Research in Air Transportation (ICRAT '18)*. 2018. Barcelona, Spain.
2. "Conflict Prediction Using Generative Model from ADS-B Data". *2019 Integrated Communications Navigation and Surveillance (ICNS)*. IEEE, 2019. Virginia, USA.
3. "Reinforcement Learning for Two-Aircraft Conflict Resolution in the Presence of Uncertainty". *The 2019 RIVF International Conference on Computing & Communication Technologies-Research, Innovation, and Vision for Future (RIVF)*. IEEE, 2019. Da Nang, Viet Nam.

4. "Learning Air Traffic Controller Strategies with Real-Time Neural and Physiological Feedback". *SESAR Innovation Days (2018)*. 2018. Salzburg, Austria.
5. "A Machine Learning Approach for Conflict Resolution with Uncertainties in Dense Traffic Scenarios", *ATM R&D Seminars 2019*. 2019. Vienna, Austria.

5.2 Future research directions

The research work in this thesis can be extended in the following directions in the future.

1. Considering complex fast-time simulator with dense traffic for evaluating and validating our approaches.
2. Strategy reconstruction from data: we attempt to establish the clear definition and structure of those strategies in conflict resolution from extracted patterns or observed actions. This knowledge will play a crucial role in supporting mining, exploring real operational data i.e. ADS-B data. Then the knowledge will be organized and structured to support machine learning approaches by guiding new designs for data structure and model's architecture.
3. Feature extraction or scenario representation: when making decision, ATCO may consider global and local information of the airspace, include but not limited to complexity of the conflict, neighboring traffic, airways structure, traffic flow, sector geometry, secondary conflicts, Letters of Agreement etc. How to process and use those information is the big challenges for automation of conflict resolution. Specially in machine learning approaches, information of airspace must be extracted, standardized and pre-processed which is usually called feature extraction. Even though, in this thesis, our extracted features can be used in proposed models, we also observe their limitation and impact on the performance of our approach. Without good features of given scenarios, there is big limitation on performance of learning model in general. We will investigate this problem by combining two approaches: surveying how ATCO interprets, relates and uses the airspace information and new machine techniques in representation learning.
4. Probabilistic conflict detection for whole sector: the multi-output problem will be investigated to increase model's performance and speed up the training. We attempt to use Neural Network with Dropout as Bayesian approximation [26] to replace Gaussian Process for modeling flight trajectory. Besides, the BayesOpt approach for probabilistic conflict detection can be generalized to work for the whole sector with multiple flights and conflicts.

5. Multi-agent conflict resolution: The proposed reinforcement learning algorithm can be extended for multi-agent in conflict resolution. Each aircraft is an agent sharing similar strategy (i.e. controller's strategy) and makes his conflict resolution in cooperative manners. The challenges for this task include but not limit to balancing global and local objectives, learning shared policy, common and individual scenario information, etc.

References

- [1] Ahmed, M., Alam, S., and Barlow, M. (2018). A cooperative co-evolutionary optimisation model for best-fit aircraft sequence and feasible runway configuration in a multi-runway airport. *Aerospace*, 5(3):85.
- [2] Alam, S., Abbass, H., Lokan, C., Ellejmi, M., and Kirby, S. (2009). Computational red teaming to investigate failure patterns in medium term conflict detection. In *8th Eurocontrol Innovation Research Workshop, Eurocontrol Experimental Center, Brtigny-sur-Orge, France*.
- [3] Alligier, R., Gianazza, D., and Durand, N. (2015). Machine learning and mass estimation methods for ground-based aircraft climb prediction. *IEEE Transactions on Intelligent Transportation Systems*, 16(6):3138–3149.
- [4] Alligier, R., Gianazza, D., and Durand, N. (2016). Predicting aircraft descent length with machine learning. International Conference on Research in Air Transportation.
- [5] Allignol, C., Barnier, N., Durand, N., and Alliot, J.-M. (2013). A new framework for solving en route conflicts. *Air Traffic Control Quarterly*, 21(3):233–253.
- [6] Allignol, C., Barnier, N., Durand, N., Gondran, A., and Wang, R. (2017). Large scale 3d en-route conflict resolution. In *ATM Seminar, 12th USA/Europe Air Traffic Management R&D Seminar*.
- [7] Ayhan, S. and Samet, H. (2016). Aircraft trajectory prediction made easy with predictive analytics. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 21–30. ACM.
- [8] Bakkes, S. C., Spronck, P. H., and van Lankveld, G. (2012). Player behavioural modelling for video games. *Entertainment Computing*, 3(3):71–79.
- [9] Bergstra, J. S., Bardenet, R., Bengio, Y., and Kégl, B. (2011). Algorithms for hyperparameter optimization. In *Advances in neural information processing systems*, pages 2546–2554.
- [10] Bloembergen, D., Tuyls, K., Hennes, D., and Kaisers, M. (2015). Evolutionary dynamics of multi-agent learning: A survey. *J. Artif. Intell. Res.(JAIR)*, 53:659–697.

- [11] Blom, H. A., Krystul, J., Bakker, G. B., Klompstra, M. B., and Obbink, B. K. (2006). Free flight collision risk estimation by sequential mc simulation. In *Stochastic hybrid systems*, pages 254–286. CRC Press.
- [12] Campbell, M., Hoane Jr, A. J., and Hsu, F.-h. (2002). Deep blue. *Artificial intelligence*, 134(1-2):57–83.
- [13] Cannel, D. and Markovitch, S. (1993). Learning models of opponent’s strategy game playing.
- [14] Chaloulos, G. and Lygeros, J. (2007). Effect of wind correlation on aircraft conflict probability. *Journal of Guidance, Control, and Dynamics*, 30(6):1742–1752.
- [15] Choi, S., Kim, Y. J., Briceno, S., and Mavris, D. (2016). Prediction of weather-induced airline delays based on machine learning algorithms. In *Digital Avionics Systems Conference (DASC), 2016 IEEE/AIAA 35th*, pages 1–6. IEEE.
- [16] Conde Rocha Murca, M., DeLaura, R., Hansman, R. J., Jordan, R., Reynolds, T., and Balakrishnan, H. (2016). Trajectory clustering and classification for characterization of air traffic flows. In *16th AIAA Aviation Technology, Integration, and Operations Conference*, page 3760.
- [17] Cowley, B., Charles, D., Black, M., and Hickey, R. (2009). Analyzing player behavior in pacman using feature-driven decision theoretic predictive modeling. In *Computational Intelligence and Games, 2009. CIG 2009. IEEE Symposium on*, pages 170–177. IEEE.
- [18] Di Ciccio, C., Van der Aa, H., Cabanillas, C., Mendling, J., and Prescher, J. (2016). Detecting flight trajectory anomalies and predicting diversions in freight transportation. *Decision Support Systems*, 88:1–17.
- [19] Dyster, T., Sheth, S. A., and McKhann, G. M. (2016). Ready or not, here we go: Decision-making strategies from artificial intelligence based on deep neural networks. *Neurosurgery*, 78(6):N11–N12.
- [20] Eerland, W. J., Box, S., and Sóbester, A. (2016). Modeling the dispersion of aircraft trajectories using gaussian processes. *Journal of Guidance, Control, and Dynamics*, pages 2661–2672.
- [21] Eurocontrol (2017). Eurocontrol Maastricht Upper Area Control Centre Annual Report 2017. <https://www.eurocontrol.int/sites/default/files/publication/files/muac-annual-report-2017.pdf>. [Online; accessed 10-April-2019].
- [22] Fogel, D. B., Hays, T. J., Hahn, S. L., and Quon, J. (2004). A self-learning evolutionary chess program. *Proceedings of the IEEE*, 92(12):1947–1954.

- [23] Fothergill, S. and Neal, A. (2008). The effect of workload on conflict decision making strategies in air traffic control. In *Proceedings of the human factors and ergonomics society annual meeting*, volume 52, pages 39–43. Sage Publications Sage CA: Los Angeles, CA.
- [24] Friedman, J., Hastie, T., and Tibshirani, R. (2001). *The elements of statistical learning*, volume 1. Springer series in statistics New York.
- [25] Fürnkranz, J. (1996). Machine learning in computer chess: The next generation. *ICGA Journal*, 19(3):147–161.
- [26] Gal, Y. and Ghahramani, Z. (2016). Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059.
- [27] Galway, L., Charles, D., and Black, M. (2008). Machine learning in digital games: a survey. *Artificial Intelligence Review*, 29(2):123–161.
- [28] Gianazza, D. (2017). Learning air traffic controller workload from past sector operations. In *ATM Seminar, 12th USA/Europe Air Traffic Management R&D Seminar*.
- [29] Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y. (2016). *Deep learning*, volume 1. MIT press Cambridge.
- [30] Hao, S., Cheng, S., and Zhang, Y. (2018a). A multi-aircraft conflict detection and resolution method for 4-dimensional trajectory-based operation. *Chinese Journal of Aeronautics*, 31(7):1579–1593.
- [31] Hao, S., Zhang, Y., Cheng, S., Liu, R., and Xing, Z. (2018b). Probabilistic multi-aircraft conflict detection approach for trajectory-based operation. *Transportation Research Part C: Emerging Technologies*, 95:698–712.
- [32] Hooshyar, D., Yousefi, M., and Lim, H. (2018). Data-driven approaches to game player modeling: A systematic literature review. *ACM Computing Surveys (CSUR)*, 50(6):90.
- [33] Hsieh, J.-L. and Sun, C.-T. (2008). Building a player strategy model by analyzing replays of real-time strategy games. In *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*, pages 3106–3111. IEEE.
- [34] Hu, J., Prandini, M., and Sastry, S. (2005). Aircraft conflict prediction in the presence of a spatially correlated wind field. *IEEE Transactions on Intelligent Transportation Systems*, 6(3):326–340.
- [35] ICAO (2007). Pans, air traffic management. *Doc-4444*.

- [36] ICAO (2017). The World of Air Transport in 2017. https://www.icao.int/annual-report-2017/Documents/Annual.Report.2017_Air%20Transport%20Statistics.pdf. [Online; accessed 10-April-2019].
- [37] Jan Hendrik, M. (2016). gp_extras. https://github.com/jmetzen/gp_extras.
- [38] Jenks, G. F. and Caspall, F. C. (1971). Error on choroplethic maps: definition, measurement, reduction. *Annals of the Association of American Geographers*, 61(2):217–244.
- [39] Jilkov, V. P., Ledet, J. H., and Li, X. R. (2018). Multiple model method for aircraft conflict detection and resolution in intent and weather uncertainty. *IEEE Transactions on Aerospace and Electronic Systems*, pages 1–1.
- [40] Kim, Y. J., Choi, S., Briceno, S., and Mavris, D. (2016). A deep learning approach to flight delay prediction. In *Digital Avionics Systems Conference (DASC), 2016 IEEE/AIAA 35th*, pages 1–6. IEEE.
- [41] Kuchar, J. K. and Yang, L. C. (2000). A review of conflict detection and resolution modeling methods. *IEEE Transactions on Intelligent Transportation Systems*, 1(4):179–189.
- [42] Kulkarni, V. B. (2015). Intelligent air traffic controller simulation using artificial neural networks. In *Industrial Instrumentation and Control (ICIC), 2015 International Conference on*, pages 1027–1031. IEEE.
- [43] Lauderdale, T. (2012). Probabilistic conflict detection for robust detection and resolution. In *12th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference and 14th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, page 5643.
- [44] Le, Q. V., Smola, A. J., and Canu, S. (2005). Heteroscedastic gaussian process regression. In *Proceedings of the 22nd international conference on Machine learning*, pages 489–496. ACM.
- [45] Lee, H., Malik, W., and Jung, Y. C. (2016). Taxi-out time prediction for departures at charlotte airport using machine learning techniques. In *16th AIAA Aviation Technology, Integration, and Operations Conference*, page 3910.
- [46] Lee, S. J., Liu, Y.-E., and Popovic, Z. (2014). Learning individual behavior in an educational game: a data-driven approach. In *Educational Data Mining 2014*. Citeseer.
- [47] Lewis, J., Trinh, P., and Kirsh, D. (2011). A corpus analysis of strategy video game play in starcraft: Brood war. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 33.

- [48] Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. (2015). Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*.
- [49] Liu, W. and Hwang, I. (2011). Probabilistic trajectory prediction and conflict detection for air traffic control. *Journal of Guidance, Control, and Dynamics*, 34(6):1779–1789.
- [50] Liu, Z., Cai, K., Zhu, X., and Tang, Y. (2017). Large scale aircraft conflict resolution based on location network. In *2017 IEEE/AIAA 36th Digital Avionics Systems Conference (DASC)*, pages 1–8. IEEE.
- [51] Machado, M. C., Fantini, E. P., and Chaimowicz, L. (2011). Player modeling: Towards a common taxonomy. In *Computer Games (CGAMES), 2011 16th International Conference on*, pages 50–57. IEEE.
- [52] Matsuno, Y. and Tsuchiya, T. (2014). Probabilistic conflict detection in the presence of uncertainty. In *Air traffic management and systems*, pages 17–33. Springer.
- [53] McNally, D., Erzberger, H., Bach, R., and Chan, W. (1999). A controller tool for transition airspace. In *Guidance, Navigation, and Control Conference and Exhibit*, page 4298.
- [54] 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.
- [55] Moravčík, M., Schmid, M., Burch, N., Lisý, V., Morrill, D., Bard, N., Davis, T., Waugh, K., Johanson, M., and Bowling, M. (2017). Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, 356(6337):508–513.
- [56] Nolan, M. (2010). *Fundamentals of air traffic control*. Cengage learning.
- [57] Odoni, A. R. (1987). The flow management problem in air traffic control. In *Flow control of congested networks*, pages 269–288. Springer.
- [58] Paielli, R. A. and Erzberger, H. (1997). Conflict probability estimation for free flight. *Journal of Guidance, Control, and Dynamics*, 20(3):588–596.
- [59] Pallottino, L., Feron, E. M., and Bicchi, A. (2002). Conflict resolution problems for air traffic management systems solved with mixed integer programming. *IEEE transactions on intelligent transportation systems*, 3(1):3–11.
- [60] Pham, D.-T., Alam, S., Yi-Lin, S., and Duong, V. (2018). A machine learning approach on past ads-b data to predict planning controller’s actions. In *8th International Conference on Research in Air Transportation (ICRAT ’18)*.

- [61] Pham, D.-T., Ngo, M.-M., Tran, N. P., Alam, S., and Duong, V. (2019a). Conflict prediction using generative model from ads-b data. In *2019 Integrated Communications, Navigation and Surveillance Conference (ICNS)*, pages 1–9. IEEE.
- [62] Pham, D.-T., Tran, N. P., Alam, S., Duong, V., and Delahaye, D. (2019b). A Machine Learning Approach for Conflict Resolution in Dense Traffic Scenarios with Uncertainties. In *ATM Seminar 2019, 13th USA/Europe ATM R&D Seminar*, Online proceedings at <http://www.atmseminar.org>, Vienne, Austria.
- [63] Pham, D.-T., Trant, N. P., Goh, S. K., Alam, S., and Duong, V. (2019c). Reinforcement learning for two-aircraft conflict resolution in the presence of uncertainty. In *2019 IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF)*, pages 1–6. IEEE.
- [64] Pollack, J. B. and Blair, A. D. (1998). Co-evolution in the successful learning of backgammon strategy. *Machine learning*, 32(3):225–240.
- [65] Prandini, M., Hu, J., Lygeros, J., and Sastry, S. (2000). A probabilistic approach to aircraft conflict detection. *IEEE Transactions on intelligent transportation systems*, 1(4):199–220.
- [66] Radanovic, M., Piera Eroles, M. A., Koca, T., and Ramos Gonzalez, J. J. (2018). Surrounding traffic complexity analysis for efficient and stable conflict resolution. *Transportation Research Part C: Emerging Technologies*, 95:105–124.
- [67] Ravizza, S., Chen, J., Atkin, J. A., Stewart, P., and Burke, E. K. (2014). Aircraft taxi time prediction: comparisons and insights. *Applied Soft Computing*, 14:397–406.
- [68] Schaeffer, J. (2001). A gamut of games. *AI Magazine*, 22(3):29.
- [69] Sethy, H., Patel, A., and Padmanabhan, V. (2015). Real time strategy games: a reinforcement learning approach. *Procedia Computer Science*, 54:257–264.
- [70] Shakarian, A. and Haraldsdottir, A. (2001). Required total system performance and results of a short term conflict alert simulation study. In *4th US/Europe Air Traffic Management R & D Seminar, Sante Fe*, pages 1–9.
- [71] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. (2016). Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489.
- [72] Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., and Riedmiller, M. (2014). Deterministic policy gradient algorithms. In *ICML*.
- [73] Snelson, E. (2006). Tutorial: Gaussian process models for machine learning. *Gatsby Computational Neuroscience Unit, UCL*.

- [74] Strohmeier, M., Schafer, M., Lenders, V., and Martinovic, I. (2014). Realities and challenges of nextgen air traffic management: the case of ads-b. *IEEE Communications Magazine*, 52(5):111–118.
- [75] Takeichi, N., Kaida, R., Shimomura, A., and Yamauchi, T. (2017). Prediction of delay due to air traffic control by machine learning. In *AIAA Modeling and Simulation Technologies Conference*, page 1323.
- [76] Tran, N. P., Pham, D.-T., Goh, S. K., Alam, S., and Duong, V. (2019). An intelligent interactive conflict solver incorporating air traffic controllers’ preferences using reinforcement learning. In *2019 Integrated Communications, Navigation and Surveillance Conference (ICNS)*, pages 1–8. IEEE.
- [77] Weiss, K., Khoshgoftaar, T. M., and Wang, D. (2016). A survey of transfer learning. *Journal of Big Data*, 3(1):9.
- [78] Wu, T. and Du, W. (2014). A distributed approach to aircraft conflict resolution based on satisficing game theory. In *Foundations of Intelligent Systems*, pages 383–393. Springer.
- [79] Yakovenko, N., Cao, L., Raffel, C., and Fan, J. (2016). Poker-cnn: A pattern learning strategy for making draws and bets in poker games using convolutional networks. In *AAAI*, pages 360–368.
- [80] Yang, L., Yang, J. H., Kuchar, J., and Feron, E. (2004). A real-time monte carlo implementation for computing probability of conflict. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, page 4876.
- [81] Yang, L. C. and Kuchar, J. K. (1997). Prototype conflict alerting system for free flight. *Journal of Guidance, Control, and Dynamics*, 20(4):768–773.
- [82] Yang, Y., Zhang, J., Cai, K., and Prandini, M. (2017). Multi-aircraft conflict detection and resolution based on probabilistic reach sets. *IEEE Transactions on Control Systems Technology*, 25(1):309–316.
- [83] Yannakakis, G. N. (2012). Game ai revisited. In *Proceedings of the 9th conference on Computing Frontiers*, pages 285–292. ACM.
- [84] Yannakakis, G. N., Spronck, P., Loiacono, D., and André, E. (2013). Player modeling. In *Dagstuhl Follow-Ups*, volume 6. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.
- [85] Yokoyama, N. (2018). *Decentralized Conflict Detection and Resolution Using Intent-Based Probabilistic Trajectory Prediction*. AIAA SciTech Forum. American Institute of Aeronautics and Astronautics.

- [86] Zook, A., Lee-Urban, S., Drinkwater, M. R., and Riedl, M. O. (2012). Skill-based mission generation: A data-driven temporal player modeling approach. In *Proceedings of the The third workshop on Procedural Content Generation in Games*, page 6. ACM.

Résumé

L'augmentation de la demande de trafic a mis à rude épreuve le système de contrôle de la circulation aérienne et les contrôleurs, d'où la nécessité d'un système novateur et efficace de détection et de résolution des conflits. Dans le cadre de cette thèse, nous nous concentrons sur les défis de la détection et de la résolution des conflits en utilisant des approches d'apprentissage automatique. Nous appliquons Random Forest pour apprendre et prédire les comportements des contrôleurs. Nous proposons également une approche axée sur les données pour la détection probabiliste des conflits en utilisant le processus gaussien hétéroscédastique comme modèles prédictifs et l'optimisation bayésienne pour la détection des conflits. Enfin, nous proposons un agent intelligent artificiel capable de résoudre les conflits, en présence de trafic et d'incertitude. La tâche de résolution de conflit est formulée comme un problème de prise de décision applicable à l'utilisation d'un algorithme d'apprentissage de renforcement. Notre travail comprend le développement d'un environnement d'apprentissage, la représentation des états des scénarios, la fonction de récompense et l'algorithme d'apprentissage. Les méthodes d'apprentissage automatique ont montré leur potentiel dans ces défis. Toutefois, d'autres études seraient menées pour améliorer leurs performances, telles que la représentation du réseau de l'espace aérien ou l'apprentissage du renforcement multi-agent.

Mots Clés

Détection et résolution de conflits, stratégie de contrôleur, apprentissage du renforcement, modélisation de trajectoire.

Abstract

The increasing in traffic demand has strained air traffic control system and controllers which lead to the need of novel and efficient conflict detection and resolution advisory. In the scope of this thesis, we concentrate on challenges in conflict detection and resolution by using machine learning approaches. We apply Random Forest to learn and predict controller behaviors. We also propose a data-driven approach for probabilistic conflict detection by using Heteroscedastic Gaussian Process as predictive models and Bayesian Optimization for conflict detection. Finally, we propose an artificial intelligent agent that is capable of resolving conflicts, in the presence of traffic and uncertainty. The conflict resolution task is formulated as a decision-making problem which is applicable for employing reinforcement learning algorithm. Our work includes the development of a learning environment, scenario state representation, reward function, and learning algorithm. Machine learning methods have shown their potential in those challenges. However, more studies would be conducted to improve their performances such as airspace network representation or multi-agent reinforcement learning.

Keywords

Conflict detection and resolution, controller strategy, reinforcement learning, trajectory modeling.