Cooperative Collision Avoidance for Unmanned Aerial Vehicles

by

Dinorego Mphogo

Thesis presented in partial fulfilment of the requirements for the degree of Master of Engineering (Electronic) in the Faculty of Engineering at Stellenbosch University

Supervisor: Dr J.A.A Engelbrecht

March 2020

Declaration

By submitting this thesis electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the sole author thereof (save to the extent explicitly otherwise stated), that reproduction and publication thereof by Stellenbosch University will not infringe any third party rights and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

Date: March 2020 . . . .

Abstract

This thesis presents the design, implementation, and verification of a cooperative collision avoidance algorithms for unmanned aerial vehicles (UAVs) in multi-aircraft conflict scenarios. Two types of collision avoidance algorithms are developed and verified in simulation: a rules-based algorithm and a cooperative path planning based algorithm.

The rules-based collision avoidance algorithm is modelled after the tactical Traffic Collision Avoidance System (TCAS) that is used on commercial passenger airliners. To enable multi-aircraft collision avoidance, two methods for combining the pairwise rules-based collision avoidance actions are proposed, namely Resolution Action Superposition (RAS) and pairwise Closest-Intruder-First (CIF).

The path planning based collision avoidance algorithm grows a search tree of admissible conflict resolution paths, and searches the tree to find the conflict-free path with the lowest cost. To enable cooperative collision avoidance, all aircraft communicate their current positions and intended flight paths to all other aircraft. A token allocation strategy is used so that the individual aircraft plan their new collision avoidance paths sequentially according to a predetermined priority order.

The rules-based and path planning based collision avoidance algorithms were implemented and verified in simulation. A simulation environment was created to test both the rules-based and path planning based collision avoidance algorithms. Set-piece conflict avoidance scenarios were performed to produce illustrative results. The simulations illustrated that both rules-based and path planning based collision avoidance can resolve both pairwise and multi-aircraft conflicts. Furthermore, Monte Carlo simulations were performed to produce statistical results and evaluate the performance of both algorithms in random conflict scenarios. The simulation results show that both the rules-based and path planning based solutions are able to successfully resolve collision scenarios involving multiple unmanned aerial vehicles. The rules-bases solution requires less computational effort but does not optimise the collision avoidance plans. The path planning based solution requires much more computational effort, but provides optimal solutions that minimises the deviation from the original flights, and minimises the control effort of the avoidance actions.

Opsomming

Hierdie tesis beskryf die ontwerp, implementasie, en verifikasie van samewerkende botsingvermyding algoritmes vir onbemande vliegtuie (UAVs) in multi-vliegtuig konflik scenarios. Twee tipes botsingvermyding algoritmes word ontwikkel en getoets in simulasie: 'n reëls-gebaseerde algoritme en 'n padbeplanning-gebaseerde algoritme.

Die reëls-gebaseerde algoritme word gemodelleer na die taktiese Traffic Collision Avoidance System (TCAS) wat gebruik word op kommersiële passasiersvliegtuie. Om multi-vliegtuig botsingvermyding te bewerkstellig, word twee metodes voorgestel om die paarsgewyse reëls-gebaseerde botsingvermyding aksies te kombineer, naamlik Resolusie Aksie Superposisie (RAS) en Naaste-Indringer-Eerste (NIE).

Die padbeplanning-gebaseerde botsingvermyding algoritme groei 'n soektogboom van toelaatbare botsingvermyding paaie, en deursoek dan die boom om die botsingvrye pad met die laagste koste te vind. Om samewerkende botsingvermyding te bewerkstellig, kommunikeer al die vliegtuie hulle huidige posisies en beplande vlugpaaie aan al die ander vliegtuie. 'n Token allokering strategie word gebruik sodat individuele vliegtuie hulle nuwe botsingvermyding paai sekwensieël beplan volgens 'n voorafbepaalde prioriteit volgorde.

Die reëls-gebaseerde en padbeplanning-gebaseerde botsingvermyding algoritmes is geïmplimenteer en getoets in simulasie. 'n Simulasie omgewing is geskep om beide die reëls-gebaseerde en padbeplanning-gebaseerde algoritmes te toets. Vooropgestelde botsingvermyding scenarios is uitgevoer om illustratiewe resultate te verkry. Die simulasies illustreer dat beide die revls-gebaseerde en padbeplanning-gebaseerde algoritmes in staat is beide paarsgewyse en multi-vliegtuig konflikte op te los. Monte Carlo simulasies is uitgevoer om statistiese resultate te lewer om die vermoë van beide algoritmes in lukrake konflik scenarios te evalueer. Die Monte Carlo simulasies wys dat beide die reëls-gebaseerde en padbeplanning-gebaseerde benaderings suksesvol lukrake multi-vliegtuig konflikte kan oplos. Die reëls-gebaseerde benadering vereis minder verwerkingskrag, maar optimiseer nie die botsinvermyding paaie nie. Die padbeplanning-gebaseerde benadering vereis meer verwerkingskrag, maar verskaf optimale oplossings wat die afwyking van die vliegtuie van hulle oorspronklike vlugplanne minimeer, en ook die beheerenergie van die vermydingsaksies minimeer.

Acknowledgements

- First and foremost, I would like to extend gratitudes to my study leader Dr JAA Engelbrecht for his consistent guidance and invaluable support during this thesis study which have allowed me to finish this research.

- My lovely parents, Matobola Mphogo and Mmatshehla Mphogo, for their dedicated support and love in my life.

- My younger two sisters, Lebogang and Ofentse for consistently asking when this research is finishing.

- To Helene Lambrechts for much assistance with grammar editing of this document.

- The JB Marks Education Trust Fund and Electronic Systems Laboratory(ESL) department bursary at Stellenbosch University for the financial assistance offered to me throughout this research.

- To Witold Buzantowicz the creator of Matlab package `flypath` useful for aircraft trajectory simulation which was used in this thesis. Simulation of UAV trajectories would have been less interesting without `flypath`.

- Gerju Goosen who made available the fixed-wing UAV flight data containing reality velocities used for the simulations.

- The Stellenbosch Electronic System Laboratory (ESL) postgraduate students who were with me in the lab between 2016 and 2019.

Dedications

To my late grandfather Mabuti Mathebula for the role he has played in my life. This is for you "Nchela mmina tshwene!".

Contents

List of Figures

List of Tables

Nomenclature

Useful Abbreviations

ADS-B	Automatic Dependent Surveillance - Broadcast
AGL	Above Ground Level
CA	Collision Avoidance
CIF	Closest Intruder First
CPP	Cooperative Path Planning
ESL	Electronic Systems Laboratory
GPS	Global Positioning System
NMAC	Near Mid-Air Collisions
RA	Resolution Alert
SAA	Sense-And-Avoid
SEEAA	See-And-Avoid
TCAS	Traffic Collision Avoidance System
TA	Traffic Alert
TCAS	Traffic Collision Avoidance System
UAV	Unmanned Aerial Vehicles

Constants

$\mathrm{x_{ra}} =$	0.35NM
$\mathrm{x_{ta}} =$	0.48NM
$\mathrm{z_{THR}} =$	650ft
$\tau_{\mathrm{ra}} =$	20s
$\tau_{\mathrm{ta}} =$	30s
$\mathrm{h_{AGL}} =$	1000ft

Variables

x	State vector	[ft × ft × s]
h	Altitude	[feet]
$\overline{V}$	Velocity Vector Magnitude	[feet/s]
$\bar{h}$	Altitude deviation mean	[feet]

$\dot{h}$	Altitude rate	[ft/s]
T	Sampling Period	[s]
t	Sampling time	[s]
τ	Time to collision	[s]
x	Horizontal position	[feet]
$\dot{x}$	horizontal position rate	[feet]
γ	Flight path angle	[degrees]

Chapter 1

Introduction

The introduction discusses the motivation and goal for undertaking the research, and formulates the research goals and objectives that we wished to achieve. We shall start with the research motivation, which is the necessity for a cooperative collision avoidance system for unmanned aerial vehicles to enable an integrated airspace with both manned and unmanned aircraft in future. We will then formulate the research goal which is to develop and evaluate cooperative collision avoidance algorithms using both rules-based and path planning based approaches. In addition, we will outline the achievable research objectives which shall be implemented to accomplish the research goal. Next, we provide an overview of the research project and the proposed cooperative collision avoidance solutions. Finally, we give an outline of the thesis report.

1.1 Research Motivation

The existing collision avoidance systems that are currently regulated within the civilian airspace have not been able to integrate both manned and unmanned aircraft. The focus in literature over the past few decades has mostly been on the development of collision avoidance systems for manned aircraft, such as the Traffic Collision Avoidance System (TCAS). However, collision avoidance systems for unmanned aircraft are gaining attention, due to the increasing usage of unmanned aerial vehicles (UAVs) for leisure and commercial purposes. The need for collision avoidance systems for unmanned aircraft is evident from the fact that an increasing number of conflict incidents in civilian airspace involving unregulated unmanned aircraft has been reported by the Federal Aviation Administration (FAA) [1].

To address this need, an intervention is required to introduce unmanned aerial vehicles into the commercial traffic management system. However, it has been reported that the existing TCAS cannot assist with collision avoidance which involves unmanned vehicle [2][3]. This is due to the fact that TCAS was introduced in 1987 after a series of reported accidents between commercial aircraft, therefore its design objective was to improve air transportation safety of large manned passenger. Furthermore, TCAS was designed to issue conflict resolution advisories that must be manually executed by the pilot, and was not designed to consider unmanned aerial vehicles.

Since TCAS was designed for manual conflict resolution, it will not be useful for collision avoidance that involves UAVs. This is due to the fact that TCAS will not be able to interface with the automatic flight control systems of unmanned aerial vehicles through its advisories. To integrate unmanned aircraft into civilian airspace, the collision avoidance systems must have the capacity to handle multiple intruder aircraft [4]. To address the issue of unmanned aircraft integration into the civilian airspace, the FAA has proposed the Automatic Dependent Surveillance Broadcast system (ADS-B) and Airborne Collision Avoidance System-X (ACAS-X) as the communication network system and the collision avoidance system to accomplish automatic collision avoidance. The existing ADS-B system provides the communications infrastructure for the implementation of cooperative collision avoidance systems, which is the topic of our research.

1.2 Research Goal

The research goal is to develop and evaluate cooperative collision avoidance algorithms for unmanned aerial vehicles in multi-aircraft conflict scenarios. Two types of collision avoidance algorithms must be investigated: a rules-based algorithm and a cooperative path planning based algorithm. It is assumed that all vehicles share their state and intent information with one another and possibly with a central node.

1.3 Research Objectives

The research objectives are as follows:

1. To develop a cooperative *rules-based* collision avoidance algorithm for unmanned aerial vehicles in a multi-aircraft conflict scenario.

2. To develop a cooperative *path planning based* collision avoidance algorithm for unmanned aerial vehicles in a multi-aircraft conflict scenario.

3. To create a simulation environment that can be used for set-piece conflict scenarios and Monte Carlo simulations.

4. To investigate and compare the behaviour of the rules-based and path planning based collision avoidance algorithms in set-piece conflict scenarios.

5. To test the performance of both the rules-based and path planning based collision avoidance algorithms using Monte Carlo simulations.

6. To compare and evaluate the performance of the rules-based and path planning based algorithms in terms of metrics such as the collision avoidance minimum time to collision, the minimum separation distance for rule-based algorithm. Furthermore, to introduce performance metrics for path planning based algorithm such as the

optimality of the solutions in terms of nominal altitude deviation costs, the path replanning rate, and the conflict resolution time.

1.4 Research Methodology

The research will be performed through the following methodology:

1. A literature study will be performed on existing collision avoidance systems for manned aircraft, previous research on collision avoidance for unmanned aerial vehicles, and cooperative path planning for autonomous vehicles.
2. The dynamics of a multi-aircraft system will be conceptualised and modelled.
3. An abstract communications framework for the unmanned aerial vehicles to share their state and intent information with one another and with a central node will be conceptualised.
4. A rules-based cooperative collision avoidance algorithm will be developed.
5. A path planning based cooperative collision avoidance algorithm will be developed.
6. A simulation environment will be created that simulates the vehicles, the terrain, the communication interface, and the cooperative collision avoidance algorithms.
7. Set-piece conflict scenarios will be simulated to produce illustrative results.
8. The illustrative simulation results will be analysed to investigate and compare the qualitative behaviour of the rules-based and path planning based collision avoidance algorithms.
9. Monte Carlo simulations will be performed to produce statistical simulation results.
10. The Monte Carlo simulation results will be analysed to evaluate and compare the quantitative performance of the rules-based and path planning based collision avoidance algorithms.

1.5 Previous Work

In this section, we review the previous works that are the basis for our rules-based and path planning based cooperative collision avoidance solutions. We first review the Traffic Collision Avoidance System (TCAS) which is the basis for our rules-based cooperative collision avoidance solution. We then review the conflict detection and resolution framework proposed by Van Daalen [5], and the cooperative path planning framework proposed by Shanmugal, Tsourdos, White, and Zbikowski [6], which are foundational for our path planning based cooperative collision avoidance solution.

1.5.1 Terminology Background

The following are terminologies adopted from previous work that will used in this thesis:

- A **Collision** is a physical collision between two or more aircraft.
- A **Conflict** is the violation of the safe separation zone around an aircraft.
- **Collision/Conflict Detection** is the process of detecting whether two or more aircraft are in violation of one another's safe separation zones at a given time instant. The safe separation zones are typically defined in terms of minimum separation distances and/or minimum allowable time to collision.
- **Collision/Conflict Prediction** is the process of predicting whether two or more aircraft will violate one another's safe separation conflict zones at some time instant in the future. Conflict prediction can be performed by calculating the time to collision between aircraft at the current time instant, or by propagating the aircraft states forward in time and performing conflict detection at regular time intervals along the predicted state trajectories.
- **Conflict Resolution** is the process of determining and executing appropriate collision/conflict avoidance actions to prevent a predicted collision/conflict from occurring.
- **Optimisation** is the process of determining the best sequence of control actions and the resulting best state trajectory that minimises some cost function or maximises some utility function. For the collision avoidance problem, we wish to minimise the deviations of the UAVs from their original flight plans, as well as the control effort of the collision avoidance actions.
- **Communication** is the sharing of the intent information, position and velocity (horizontal and vertical) of each aircraft in the communication network. The communication network also shares the token list to each UAV.
- **Cooperation** is the broadcasting of an aircraft's control inputs from TCAS rules-based algorithm or path planning token allocation (from [7]) algorithm for global collision avoidance.

1.5.2 TCAS, EGPWS, and ADS-B

Manned aircraft currently use TCAS (Traffic Collision Avoidance System), EGPWS (Enhanced Ground Proximity Warning System), and ADS-B (Automatic Dependent Surveillance) for collision avoidance. TCAS is a rules-based collision avoidance system that uses only vertical manoeuvres (climb and descend) to avoid aircraft-to-aircraft collisions. TCAS does not control the aircraft directly but advises resolution actions for the human pilot to follow. The resolution actions for the host aircraft are issued based on the instantaneous positions of the intruder aircraft relative to the host aircraft [8]. Each aircraft involved in the conflict scenario considers itself to be the host aircraft and all other aircraft to be the intruder aircraft. Each aircraft determines the position of other aircraft by interrogating their transponder units. Possible resolution actions for the host aircraft are "maintain level flight", "climb", "descend", "steep climb" and "steep descend". The TCAS unit applies a pre-defined set of collision avoidance rules to determine the instantaneous avoidance action for the host aircraft as a function of the instantaneous relative positions of the intruder aircraft. TCAS can resolve conflicts scenarios involving up to 17 aircraft simultaneously [9].

EGPWS is a rules-based ground collision avoidance system that uses only vertical manoeuvres to avoid aircraft collisions with terrain. The original GPWS (Ground Proximity Warning System) used the aircraft's altitude and a database of man-made structures and natural terrain to warn the pilot if the aircraft is too near to the ground [10][4]. However, since GPWS only considered the terrain under the aircraft, it could not warn the pilot of future collisions resulting from steep increases in the altitude of the terrain. EGPWS improves on GPWS by using the aircraft's GPS position to predict future changes in terrain.

ADS-B is a surveillance technology in which an aircraft determines its position via satellite navigation and periodically broadcasts it, enabling it to be tracked [3]. The information can be received by air traffic control ground stations, and can also be received by other aircraft to provide situational awareness and to allow self-separation.

It should be noted that TCAS, EGWPS, and ADS-B are separate and decoupled systems. Aircraft-to-aircraft avoidance, terrain avoidance, and communication of intent are therefore not currently integrated on collision avoidance systems for manned aircraft [11].

1.5.3 Conflict Detection and Resolution Framework (Van Daalen)

The conflict detection and resolution framework proposed by Van Daalen is shown in Figure 1.1. The framework consist of three modules: a Modelling module, a Conflict Detection module, and a Conflict Resolution module. The Modelling module contains models for the vehicle motion, the static environment, and the dynamic obstacles. The Modelling module may contain representations of uncertainties in the vehicle, environment, or dynamic obstacle models. The Conflict Detection module uses the Modelling module to propagate the vehicle state and detect future conflicts between the vehicle and the environment or with dynamic obstacles. Given the vehicle's planned path and the predicted paths of the dynamic obstacles, the Conflict Detection module determines whether the probability of a conflict exceeds a given threshold. The Conflict Resolution module determines a conflict-free path that minimises some cost function, while avoiding conflicts, and obeying the dynamic constraints of the vehicle. The Conflict Resolution module uses the Conflict Detection module to determine whether there are any conflicts predicted along a proposed path, and it uses the Modelling module to ensure that the proposed path obeys the vehicle's dynamic constraints.

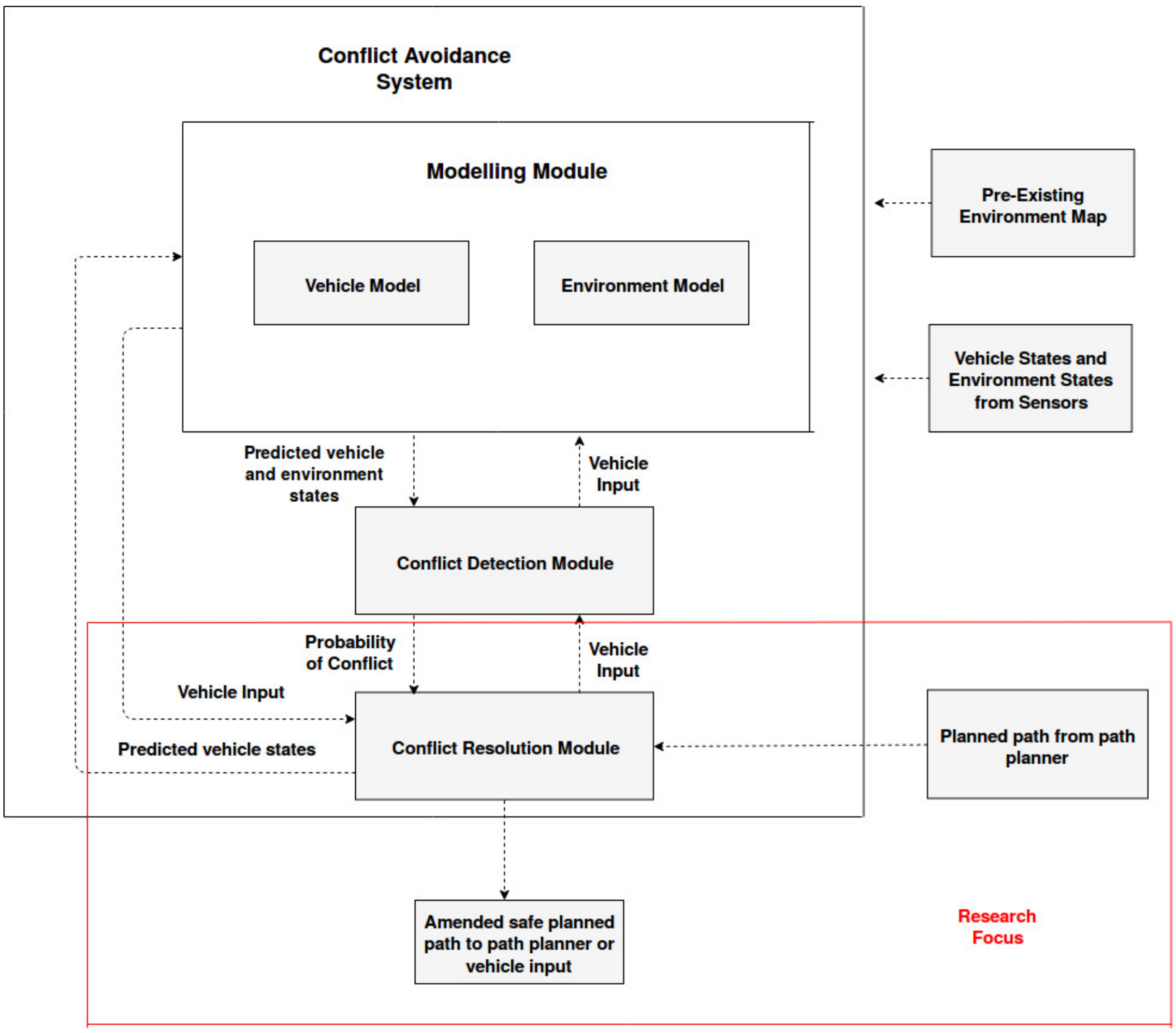

Conflict Avoidance System
Modelling Module
Vehicle Model
Environment Model
Pre-Existing Environment Map
Vehicle States and Environment States from Sensors
Predicted vehicle and environment states
Vehicle Input
Conflict Detection Module
Probability of Conflict
Vehicle Input
Vehicle Input
Predicted vehicle states
Conflict Resolution Module
Planned path from path planner
Amended safe planned path to path planner or vehicle input
Research Focus

Figure 1.1: Conflict detection and resolution framework proposed by Van Daalen [5]

The Modelling module may implement any suitable models for the vehicle, environment, and dynamics obstacles. The Conflict Detection module may implement any suitable conflict detection algorithm, and the Conflict Resolution module may implement any suitable path planning algorithm. The path planning algorithm may attempt to find only a feasible path, or may attempt to find the optimal path that minimises some cost function.

Van Daalen implemented a conflict detection algorithm based on probability flow for the Conflict Detection module, and implemented a kinodynamic sampling-based path planning algorithm for his Conflict Resolution module. The sampling-based path planning together with the probability flow conflict detection produced a conflict detection and resolution solution that is suitable for "cluttered, uncertain, and dynamic" environments.

The advantages of Van Daalen's framework is that it separates the conflict detection function and the conflict resolution function, whilst using modular components for modelling, conflict detection, and conflict resolution so that different algorithms can be implemented and evaluated.

Our proposed "path planning based cooperative collision avoidance" solution will extend Van Daalen's framework to perform cooperative path planning for unmanned aerial vehicles in multi-aircraft conflict scenarios. (Our solution will therefore be implemented in the components of Figure 1.1 highlighted in red.)

1.5.4 Cooperative Path Planning Framework (Shanmugal et al. [6])

The cooperative path planning framework proposed by Shanmugal et al. [6] is shown in Figure 1.2. The purpose of the framework is to coordinate the mission planning and task allocation for a group of unmanned aerial vehicles, to facilitate the cooperative trajectory planning for all of the UAVs in the group, as well as to enable each individual UAV to execute its planned reference trajectory. The framework consists of three layers. The first layer performs the cooperative mission planning and task allocation. The Mission Planning module receives the current states of all of the UAVs and assigns the missions and the tasks for each UAV. The second layer performs the cooperative trajectory planning for all UAVs. The Cooperative Trajectories module receives the assigned missions and tasks for each UAV and plans the reference trajectories for all of the UAVs. The third layer is responsible for executing the planned trajectories. Therefore, each UAV has its own individual controller which allows it to follow its individual reference trajectory using feedback control.

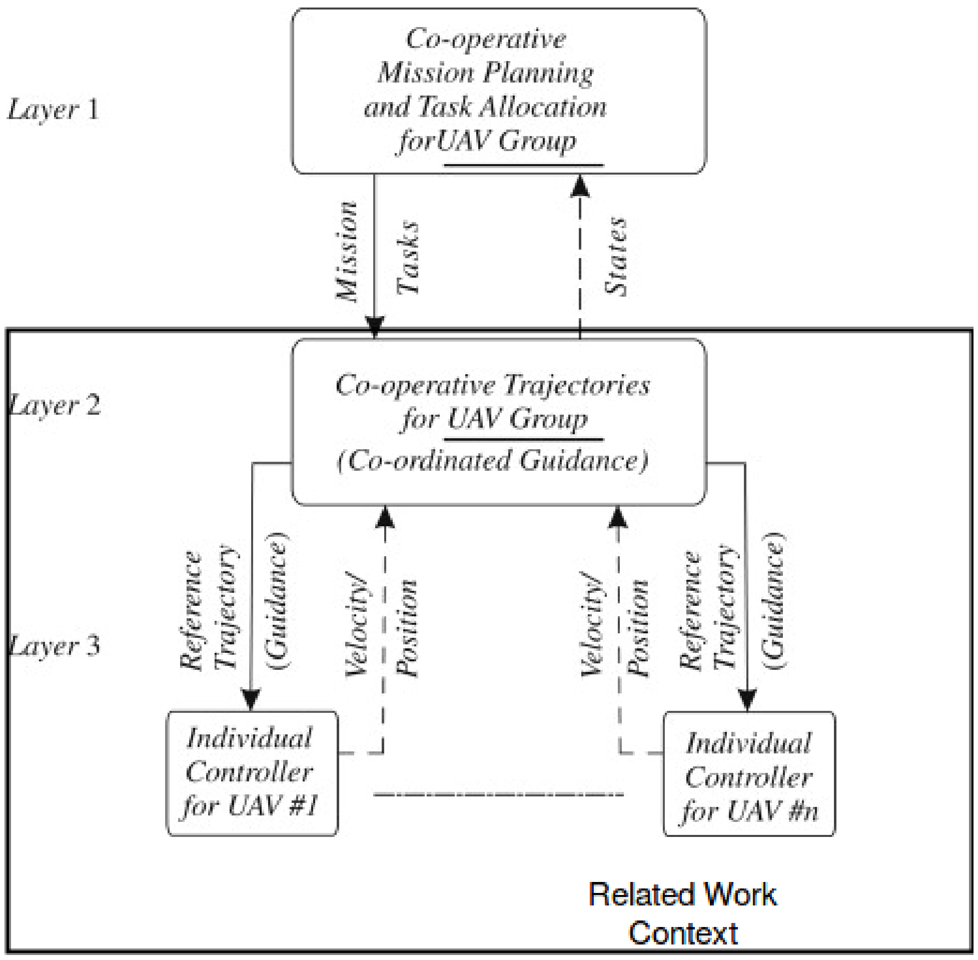

Layer 1
Co-operative Mission Planning and Task Allocation forUAV Group
Mission
Tasks
States
Layer 2
Co-operative Trajectories for UAV Group
(Co-ordinated Guidance)
Layer 3
Reference Trajectory (Guidance)
Velocity/ Position
Velocity/ Position
Reference Trajectory (Guidance)
Individual Controller for UAV #1
Individual Controller for UAV #n
Related Work Context

Figure 1.2: System architecture for Cooperative Path Planning of multiple UAVs [6]

1.6 Overview of Proposed Solutions

We propose two solutions for cooperative collision avoidance for unmanned aerial vehicles: a rules-based algorithm and a cooperative path planning based algorithm. The rules-based collision avoidance algorithm is modelled after the Traffic Collision Avoidance System (TCAS) that is used on commercial passenger airliners. To enable multi-aircraft collision avoidance, two methods for combining the pairwise rules-based collision avoidance actions are proposed, namely Resolution Action Superposition (RAS) and pairwise Closest-Intruder-First (CIF). The path planning based collision avoidance algorithm grows a search tree of admissible conflict resolution paths, and searches the tree to find the conflict-free path with the lowest cost. To enable cooperative collision avoidance, all aircraft communicate their current positions and intended flight paths to all other aircraft. A token allocation strategy is used so that the individual aircraft plan their new collision avoidance paths sequentially according to a predetermined priority order. Aircraft with higher priority ignore lower priority aircraft when planning their new paths, while aircraft with lower priority treat higher priority aircraft as dynamic obstacles that must be avoided.

1.6.1 Rules-Based Cooperative Collision Avoidance Solution

The rules-based collision avoidance algorithm is modelled after TCAS that is used on commercial passenger airliners. The rules-based collision avoidance solution uses only vertical manoeuvres (climb and descend) to avoid aircraft to aircraft collisions. All co-operative aircraft continuously communicate their position and velocity information to one another. In a pairwise collision avoidance scenario, each aircraft determines its own avoidance action as a function of the time to collision and the relative altitude of the other aircraft. Figure 1.3 shows the rules that are used to determine the avoidance action for the host aircraft as a function of the relative position of the intruder aircraft.

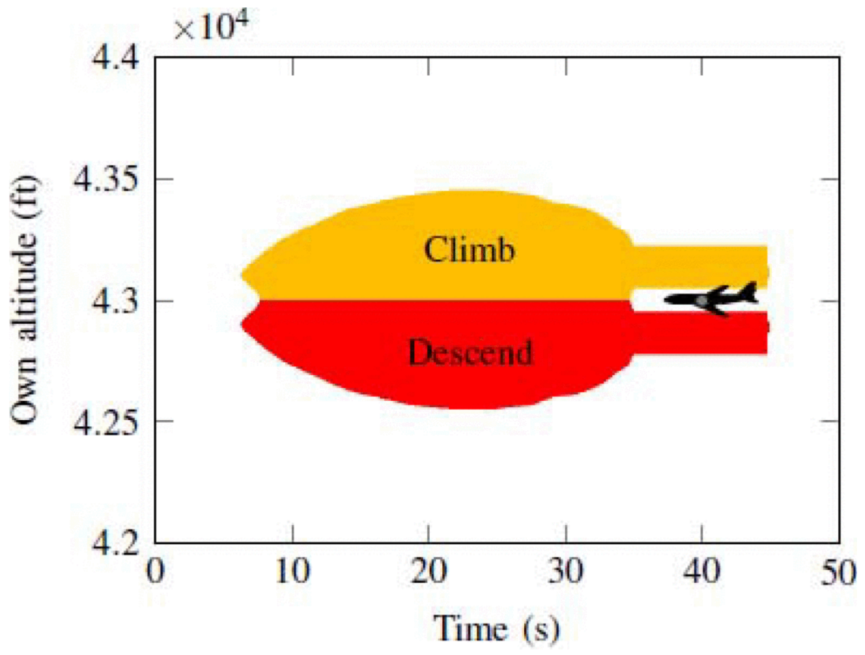


Figure 1.3: TCAS conflict resolution policy with vertical distance (y-axis) and time-to-collision (x-axis) from [12]

Five possible collision avoidance actions are available, namely "maintain level flight", "climb", "descend", "steep climb" and "steep descend". If the intruder aircraft is outside the collision avoidance region, then the action for the host aircraft is "maintain level flight". If the intruder aircraft is inside the collision avoidance region, but still far enough away, then the avoidance action for the host aircraft is either "climb" or "dive", depending on whether the intruder aircraft is at a lower or higher altitude than the host aircraft. If the intruder aircraft is inside the collision avoidance region and close to the host aircraft, then the avoidance action is either "steep climb" or "steep dive". Once the intruder aircraft has passed the host aircraft, the host aircraft returns to its original flight path.

In order to extend the rules-based solution from pairwise collision avoidance to multi-aircraft, two methods for combining the pairwise rules-based collision avoidance actions are proposed, namely Resolution Action Superposition (RAS) and pairwise Closest-Intruder-First (CIF). The RAS approach first determines the individual pairwise collision avoidance actions for the host aircraft with respect to each individual intruder aircraft, and then combines the individual pairwise collision avoidance actions into a single multi-aircraft collision avoidance action. The CIF approach only determines and executes the pairwise collision avoidance action with respect to the closest intruder aircraft (smallest time to collision).

1.6.2 Path Planning Based Cooperative Collision Avoidance Solution

The path planning based collision avoidance solution grows a search tree of admissible conflict resolution paths, and searches the tree to find the conflict-free path with the lowest cost. Each node of the search tree represents the state of the aircraft at a discrete time instant. Nodes are added to the tree by iterating through a finite set of actions (the same set of actions used by TCAS) at discrete time instants, propagating the aircraft state, checking for predicted collisions, and only adding conflict-free nodes to the tree. To enable cooperative multi-aircraft collision avoidance, all aircraft communicate their current positions and intended flight paths to all other aircraft. A token allocation strategy is used to select a single aircraft that is then given a turn to re-plan its own flight path while assuming that all the other aircraft will continue to follow their published flight plans. Aircraft with higher priority ignore lower priority aircraft when planning their new paths, while aircraft with lower priority treat higher priority aircraft as dynamic obstacles that must be avoided. The aircraft then communicates its new flight path (replacing its previous flight path) and gives the next aircraft a turn.

The architecture of the path planning based cooperative collision avoidance solution is shown in Figure 1.4. The first layer contains the Communication Hub that stores and distributes the published collision avoidance paths for all of the cooperative aircraft. The Communication Hub also stores the priority order for all of the UAVs and facilitates the token passing. The second layer contains the cooperative search-based path planning algorithm. The cooperative path planner receives the original flight paths of all the aircraft, their initial states when the conflict avoidance is triggered, and their priority order. The path planner then sequentially plans the new collision avoidance paths for all of the aircraft and stores them in the Communication Hub. The new collision avoidance flight paths are stored in the Communication Hub. The third layer contains the Aircraft Controllers that control the individual aircraft. Each aircraft has its own individual altitude controller. When the aircraft is in normal flight mode, the altitude controller controls the aircraft to follow the original flight path; when the aircraft is in collision avoidance mode, the altitude controller controls the aircraft to execute the vertical collision avoidance actions planned by the cooperative path planner. The fourth layer contains the original flight paths that were planned for normal flight.

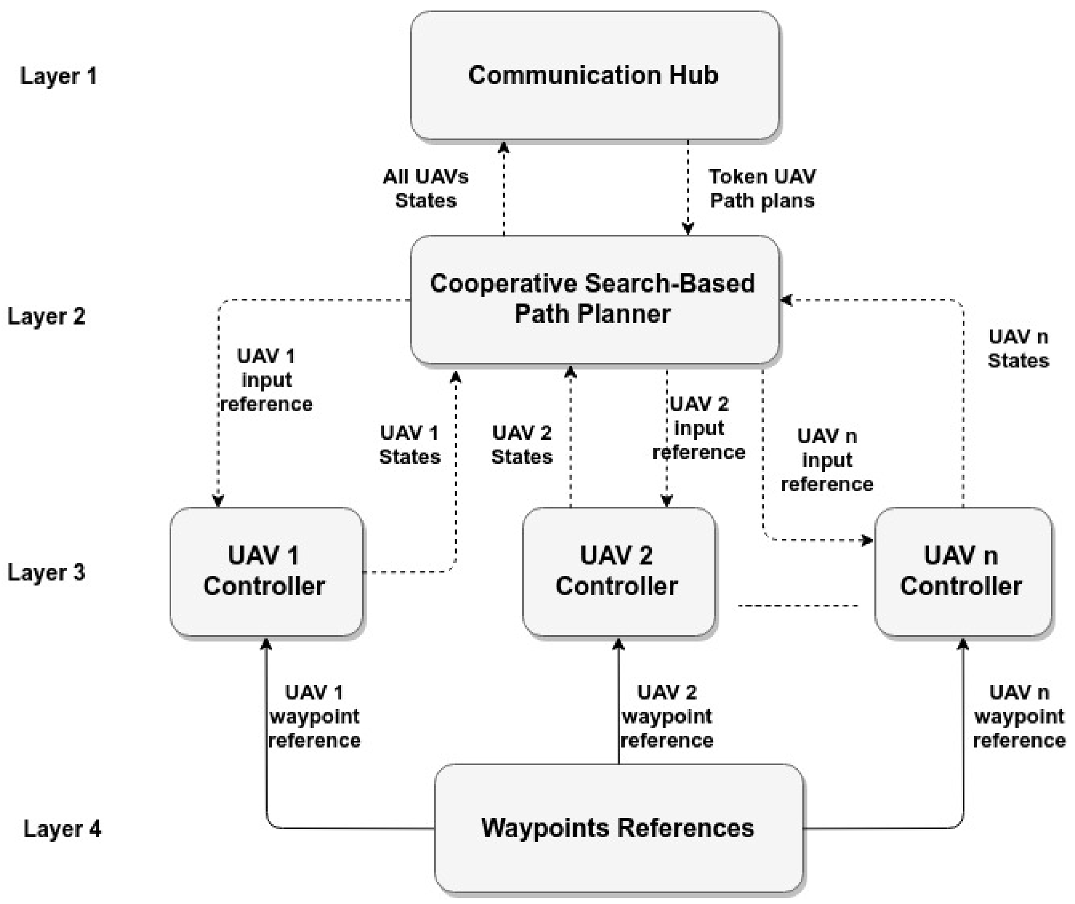


Figure 1.4: Proposed high level view of a Token Allocation communication framework for a Cooperative Path Planning of Multiple UAVs

1.6.3 Thesis Overview

- **Chapter 1: Introduction** discusses the motivation for undertaking the research, and formulates the research goals and objectives that we wished to achieve, it also provides an overview of the research methodology, briefly reviews the previous works that are the basis for our solution, and further gives an overview of our proposed solutions, finally outline the research report.

- **Chapter 2: Literature Review** presents a literature review of previous research on collision avoidance. The literature review covers terminology and definitions, the state of the art of existing collision avoidance systems for manned aircraft, proposed collision avoidance systems for unmanned aircraft, and related research on cooperative path planning algorithms.

- **Chapter 3: Conceptualisation and Modelling** conceptualises modelling the multi-aircraft collision avoidance problem and establishes mathematical models for the unmanned aircraft, their flight control systems, and their static and dynamic environment.

- **Chapter 4: Rules-Based Collision Avoidance** presents our rules-based cooperative collision avoidance solution. Two solutions are proposed for multiple aircraft collision avoidance, namely Resolution Action Superposition (RAS), and pairwise Closest-Intruder-First (CIF). Simulations are performed for set-piece conflict scenarios to evaluate the behaviour of the algorithms.

- **Chapter 5: Path Planning Based Collision Avoidance** introduces the path planning based approach to collision avoidance for unmanned aerial vehicles. Simulations are performed for pairwise collision avoidance scenarios to demonstrate the behaviour of the algorithm.

- **Chapter 6: Cooperative Collision Avoidance** extends the path planning algorithm to enable cooperative collision avoidance for multiple unmanned aircraft. A token allocation strategy is defined so that the individual aircraft plan their new collision avoidance paths sequentially according to a predetermined priority order. Simulations are performed to demonstrate the cooperative collision avoidance.

- **Chapter 7 : Monte Carlo Simulations** presents Monte Carlo simulations that were performed to evaluate and compare the rules-based and path planning based solutions. The simulation setup is described and the simulation results are analysed and discussed.

- **Chapter 8 : Conclusions** provides a summary of the work done, presents the main findings, and gives recommendations for future research.

Chapter 2

Literature Review

This chapter presents a literature review of previous research on aircraft collision avoidance. The literature review covers terminology and definitions, the state of the art of existing collision avoidance systems for manned aircraft, proposed collision avoidance systems for unmanned aircraft, and related research on cooperative path planning algorithms. In Section 2.1 we will study different stages of Conflict Modelling (from a survey [4]) for conflict detection and a resolution process known as States Propagation and Conflict Metrics. Thereafter, in Section 2.2 we shall consider the existing collision avoidance systems such as TCAS and GPWS for manned aircraft. Afterwards, an unmanned aircraft conflict formulation is discussed in Section 2.3. Then, several general planning approaches to collision avoidance for multiple unmanned aircraft are furthermore examined in Section 2.4. These general planning approaches include configuration-space planning, search space solutions, numerical Markovian decision making processes and sampling-based path planning. Thereafter in Section 2.5, our discussion will look into multi-robot path planning approaches such as the principled negotiation, centralised planning, and decentralised planning. Section 2.6 then provides an overview of performance metrics that are used in literature to evaluate the performance of collision avoidance systems, both for manned and unmanned aircraft. Finally, we discuss the key highlights of the literature study and our subsequent research decisions.

2.1 Conflict Modelling Background

In this section we consider the definition of conflict, different conflict detection techniques, and different conflict resolution techniques for aircraft as found in literature. The full process of conflict detection and resolution as discussed by Kuchar and Yang in their survey [4] is shown in Figure 2.1. Unmanned and manned aircraft use two different technologies for conflict detection and resolution, namely the Sense-And-Avoid (SAA) and See-And-Avoid (SEEAA). A conflict defines the abstraction of static or dynamic margins for safe separation between aircraft. There are three types of approaches to a conflict detection based on how the aircraft states are propagated forward in time, namely the Nominal, Worst-Case, and Probabilistic approaches. Two general approaches to aircraft conflict resolution will be considered, namely pairwise resolution and multi-aircraft resolution.

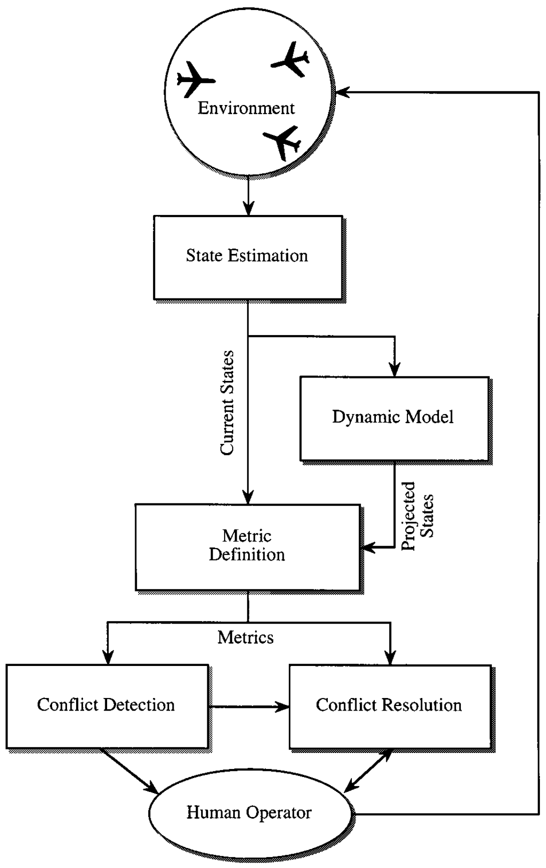


Figure 2.1: Example conflict detection and resolution process modelling [4]

2.1.1 Definition of Conflict

The term *conflict* can be defined as the state in which the minimum separation between aircraft is violated. The conflict could occur between two or more aircraft that are on collision courses. The conflict conditions are derived from a set of self-separation rules for aircraft which are regulated by aviation authorities such as the International Civil Aviation Organization (ICAO). Therefore, regulation bodies use two separate rules for SAA and SEEAA for the conflict detection and avoidance of manned and unmanned aircraft respectively. The fundamental difference between SAA and SEEAA is the level of human intervention in the control loop of aircraft during the conflict resolution stage. Hence, the SAA rules are for unmanned aircraft conflict detection and resolution which relies on surveillance technology for automation [2]. Due to the differences between communication technologies that are used for SAA purposes, different definitions of conflict exist, particularly for the coordination of multi-aircraft conflicts.

Each UAV has an abstract conflict zone with dimensions which define where other aircraft are not allowed to enter. Communication networks therefore play an important role in tracking the positions and velocities of all aircraft in the airspace which enables each UAV to independently maintain the safe separation. The process of monitoring the separation distance to detect whether an aircraft's conflict zone has been violated is called conflict detection and will be explored in detail in the next section. The self-separation conflict zone for an aircraft is nominally defined in literature as shown in Figure 2.2 as a cylindrical shape with a height of 2000 feet and a radius of 5 NM. However, other confict-zone dimensions with different safety margins are used for aircraft at other altitude levels [25] and different conflict-zone dimensions are also used in congested areas such as airport environments where airborne collision avoidance systems are not activated [13]. For the research performed for this thesis, we will adopt conflict- zone dimensions that are similar to the one shown in Figure 2.2 for our design and simulations.

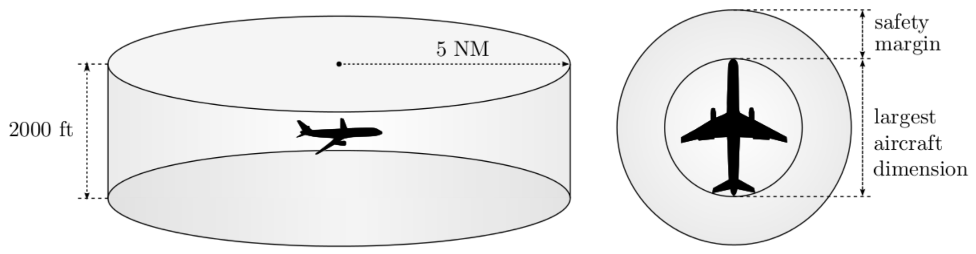


Figure 2.2: Example of self-separation conflict region of a large aircraft [13]

2.1.2 Conflict Prediction Techniques

Conflict prediction is performed by thresholding the overlap between the conflict zones of two or more aircraft after simulating the aircraft states into the near future. Conflict prediction determines the quantity of the overlap between the conflict zones of the aircraft over the course of their planned trajectories. The output of the conflict prediction is the point in time in the future when the states of two or more aircraft will be in conflict [14][13]. The aircraft state simulation is used to estimate over the short or long term when safe separation will be violated, either because the minimum separation distance or the minimum time to collision is violated [15]. Therefore, a conflict detection algorithm decides *when* an aircraft should start with conflict resolution actions to avoid a collision with a terrain as a static obstacle, or with another aircraft as a dynamic obstacle.

The simulation methods used for aircraft conflict prediction are shown in Figure 2.3. There are three aircraft states projection techniques that are used to predict conflict with multiple other intruder aircraft in the airspace. The state propagation technique enables an aircraft to detect conflict with other aircraft forward in time, which is called conflict prediction. The three states propagation techniques are the Nominal, Worst Case and Probabilistic approaches [4]. Each of these methods uses a different level of modelling complexity for the ownship and the intruder aircraft. Therefore, should uncertainty be introduced into the aircraft state propagation, it will lead to additional complexity of conflict prediction.

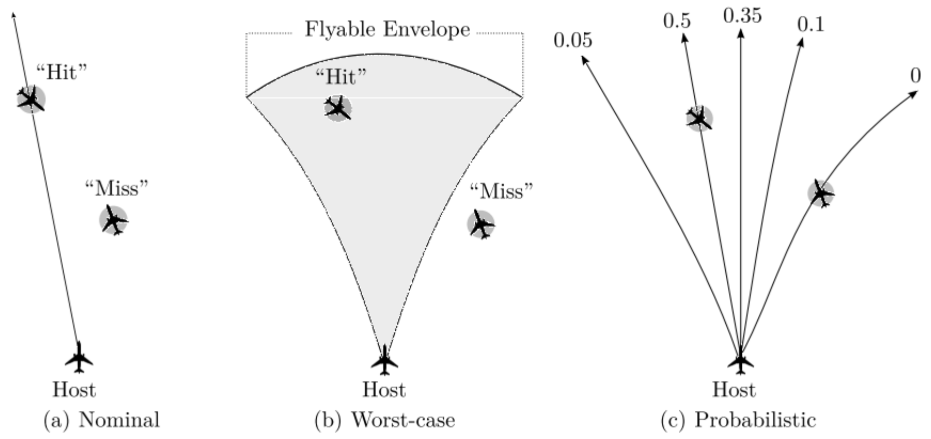


Figure 2.3: Existing trajectory propagation approach from survey by Kuchar and Yang [4]

The *Nominal* dynamic states projection for aircraft is shown in Figure 2.3(a). The Nominal trajectory projection approach assumes that there is no uncertainty in the state propagation of the host aircraft or the intruder aircraft. Hence, conflict prediction using the nominal trajectory approach is less computationally expensive to implement. The *Worst-case* approach as shown in Figure 2.3(b) assumes that both the host aircraft and the intruder aircraft are equally likely to perform all possible manoeuvres within their dynamic constraints and times. The result of the Worst-case states projection, as shown in Figure 2.3(b), is a "flyable envelope" of all possible manoeuvres within the simulation time horizon, which is then used for conflict prediction. Hence, the Worst-case state projection is more computationally expensive compared to the simple nominal approach because the method has to cover a larger state-space for conflict detection.

Both the Nominal and Worst-case states propagation for conflict detection do not make uncertainty assumptions regarding the states propagation of the host aircraft. Hence, the *Probabilistic* approach as shown in Figure 2.3(c) assumes that there is some uncertainty in the state propagation [13]. The "Probabilistic Host" approach combines aspects of both of the Nominal and Worst-Case approaches into the state propagation while considering the uncertainty. The probabilistic approach assumes that the actual host aircraft trajectory follows the nominal trajectory, but with an uncertainty described with a known probability distribution function. The probabilistic approach therefore calculates the probability of the conflict, given the nominal flight plan and the uncertainty in the state propagation. For our research, we will use the nominal state propagation approach, with the assumption that the conflict zone already contains sufficient margin for safety to accommodate the uncertainty in the state propagation.

2.1.3 Conflict Resolution Techniques

Conflict resolution is the processes of finding a set of conflict-free paths for an aircraft to avoid a predicted conflict. Conflict resolution takes place after a conflict has been predicted using either a rules-based prediction technique or one of the state propagation techniques described in the previous section. Thereafter, conflicts that have been detected have to be avoided through either tactical response or strategic planning [16]. When a conflict involves a manned aircraft, the pilot receives conflict advisories in the form of audio feedback that have to be executed manually [3]. Unmanned aircraft require a sequence of collision avoidance actions that can be executed by the onboard autopilot system [17]. Generating conflict-free paths in form of aircraft command inputs is called path planning and it is the focus of our research.

Aircraft conflict resolutions are dynamic in nature due to the fact that conflict detection processes dynamically simulate the states. Conflict resolution for more than two aircraft require coordination and multi-objective trajectory planning. Therefore, predicted conflicts involving multiple aircraft can either be resolved as a strategic long-term or a tactical short-term problem [4]. The strategic long-term solution takes longer to compute and the state propagation is subject to uncertainty as flight plans are estimated as shown in Figure 2.3. Hence, either the Worst-Case or the Probabilistic state propagation approaches should be used to perform conflict prediction when strategic conflict resolution is performed for multiple aircraft. On the other hand, the Nominal state propagation approach should be used to perform conflict prediction when short-term tactical conflict resolution is performed, since it takes a shorter time to compute and the state propagation contains less uncertainty. Two approaches to conflict resolution for multiple aircraft have been proposed in literature, namely pairwise resolution and global resolution. These two approaches are illustrated in Figure 2.4. In the following subsection we will elaborate on the fundamental difference between the two approaches in terms of computational complexity.

2.1.3.1 Pairwise Conflict Resolution for Multiple Aircraft

The pairwise solution approach performs conflict resolution for pairs of aircraft at a time. Pairwise solutions take less time to compute and are intended for short-term conflict resolution. Pairwise solutions only focus on a single intruder, and do not offer coordination with more than one intruder. The pairwise solution approach therefore focusses on two aircraft at a time, ignoring all other aircraft in the environment [18][4].

The pairwise approach is less computationally intensive, and sub-optimal solutions for multi-aircraft conflict scenarios can be generated relatively quickly. However, pairwise solutions can be limiting or short-sighted for multi-aircraft conflict resolution. Therefore, multiple pairwise solutions have to be combined to find a global solution and checks must be performed to determine whether new conflicts were introduced [4].

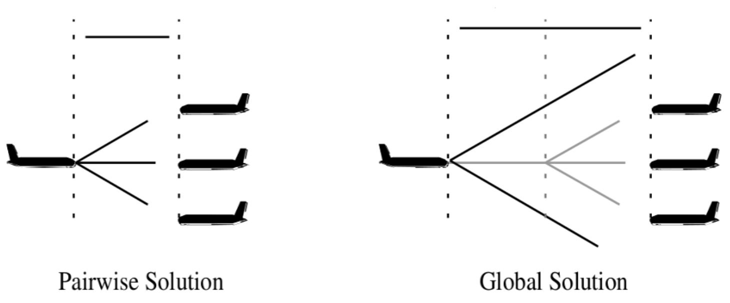


Figure 2.4: Pairwise approach in comparison to global approach for collision avoidance resolution adopted from Kuchar and Yang's survey [4].

2.1.3.2 Global Conflict Resolution for Multiple Aircraft

The Global Solution approach is a more time-consuming conflict resolution approach. The global solutions for multiple aircraft conflict scenarios take time to compute because they globally optimise the resolutions [9]. Global conflict resolution is a holistic approach in which aircraft are cooperative. Therefore, aircraft take turns to propose a solution for the predicted conflict. Hence, in the global solution approach more than one conflict manoeuvre will be considered and thus, the global solution approach performs a global multi-objective conflict resolution optimisation that considers all of the aircraft involved in the predicted conflict.

The limitation of the global solution approach is that it has to take all of the aircraft into account when resolving the conflict, and it therefore takes much longer to execute [12].

2.2 Existing Systems for Manned Aircraft

In this section we discuss the existing conflict detection and resolution systems for manned aircraft. We begin with an overview of the existing systems that have historically been used for both airborne and ground-based alert systems [18]. Thereafter, we shall briefly discuss the Ground Proximity Warning System (GPWS) as one of the early primary surveillance systems which were introduced with radio altimeter systems to be used for terrain collision avoidance. Next, we will look into the wireless communication technologies which were later introduced for secondary surveillance for conflict detection with signal sharing between aircraft. This enabled aircraft to establish coordination between the ground station and other aircraft in systems such as the TCAS. Finally, we shall discuss the proposed NextGen collision avoidance system intended for the integrated civilian airspace. The NextGen system was introduced with the goal to enable automated airborne collision avoidance in which the Free-Flight concept is used and both manned and unmanned aircraft can operate together in airspace.

2.2.1 Manned Aircraft Systems Approach

Collision Avoidance systems for manned aircraft can be classified into three categories: short term, medium term and long term. Short term collision avoidance systems are intended to be used whilst airborne with the aide of on-board communication system and Air Traffic Control (ATC). Conflict resolutions for short-term collisions are expected to resolve conflicts in less than a minute [4]. The midterm conflict resolutions systems are also designed for airborne usage. However, these resolutions are designed to resolve conflicts that are predicted a few minutes into the future. Lastly, the long-term collision avoidance system works closely with ground-based systems such as GPWS, and alerts aircraft pilots of conflicts that are predicted longer than 30 minutes into the future [11]. Therefore, long-term collision avoidance systems are useful for separation assurance, because conflict advisories are issued by ground-based flight management which does not demand expensive onboard electronics [3].

Secondary surveillance is a radar system that not only detects and measures the position of aircraft, such as bearing and distance, but also requests additional information from the aircraft itself, such as its identity and altitude. Unlike primary radar surveillance, secondary surveillance enables coordination between aircraft and flight ground stations where a broadcast communication network shares the navigation data of all the aircraft in the airspace. TCAS is one of the systems which introduced the aircraft state broadcast mechanism for conflict detection and resolution between large aircraft. Therefore, we shall discuss TCAS conflict avoidance as one good example of systems based on a bi-direction Mode-S communication secondary surveillance device [3]. This is due to the fact that Mode-S transponders can handle coordination between one or more aircraft.

2.2.2 Ground Proximity Warning System (GPWS)

GPWS is a non-cooperative system which is one of the first generation systems to use primary radar surveillance for non-communication conflict detection with static objects such as terrain within a limited range. This system is useful for high altitude terrain detection as a warning system to alert pilots. However, the fact that GPWS is non-communicating means that it has limited application to conflict resolution for multiple aircraft because the aircraft would not be able to cooperate. In the review of conflict detection and resolution by Kuchar and Yang, the GPWS is described as a basic altitude terrain detection system which advises the aircraft pilot with alerts such as "Pull up". The following quote was used to describes the GPWS [4]:

> "GPWS does not perform additional computation to determine an optimal escape maneuver"

The basic functionality of the GPWS is to detect the altitude level of the aircraft above the terrain using an onboard radio altimeter. The Honeywell company later introduced an improved system called the Enhanced Ground Proximity Warning System (EGPWS) [10]. EGPWS improved on GPWS by including a GPS database of terrain obstacles. However, EGPWS only detects terrain and does not detect other aircraft within range. Hence, GPWS and EGPWS are only useful for terrain awareness and for alerting the pilot to avoid collisions with terrain obstacles. EGPWS provides the following functions for terrain collision avoidance [10]:

- Forward Looking Terrain Avoidance (FLTA) is a function which looks ahead of time below the aircraft's lateral and vertical flight path to provides suitable alerts if a potential controlled flight into terrain (CFIT) threat exists.
- Descent Alert (DA) uses the aircraft's current position and flight path information as determined from a suitable navigation source and airport database to determine if the aircraft is hazardously below the normal approach path for the nearest runway (typically with a flight path angle of 3 degrees) as defined by the alerting algorithm.
- A class of equipment which provides indication of imminent contact with the ground under certain conditions; excessive rates of descent, excessive closure rate to terrain, negative climb rate or altitude loss after take-off, flight into terrain when not in landing configuration and voice call-out "Five Hundred" when the aircraft descends to 500 feet above the nearest runway elevation.

2.2.3 Traffic Collision Avoidance System (TCAS)

TCAS is one of the first communication-based systems to provide collision avoidance functions for manned aircraft. This system was introduced after several historic accidents in civilian aviation involving passenger aircraft [18]. Thereafter, TCAS was introduced with the goal to use it in aircraft to aircraft communication as an interrogation mechanism to determine intruder state variables for conflict detection over the secondary radar surveillance system [19].

The interrogation mechanism used by TCAS is for interrogation between aircraft which is useful for determining the relative range, and altitude every second. Because the host aircraft receives intruder aircraft surveillance data each second, TCAS is able to numerically derive other useful dynamic data including the range, relative closure rate, and relative altitude rate. The derived dynamic parameters are used to compute an important concept to TCAS, namely the time to collision, or tau. Time to collision (τ) is an important variable used by TCAS for conflict prediction. In the following sections, we will discuss how TCAS was developed and how it functions in more detail.

2.2.3.1 Interrogation and Coordination

The state estimation implemented by TCAS starts by performing interrogation and coordination using the aircraft's onboard transponder system, as shown in Figure 2.5. An aircraft that is equipped with a Mode-S transponder can interrogate other TCAS equipped intruder aircraft (acting as signal interrogation receiver). The interrogation is transmitted at a frequency of 1090 MHz, and the reply is received at a frequency of 1030 MHz [19]. In addition other aircraft which are equipped with an ordinary transponder but without on-board TCAS can also be interrogated. However, all aircraft with Mode-S transponders receive conflict resolution advisories at a frequency of 1030 MHz and send replies at a frequency of 1090 MHz [19]. The coordination scheme used by TCAS is shown in Figure 2.5.

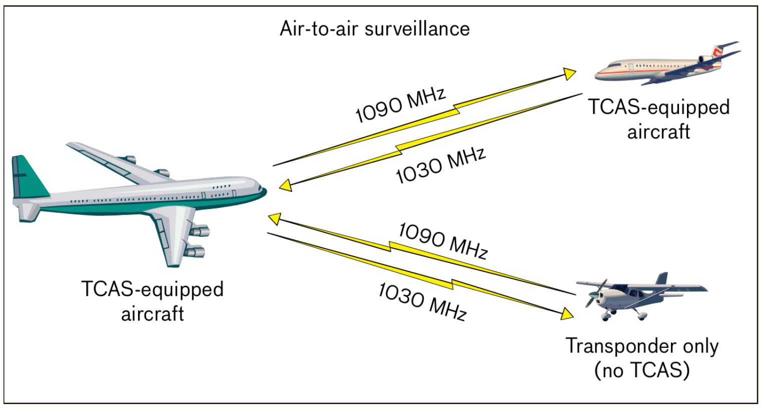


Figure 2.5: Large aircraft conflict interrogation and coordination [3]

The conflict resolution coordination scheme enables TCAS to select a single aircraft (with the lowest transponder number) to be nominated as the host to initiate intruder interrogations [8]. Using the concept of host aircraft nomination, TCAS will be able to coordinate complementary conflict resolution actions. TCAS therefore performs interrogation and coordination each second to issue conflict resolution advisories that will ensure that each aircraft in the predicted conflict arrives at a safe separation state.

2.2.3.2 Conflict Resolution Advisories

The given aircraft conflict scenario in Figure 2.6 illustrates the simple numerical safety logic which TCAS uses to predict and resolve conflict. The TCAS safety logic uses Traffic Advisory (TA) and the Resolution Advisory (RA) envelopes, are separation regions that trigger the different stages of conflict prediction an resolution [3]. If an intruder aircraft enters the traffic advisory (TA) envelope of the host aircraft, then TCAS issues an audio alert to the pilot with the warning "Traffic, Traffic" to alert the pilot to the presence of the intruder aircraft. However, at this stage TCAS does not issue a recommended collision avoidance manoeuvre yet. If an intruder aircraft enters the resolution advisory (RA) envelope of the host aircraft, then TCAS calculates the appropriate collision avoidance action and issues an audio conflict resolution advisory to the pilot.

The objective of the conflict resolution advisories computed by TCAS is to maximise the vertical or altitude separation distance at the closest point of approach. TCAS has

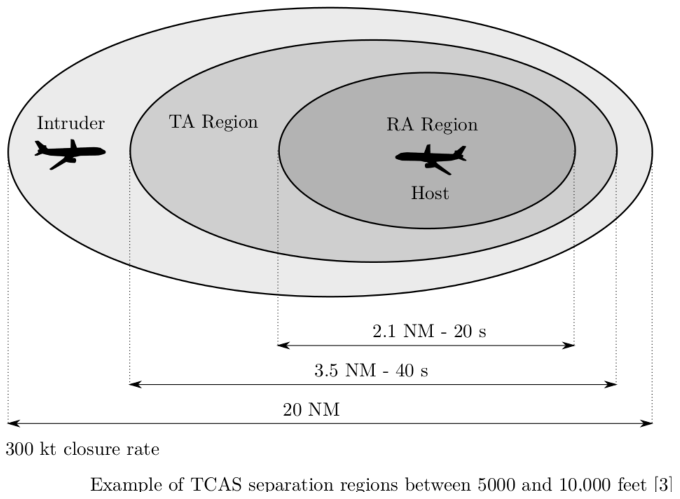


Figure 2.6: TCAS Conflict Traffic Alert and Resolutions Alert Regions for two aircraft

historical design limitations because it only uses vertical manoeuvres to avoid collision, and does not use horizontal manoeuvres such as heading or speed adjustments. The resolution advisory approach used by TCAS is similar to other reactive short-term collision avoidance approaches discussed in a survey by Kuchar and Yang [4]. Ultimately, TCAS operates by issuing instantaneous climb rate / descent rate commands based on the current time to collision and relative altitude separation of the host and intruder aircraft, with the aim of maximising the vertical separation distance at the Closest Point of Approach (CPA) [3].

2.2.4 NextGen Air-Traffic Control

The Next Generation of Airborne Collision Avoidance (NextGen) system is the FAA's future integrated airspace with automated air traffic control. The NextGen systems would be integrated by a communication network in order to implement Sense-And-Avoid (S&A) shown in Figure 2.7. The objective of NextGen is to use the communication network with the onboard flight controllers to implement the Free Flight concept, and to provide an integrated airspace that contains both manned and unmanned aircraft.

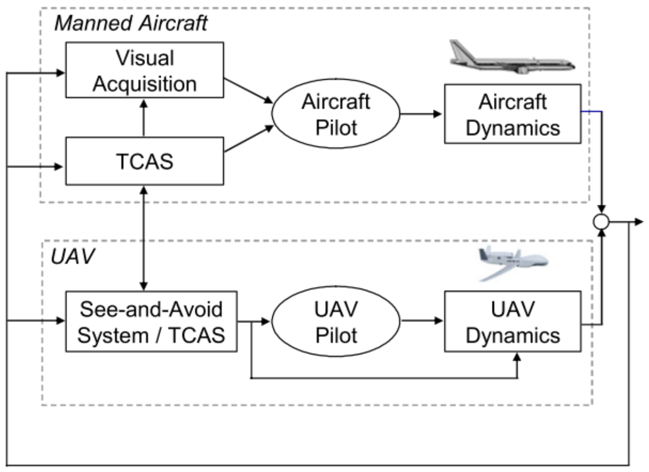


Figure 2.7: Integrated airspace collision avoidance systems for unmanned and manned aircraft systems adopted from [10]

The literature on NextGen technology towards the integration of both manned and unmanned aircraft covers the digitalization of the current widely deployed TCAS system. In 2008 a proposal to replace TCAS with the Airborne Collision Avoidance System-X (ACAS-X) for manned aircraft was made [18]. The approach proposed by NextGen is to equip UAVs with secondary surveillance technology (See and Avoid) to enable them to communicate and select trajectories freely without the control of ATC. NextGen relies on cooperative technologies such as the ADS-B communication system for accurate surveillance data to make decisions; these we shall discuss in the next Section 2.2.4.1 [3].

2.2.4.1 Sensor Measurements

The sensor measurements are used to obtain surveillance information about threats to host the aircraft within sensor range. Sensor measurements are further used to obtain the real-time positions and velocities of the host aircraft and the intruder aircraft so that they can be used for conflict prediction. Currently, surveillance sensors measurements are mainly used by the TCAS system, which uses the secondary surveillance to obtain aircraft intent information, such as time, altitude, vertical rate, and heading. The proposed ACAS-X would reuse the same on-board radar Mode-S protocol hardware as TCAS, but with a different approach to the resolution advisory logic [3]. The other major improvement to ACAS-X surveillance data, which is different from TCAS, is the introduction of an automatic surveillance system called ADS-B as part of the navigation sensor technologies.

ADS-B is a secondary surveillance technology which is bi-directional in communication between aircraft, and also offers fast and more accurate GPS positioning surveillance data. The communication network does not only offer cooperation between aircraft, but also shares data with the ground station which is then distributed to remote air-traffic controls centres for higher decision making. Aircraft that are equipped with ADS-B are capable of broadcasting position data which is immediately made available to other aircraft. The broadcasting is useful to other systems such as the ACAS-X, because such a system will be able to propagate a probabilistic state model from sensor data [3].

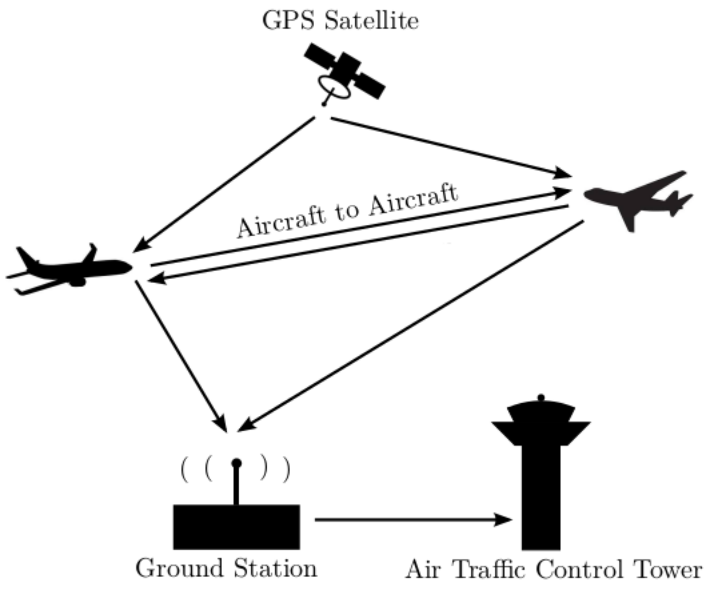


Figure 2.8: ADS-B Communication system for NextGen systems from [13]

There are a few functionality benefits when using ADS-B for conflict surveillance. According to the ADS-B design objectives, communication sensors benefits civilian airspace Air Traffic Control functions as follows:

- Cooperative Dependent Surveillance (CDS) technology will be fully certified by the FAA and ICAO.
- ABS-B-In and ADS-B-Out in each aircraft will be capable of providing accurate and long range GPS positioning suitable for conflict detection and resolution services.
- Both horizontal and vertical safe separation with improved safety to manage large traffic and aircraft fleet will be provided.

2.2.4.2 Future ACAS-X Conflict Avoidance System

The Lincoln Laboratory research team at MIT studied the ACAS-X program with the aim of overcoming the existing logic limitations of TCAS [3]. While the conflict resolution process for TCAS is rules based, the ACAS-X conflict resolution process uses an optimised numeric lookup table, as shown in Figure 2.9. When ACAS-X is eventually implemented, the role of Air-Traffic-Management (ATM) would be affected and the traffic complexity reduced to make improvements to the surveillance architecture. Therefore, ACAS-X will replace the TCAS conflict resolution advisory rules with a numerical logic table that uses a computational optimisation technique called Dynamic Programming (DP) to optimise the resolution actions in the lookup table.

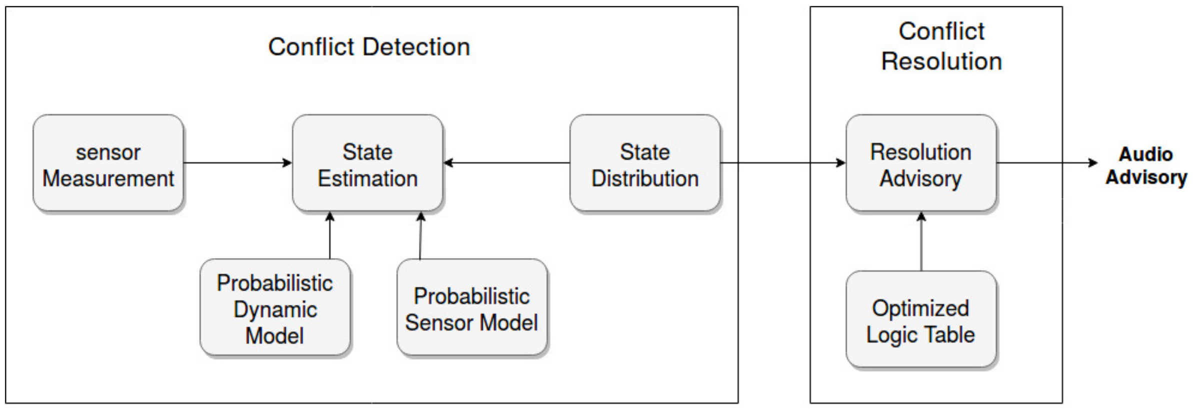


Figure 2.9: Logic based NextGen collision avoidance system

The state estimation block uses the sensor measurements, a probabilistic dynamic model of the aircraft, and a probabilistic sensor models. The state estimation used by TCAS utilizes nominal state propagation, and is therefore limited compared to the probabilistic state estimation used by ACAS-X [3]. The state distribution block passes the state information at the current and previous time steps to the resolution advisory block. The resolution advisory block uses the state information as inputs into an optimsed lookup table to determine the optimal resolution advisories. (The lookup table is generated off-line using the dynamic programming optimisation algorithm.) The optimisation objectives range from minimisation of false alerts, minimisation of reversal of advisories by the system to maximisation of safety.

2.3 Conflict Prediction and Resolution for Unmanned Aircraft

The main purpose of collision avoidance systems on unmanned aircraft is to enable the UAV to navigate the environment. Recently, a new installation of TCAS has linked its vertical avoidance manoeuvre commands to the autopilot system so that the manoeuvres can be executed automatically [20]. Two approaches for multi-agent conflict resolution are highlighted in literature; these are the Tactical Responsive [21] and the Long-Term Strategic approaches [22][23]. From recent work, an intelligent agent approach to representation of information sharing has been previously studied for unmanned aircraft decision making process such as the reward policy [24][25]. The following sections explore earlier approaches to conflict detection and resolution by mobile robotics.

2.3.1 Unmanned Aircraft Systems Approach

The conflict avoidance problem for unmanned aerial vehicles is similar to negotiation between intelligent agents in a multi-agent system. The approaches that are widely used in literature for multi-agent systems are the Markov Decision Process (MDP) [26] and cooperative multi-agent path planning [26]. The conflict formulation with MDP and Motion Path planning were done in [27]. At the highest level, both methods are computational in nature and the conflict resolution uses a state-control policy to determine the appropriate control action based on the estimated state which is again based on observations from the environment [26]. An agent is therefore a process that performs actions based on states observed in the environment [28], as shown in Figure 2.10. This agent process is similar to the conflict detection and resolution process described in Figure 2.1 in the first section of this chapter, only without the "Human Operator".

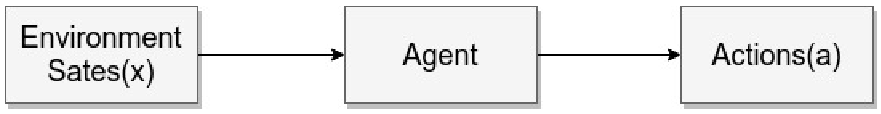


Figure 2.10: Agent in an environment making observations of a single agent [28]

Literature thus far has considered two main conflict sensor technologies suitable for unmanned aircraft, Electro-Optical and Passive Radar [29]. The measurements obtained from the two sensors are for non-cooperative conflict resolution and each has limitations in conflict detection alert accuracy. ACAS-X assumes a probabilistic state distribution of the aircraft states to address the sensor model uncertainty introduced by the measurement or process stages [3]. Therefore, the state estimation makes use of aircraft dynamic modelling and sensor model information to better estimate the states.

2.3.2 Conflict Zone and Detection

The geometrical approach to conflict detection and resolution uses geometric shapes to represent paths that are flyable by a UAV given its dynamic constraints. A nominal geometric path replanning approach was discussed in [23], to perform pairwise conflict resolution for UAVs using a cylindrically-shaped conflict zone which is defined around each aircraft. Therefore, a pair of aircraft are said to be in a geometric conflict if both the vertical and the horizontal separation distances are predicted to fall below the thresholds [30][21]. The geometric approach is also used to estimate the time t when two aircraft will lose safe separation at the Closest Point of Approach (CPA) [21]. Furthermore, geometric conflict detection between pairwise aircraft with constant ground speeds v_o and v_i for aircraft o and i can be used to estimate the times $t_1 \neq t_2$ when the conflict starts and ends respectively. The following is a conflict detection inequality of a distance threshold D projected to a closest separation time τ from [21]:

$$|(p_o - p_i) + \tau(v_o - v_i)|^2 < D^2 \tag{2.1}$$

The general Velocity Obstacle (VO) technique proposed by Wilkie, Berg and Manocha [31], considers both a geometric and dynamic approach between two agents in motion. VO computes a set of all the velocity values for each agent that leads to a conflict. The algorithm creates a dynamic disk shape that represents the conflict region around the aircraft or its sensing agent. Furthermore, it is expected that the agent will be surrounded with both static and dynamic obstacles. Therefore, the dynamic state variables (position, velocity, and time) of an agent A is used to describe a conflict zone defined by the velocity field $V_{A|B} \subseteq \mathcal{V}$. This can also be interpreted by the agent as a geometric conflict region between two agents A and B as defined in [31], shown in Figure 2.11, and outlined by Equation 2.2.

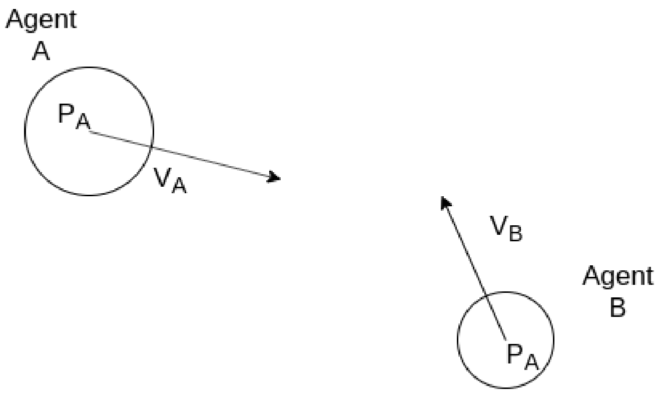


Figure 2.11: Velocity obstacle of two agents in motion

The two agents which are shown in Figure 2.11 have their own radius distance r_A and r_B respectively that they use for conflict detection in their environment. Therefore, the geometric abstract region $\mathcal{B}$ is a relative distance disk region located at $P_B - P_A$ which is likely to lead to a collision with aircraft B within time horizon t .

$$V_{A|B} = \{v \mid \exists t > 0,\ P_A + t(v - v_B) \in \mathcal{B}\} \tag{2.2}$$

$$\mathcal{B} = \{v \mid d(P_B - P_A, r_A + r_B)\} \tag{2.3}$$

2.3.3 Collision Avoidance

A reactive collision avoidance approach is short-term in that it only reacts to avoid conflict condition violation. The reactive approach does not consider longer-term conflicts, and is therefore likely to create new medium-term or long-term conflicts in the process of avoiding short-term conflicts. A long-term solution to the limitation of reactive conflict resolution is a path planning based strategic conflict resolution, such as discussed by Kuchar [4] which we will deal with later. One of the earlier reactive collision avoidance approaches for indoor robotic systems was proposed by Fox, Burgard and Thrun in [32].

Their approach is known as Dynamic Window Approach (DWA) for dynamic obstacle environments. DWA only considers the search space of robot velocity commands which can be executed over a short time of 0.25 seconds to "avoid the enormous complexity of the general motion planning problem" [32]. Collision avoidance is achieved through the optimization of an objective function $J(\cdot)$ using the admissible velocity commands v, ω and the trajectories that are reachable under the dynamic constraints of the planning robot.

The cost function J is formulated as:

$$J(v,\omega) = \sigma(\alpha \cdot A(v,\omega) + \beta \cdot B(v,\omega) + \gamma \cdot C(v,\omega)) \tag{2.4}$$

where $A(v,\omega)$ is a measure towards the goal location, $B(v,\omega)$ is the closest distance penalty towards an obstacle and $C(v,\omega)$ is the forward velocity maximization towards the goal.

Yasunki and Yoshiki developed a collision avoidance method for multiple autonomous mobile agents that performs cooperative collision avoidance using reactive collision resolution based on the foundations of the velocity obstacle technique [33]. Their cooperative collision avoidance system makes an assumption that the agents have same geometric description and that they implement the the same conflict resolution algorithm. A modified version of the Velocity Obstacle (VO) approach, called the Cooperative Velocity Obstacle (CVO) approach, was introduced to allow implicit agent cooperation.

2.4 General Path Planning

This section considers path planning as a strategic collision avoidance framework for unmanned aircraft that allows the paths to be optimised. First, we provide an overview of a hierarchical autonomous path planning framework for aircraft that is partitioned into different levels of planning, namely mission requirement, goalpoint planning, high-level trajectory planning, trajectory planning, and safety planning. The path planning framework also involves modelling of the aircraft dynamics, the static obstacles, and the dynamic intruder aircraft. Thereafter, we review different path planning approaches, namely Configuration Space, Search Space, Numerical Optimization, Sampling-Based, Artificial Heuristic, Planning Under Differential Constraint, and Path Planning Uncertainty.

2.4.1 Overview

A hierarchical autonomous path planning framework for unmanned aircraft is shown in Figure 2.12 [17]. The mission requirements tier defines a starting location, a finish location, mission waypoints, threat regions, and restricted regions. The goalpoint planning tier plans the path from the start to the finish via the mission waypoints without ignoring the threat regions and the restricted region. The high-level trajectory planning tier modifies the path to avoid threat regions. The trajectory planning tier modifies the planned path further to create flyable paths that adhere to the constraints of the vehicle dynamics.

Finally, the safety planning tier modifies the planned trajectory to avoid local traffic. The following subsections will review different path planner design methods and path search optimization approaches found in literature.

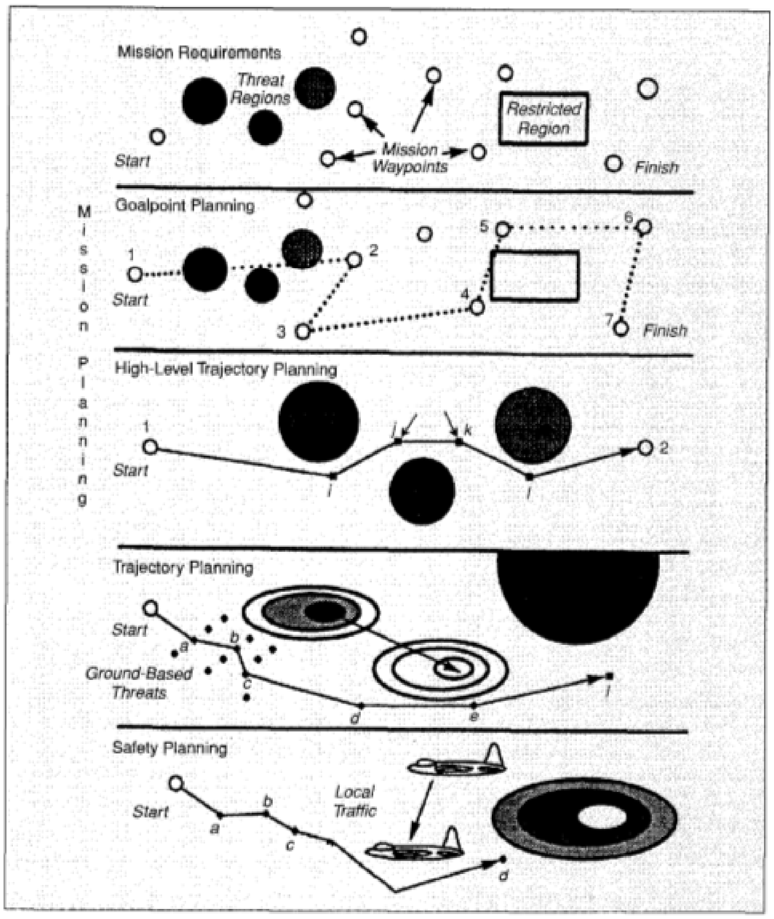


Figure 2.12: Autonomous Path Planning framework from [17]

2.4.2 Configuration Space Planning

The configuration space (C-space) is a data structure which enables a path planning algorithm to computationally explore each possible manoeuvre geometrically with or without vehicle dynamics constraints [34]. The data structure used by a path planning algorithm is a graphs or trees, and a search algorithm for optimization of path segments for mobile robots in [35]. An existing work approach to configuration space path planner in 2.13 by Shanmugavel, Tsourdos, White and Zbikowski in a study of multiple unmanned aircraft path planning [6].

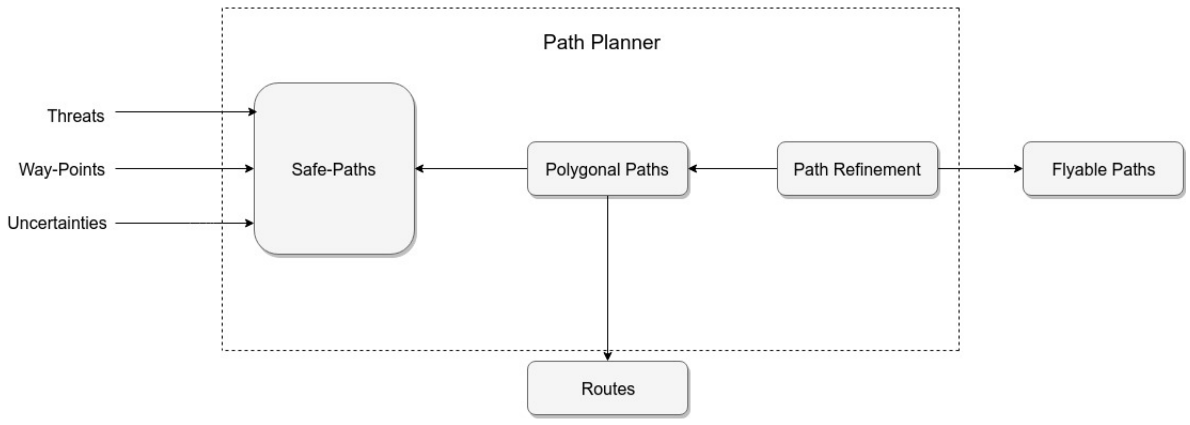


Figure 2.13: Path planning framework as defined in literature [6] and [36]

Tsourdos et al. [36] proposed an integrated goal-orientated path planner to generate flyable, conflict-free paths for unmanned aircraft. A block diagram of their path planning framework is shown in Figure 2.13. A C-space data structure was created that takes threats, waypoints, and uncertainties in the environment into account. A search algorithm was then used to find flyable paths in the C-space, that considers both the vehicle dynamic constraints and the obstacles in the environment.

2.4.3 Search-Based Approaches

Search-based approaches use tree or graph structures and search algorithms to plan collision-free paths. The tree or graph structure is created to take the obstacles and dynamics constraints of the vehicle into account, and the search algorithm is used to explore the tree or graph structure to find the feasible or optimal path from some initial state to some goal state.There are two categories of searching called informed and uninformed [37], which are useful for exploration of graph and tree data structures. The uninformed search algorithms are exploratory in nature and the aim is to discover the connectivity of the obstacle-free configuration space without prior knowledge and without assigning costs to compare the costs of paths. The Breadth First Search (BFS) and Depth First Search (DFS) are two notable uninformed search methods [37]. The goal of an uninformed search is to find best connected nodes within the configuration planning space.

An informed search attaches a cost to each transition between the nodes of a data structure tree, so that the costs of candidate paths can be determined and compared, and a heuristic function is used to guide the search using domain knowledge of the search problem. The heuristic function reduces the number of iterations taken to find the optimal path by ensuring that the lowest cost paths are explored first. Pruning techniques may also be used to reduce the size of the search tree without affecting the optimality of the solution found [37]. The A-star algorithm is an example of an informed search algorithm that is used to find optimal paths in a grid configuration space [38].

2.4.4 Numerical Optimisation Approaches

Numerical optimisation approaches are suitable for solving optimal planning problems with multiple objectives and constraints. Examples of numerical optimisation techniques include Sequential Quadratic Programming (SQP) [39]and Mixed Integer Linear Programming (MILP) [40]. Mellinger et al. [40] used mixed integer linear programming to perform aircraft trajectory planning with collision avoidance.

The application of MILP is suitable for high-level planning for the optimisation of global objectives. Challenges to collision avoidance with a MILP formulation are computational time complexity and poor scalability, the limitations of linear constraint problems, and static obstacles. A collision avoidance problem with MILP was largely successful in the study by Richards [40]. The multi-objective optimal collision avoidance problem can be written in the following general form :

$$\text{minimize} \sum_{i=1}^{N} J(\cdot) \tag{2.5}$$

$$\text{subject to } D_{ij} - d_{ij} \geq 0 \tag{2.6}$$

$$U_{min} \leq u_i \leq U_{max} \tag{2.7}$$

Where J is a cost function, d_{ij} is minimum separation distance D_{ij} of pairwise UAVs, U_{min} and U_{max} are minimum and maximum control effort u_i.

A method to optimise the TCAS safety logic using a Markov Decision Process formulation was proposed by Kochenderfer and Chryssanthacopoulos [41]. The logic optimization discretize the aircraft states and the actions of an aircraft pilot to form a policy of dynamic states transition through a closed-loop Markov Decision Process (MDP) model shown in Figure 2.14. The MDP decisions are optimised using a numerical Dynamic Programming (DP) algorithms as discussed in [42][26]. The MDP transition to future states depends only on the current state, therefore discrete DP is used to optimize the MDP reward-cost function such as vertical separation distance [43].

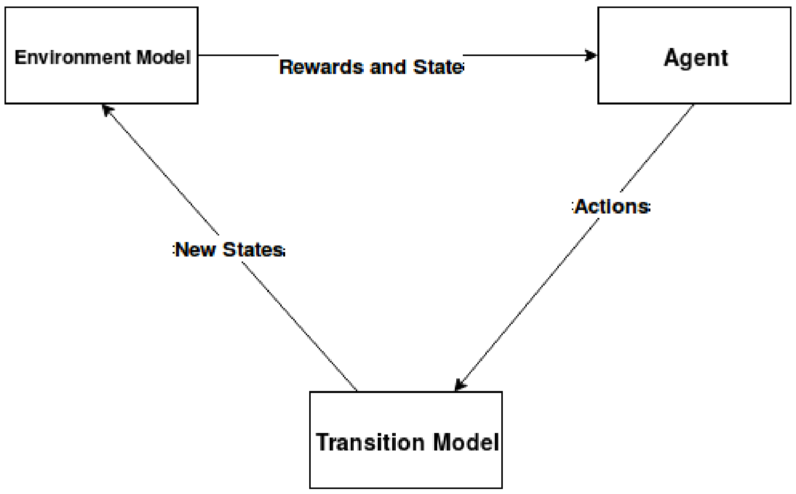


Figure 2.14: Markov Decision Process cycle for artificial intelligent agent environment [28]

Conflict resolution using MDP with Dynamic Programming discretizes the continuous states dynamics to produce a table of decision making actions for the UAV [43]. A similar example is the estimation of continuous states dynamics with once a second interrogations in TCAS. The following states and actions are typically used for unmanned aircraft conflict detection and resolution [44]:

- Relative altitude to own-intruder aircraft, $h_i - h_h$
- Vertical rate of own aircraft, $\dot{h}_h$
- Vertical rate of intruder aircraft, $\dot{h}_i$
- Time to vertical closest approach distance, τ
- Time to zero horizontal separation,τ and conflict state, s_{RA}

2.4.4.1 Reward Cost Functional

The agent environment shown in Figure 2.14 defines a numerical reward feedback after taking actions in a state. The cost function $J(x, a)$ is used as a numerical judgement of an agent's action choice a in a state x to penalise or reward the agent. Some of the objective metrics which MDP can se are: maximizing separation distance, minimising the rates of false alerts and late alerts, and minimizing near misses [43]. Therefore, the rewards in Table 2.1 are assigned to the MDP state transitions as rewards and penalties for state-action choices at each time step. The aircraft actions at each time step are penalised or rewarded based on effort. The highest penalty is awarded if a conflict would occur. Alerts are penalised to reduce the number of false alerts. Strengthening of resolution actions and reversal of resolution actions are also penalised.

Conflict	Alert	Strengthening	Reversal	Clear of Conflict
1	0.01	0.009	0.01	1×10^{-4}

Table 2.1: TCAS states-cost for actioon policy evaluation [28]

The optimisation objective for the MDP formulation of the collision avoidance problem is the maximization of the performance metrics for the collision avoidance logic. MDP problem is solved using Dynamic Programming(DP). DP is a backpropagation-based computation technique which creates a discrete numerical look-up table of actions for given states with the associated cost. One of the known evaluation technique for the DP policy is Value Iteration [41], this approach is useful since aircraft states-action pair are made up of millions values which are iterated before algorithm convergence [28]. However, multiple agent dynamic programming is inefficient for multi-aircraft conflict avoidance.

Another example of rewards-based approaches is Reinforcement Learning (RL) In a reinforcement learning framework, an agent takes an action in an environment and receives a reward for the action. Through iteratively taking actions and receiving rewards, the agent learns a state-action policy that maximises its reward [28].

2.4.5 Sampling-Based Approaches

The sampling-based algorithms approach is a Monte-Carlo simulation strategy for the sampling of the configuration space (C-space) in a finite amount of time. The sampling-based algorithm does not need an explicit representation of the configuration space. As illustrated in Figure 2.15, the sampling-based algorithm randomly samples states in the configuration space C which contains obstacle regions $\mathcal{O}$. An implicit representation of the configuration space is used to perform conflict detection for the sampled state. If a conflict is not detected for the sampled state, then it is added to a graph or tree structure that represents the free configuration space $\mathcal{C}_{free}$; if a conflict is detected for the sampled state, then it is discarded.

The sampling-based algorithm is iterative. Consequently, after some finite amount of time a representation of the free configuration space will be obtained in the form of a graph or tree data structure, which a robot can then search for a conflict-free path to its goal [45]. Therefore, the goal with the sampling-based approach is to construct the conflict-free configuration space, as a subset of the full configuration space, without knowing the representation of the obstacle region, but relying on a conflict detection algorithm. The sampling of the configuration space is performed iteratively until some stopping criteria is reached, and then the robot performs a search to find a path. The sampling-based approach has a history of success with the two main algorithms being Probabilistic RoadMap (PRM) [46] and Rapidly-exploring Random Tree (RRT) [47].

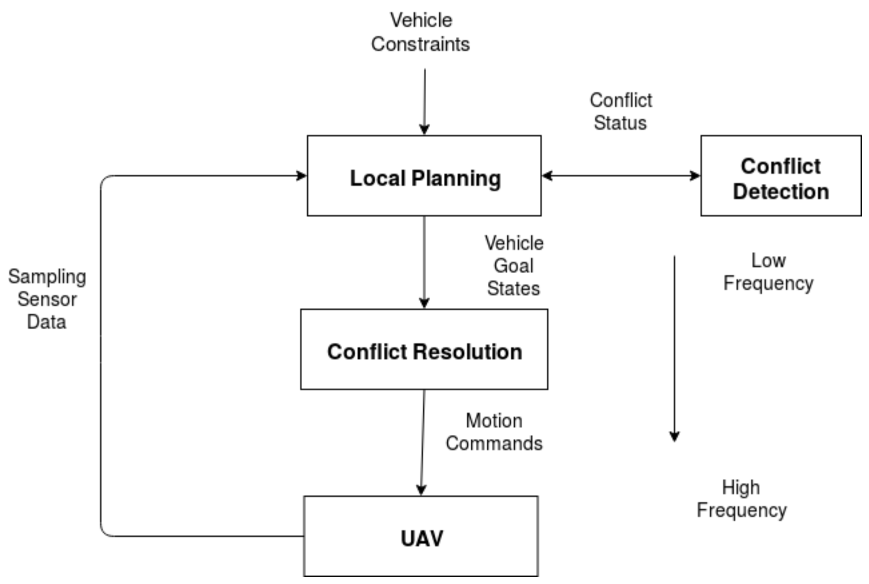


Figure 2.15: Two-Layered architecture for collision avoidance from [38]

The sampling-based PRM algorithm uses a line of sight local planner to connect a roadmap of random configurations. The random configurations are states of a vehicle in the C-space. There are two phases of the PRM algorithm, namely a learning phase and a query phase. During the learning phase, collision-free random samples are added into the roadmap, which is an undirected graph. Thereafter, any node in the graph is connected to the nearest configurations by a local planning metric which may also result in a disconnected graph [46]. The objective is to sample configurations from the C-space until it is possible to query graph G for a path between two configurations. The results of the PRM algorithm are shown in Figure 2.16.

The sampling-based Randomly Rapid Trees (RRT) algorithm uses iterative random sampling of the configuration space to build a tree structure [48]. This approach is suitable for generating a connected tree for systems with nonlinear dynamics in high-dimensional spaces [?]. An RRT tree has a root configuration which iteratively expands the C-Space with the help of a conflict detection algorithm. The tree data structure $\mathcal{T}$ is kept connected by a control law which generates inputs for new configurations from the existing nodes in the tree. Following that, a control law input produced by a RRT planner should meet the differential constraints of non-holonomic robots [47]. Therefore, the search of the RRT tree is limited to a single query from an initial root configuration to a goal configuration as shown in Figure 2.17.

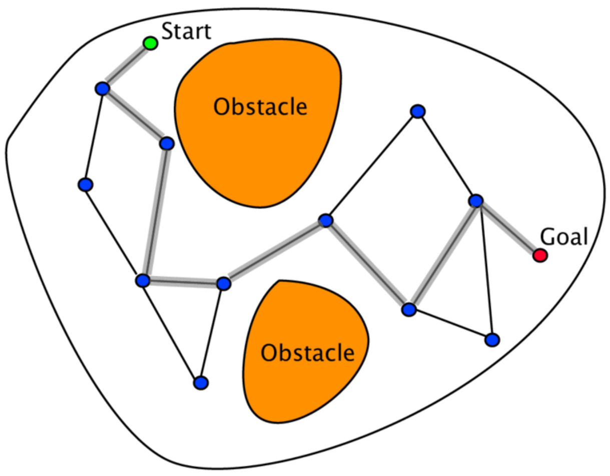


Figure 2.16: Sampling-based approach using Probabilistic Roadmaps to explore connectivity from start to goal configurations with static obstacles [46]

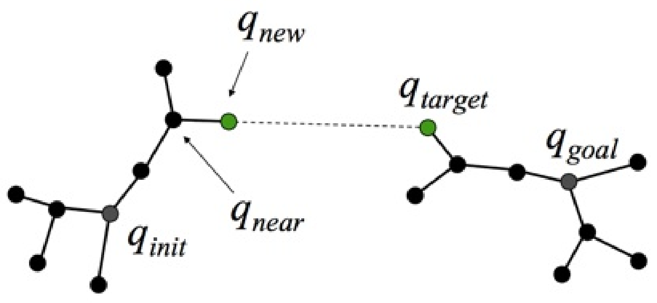


Figure 2.17: RRT search tree expansion for path planning [49]

The sampling based RRT-star [50] and kinodynamic planning [51] are a few of the recent sampling-based algorithms which include the system dynamics of a robot for optimal planning. The RRT-star algorithm guarantees asymptotic optimality. The kinodynamic planning is suitable for problems with differential constraints. Kindel et al. [35] applied kinodynamic planning for real-time collision avoidance with dynamic obstacles.

2.4.6 Artificial Potential Field Approaches

The Artificial Potential Field approach takes a different approach to the C-space representation for path planning algorithms. It performs local planning using a potential field function and attractive and repulsive particles [52]. The method requires a explicit representation of the C-Space in the form of a discrete grid representation with obstacle configurations. The goal is to create a navigation function which represents an artificial force-field which is used in planning a path for a vehicle. The vehicle and local obstacles are modelled as positive charge particles that emulate the behaviour of electrical charges which repel other electrical charges with the same polarity. The goal is modelled as a negative charge particle that attracts the positively charged vehicle. The artificial potential field method has been to perform collision avoidance for multiple self-organising UAVs by (insert authors) [53], as shown in Figure 2.18. Therefore, we have seen voltage potential field method for self organising multiple UAVs was discussed in [53] as shown in Figure 2.18.

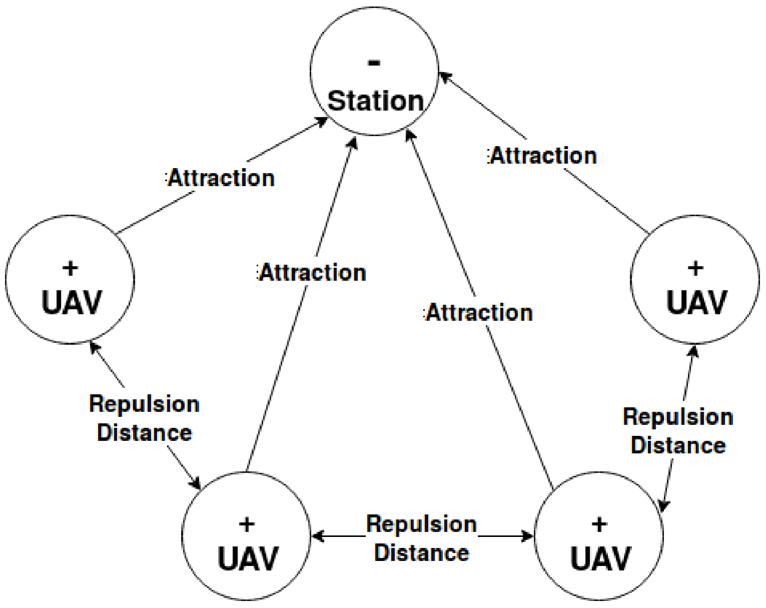


Figure 2.18: Voltage potential field method for collision avoidance [53]

2.4.7 Planning Under Differential Constraints

The conflict resolution trajectory and actions must respect the vehicle's dynamic constraints [6]. For example, the manoeuvre dynamics of an aircraft constrains the maximum curvature of the flight path. The collision avoidance paths must therefore meet the vehicle kinematic constraints in real time [4]. The Dubin's path planning method is a geometric optimisation method that calculates the shortest path length between two points in a two-dimension space. Lester Dubin proposed a robot vehicle model that uses three types of shortest path commands for path planning, namely turn left (L), turn right (L), and continue straight (S). The commands result in a path which can be described in one of six combination of motion primitives; SRL, RSL, LSR, RLS, LRL, RLR. Figure 2.19 shows how a path is optimized using RSR motion primitives to move a vehicle from an initial configuration $A = (x_a, y_a, \theta_a)$ to a destination configuration $B = (x_b, y_b, \theta_b)$.

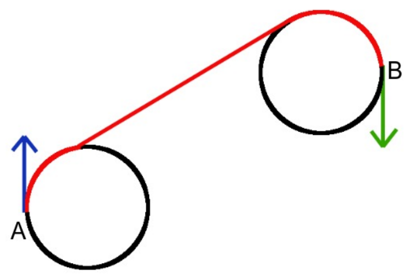


Figure 2.19: RSR motion primitives from [54]

The circle radius of each path is constrained by the vehicle being modelled. It is assumed that the vehicle can only move forward with a constant velocity v, and can only make turns with a constant turning radius r_{min} [54]. The vehicles states over time are determined based on the forward velocity and the turning radius r.

2.4.8 Conflict Prediction Uncertainty

The trajectory path planning uncertainties in airspace between aircraft were addressed with random encounter variables in [5][55][56]. Probability modelling measures the likelihood that a conflict will occur between pairs of aircraft for multiple aircraft trajectories [13]. The conflict zones or protection zones around the host aircraft in the geometric conflict prediction method is the measure of the uncertainty related to a collision risk [57]. The geometric method which consider modelling the presence of another aircraft with uncertainty was interpreted in [56]. Furthermore, a broad commercial airspace analysis of uncertainty for strategic path planning to minimize conflicts and reduce airspace complexities was considered by Nguyen-Duc et al. [58].

2.4.8.1 Uncertainty in Airspace

Intent information can also be a source of uncertainty for trajectory propagation as discussed [59], even if the Nominal or Worst Case approach are chosen. Uncertainties relating to state propagation from the intent information is a topic which is outside the scope of our project, because mechanisms which introduce uncertainties could be solved at the highest flight planning level by introducing arrival time at a waypoint as a requirement. However, Yang and Kuchar [59]studied the effect of intent informaton uncertainty on probabilistic conflict detection as shown Figure 2.12.

2.4.8.2 Probability Ellipses

In a study by Paielli and Erzberger [60], a conflict probability estimation of the aircraft states under the free-flight concept were discussed. The aim of this work was to research whether a probability of a conflict will fall below a certain safe separation thresholds, when it is propagated forward in time using the Geometric Monte Carlo (GMC) approach. The results of the GMC probability algorithm method was intended for long-term conflict prediction for trajectories over a period of between 20 to 30 minutes.

The conflict prediction error for pairs of aircraft is performed using a combination of aircraft states (position and velocity), resulting in normally distributed error ellipses or confidence ellipses at an estimated point of minimum separation [61]. A joint covariance position error is obtained from normally distributed combined modelling of a pair of aircraft. As shown in Figure 2.20, a reference aircraft R is chosen and its conflict zone is modelled to have no uncertainty, while the conflict zone of the intruder aircraft is modelled to have the combined uncertainty of both the reference aircraft and the intruder aircraft [61].

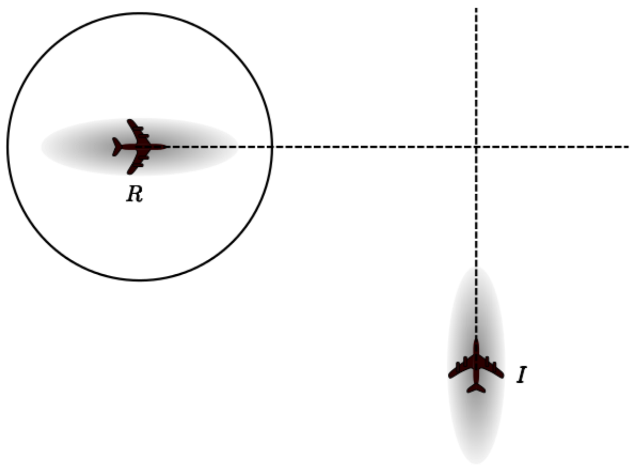


Figure 2.20: Pair of aircraft, "Stochastic" intruder and "Reference" host [5]

2.4.8.3 Probability Flow

The probability flow approach to conflict detection was developed by Van Daalen [5] based on earlier work performed by Jones [62]. Jones used Monte Carlo simulations and volumetric integration of the probabilistic distribution of the aircraft states that violate the conflict zone to estimate the probability of conflict. Van Daalen improved the efficiency of the volumetric integration by integrating the probability flow through the surfaces of the conflict zones, instead of integrating the probability in the conflict zone volumes. Van Daalen also developed path planning algorithms for autonomous vehicles that achieve reduced computational complexity, for vehicles with nonlinear dynamics and clustered environments. Furthermore, as shown in Figure 2.21, Probability Flow was used a conflict detection algorithm which has no knowledge of how conflict resolution (path planning) executes path plans, an approach suitable for sampling-based path planning.

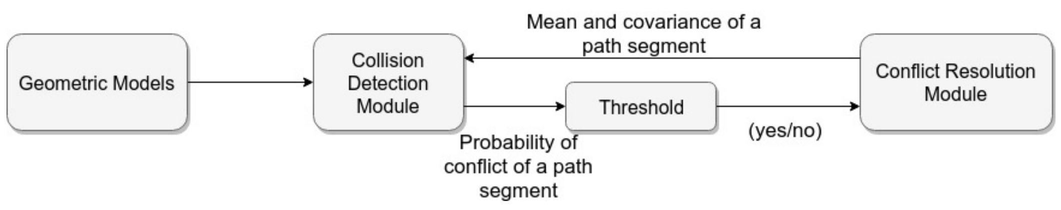

Geometric Models
Collision Detection Module
Mean and covariance of a path segment
Threshold
Probability of conflict of a path segment
(yes/no)
Conflict Resolution Module

Figure 2.21: Probabilistic approach for conflict detection and resolution for autonomous vehicle [5]

2.4.8.4 Monte-Carlo Simulation

The Monte Carlo simulation is used for empirical analysis of aircraft encounters to estimate the probability of conflict [60]. Pienaar performed simulaton studies to compare the Monte Carlo simulation approach against the Probability Flow approach for conflict prediction in airport environments [13]. Pienaar performed simulations for large aircraft in airport environments and implemented probabilistic conflict prediction using both Monte Carlo and Probability Flow algorithms to predict conflicts with other aircraft and terrain. Ultimately, it was concluded by Pienaar that the Monte-Carlo probabilistic conflict prediction is not feasible for real-time collision avoidance because a large number of simulations is required to have a particular guarantee of solution completeness resulting in long computation times.

> "*Monte Carlo does not have an explicit knowledge of the geometric conflict model of an aircraft, but the aerodynamic dynamics are used for the simulation of trajectory in which probability of a aircraft is the ratio of the conflict samples to the total number of simulations*" from [13]

2.5 Multi-Robot Path Planning

The control and negotiation between multiple entities known as agents is a concept applicable to multiple UAV collision avoidance problems, as previously proposed by Wangermann [16]. Principled Negotiation as a multi-agent collision avoidance scheme is applicable to a multi-aircraft conflict resolution. In such an environment, an agent has the ability to use a communication network to make proposals for conflict resolution, make adaptive decisions, or make changes to path plans to benefit a multi-agent system. Hence, the benefit of the approach is the use of negotiation as a resolution scheme, unlike the responsive non-cooperative intruder aircraft approach to conflict avoidance.

2.5.1 Principled Negotiation

Principled Negotiation is a system for an independent multi-agent environment decision making process as shown in Figure 2.22. The aircraft and flight control systems are viewed as agents as proposed by Wangermann et al. [16]. An agent is defined as an entity with the ability to make observations in an environment, perform complex tasks or actions, unilaterally or in coordination with other agents, to achieve a set of common goals. Using the Principled Negotiation , an Integrated Aircraft/AIrspace System (IAAS) was designed as a multi-agent environment which focusses on cooperative planning of tasks between agents. The multi-agent environment is shown in Figure 2.22. Multiple agents are designed to perform decision making using their own set of actions to share the workload collaboratively, thus making this approach suitable for solving multi-aircraft conflict scenarios [16].

"Agents only propose options that satisfy their constraints and have increased utility" - Principled Negotiation [16]

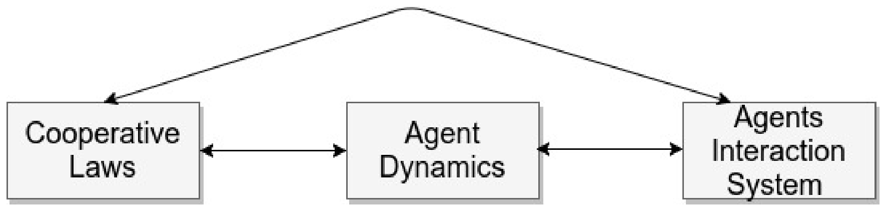


Figure 2.22: Configuration of a Multi-agent system adopted from Cooperative Control of Multi-Agent[16]

2.5.2 Centralised and Decoupled Planning

Path planning for multiple robots is an implementation of the artificial intelligence principle where negotiation between agents is useful because the trajectories already planned by other robots have to be considered. Furthermore, the path plannersfor multi-robot path planning are either coupled or decoupled [63][?][64]. Figure 2.23 illustrates why coordination between robots is crucial for path planning and collision avoidance.

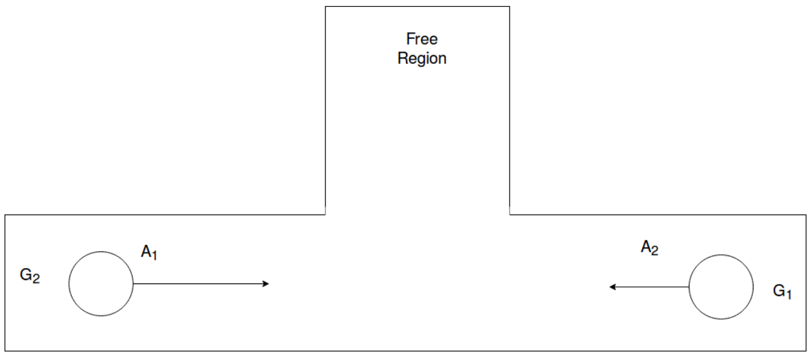


Figure 2.23: Path planning where priority planning between two robots is applicable for conflict resolution

Coupled path planning uses the combined configuration space of all the robots, which is the union of the configuration spaces of the individual robots. However, searching for a path in a combined configuration space has a computational complexity that increases exponentially with the number of individual configuration spaces. Therefore, searching for conflict-free trajectories for aircraft using coupled path planning does not scale well according to a study by Ong and Kochenderfer [28].

Decoupled path planning decentralises the search by performing single robot path planning sequentially. Each robot performs its own path planning using its individual configuraration space, and then becomes a dynamic obstacle for the next robot. This reduces planning time as compared to the coupled planning as shown in Figure 2.23. However, coordination between the host aircraft and the intruder aircraft is still necessary because the execution time of the planning algorithm and the completeness of the solution still depends on the order of the sequential planning. The individual path planning for each robot is performed using planning algorithms such as sampling-based planning or potential field based planning. Thereafter, the order of the sequential planning is determined by a function known as prioritised planning or token allocation [65].

2.6 Conflict Resolution Performance Metrics

In this section we provides an overview of performance metrics for evaluation of Monte Carlo simulation of conflict resolution algorithms. In this thesis we shall consider performance metrics found literature to evaluate the rules-based and path planning algorithms. An extensive evaluation of performance metrics for TCAS of multiple aircraft were discussed in [9]. The multiple conflict resolution of path planning algorithms with coordination will trigger replanning during a new conflict encounters discussed by [66].

2.6.1 Collision Avoidance Success Rate

The evaluation of path planning algorithm shall consider the success rate as the ratio of the difference in the total number of non-flyable paths and collision avoidance path to the total number of simulated paths.

$$\text{Success rate} = \frac{\text{number of planned paths - number of non-flyable paths}}{\text{number of planned paths}} \times 100 \tag{2.8}$$

The rules-based success rates shall will be the number of ratio of encountered distances to the number of NMAC distances.

$$\text{Success rate} = \frac{\text{number of near encounters - number of NMAC encounters}}{\text{number of near encounters}} \times 100 \tag{2.9}$$

2.6.2 Maximum Path Deviations

The path deviation of vertical manoeuvre were considered to keep an aircraft on its nominal path for as long as possible. The aim is to minimise the integral time distance from the nominal altitude as considered in [67]. This performance metric evaluate the vertical local motion planner that generate vertical manoeuvre $\dot{h}$ which will deviate the aircraft with $f(h_i)$ as perpendicular distance from nominal altitude between times t_0 and t_1. The manoeuvres directions are both climb and descend similar to TCAS which are evaluated in a cost function J_{dev}.

$$J_{dev} = \int_{t_0}^{t_1} f(h_i)dt \tag{2.10}$$

2.6.3 Number of UAV Path Replanning

The multiple conflict resolution of path planning algorithms with coordination will trigger replanning during a new conflict encounters discussed by [66]. This implies that the aircraft will need to a path replanning after each conflict encounter. Thus, the priority path planning execution algorithm chooses from conflict free trajectories which are determined independently. A few of the following algorithms from literature that could be used for counting the number of aircraft replanning are as follows:

- Fixed-path Coordination from [68]
- Decoupled Planning from [69]
- Token Allocation from [16] and [7]

2.6.4 Number of Near Mid-Air Collisions

The number of Near-Mid-Air Collisions (NMAC) distance between multiple aircraft for TCAS collision avoidance was modelled to estimate the probability of loss of separation [9]. In the multi-thread analysis, the NMAC distance was defined as the number of loss of separation of 500 ft vertically and 1000 ft horizontally between pairwise UAVs during simulation of collisions encounters. Conflict encounter between a pairwise or a multiple aircraft were surveyed by Jamoom et al. [70] to calculate the probability of NMAC. An illustration of NMAC threshold defined 100 feet vertically and 500 feet is shown in Figure 2.24.

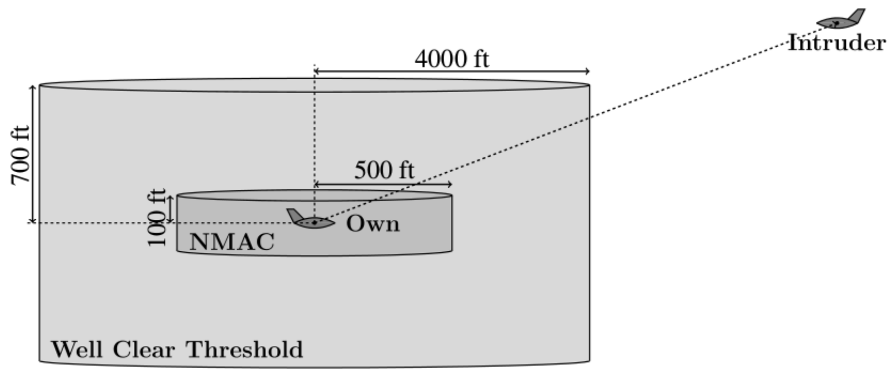


Figure 2.24: Near-midair collision threshold for pairwise aircraft [70]

2.7 Summary and Discussion

In the first part of our background study we reviewed at See-and-Avoid technology for conflict prediction and resolution for manned aircraft. This literature has identified TCAS and GPWS as two existing systems used in civilian airspace systems. We also reviewed Sense-and-Avoid technologies for conflict prediction and resolution for unmanned aerial vehicles and learned that such systems are dependent on a communication sharing a network. Then, we established that the existing Mode-S transponder remains the main protocol currently used by TCAS and ACAS-X for manned aircraft collision avoidance systems. The ADS-B network can also be used to share data for cooperative conflict prediction and resolution, and for cooperative mission planning.

In the second part of our background study, we reviewed general path planning algorithms, multi-robot path planning, and performance metrics for conflict prediction and resolution systems. The review of general path planning algorithms covered configuration space planning, search-based planning, numerical optimisation approaches, sampling-based planning, and artificial potential field approaches. Path planning under differential constraints, and conflict prediction under uncertainty were also considered. The review of multi-robot path planning covered the principled negotation approach, as well as centralised and decoupled planning. Token passing was identified as a mechanism to coordinate sequential decoupled planning for multi-aircraft conflict resolution. The conflict resolution performance metrics found in literature included the number of near mid-air collisions, the probability of near mid-air collisions, the number of false alerts, the number of resolution advisory reversals, and the total deviation of the vehicles from their nominal paths.

For our research project, we shall therefore develop two types of collision avoidance algorithms for unmanned aerial vehicles: a rules-based algorithm and a cooperative path planning based algorithm. The rules-based algorithm will be modelled after the TCAS system that is used for manned aircraft; the path planning based system will use decoupled multi-robot path planning. and will use token passing to coordinate the sequential path planning for individual UAVs. The rules-based and path planning based algorithms will be evaluated in simulation using performance metrics similar to those found in literature.

Chapter 3

Conceptualisation and Modelling

This chapter presents the conceptualisation and modelling of a multi-aircraft system which serves as the basis for the development of our cooperative collision avoidance solutions. In Section 3.1, a conceptualisation of the system of multiple unmanned aerial vehicles (UAVs) and static terrain will be summarised. In the section that follows, Section 3.2, we look into the UAV dynamic model which provides an overview of the guidance autopilot system that performs the flight control of a UAV for dynamic modelling of point mass translational motion of the flight control system in three-dimensional space. Additionally, in the same section, a set of three-dimensional equations of motion for the UAV is reduced to equation of two-dimensional space. The two-dimensional of translational motion are thereafter used for state propagation and conflict prediction. In Section 3.3, the definition of conflict is considered, including the modelling of a safe exclusion zone to detect conflict with other UAVs, and a minimum altitude to prevent conflict with terrain. In Section 3.4, pairwise conflict detection and resolution using an altitude flight guidance system is conceptualised. Finally, in Section 3.5, a multi-aircraft collision prediction and avoidance framework is conceptualised to provide the framework for the rules-based and path planning based collision avoidance algorithms that will be developed in Chapter 4 and 5, respectively.

3.1 Multiple Aircraft Collision Avoidance

The conflict conceptualisation for multiple UAVs presents many scenarios where short-term conflict without resolution cooperation and communication is challenging. As discussed in Section 2.5.1, Principled Negotiation between multiple decision-making agents can be used as a model for cooperative conflict resolution between multiple unmanned aircraft. As shown in Figure 3.1, five UAVs in a conflict scenario are at different levels of altitude with a certain impending conflict. In the illustrated multi-aircraft conflict more than one aircraft (or all of them) would have to manoeuvre to avoid the collision *cooperatively.*

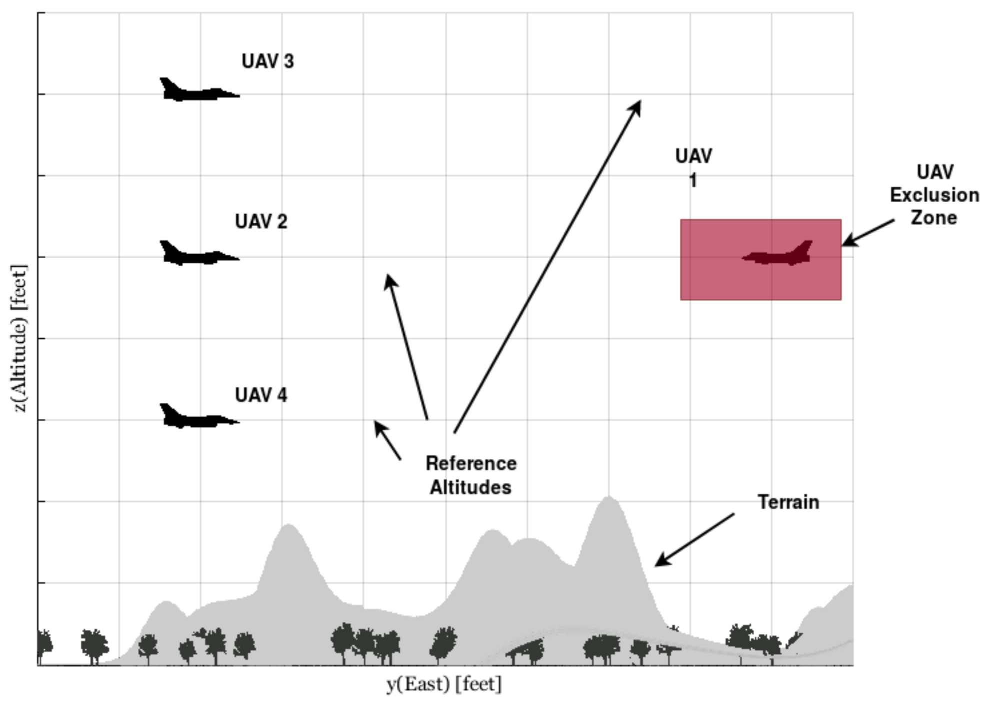


Figure 3.1: Multiple of aircraft conflict example

The system contains a number of UAVs travelling along the same route between two destinations, and flying over static terrain. We consider a window along the route where all UAVs are in the cruise phase of their flight. The UAVs are therefore flying at their individual cruise speeds and at their individually assigned cruise altitudes. It is assumed that a finite number of altitude "lanes" are available in the airspace, and that different UAVs have been assigned different cruise altitudes by air traffic control. The UAVs are travelling in both directions along the route, and an individual UAV may appear at the left border of the window and travel from left to right, or may appear at the right border of the window and travel from right to left. The UAVs must fly at their assigned cruise altitude, but must also avoid collisions with one another, and collisions with the terrain.

It is assumed that cooperative UAVs determine their own states, and communicate their current states and intents to one another. The state of a UAV at a given time instant is defined by its position, altitude, speed, heading, and flight path angle (or climb rate) at that time instant. The intent of a UAV is represented by its planned flight path or its planned actions. Each UAV determines its own state from a combination of sensors, including GPS sensors, barometric pressure sensors, inertial sensors, and magnetometers. The UAVs communicate their states to one another and to ground control stations using Mode S Transponders, and communicate their intent (planned flight path or planned actions) to one another through ADS-B.

In addition, it is assumed that dynamic obstacles are tracked by ground stations using ground-based radar. The ground stations communicate the states of the dynamic obstacles to the UAVs. The intentions of the dynamic obstacles are unknown, but the ground stations and UAVs can extrapolate the flight paths of the dynamic obstacles based on their current position and altitude, assuming that they will hold their current speed, heading, and climb rate.

3.2 UAV Dynamic Modelling

In this section we discuss the dynamic model of a UAV that will be used for propagation in collision prediction and avoidance algorithms. Firstly, an overview will be given of the UAV system which includes the aircraft, the autopilot, and the guidance system. Thereafter, the three-dimensional equations of translational motion for a UAV are discussed assuming a rigid-body aircraft model. Thereafter, the equations of translational motion in three dimensions are reduced to translational motion in two dimensions, which will be used for modelling conflict detection and avoidance between UAVs in this thesis. Finally, the discrete-time state propagation of the UAVs two-dimensional equations of translational motion is presented to be used for forward simulation of intruder UAV trajectories based on their intent in cooperative systems.

3.2.1 Overview

The models of the UAV's control loops are implemented at different levels of control for the waypoints navigation, path planning and control surfaces command. As shown in Figure 3.2, the outer control loop is for the lower frequency control of guidance rules at the higher level of the UAV. The guidance system receives a higher level path plan and then issues acceleration commands to the lower level autopilot of the UAV. These guidance control command inputs of a UAV are designed to meet formulated path plans [37].

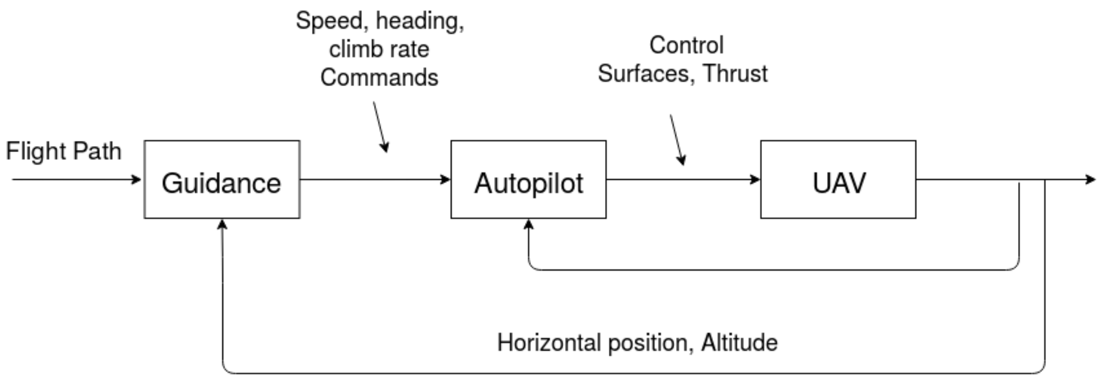

Speed, heading, climb rate Commands
Control Surfaces, Thrust
Flight Path
Guidance
Autopilot
UAV
Horizontal position, Altitude

Figure 3.2: Guidance and Flight Control Architecture for a UAV as discussed in [36]

The flight control system is the inner-loop control system that issues acceleration commands to the UAV actuators (control surfaces). The guidance system is executed at a lower frequency than the autopilot commands. The control surfaces command the UAV actuators with acceleration inputs from the guidance system [36].

The point mass kinematics provides the UAV guidance system with the current velocity vector in the inertial axis. The UAV will then generate acceleration commands for the lower level autopilot controller [36]. The kinetics are differential equations that are integrated in the lateral and longitudinal directions in the body-axis system. Therefore, the integrated acceleration vectors produce the velocity vector of the UAV in an inertial axis system (Newton's Second Law).

The guidance system controls the UAV's trajectory to follow a planned flight. The planned flight path may be separated into a planned ground track trajectory and a planned altitude trajectory. The guidance system receives the planned flight path as a reference signal, and the measured horizontal position and altitude of the UAV as feedback signals, and then outputs speed, heading, and flight path angle (or climb rate) references to the flight control system. The guidance system may control the vertical motion of the UAV either in climb rate control mode, or in altitude control mode. Altitude control mode is typically used in the cruise flight phase to maintain a reference altitude; climb rate control mode is typically used to control the UAV's climb rate directly during take-off and landing, or during collision avoidance manoeuvres.

3.2.2 Three-Dimensional Motion

The majority of aircraft dynamics models in literature are point-mass models [71]. Point-mass models reduce the rigid-body models of the aircraft to translation equations of motion which are derived in the inertial axis coordinate system [72]. The assumption we make with respect to translational motion is that the Earth is flat and non-rotating. The time variable t is often included as an extra state variable in dynamic obstacle modelling cases [73]. Therefore we can represent the translational state of the i'th UAV in three dimensions by its state vector $\mathbf{x}_i(t)$:

$$\mathbf{x}_i(t) = \begin{bmatrix} N_i(t) & E_(t) & D_i(t) & \overline{V}_i(t) & \Phi_i(t) & \gamma_i(t) \end{bmatrix}^T \tag{3.1}$$

where N_i, E_i and h_i are the north, east and altitude positions and $\overline{V}_i$, Φ_i and γ_i are the velocity magnitude, heading, and flight path angles of the i'th UAV. Therefore, the equations of motion for the i'th UAV in three dimensional space are given by:

$$\dot{N}_i(t) = \overline{V}_i(t) \cos \Psi_i(t) \cos \gamma_i(t) \tag{3.2}$$

$$\dot{E}_i(t) = \overline{V}_i(t) \sin \Psi_i(t) \cos \gamma_i(t) \tag{3.3}$$

$$\dot{h}_i(t) = \overline{V}_i(t) \sin \gamma_i(t) \tag{3.4}$$

Furthermore, the UAVs are assumed to be of fixed-wing type, which means that the vehicle must maintain a minimum speed, otherwise it will stall.

$$||\overline{V}_i(t)|| \quad \geq \quad V_{stall} \tag{3.5}$$

where V_{stall} is the stall speed of a specific UAV. In addition it is assumed that the velocity, heading, and flight path angle are controlled by the flight control system, and that the velocity, heading, and flight path angle controller responses can be modelled as first-order transient responses with zero steady-state error, as follow :

$$\bar{V}_i(t) = \frac{1}{\tau_{\overline{V}}}\left[\overline{V}_{i,ref}(t) - \bar{V}_i(t)\right] \tag{3.6}$$

$$\dot{\Psi}_i(t) = \frac{1}{\tau_{\Psi_i(t)}}\left[\Psi_{i,ref}(t) - \Psi_i(t)\right] \tag{3.7}$$

$$\dot{\gamma}_i(t) = \frac{1}{\tau_{\gamma_i}}\left[\gamma_{i,ref}(t) - \gamma_i(t)\right] \tag{3.8}$$

where $\overline{V}_{ref}$, $\Psi_{i,ref}$ and $\gamma_{i,ref}$ are the reference commands for the velocity magnitude, heading, and flight path angle respectively, with $\tau_{\overline{V}_{ref}}$, $\tau_{\Psi_{i,ref}}$, $\tau_{\gamma_{i,ref}}$ as the time constants of the controller responses for the velocity magnitude, heading, and flight path angle, respectively.

The altitude of the UAV may also be controlled by an altitude controller, given by the following control law equations:

$$\dot{h}_i(t) \quad = \quad -\frac{1}{\tau_h}\left(h_{i,\text{ref}}(t) - h_i(t)\right) \tag{3.9}$$

$$\gamma_i(t) \quad = \quad \arcsin\frac{\dot{h}_{i,\text{ref}}(t)}{\overline{V}_i(t)} \tag{3.10}$$

where $h_{i_{\text{ref}}}$ is the reference altitude, $\dot{h}_{i_{\text{ref}}}$ is the commanded climb rate, and τ_h is the time constant of the altitude controller.

3.2.3 Two-Dimensional Motion

In this project, we will only consider collision avoidance using vertical (climb rate) manoeuvres. We can therefore reduce the three-dimensional equations of motion to two dimensions as follows :

$$\dot{x}_i(t) = \overline{V}_i(t)\cos\gamma_i(t) \tag{3.11}$$

$$\dot{h}_i(t) = ||\overline{V}_i(t)||\sin\gamma_i(t) \tag{3.12}$$

where x_i is the horizontal position, and $\overline{V}$ is positive when travelling in the positive horizontal axis direction, and is negative for travelling in the negative horizontal axis direction.

Going forward, we will make several assumptions to simplify the UAV dynamic model. The first assumption is that the UAVs are in cruise flight, and therefore a constant cruise speed $\overline{V}_{i,ref}$ and cruise altitude $h_{i,ref}$ are maintained. In addition we will assume that the flight path angle controller dynamics may be abstracted away through time scale separation, and that we may assume that the flight path angle γ_i changes instantaneously from the perspective of the point mass translational motion. The following simplifications are made because the time constant of the flight path angle controller is an order of magnitude smaller than the time constant of the altitude controller, and also an order of magnitude smaller than the time scales over which the collision avoidance system operates. The flight controller dynamics therefore reduces to the following:

$$\dot{x}_i(t) \approx \overline{V}_{ref}(t) \tag{3.13}$$

$$\gamma_i(t) \approx \gamma_{ref}(t) \tag{3.14}$$

$$\dot{h}_i(t) \approx \dot{h}_{ref}(t) \tag{3.15}$$

We shall therefore assume that the horizontal velocity equals the reference cruise speed and remains constant, that the flight path angle instantaneously follows the commanded flight path angle, and that the climb rate instantaneously follows the commanded climb rate.

3.2.4 UAV State and Intent

A discrete-time dynamic model of the UAV translational motion is used to propagate the UAV states forward in time for the purposes of collision prediction and testing candidate conflict avoidance paths. The intent of a UAV is obtained by propagating the UAV state forward in time from its current state based on its intended flight path or its intended actions, as shown in Figure 3.3. The UAV state is typically propagated using a fixed sampling period Δt and using a finite time horizon in the future. A constraint on the state propagation is that the vehicle should maintain a constant forward velocity in one direction during intent construction. The velocity vector $\overline{V}$ cannot be zero, due to the nature of fixed-wing design [45].

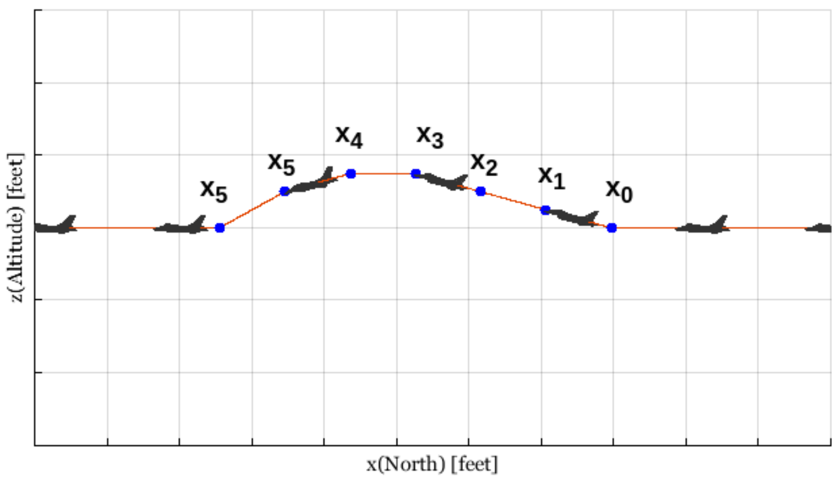


Figure 3.3: Planning of UAV states showing intent

The planned flight path may be expressed as the UAV's future horizontal position $x(t)$ and altitude $h(t)$ as a function of time t. Alternatively, it may be expressed more concisely as a sequence of position and altitude waypoints $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$, at given time instants $t_1, t_2, \ldots, t_n$. Similarly, the planned actions may be expressed as the UAV's future climb rate command $h_{iref}(t)$ as a function of time t, or more concisely as a sequence of climb rate actions $\mathbf{u_1}, \mathbf{u_2}, \ldots, \mathbf{u_n}$, at given time instants $t_1, t_2, \ldots, t_n$. Therefore for the purpose of collision prediction and collision avoidance, we will assume that the time instants are equally spaced and separated by a fixed sampling period Δt which is useful for communicating intent in a concise way.

The simulation of intent of a UAV in Figure 3.3 is an important stage of conflict prediction of multiple UAVs. In this thesis, the conflict resolution algorithm will rely on the conflict detection to generate a unique plan of actions that are conflict-free as shown in Figure 3.4. Therefore, the communication network would require a coordination data structure to broadcast the planned action set to all UAVs. Furthermore, in a multi-aircraft conflict scenario the path planning algorithm requires reliable communication and efficient conflict detection algorithm between the UAVs.

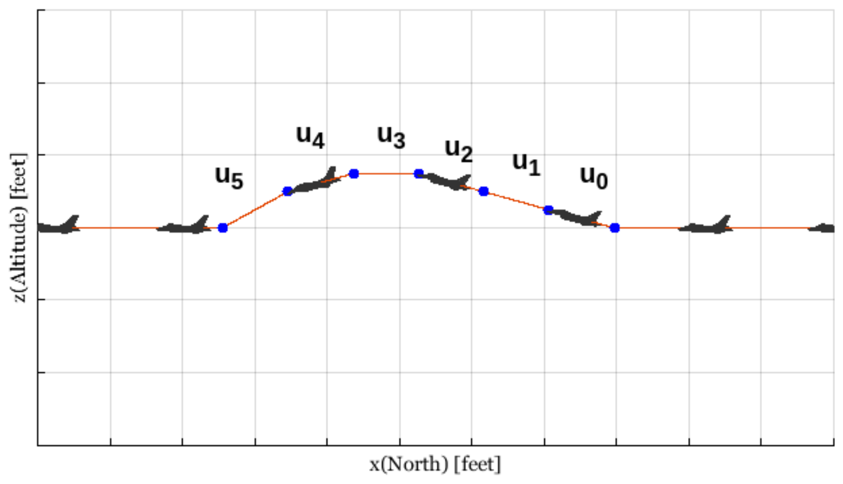


Figure 3.4: Planned actions generating UAV states showing intent

Since the horizontal velocity of the UAV is kept constant $\dot{x} \in \{\dot{x}_{min}, \dot{x}_{max}\}$ as shown in Figure 3.4, our approach in this thesis will only focus on the inputs sampled from a finite set of climb rates that the UAV can execute. The action space $\mathcal{U}$ is therefore predefined by Equation 3.16. The goal is to use these inputs to sample state configurations $q \in \mathcal{C}_{free}$ for a UAV at different altitudes, where the climb and descend rate inputs are the TCAS conflict advisories $\mathcal{U} \subseteq \mathbb{R}^5$.

Given the fixed-wing aircraft model, the initial state, and a sequence of altitude rate actions which satisfy Equation 3.16, the UAV state can be propagated to perform collision prediction with future states of other UAVs. The state propagation can be performed using either flight path angle γ commands or altitude rate $\dot{h}$ commands to the UAV flight control system. The altitude controller can also command either flight path angles or altitude rates to control the UAV altitude. Finally, a sequence of flight path angle or altitude rate commands can be provided at fixed time intervals Δt to make the UAV fly a desired flight path. In Chapter 5, we will present a path planning based collision avoidance algorithm that generates altitude rates to steer the aircraft to follow a desired collision avoidance flight path.

$$\mathcal{U} = \begin{bmatrix} u_1 \\ u_2 \\ u_0 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} 2500 \text{ ft/min} \\ 1500 \text{ ft/min} \\ 0 \text{ ft/min} \\ -1500 \text{ ft/min} \\ -2500 \text{ ft/min} \end{bmatrix} \tag{3.16}$$

$$H_{dot} = \{ \, \dot{h}_k \in \mathcal{U} \mid \dot{h}_k = \frac{h_k - h_{k-1}}{\Delta t} \, \} \tag{3.17}$$

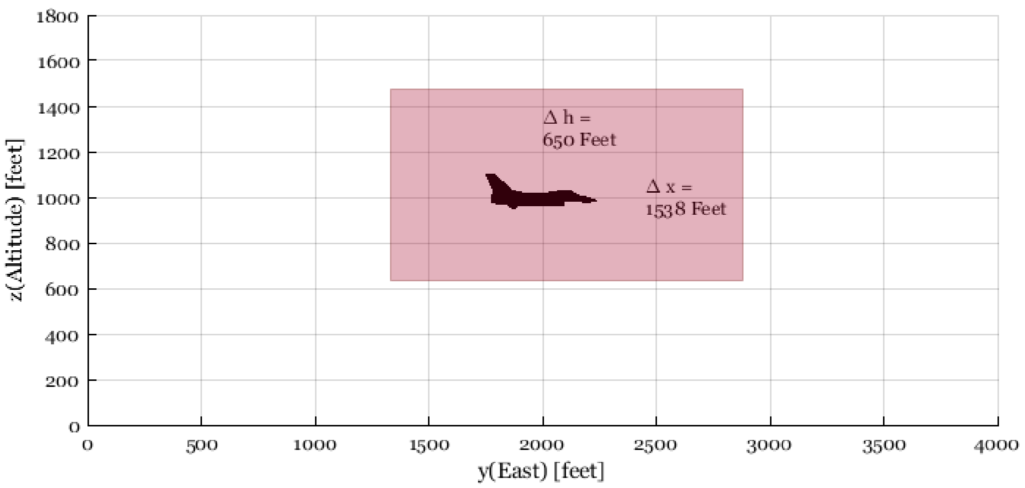


Figure 3.5: Conflict separation region of an unmanned aircraft (not drawn to scale)

3.3 Definition of Conflict

In this section we shall conceptualise the definition of conflict for conflict detection between UAVs and with static terrain obstacles. The definitions given in this section shall be used throughout this thesis. We shall begin by defining a conflict zone, or a UAV exclusion zone, around each UAV. Next, a generic conflict detection and prediction procedure between pair of UAVs is presented. Finally, a simplified low altitude terrain conflict detection shall follow.

3.3.1 UAV Exclusion Zone

The UAV exclusion zone, or conflict zone, of the host UAV is a rectangular zone around the UAV specified as a minimum longitudinal separation distance Δx and a minimum relative altitude Δh. The shape of the conflict zone is shown in Figure 3.5. Conflict prediction and avoidance should prevent any intruder UAVs from entering the conflict zone of the host UAV. The UAV exclusion zone may be expressed mathematically as :

$$\mathcal{C}(x,h) = \{x,h \,:\, ||x - x_i|| \leq \Delta x \cap ||h - h_i|| \leq \Delta h\} \tag{3.18}$$

A simplified conflict detection function is shown in Algorithm 1. The function takes the positions of the host and intruder aircraft as inputs, and its output indicates whether a conflict is detected. A conflict prediction algorithm propagates the UAV states forward in time and calls the conflict detection function at every time step. A conflict is predicted if the conflict detection function detects a conflict for any of the propagated states. For the conflict detection function shown in Algorithm 1, a conflict is defined in relation to pairwise UAV altitude and range difference thresholds in line 1-2 then used in line 6.

Therefore, a conflict status result is returned by the algorithm in line 9.

Algorithm 1 Simplified Conflict Detection *ConflictDetect*(s_h, s_i)

```
Require: Two aircraft 2D state position $s_h = (x_h, h_h)$ for host and $s_i = (x_i, h_i)$ for
    intruder
1: $x_{dmod\_}$ra $\leftarrow$ 1000 feet
2: $z_{thr} \leftarrow$ 650 feet
3: $\Delta x \leftarrow x_h - x_i$
4: $\Delta h \leftarrow h_h - h_i$
5: conflict $\leftarrow 0$                                        ▷ Default conflict status
6: if $||\Delta z|| \leq z_{thr}$ and $||\Delta x|| \leq x_{dmod\_}ra$ then
7:     conflict $\leftarrow 1$
8: end if
9: return conflict
```

3.3.2 Conflict with another UAV

Conflict between two UAVs is defined as shown in Figure 3.6 where an exclusion zone is the logic for conflict detection. Conflict is detected based on the instantaneous position states of both UAVs. Two UAVs are said to be in conflict if the position of one of the UAVs is inside the exclusion zone of the other UAV. Conflict with another UAV may be expressed mathematically as follows :

$$\mathbf{x}_j(t) \quad \in \quad \mathcal{C}_{\mathrm{UAV}}(\mathbf{x}_{\mathrm{i}}(\mathrm{t})), \text{ for j} \neq \text{i} \tag{3.19}$$

where $\mathbf{x}_j(t)$ is the position of the intruder UAV, $\mathbf{x}_i(t)$ is the position of the host UAV, and $\mathcal{C}_{\mathrm{UAV}}$ is the UAV exclusion zone around the host UAV.

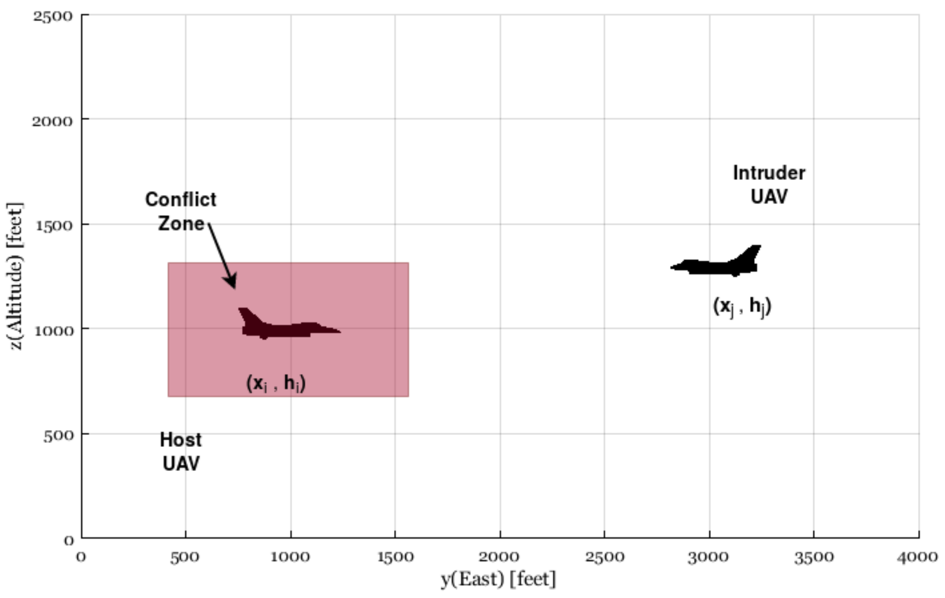


Figure 3.6: Two aircraft conflict scenario with a minimum separation distance for conflict detection

3.3.3 Conflict with Terrain

Terrain features are static obstacles which a UAV is likely to encounter during flight and serves as a constraint for the collision avoidance algorithm. Obstacles fall in one of two categories: static obstacles and dynamics obstacles. Static obstacles include terrain features, while dynamic obstacles included other flying aircraft. It is assumed that a database of terrain is available to the system with interaction with other existing systems such as EGPWS [10]. However, the integration with such of terrain detection systems is beyond the scope of this project. For our project, terrain avoidance will be performed by specifying a minimum operating altitude below which the UAVs are not allowed to operate. The minimum operating altitude will ensure sufficient clearance above all terrain features.

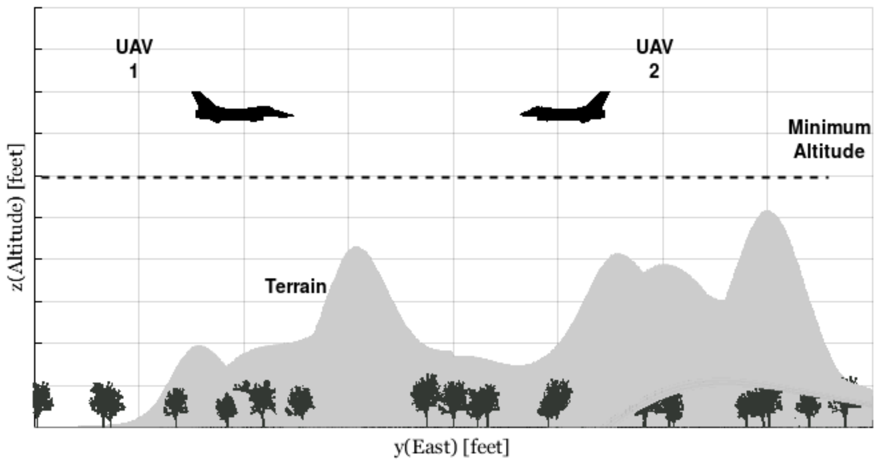


Figure 3.7: Terrain Model

A conflict exists between the host UAV and the terrain if the position of the UAV is below the minimum operating altitude, as shown in Figure 3.7. Conflict with terrain may be expressed mathematically as

$$h_i(t) \quad \leq \quad h_{min} \tag{3.20}$$

where h_i is the altitude of the host UAV, and h_{min} is the minimum allowable altitude to ensure terrain avoidance.

3.4 Collision Prediction and Avoidance

In this section, we conceptualises the collision prediction and avoidance process for unmanned aerial vehicles. First, we describe the general framework and the concept of operation of the collision prediction and avoidance system. Next, we discuss the stages of pairwise collision detection and avoidance. We then conceptualise the conflict prediction zones around the host UAV, and describe two approaches to conflict prediction. The first approach performs conflict prediction based on the instantaneous positions and velocities of the host UAV and the intruder UAV. The second approach propagates the UAV positions forward in time, and checks for conflict at each time step in the future. Next, we conceptualise the UAV flight controller which switches between normal flight mode and collision avoidance mode. Finally, a method for the estimation of the conflict resolution during for UAVs with unequal horizontal velocities is presented.

3.4.1 Collision Prediction and Avoidance Framework

The collision prediction and avoidance problem can be divided into two sub-problems: collision prediction and collision avoidance. The goal of collision prediction is to predict future collisions between UAVs, as well as collisions with terrain and dynamic obstacles. The goal of collision avoidance is to execute avoidance actions to avoid collisions between UAVs, as wells as collisions with terrain and dynamic obstacles, while also obeying the differential constraints of the individual UAVs.

The collision prediction and avoidance framework consists of three modules, a modelling module, a collision prediction module, and a collision avoidance module, that work together to perform collision avoidance. The collision prediction module executes continuously to predict future collisions within a short-term prediction horizon. If a future collision is predicted, then the collision avoidance module is activated to execute collision avoidance actions. The collision prediction and collision avoidance modules both use the modelling module to propagate the UAVs and dynamic obstacles forward in time, to determine future collisions between UAVs, and future

3.4.2 Stages of Collision Avoidance

The stages of collision prediction and avoidance for two UAVs in a pairwise conflict scenario is illustrated in Figure 3.8. Both UAVs are flying at the same altitude, one from the left and the other from the right, and are on a collision course with one another. The two UAVs both start in "normal flight" mode and their guidance systems use their altitude controllers to control their altitudes to follow their reference altitudes. When they reach their minimum safe horizontal separation distance, their collision prediction modules predict that a collision will occur and activates their collision avoidance modules. Their guidance systems then switch to "collision avoidance" mode and their collision avoidance modules execute collision avoidance climb rate actions. In this case, the UAV travelling from left to right maintains its level flight, while the UAV travelling from right to left climbs and then maintains its new altitude until the predicted collision has passed. Once the predicted collision has passed, the guidance system switches back to "normal flight" mode, and both UAVs return to their nominal reference altitude using their altitude controllers.

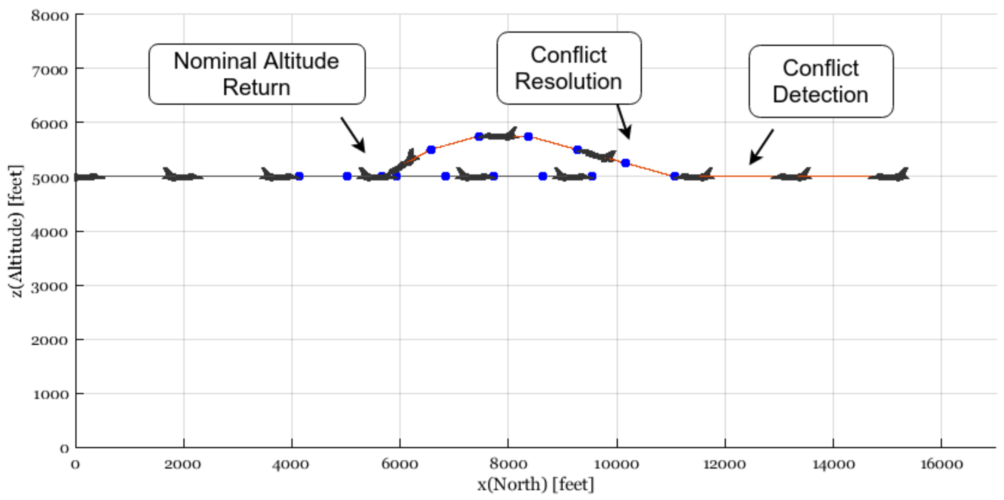


Figure 3.8: Stages of the Pairwise Conflict Detection and Resolution for UAVs

3.4.3 Collision Prediction

The different collision prediction zones for a UAV are conceptualised in Figure 3.9. If an intruder UAV enters the Traffic Advisory Zone, then the host UAV is alerted of the presence of the intruder UAV, but does not need to take avoidance actions yet. If an intruder UAV enters the Resolution Advisory Zone, then the host UAV is advised to take resolution actions to avoid a conflict with the intruder UAV. If an intruder UAV enters the Conflict Zone, then a Near Mid-Air Collision occurs, meaning that the conflict avoidance was unsuccessful.

The purpose of the conflict prediction function is to predict if the intruder UAV will enter the Conflict Zone of the host UAV, and if so, to estimate at what point in time the conflict will occur. TCAS performs collision prediction by using the instantaneous positions and velocities of the host and intruder aircraft to estimate the time-to-closest-approach τ_{ij} and the miss distance r. The miss distance is used to determine if a Near Mid-Air Collision is predicted, and the time-to-closest approach is used to determine when the predicted collision will occur.

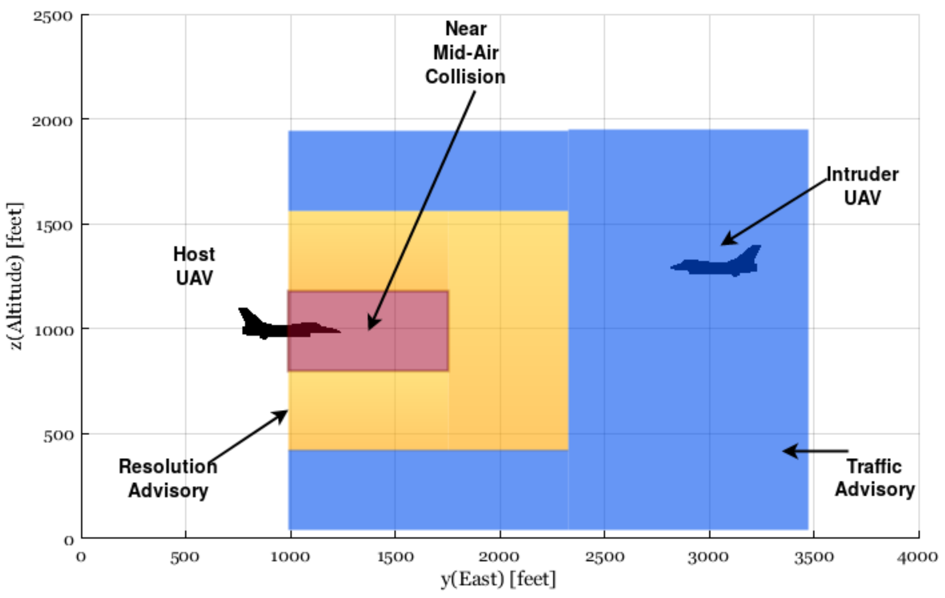


Figure 3.9: Conflict Prediction Zones

Another approach to conflict prediction, is to propagate the position states of the host UAV and the intruder UAV forward in time using their dynamic models, and to check at each time step if the intruder UAV enters the Conflict Zone of the host UAV. Both UAVs are propagated forward in time from their initial states using their intended flight paths or intended actions. This approach to conflict detection will be used in our path planning based conflict avoidance algorithm, and will be discussed in more detail in Chapter 5.

3.4.4 Flight Control Switching

The UAV system which is used to switch between normal flight mode and conflict avoidance mode dynamically is shown in Figure 3.10. The conflict resolution stage described in Section 3.4.2 will deviate a single UAV as shown in Figure 3.8 by executing the collision avoidance actions recommended by the rules-based or the path planning based collision avoidance algorithms.

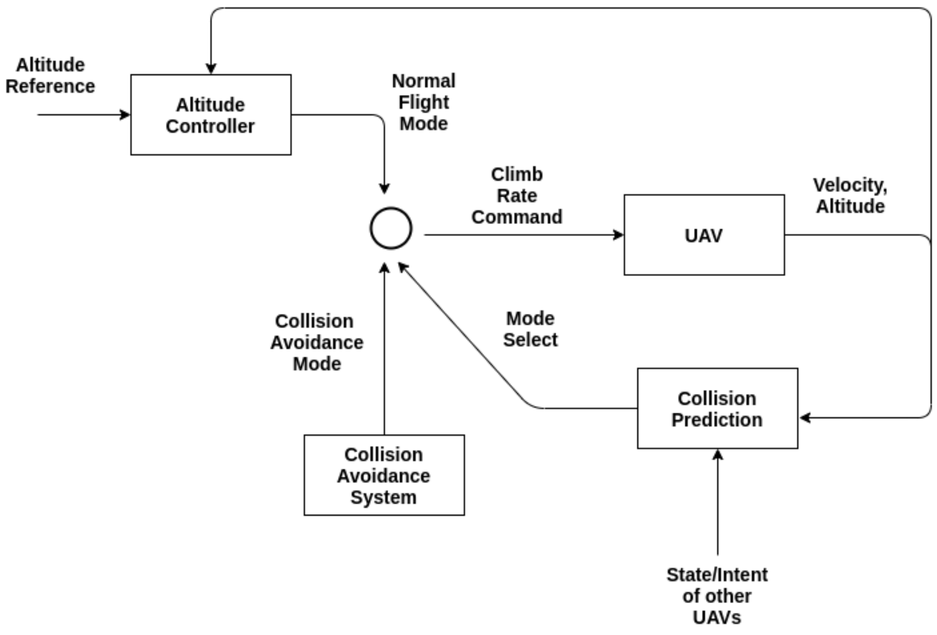


Figure 3.10: Flight Control for switching between Normal Flight mode and Collision Avoidance mode

The objective of the altitude controller is to have the UAV continue to follow its nominal altitude. UAVs will fly at different levels of constant altitudes when executing long-term mission plans [61], therefore during Collision Avoidance mode the altitude control commands will be replaced with conflict resolution control inputs. In Normal Flight mode, the altitude control law below is used to provide the altitude rate commands for a UAV.

$$\dot{h}_i(t) = \frac{1}{\tau_h}\left[h_{i,ref} - h_i(t)\right] \tag{3.21}$$

$$\gamma_i(t) = \arcsin\frac{\dot{h}_i(t)}{\overline{V}_i(t)} \tag{3.22}$$

In Collision Avoidance mode, the altitude controller is disabled and the collision avoidance system issues direct altitude rate commands to the flight control system. The collision avoidance actions assume that the aircraft is travelling horizontally at its cruise speed $\overline{V}_{i,ref}$, and that it only adjusts its climb rate $\dot{h}_{i,ref}$. Following the example of TCAS (to be discussed in Chapter 4), the available actions for each aircraft are chosen as "maintain level flight", "climb", "dive", "steep climb", and "steep dive".

$$U_i = \{\dot{h}_{i,\text{steep climb}}, \dot{h}_{i,climb}, \dot{h}_{i,level}, \dot{h}_{i,\text{steep dive}}, \dot{h}_{i,climb}\} \tag{3.23}$$

with the following

$$\dot{h}_{i,\text{steep climb}} = 2\dot{h}_{climb} \tag{3.24}$$

$$\dot{h}_{i,climb} = +\dot{h}_{climb} \tag{3.25}$$

$$\dot{h}_{i,level} = 0 \tag{3.26}$$

$$\dot{h}_{i,dive} = -\dot{h}_{climb} \tag{3.27}$$

$$\dot{h}_{i,\text{steep dive}} = -2\dot{h}_{climb} \tag{3.28}$$

The altitude changes from the initial altitude h_i due to the commanded altitude rates $\dot{h}_i(t)$. The altitude change Delta h (t) is obtained by integrating the altitude rate command $\dot{h}_i(t)$ with respect to time. The collision avoidance system issues a sequence of altitude rate commands, where each command is held constant for a sampling period T. The following Equations in 3.29 and 3.30 will update altitude states of a UAV during simulation of conflict resolutions.

$$\Delta h_k = \int_{(k-1)T}^{kT} \dot{h}(t)dt \tag{3.29}$$

$$h_k = h_{k-1} + \Delta h_k \tag{3.30}$$

Algorithm 2 shows the altitude controller that is used in Normal Flight mode. During the conflict-free period the UAV altitude controller provides reference altitude control commands. However, after a conflict is detected, conflict resolution system deviates the UAV using altitude rate commands which avoids a collision.

Algorithm 2 Nominal Flight Altitude Controller $AltitudeControl(h, h_{ref}, h_{thr})$

```
Require: Current UAV altitude $h$, saturation distance $h_{thr}$ and reference altitude $h_{ref}$
 1: $h_{max} \leftarrow 50$                                   ▷ saturation limiter ft/min
 2: k ← 0.5
 3: if $||(h_{ref} - h|| < h_{thr}$ then
 4:     $\dot{h} \leftarrow k \cdot (h_{ref} - h)$
 5: else
 6:     if $h_{ref} >$ h then
 7:         $\dot{h} \leftarrow h_{max}$
 8:     else
 9:         $\dot{h} \leftarrow -h_{max}$
10:     end if
11: end if
12: return $\dot{h}$
```

3.4.5 Conflict Duration

In this section, we discuss the analysis of conflict resolution duration for two UAVs manoeuvring in the same direction and an overtaking flight is predicted. When two UAVs are travelling in the same direction at different magnitude horizontal rates, there is a longer conflict resolution period. Therefore, below we shall analyse two UAVs with initial horizontal rates at the start of the simulation that are kept constant throughout. The horizontal rates are kept constant during simulation because state propagation there are no state accelerations. Therefore, the UAV positions are propagated assuming a horizontal velocity $\dot{x}$ which is sampled from a uniform normal distribution.

3.4.5.1 Overtaking Equations of Motion

The horizontal positions of the two UAVs in the overtaking flight scenario are propagated forward in time from their initial positions to estimate the time when the difference in their horizontal positions will fall below the resolution advisory threshold x_{ra}. The UAV horizontal positions x_i and x_j are propagated using equations 3.31 3.32 must be true.

$$x_i(t) = x_i(t = 0) + \dot{x}_i \cdot t \tag{3.31}$$

$$x_j(t) = x_j(t = 0) + \dot{x}_j \cdot t \tag{3.32}$$

The inequality in Equation 3.33 holds for all values of Delta t for which the two UAVs are potentially in conflict. We wish to estimate the conflict duration $\Delta T = t_2 - t_1$ where t_1 and t_2 are the times at the beginning and end of the potential conflict.

$$|(\dot{x}_i \cdot t - \dot{x}_j \cdot t) + (x_i(0) - x_j(0)| < x_{ra} \tag{3.33}$$

3.4.5.2 Resolution Duration

To calculate the time it takes for the one UAV to overtake the other we shall sample randomly two horizontal velocities $\dot{x}_i$ and $\dot{x}_j$. The velocities are sampled randomly from a normal distribution thus two horizontal velocities can be equal or non-equal. Therefore, numerically there exists ϵ and Δx such that,

$$\dot{x}_i - \dot{x}_j = \epsilon \tag{3.34}$$

$$x_i(0) - x_j(0) = \Delta x \tag{3.35}$$

The two UAVs will replan a path after each plan until there exists two time values $t_1, t_2 \in T$ with sufficient horizontal separation distance x_{ra} distance between the UAVs. The system will begin conflict resolution at time t_1 and will return to conflict-free status at time t_2.

$$\epsilon \cdot t_1 + \Delta x = x_{ra} \tag{3.36}$$

$$\epsilon \cdot t_2 + \Delta x = -x_{ra} \tag{3.37}$$

The horizontal ranges and the horizontal rates which lead to UAV j overtaking i are shown in Table 3.1.

ϵ	**Range(Δx)**	Horizontal rates
$\epsilon < 0$	$x_i - x_j > 0$	$\dot{x}_i > 0$
$\epsilon < 0$	$x_i - x_j < 0$	$\dot{x}_i < 0$
$\epsilon > 0$	$x_i - x_j > 0$	$\dot{x}_j < 0$
$\epsilon > 0$	$x_i - x_j < 0$	$\dot{x}_j > 0$

Table 3.1: Conditions which leads to one UAV overtaking another UAV

Case 1: $\epsilon = \dot{x}_i - \dot{x}_j > 0$

$$\begin{aligned}(\dot{x}_i - \dot{x}_j) \cdot t_1 - (x_i - x_j) &= x_{ra}\\ (\dot{x}_i - \dot{x}_j) \cdot t_1 &= x_{ra} + (x_i - x_j)\\ t_1 &= \frac{x_{ra} + (x_i - x_j)}{\dot{x}_i - \dot{x}_j}\end{aligned} \tag{3.38}$$

Case 2: $\epsilon = \dot{x}_i - \dot{x}_j < 0$

$$\begin{aligned}-\Big(\dot{x}_i - \dot{x}_j\Big) \cdot t_2 + (x_i - x_j) &= x_{ra}\\ -(\dot{x}_i - \dot{x}_j) \cdot t_2 &= x_{ra} - (x_i - x_j)\\ t_2 &= \frac{x_{ra} - (x_i - x_j)}{-(\dot{x}_i - \dot{x}_j)}\end{aligned} \tag{3.39}$$

Case 3: $\epsilon = \dot{x}_i - \dot{x}_j = 0$

$$\begin{aligned}0 \cdot t + |x_i - x_j| &= x_{ra}\\ t &= \frac{x_{ra} - |x_i - x_j|}{0}\end{aligned} \tag{3.40}$$

In case 3 there is a contradiction since there is no such $t \in \mathbb{R}$ for this to be true.

The following time and relative horizontal position between a pair of UAVs i and j are shown in Figure 3.11 and 3.12. Figure 3.11 shows conflict beginning at $t_1 = 692.38s$ and ending at $t_2 = 1307.6s$ with $|\dot{x}_i - \dot{x}_j| = 5$ feet/s. Figure 3.12 shows a conflict beginning at $t_1 = 86.5s$ and ending at $t_2 = 163.45s$ with $|\dot{x}_i - \dot{x}_j| = 40$ feet/s. Both examples illustrate the existence of times t_1 and t_2 where the potential conflict starts and ends, with the conflict duration equal to $t_2 - t_1$. Figure 3.13 illustrates a scenario where there is no overtaking (and no conflict) because the two UAVs start with sufficient horizontal separation and equal horizontal velocities.

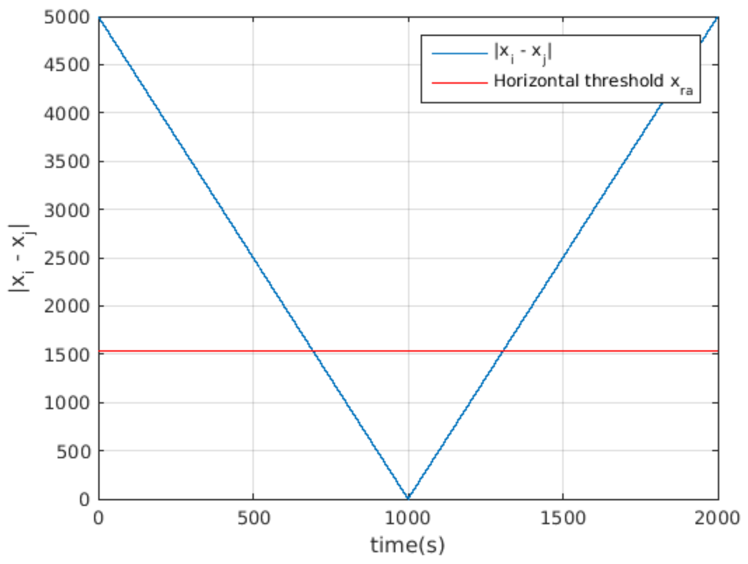


Figure 3.11: Existence of conflict resolution for different horizontal rates at 5 feet/s

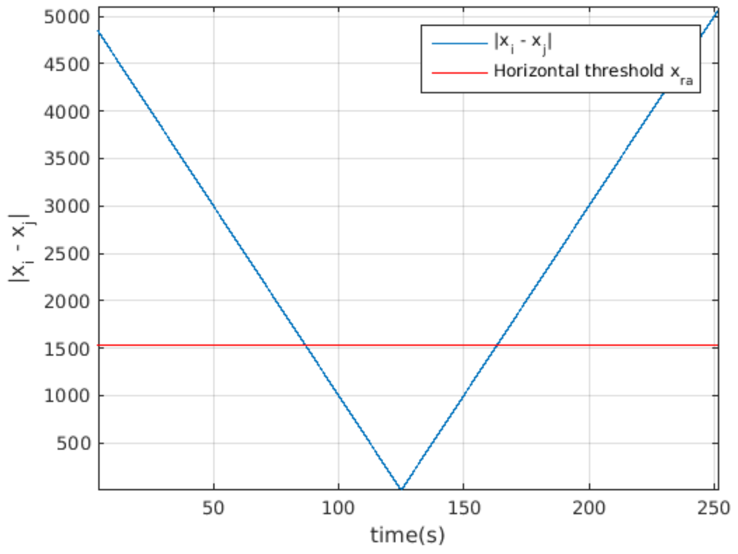


Figure 3.12: Existence of conflict resolution for different horizontal rates at 40 feet/s

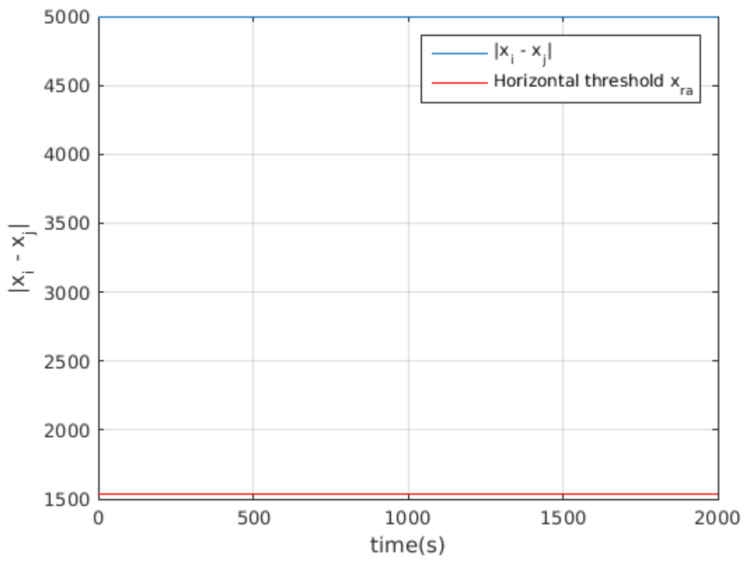


Figure 3.13: No overtaking between UAVs as the two roots $t_1 = \infty$ and $t_2 = \infty$ does not exist with $|\dot{x}_i - \dot{x}_j| = 0$ feet/s

3.5 Cooperative Collision Prediction and Avoidance

Cooperative collision prediction and avoidance for UAVs is conceptualised as shown in Figures 3.14 and 3.15. Figure 3.14 shows the initial positions and velocities of three UAVs in a multi-aircraft conflict scenario. Figure 3.15 shows the intended flight paths of each of the three UAVs.

The cooperative collision prediction and avoidance framework assumes that all cooperative UAVs communicate their current state and intent to all other UAVs. The framework assumes that that dynamic obstacles are tracked by ground stations using ground-based radar, and that the ground stations communicate the states of the dynamic obstacles to the UAVs. The host UAV exclusions zones, the minimum terrain avoidance altitude, and the predicted states of the other cooperative UAVs and dynamics obstacles may therefore be used both for cooperative collision prediction and for cooperative collision avoidance.

The cooperative collision prediction and avoidance system may predict the future states of the cooperative UAVs based on their current states and communicated intent, and may predict the future states of dynamic obstacles based on their current states, but without knowledge of their intent. If a collision is predicted, then the UAVs also coordinate their collision avoidance actions. The cooperative collision prediction and avoidance framework will be realised using a rules-based approach in Chapter 4 and using a path planning based approach in Chapter 5.

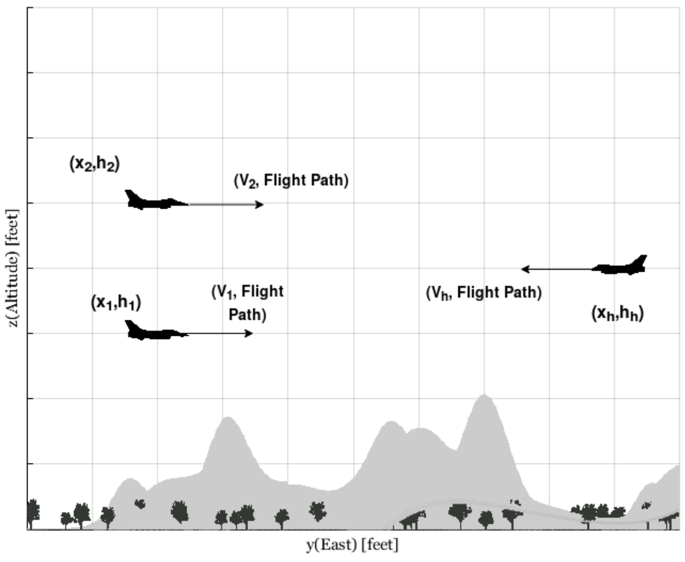


Figure 3.14: Three simultaneous conflict prediction example

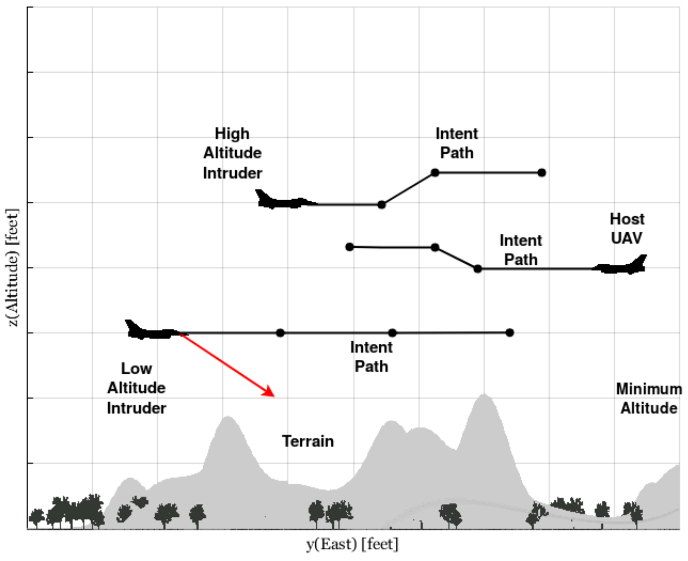


Figure 3.15: Three UAVs which shares intent information for conflict resolution near terrain altitude

3.6 Summary and Discussion

The discussion in this chapter was on the modelling and conceptualization of the UAV translational motion in Section 3.2. The translational equations of UAV in three-dimensional space were then reduced to discrete-time dynamics in two-dimensional state modelling. The two-dimensional states of a UAV are then used for a low time complexity state propagation which discretises a continuous model to obtain the intended flight plan for collision prediction in Section 3.2.4. The flight control system of the UAV switches between two states of a state machine diagram, namely normal flight mode and conflict avoidance mode. The flight control state machine switches between the two states based on the conflict detection status. The conflict detection process is modelled as a deterministic two-dimensional safe separation exclusion zone defined around each UAV in Section 3.3. Each conflict zone region of a UAV can detect and predict a conflict using forward simulation of the state dynamics. The conflict prediction was conceptualised in Section 3.2.4 propagating the states of the UAVs forward in time and checking for conflict with other UAVs and terrain static obstacles below the minimum operating altitude threshold of 1000 feet. In conflict resolution mode, the UAV flight controller will use an altitude rate control system to deviate the UAV from its nominal altitude and return it after the conflict is resolved. Therefore, a discrete set of control inputs with constant horizontal rates were introduced in Section 3.4.3 for trajectory modelling of tactical rules-based and strategic path planning collision avoidance modelled after TCAS. Moreover, the fixed input set can be shared for cooperative collision avoidance of multiple UAVs when using the path planning based approach. The cooperative path planning algorithm will use intent sharing for the simulation of future states to resolve multi-aircraft conflict scenarios in Section 3.5. Finally, this chapter has demonstrated how to conceptualise a cooperative conflict prediction and avoidance system from pairwise and multi-aircraft conflict resolution.

Chapter 4

Rules-Based Collision Avoidance

In this chapter we present a tactical rules-based collision avoidance solution that is modelled after the Traffic Collision Avoidance System (TCAS) that is used on commercial passenger airliners. Our rules-based collision avoidance solution uses only vertical manoeuvres (climb and descend) to avoid aircraft-to-aircraft collisions, as well as collisions with terrain. We first provide the necessary background information on how TCAS operates on manned aircraft, and then describe our rules-based collision avoidance solution which is applied to unmanned aircraft. Next, we describe how the rules-based solution performs collision prediction, thereafter we provide simulations of pairwise collision avoidance, and finally simulate multi-aircraft collision avoidance. Two methods for combining the pairwise rules-based conflict avoidance actions into multiple intruders conflict resolution are proposed, namely Resolution Action Superposition (RAS) and pairwise Closest-Intruder-First (CIF). Simulation results are presented to illustrate the operation of the rules-based collision avoidance solution in both pairwise and multi-aircraft conflict scenarios. The rules-based solution serves as the benchmark for the path planning based solution that will be presented in Chapter 5.

4.1 TCAS Background

This section presents the necessary background information on the Traffic Collision Avoidance System (TCAS) that is used on commercial passenger aircraft. Firstly, the TCAS conflict prediction regions surrounding the host aircraft are defined. Next, the method that TCAS uses to perform conflict prediction based on the estimated time-to-collision and the relative altitude separation is described. Finally, the rules-based approach that TCAS implements to performance collision avoidance using vertical manoeuvres is discussed in depth.

4.1.1 TCAS Regions

The TCAS unit of the host aircraft performs collision prediction and avoidance by interrogating the transponders of all intruder aircraft, determining the time to collision to each aircraft, and issuing traffic advisories and resolution advisories for the pilot to follow.

TCAS defines a protected volume of airspace surrounding the host aircraft. The protected volume given in Figure 4.1 is subdivided into the traffic advisory region and the resolution advisory region. If an intruder aircraft enters the traffic advisory region, then the TCAS unit issues a traffic advisory to alert the pilot to the presence of intruder aircraft, and to prepare the pilot for a potential resolution advisory. If an intruder aircraft enters the resolution advisory region, then the TCAS unit issues a resolution advisory with a collision avoidance action for the pilot to execute. TCAS uses only vertical manoeuvres (climb and descend) to avoid aircraft to aircraft collisions. Traffic advisories are typically issued when the time to collision is less than 40 seconds, and resolution advisories are typically issued when the time to collision is less then 20 seconds.

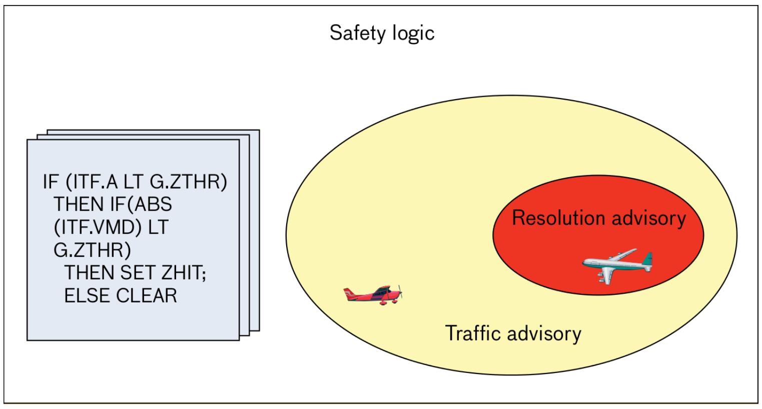


Figure 4.1: TCAS pseudocode for conflict detection adopted from [3]

The conflict regions around the host aircraft in Figure 4.1 are the Traffic and Resolution advisory region in which an intruder aircraft is detected as a potential collision. The Traffic Advisory is the region where the system produces alerts. However, in the Resolution Advisory layer, action manoeuvres are required. Conflict resolution advisories are issued to an aircraft to maintain separation distance and time-to-collision for aircraft travelling at different rates. The time-to-closest approach τ and range rates are computed each second in the system. This we shall discuss in more detail in Section 4.2.1. Additionally, the conflict detection and resolution are determined from the altitude sensitivity, as an example in level 5 aircraft between 5000 feet to 10000 feet above ground level evaluate conflict detection as follows from [19];

- Sensing : Range between aircraft of less than 37 km(20 NM)
- Traffic Alert : Range starting from 6.482 km(3.5 NM) time to collision of 40s until RA starts
- Resolution Alerts : Range less than 2.1 km(3.8892 NM) with time to collision of 20s

4.1.2 TCAS Collision Prediction

The TCAS unit of the host aircraft performs collision prediction by interrogating the transponders of all intruder aircraft and tracking their slant range, altitude, and relative heading. The conflict prediction algorithm calculates the time to reach the closest point of approach and the time to reach co-altitude with the intruder aircraft, and uses these values to issue traffic advisories and resolution advisories. The traffic advisory and resolution advisory regions are defined in terms of the time to closest point of approach (range tau) and time to co-altitude (vertical tau), in conjunction with a modified minimum distance DMOD, and a fixed altitude threshold ZTHR to accommodate slow closure rates. The thresholds range tau, vertical tau, DMOD and ZTHR are chosen to ensure a minimum vertical separation ALIM at the point of closest approach. Therefore, TCAS uses the time to closest point of approach rather than the distance to determine whether a traffic advisory or a resolution advisory should be issued.

The full technical detail of the TCAS conflict prediction algorithm is described in the TCSA II booklet [19]. The range tau is equal to the slant range divided by the closing speed. The vertical tau is equal to the altitude separation divided by the vertical closing speed. A traffic advisory or resolution advisory is only issued when both the range tau and the vertical tau are less than the specified threshold values for the sensitivity level. A problem with this simple definition of tau is that in encounters where the rate of closure is very low, an intruder aircraft can come very close in range without crossing the range tau. To provide protection in these types of encounters, a modified definition of range tau is used. At larger ranges and higher closure rates these boundaries are essentially equal to those defined by the basic tau concept. However, at close ranges and at slower closure rates the modified tau boundaries converge to a non-zero range called DMOD. This modification allows TCAS to issue TAs and RAs at or before the fixed DMOD range threshold in these slow-closure-rate encounters. There is a similar problem when the vertical closure rate of the TCAS and the intruder aircraft is low, or when they are close but diverging in altitude. To address that problem, TCAS uses a fixed altitude threshold, referred to as ZTHR, in conjunction with the vertical tau, to determine whether a TA or an RA should be issued. TCAS operates at different sensitivity levels, determined by the altitude of the host aircraft. The thresholds for traffic advisories and resolution advisories, at the different sensitivity levels, are shown in Table 4.1.

Table 4.1: TCAS Sensitivity Level TA and RA [19]

Ownship Altitude (feet)	**SL**	**TAU(sec)**		**DMOD(NM)**		**ZTHR(feet)**		**ALIM(feet)**
		TA	**RA**	**TA**	**RA**	**TA**	**RA**	
1000 - 2350	3	25	15	0.33	0.20	850	600	300
2350 - 5000	4	30	20	0.48	0.35	850	600	300
5000 -10000	5	40	25	0.75	0.55	850	600	350
10000 - 20000	6	45	30	1.00	0.80	850	600	400
20000 - 42000	7	48	35	1.30	1.10	850	700	600
$>$ 42000	7	48	35	1.30	1.10	800	1200	700

The physical interpretation of the variable τ is an instantaneous indication in seconds of how much time is left before the two UAVs have a near mid-air collision. According to literature, we formally express τ as the time to closest distance, as a ratio of the pairwise aircraft relative range $r \gg 0$ to the closure rate value $-\dot{r} \neq 0$.

$$\tau = -\frac{r}{\dot{r}} \in \mathbb{R} \tag{4.1}$$

The Equation 4.1 is the original TCAS time-to-collision (tau) parameter which was later discovered to be limiting [19]. Consequently, a reviewed time-to-collision was then proposed as part of the TCAS II design which introduces the horizontal separation thresholds DMOD and altitude threshold ZTHR [19]. These improvements were proposed to make the TCAS conflict detection and resolutions robust. Some of the major issues addressed were as follows; low relative vertical rate $\dot{h}$, closure rate $-\dot{r}$ and very close undetected range r. The improved tau was called modified-tau τ_{mod}. Therefore, the modified-tau (τ_{mod}) design would issue alerts to aircraft at both horizontal threshold DMOD and vertical threshold ZTHR, without considering the value of tau. Furthermore, these two thresholds were added to a tau expression in Equation 4.2 [74]. The modelling of the time-to-collision will be demonstrated in the next section.

$$\tau_{mod} = -\frac{(r - \text{DMOD})^2}{\dot{r}} \tag{4.2}$$

4.1.3 TCAS Collision Avoidance

The rules for conflict detection were determined as nominal state propagation of the ownship and intruder aircraft as shown in Figure 4.2. The relative vectors of pairwise UAVs in Section 4.2.1 are vertical and horizontal axes systems: distance vector r, velocity vector $\dot{r}$. The vectors r and $-\dot{r}$ extrapolates the closest-point-approach to evaluate two useful concepts known as closure-rate and time-to-closest distance. Therefore, the following equations of the conflict detection are derived between a pairwise of unmanned aircraft i and j having i as a reference host UAV.

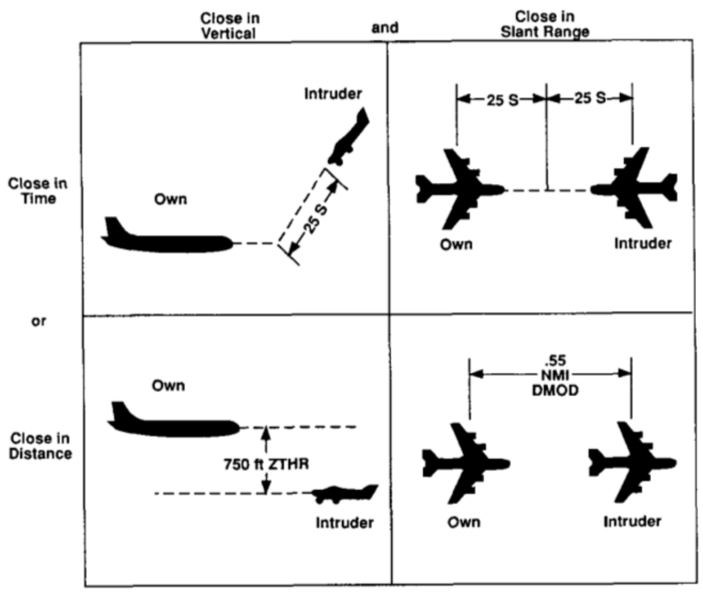


Figure 4.2: TCAS conflict detection system in time and distance from [8]

The miss distance between conflict detection parameter is calculated as follows:

$$r_{miss} = r_{i,j} + (\Delta\dot{x} + \Delta\dot{y})\tau_{i,j} \tag{4.3}$$

Where $\tau_{i,j}$ is time to the miss distance and $c_{i,j}$ is closure rate between two aircraft i and j .

$$\tau_{i,j} = \frac{r_{i,j}}{c_{i,j}} \tag{4.4}$$

$$c_{i,j} = -\dot{r}_{i,j} \tag{4.5}$$

Using the time to collision τ as a strategy for conflict detection, we have formulated the following numerical analysis rules:

1. $\tau > 0$ and $\dot{r} < 0$ aircraft are increasingly getting closer to each other, r distance reducing.

2. $\tau < 0$ and $\dot{r} > 0$ aircraft are increasingly getting away from each other, r distance increasing.

3. $||r_{miss}|| \leq r_{min}$ conflict detected using a minimum distance threshold $r_{min} > 0$.

The conflict prediction rules can now be classified into six cases. Each of these classifications describe a different scenario or rules which an aircraft will encounter during conflict. These rules are applicable to the rules-based conflict scenario of the time-to collision method.

Case	$-\dot{r}$	$\lvert r - r_{min} \rvert$	τ	**Conflict Classification**
0	≈ 0	≈ 0	0	Parallel intruder aircraft
1	Infinity	≈ 0	> 0 or < 0	Intruder is much closer
2	> 0	$\gg 0$ or ≈ 0	< 0	Intruder moving away, increasing
3	< 0	$\gg 0$ or ≈ 0	> 0	Intruder moving closer, decreasing
4	> 0 or < 0	≈ 0	> 0 or < 0	Threshold r_{min} conflict
5	$\gg 0$	$\gg 0$	$\gg 0$ or $\ll 0$	Clear of Conflict

Table 4.2: Conflict Detection Scheme using time-to-conflict(τ) [24]

TCAS performs collision avoidance using only vertical manoeuvres (climb and descend) to avoid aircraft to aircraft collisions. TCAS does not control the aircraft directly but advises resolution actions for the human pilot to follow. If an intruder aircraft enters the Resolution Advisory region of the host aircraft, then a conflict is detected, and the TCAS unit advises a conflict resolution action for the human pilot to execute. The objective of the resolution advisory is to maximise the vertical separation between the host aircraft and the intruder aircraft. In a pairwise collision avoidance scenario, the TCAS unit of each aircraft chooses the resolution action based on the time to collision and the relative altitude of the other aircraft, as shown in Figure 4.3. Possible resolution actions for the host aircraft are "maintain level flight", "climb", "descend", "steep climb" and "steep descend".

The climb rates for the TCAS advisories are summarised in Table 4.3. If the host aircraft is at a higher altitude than the intruder aircraft, the resolution advisory for the host aircraft is to climb. This is given that time-to-collision is longer. But when time to collision is shorter a steep climb command is issued. In a scenario when the host aircraft is at a lower altitude than the intruder aircraft, then the resolution advisory for the host aircraft is to descend. This occurs when the time to collision is longer. Additionally, steep descend altitude commands are issued for advisory when the time to collision is shorter, this will continue until there is sufficient vertical separation. Finally, when either the host or an intruder aircraft are below the minimum TCAS altitude, the resolution advisory for the host aircraft is to maintain level flight.

Table 4.3: TCAS command advisories [75][19]

Separation	**Advisory Text**	**Altitude rate**
$\Delta h < 0$	Climb any between	$\dot{h} \in [1500, 3000]$ ft/min
$\Delta h > 0$	Descend any between	$\dot{h} = [-1500, -3000]$ ft/min
$\Delta h \gg h_{DMOD}$	Level Off at	$\dot{h} = 0$ ft/min
$\Delta h > h_{DMOD}$	Maintain climb or descend	$\dot{h}_k = \dot{h}_{k+1}$
$\Delta h \gg h_{DMOD}$	Clear of Conflict	$\dot{h} = \dot{h}_{command}$

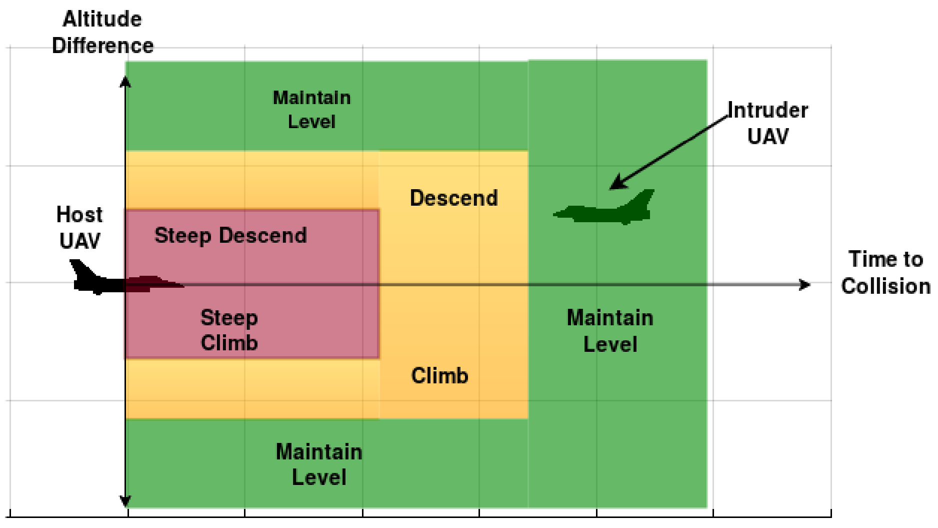


Figure 4.3: Graphical description of TCAS host aircraft conflict resolution advisories

4.2 Rules-Based Collision Avoidance for UAVs

This section describes our rules-based collision avoidance system for unmanned aerial vehicles, which is modelled after TCAS for piloted commercial passenger aircraft. Similar to TCAS, our system for unmanned aerial vehicles uses only vertical manoeuvres to avoid aircraft to aircraft collisions. TCAS issues resolution advisories for a human pilot to follow; our system issues resolution actions for the autopilot to follow. Unlike TCAS, our system does not issue traffic advisories. This is because, unlike a human pilot, the autopilot does not have to be alerted to the presence of intruder aircraft, and does not have to be prepared for a potential resolution advisory. Our system therefore only defines a conflict region similar to the TCAS resolution advisory region which does not include a "traffic advisory" region.

In section4.2.1, the conflict region for our rules-based approach is defined in terms of a horizontal time to collision and a relative altitude separation. In Section 4.2.2 a set of analytical rules of conflict detection with time-to-collision are formulated in a table with common six cases of conflict configurations. The stages of flight for each UAV for control is given as a state machine diagram showing two states of flight; Nominal Flight and Collision Avoidance. Finally, a discretised Collision Avoidance control of a UAV flight during conflict resolution of vertical manoeuvre profile are graphically illustrated in Section 4.2.3. The method which determines the vertical direction and magnitudes of conflict resolutions are given in Algorithm 4 and 5.

4.2.1 Conflict Region

Our rules-based collision avoidance system defines the conflict region as shown in Figure 4.4. The conflict region is defined in terms of a horizontal time to collision threshold and a relative altitude threshold. If an intruder aircraft enters the conflict region of the host aircraft, then the rules-based collision avoidance system issues a resolution action for the autopilot to follow. The resolution action issued to the host UAV is chosen based on the location of the intruder UAV within the conflict region of the host UAV.

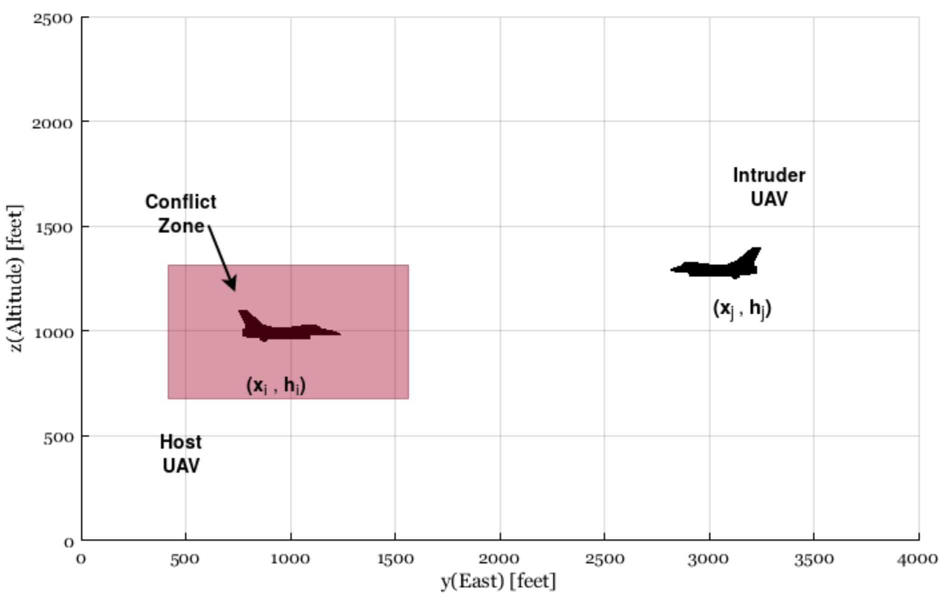


Figure 4.4: Two aircraft on a collision path example (not drawn to scale)

The conflict region is defined by both the time to collision tau and the relative altitude Δh. The time to collision tau is calculated with

$$\tau = \frac{-r}{\dot{r}} \tag{4.6}$$

where r is the range from the host aircraft to the intruder aircraft, and $\dot{r}$ is the range rate. The range r is calculated with

$$r = \sqrt{\Delta x^2 + \Delta y^2} \tag{4.7}$$

with

$$\Delta x = x_j - x_i \tag{4.8}$$

$$\Delta y = y_j - y_i \tag{4.9}$$

where Δx is the horizontal separation and Delta y is the vertical separation between the host and the intruder. The range rate $\dot{r}$ is calculated with

$$\dot{r} = \frac{\Delta\dot{x}(x_j - x_i) + \Delta\dot{y}(y_j - y_i)}{r} \tag{4.10}$$

with

$$\Delta\dot{x} = \dot{x}_j - \dot{x}_i \tag{4.11}$$

$$\Delta\dot{y} = \dot{y}_j - \dot{y}_i \tag{4.12}$$

where $\Delta\dot{x}$ is the horizontal separation rate and $\Delta\dot{y}$ is the vertical separation rate. The relative altitude Δh is calculated as :

$$\Delta h = y_j - y_i \tag{4.13}$$

where y_j is the altitude of the intruder, and y_i is the altitude of the host. The range r is always positive and cannot become negative. The range rate is negative when the host aircraft and intruder aircraft are approaching each other, and the range rate is positive when the two aircraft are receding from one another. The time to collision is positive when the two aircraft are approaching each other, and is negative when the two aircraft are receding from each other. The range, range rate, and time-to-collision versus time is shown in Figures 4.5, 4.6, and 4.7 for an example scenario where two aircraft are flying in opposite directions, but at different altitudes. The aircraft approach each other at a constant horizontal separation rate, pass each other midway, and then continue along their way.

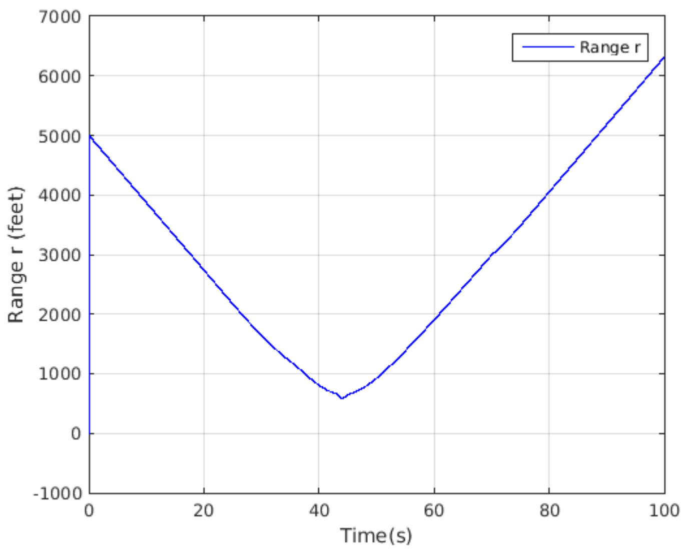


Figure 4.5: Pairwise aircraft relative range r function

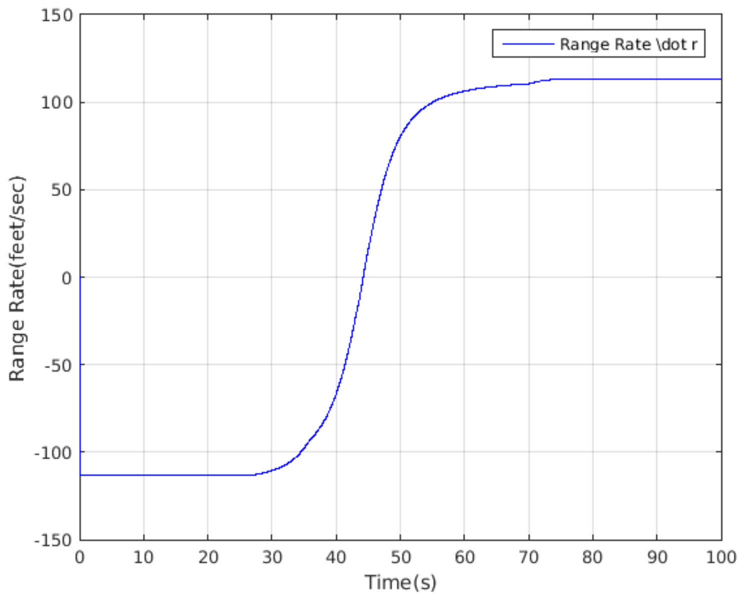


Figure 4.6: Pairwise aircraft range rate($\dot{r}$) function

In Figure 4.7 the time-to-collision function spikes when the relative range rate goes through zero at t = 45 seconds when the two aircraft pass each other, and range is divided by zero at the point of closest approach, leading to the large negative time to collision.

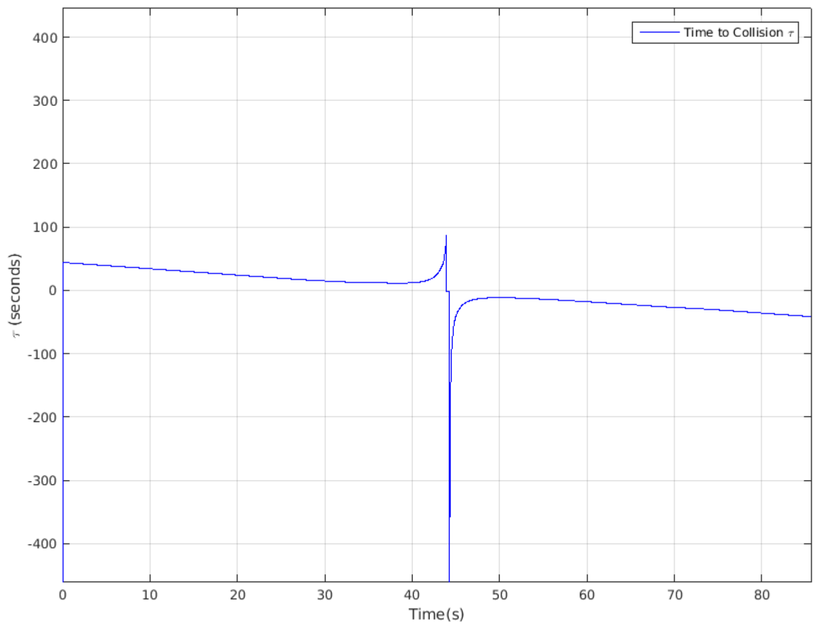


Figure 4.7: Pairwise aircraft time to conflict ($\tau_{i,j}$) function

4.2.2 Rules-Based Conflict Detection

Our rules-based collision prediction module calculates the time to collision tau and the vertical separation Δy based on the positions and velocities of the host aircraft and the intruder aircraft. The host aircraft obtains its own position and velocity from its onboard sensors, and obtains the position and velocity of the intruder aircraft by interrogating its transponder. If both the time-to-collision τ and the vertical separation Δy between the host aircraft and the intruder aircraft are less than the thresholds of the conflict region, then the rules-based system predicts that a collision will occur, and triggers the conflict resolution function.

4.2.2.1 Collision Avoidance Procedure

The rules which were defined previously in Section 3.3.2 in Algorithm 3 are executed each second after communication with intruder aircraft interrogation. The algorithm calculates range dynamics for conflict detection status of each UAV. Therefore, the outcome of the conflict detection each second is from states propagation of pairwise UAV which can be classified as shown by $conflict$ below.

$$conflict = \begin{cases} 0 & \text{no-conflict} \\ 1 & \text{conflict-alert} \\ 2 & \text{conflict-resolution} \end{cases} \tag{4.14}$$

The conflict detection algorithm procedure is given in Algorithm 3. The algorithm from line 1 to 8 initializes the TCAS constants and thresholds from the table found in Appendix 4.1. Furthermore, the algorithm below ensures that the TCAS conflict detection alerts are activated at an altitude of 1000 feet above ground. Between line 3 and 22 the conflict detection is activated for an aircraft. Therefore, the relative dynamics includes the relative vertical and horizontal range in line 9 until 14 for computation of time-to-collision (τ) in line 15. Thereafter, the algorithm will use the time-to-collision and DMOD threshold for the conflict detection between the aircraft.

4.2.3 Rules-Based Conflict Avoidance

The function for altitude direction of conflict advisories is simulated shown in Algorithm 4. The collision avoidance module issues one of five possible climb rate actions, namely "Level Off", "Climb", "Descend", "Steep Climb" and "Steep Descend". The direction of the resolution action is determined by the sign of the relative altitude between the host aircraft and the intruder aircraft. In a scenario when the host aircraft is at a higher altitude than the intruder aircraft, then a "climb" or "steep climb" resolution action is chosen. Secondly if the host aircraft is at a lower than the intruder aircraft, then a "descend" or a "steep descend" resolution action is chosen. In a third condition when the host aircraft is at exactly the same altitude as the intruder aircraft, then theoretically there is the risk that both aircraft will choose a resolution action in the same direction. However, in practice it is unlikely that both aircraft will be exactly at the same altitude. In addition,

Algorithm 3 Pairwise Aircraft Conflict Detection Scheme

```
Require: host $(x_h, y_h, h_h)$ and intruder $(x_i, y_i, h_i)$
 1: AGL ← 1000 feet
 2: conflict ← 0                                   ▷ Default conflict status
 3: if $h_h > AGL$ then
 4:     Sensitivity ← k                            ▷ Choose sensitivity $k \in [2, 7]$
 5:     $\tau_{ta} \leftarrow TA[k]$
 6:     $\tau_{ra} \leftarrow RA[k]$
 7:     $DMOD \leftarrow DMODS[k]$
 8:     $ZTHR \leftarrow ZTHRS[k]$
 9:     $\Delta x \leftarrow x_i - x_h$
10:     $\Delta h \leftarrow h_i - h_h$
11:     $\dot{x} \leftarrow \dot{x}_i - \dot{x}_h$
12:     $\dot{h} \leftarrow \dot{h}_i - \dot{h}_h$
13:     $r \leftarrow \sqrt{\Delta x^2 + \Delta h^2}$
14:     $\dot{r} \leftarrow \frac{\dot{x} \cdot \Delta x + \dot{h} \cdot \Delta h}{r}$
15:     $\tau \leftarrow -\frac{\dot{r}}{r}$
16:     if $\tau \leq \tau_{ta}$ or $(\Delta x < ZTHR \;\&\; \Delta x < DMOD)$ then
17:         conflict = 1
18:         if $\tau \leq \tau_{ra}$ then
19:             conflict ← 2
20:         end if
21:     end if
22: end if
23: return conflict
```

the altitude sensors of both aircraft will contain random sensor noise, which means that even if they happen to report exactly the same altitude at a given time instant, they will report slightly different altitudes at the next time instant. The random altitude sensor noise will therefore ensure that one aircraft will always read a slightly higher or lower altitude than the other aircraft, providing a tie-breaker for the complementary resolution actions. The direction for the host aircraft's vertical collision avoidance manoeuvre is dictated by the altitude of the intruder aircraft relative to the host aircraft, and is calculated using Algorithm 4 . Furthermore, Algorithm 4 introduces noise as discussed below for decisions on the direction of altitude conflict advisories. The random noise is added to each of the aircraft altitude because there is in no deterministic direction when the altitude difference is zero ($\Delta h = 0$). Consequently, we have resolved to add into the normal altitude a noise w_k to a UAV altitude h_k in order to break the equal altitude numerical issue.

$$h_k = h_k + w_k \tag{4.15}$$

where $w_k \in [0, 1]$ is a Uniform Distributed random variable,

$$w_k \sim \mathcal{U}(0,\, 1)\,. \tag{4.16}$$

Algorithm 4 Manoeuvre Direction $AltitudeDirection(status, h_h, h_i)$

```
Require: conflict $status$ and aircraft altitudes of host $h_h$ and intruder $h_i$
 1: AGL ← 1000 feet
 2: $w_h \leftarrow \mathcal{N}(h_h, 1)$
 3: $h_h \leftarrow h_h + w_k$
 4: Direction ← 0                                  ▷ Default resolution is to level off
 5: $\Delta h \leftarrow h_h - h_i$
 6: if $h_h > AGL$ and $status$ equals to 2 then
 7:     if $h_h \geq h_i$ then
 8:         Direction ← 1                          ▷ Climb
 9:     else
10:         Direction ← -1                         ▷ Descend
11: end if
12: end if
13: return Direction
```

The magnitude of the altitude rate for the host aircraft's vertical collision avoidance manoeuvre is dictated by the altitude difference between the intruder aircraft and the host aircraft, and is calculated using Algorithm 4 5. We choose a minimum vertical separation threshold as 600 feet. For relative altitudes of 0 to 300 feet, the collision avoidance module issues a "steep climb" or "steep descend" command that correspond to climb rate commands of +/-2500 feet/minute. For relative altitudes of 300 to 600 feet, the collision avoidance module issues a "climb" or "descend" command that correspond to climb rate commands of +/-1500 feet/minute. For relative altitudes of greater than 600 feet, the collision avoidance module issues a "level off" command that corresponds to a climb rate of 0 feet/minute.

Algorithm 5 Conflict Magnitude $Magnitude(Direction, \Delta h)$

```
Require: Conflict resolution set $R$, Direction of resolution $D \in \{1, -1\}$ and aircraft
    altitudes difference $\Delta h \in \mathbb{R}$
 1: $h_{thr} \leftarrow 600ft$
 2: Resolutions ← $[\frac{2500}{60}, \frac{1500}{60}]$        ▷ feet/s
 3: if $|\Delta h| \geq \frac{h_{thr}}{2}$ then
 4:     magnitude ← Direction · Resolutions(1)
 5: else
 6:     magnitude ← Direction · Resolutions(2)
 7: end if
 8: return magnitude
```

4.2.4 Flight Control Modes

The rules-based collision avoidance system operates in two modes, namely Normal Flight mode and Collision Avoidance mode. In Normal Flight mode, the UAV uses the altitude controller to follow its planned flight path. In Collision Avoidance mode, the UAV uses the climb rate controller to follow the climb rate commands provided by the collision avoidance system. The state machine that controls the transitions between the flight control modes is shown in Figure 4.8.

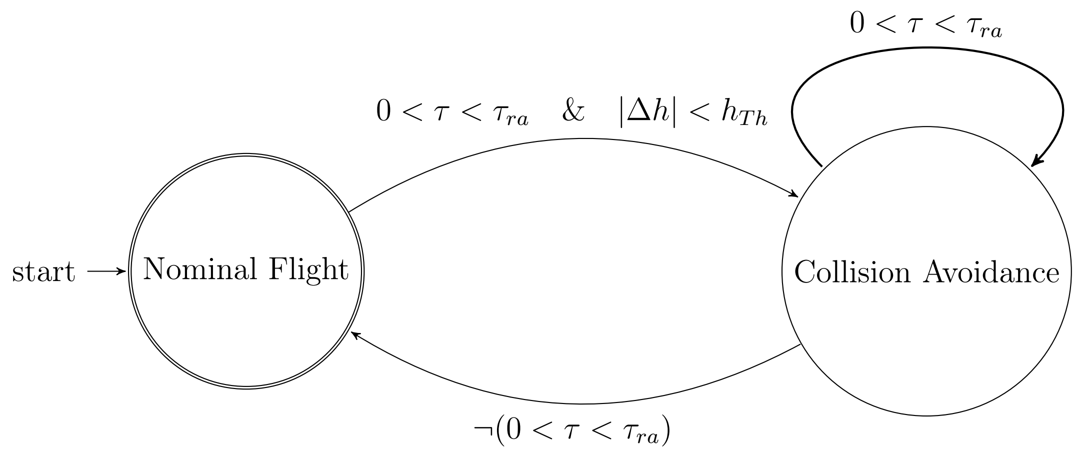

$0 < \tau < \tau_{ra}$
$0 < \tau < \tau_{ra} \quad \& \quad |\Delta h| < h_{Th}$
start
Nominal Flight
Collision Avoidance
$\neg(0 < \tau < \tau_{ra})$

Figure 4.8: Conflict Detection State Machine of TCAS rule-based system

The UAV nominally operates in Normal Flight mode. If both the time to collision and the relative altitude of an intruder aircraft is inside the host aircraft's conflict region, then the flight control mode transitions from Normal Flight mode to the Collision Avoidance mode. The flight control mode remains in Collision Avoidance mode until the time to collision becomes negative (meaning that the intruder aircraft has passed and is withdrawing) or the time to collision becomes greater than the conflict region threshold. Note that the transition from Collision Avoidance mode back to Normal Flight mode only checks the time to collision, and does not check the relative altitude. This means that the UAV will not return to normal flight if relative altitude meets the minimum vertical separation, but the time to collision is still inside the conflict region.

4.3 Pairwise Collision Avoidance

In this section we present some illustrative simulation results for our rules-based collision avoidance system in different pairwise conflict scenarios.

4.3.1 Illustrative Simulation Results: Pairwise Conflict Scenarios

Simulations were performed to illustrate the behaviour of our rules-based collision avoidance system in pairwise conflict scenarios. Simulation results are presented for the following scenarios: head-on conflict, head-on conflict at minimum altitude, overtaking, and parallel flight.

4.3.1.1 Pairwise Head-on Conflict

The conflict scenario in Figure 4.9 illustrates the simulation results of a head-on conflict with one aircraft approaching from the left and the other from the right. The simulation results illustrate the typical behaviour of the rules-based collision avoidance system in a pairwise aircraft to aircraft conflict scenario.

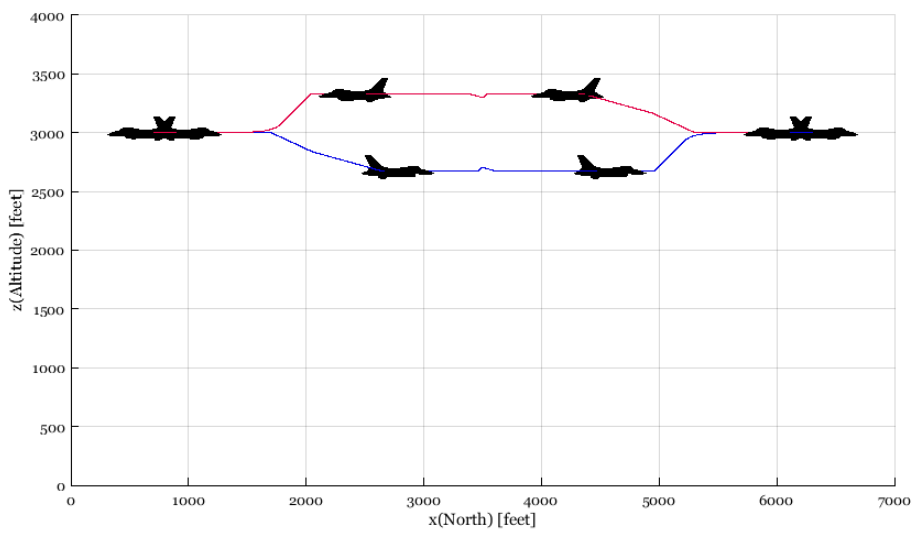


Figure 4.9: Head-on collision avoidance illustration

In the conflict in Figure 4.9 both UAVs are flying at an altitude of 3000 feet, which is well above the minimum altitude of 1000 feet for terrain avoidance. The approaching aircraft from the left is at a slightly higher altitude than the aircraft approaching from the right, then the two aircraft are approaching one another at a closure rate of 175 feet per second. Then when the host aircraft reach a time-to-collision of $\tau = 20$ seconds (with a vertical separation of less than 600 feet), the conflict is detected, and both aircraft switch to Collision Avoidance mode. From that point the conflict resolution module of both aircraft are activated to issue collision avoidance commands as shown in Figure 4.10 and 4.11. Since vertical separation between the aircraft is initially less than 300 feet, the aircraft at the higher altitude performs a steep climb, and the aircraft at the lower altitude performs a steep descent. When the vertical separation reaches 300 feet, the aircraft at the higher altitude switches to a normal climb, and the aircraft at the lower altitude switches to a normal descent. Ultimately, when vertical separation reaches 600 feet, both aircraft level off and maintain their respective altitudes.

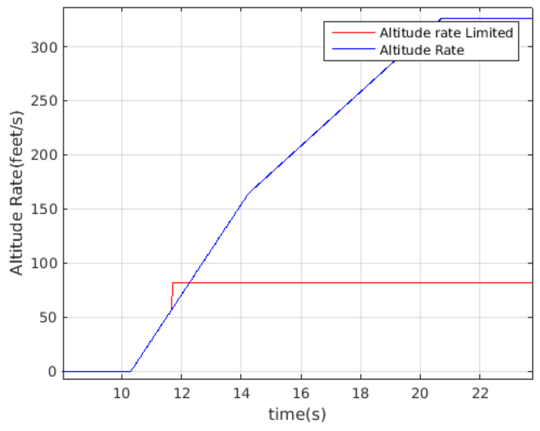


Figure 4.10: Altitude Descend Command

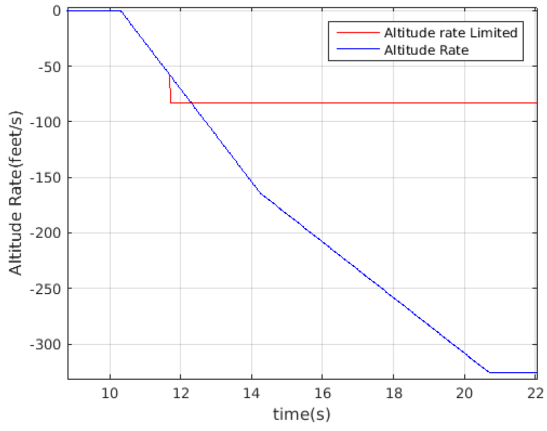


Figure 4.11: Altitude Climb Command

When the time-to-collision reaches $\tau = 0$ seconds, the vertical separation is 600 feet, and the collision is avoided. When the time to collision becomes negative $\tau < 0$, the conflict has passed, and both UAVs switch back to Normal Flight mode. Thereafter, we observe that during Normal Flight mode, both aircraft use their altitude controllers to return to their nominal altitudes exhibiting an exponential transient response shown in Figure 4.12 and 4.13.

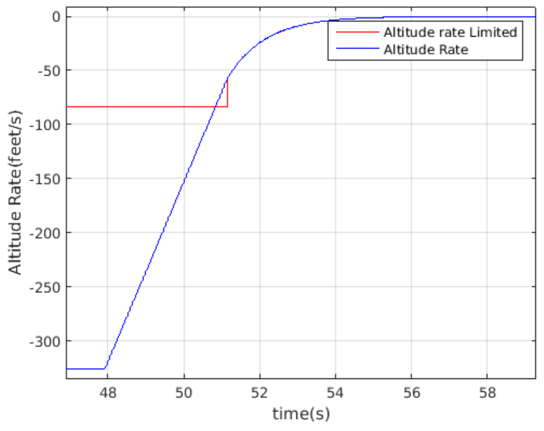


Figure 4.12: Normal Flight descend

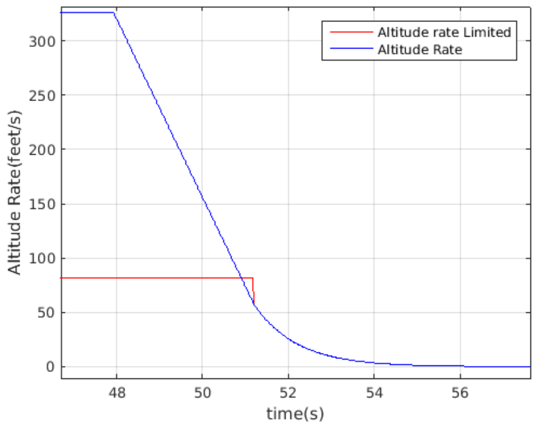


Figure 4.13: Normal Flight climb

4.3.1.2 Pairwise Head-On Conflict at Minimum Altitude

The following conflict scenario in Figure 4.14 shows the simulation results for a head-on conflict scenario, but with one of the aircraft already operating at its minimum altitude for terrain avoidance. The simulation results illustrate the typical behaviour of the rules-based collision avoidance system in a pairwise conflict scenario when terrain avoidance is also active.

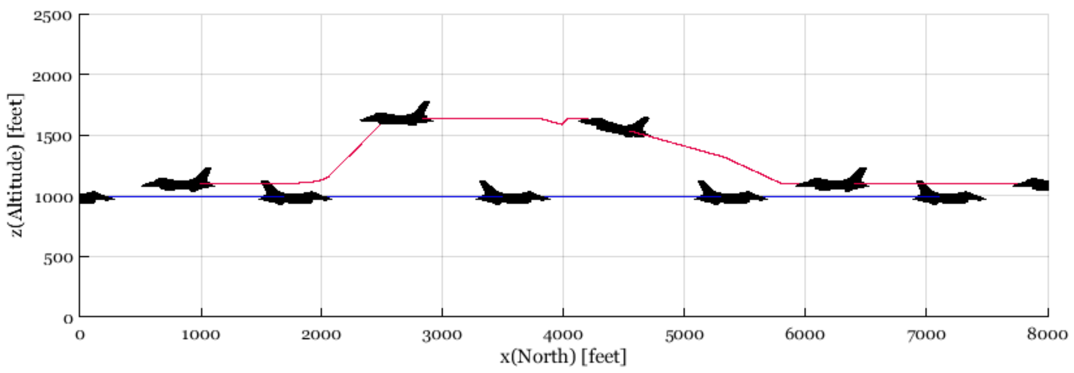


Figure 4.14: Close Encounter conflict between only one manoeuvring aircraft

The simulation shows two UAVs that are flying at an altitude of about 1000 feet, which is the minimum altitude for terrain avoidance. There is an aircraft approaching from the right at a slightly higher altitude than the aircraft approaching from the left. The conflict closure rate of approach is 175 feet per second. Then, at the time when the aircraft reach a time-to-collision of $\tau = 20$ seconds (with a vertical separation of less than 600 feet), a conflict is detected, and both aircraft switch to Collision Avoidance mode. The conflict resolution module of both aircraft are therefore activated to issue collision avoidance commands. Since the vertical separation between the aircraft is initially less than 300 feet, the aircraft at the higher altitude performs a steep climb. However, the aircraft at the lower altitude is already at its minimum altitude and therefore is not allowed to descend further. Therefore, the aircraft maintains its level flight because it is below the minimum altitude. When the vertical separation reaches 300 feet, the aircraft at the higher altitude switches to a normal climb. (The aircraft at the minimum altitude keeps on maintaining level flight.) When the vertical separation reaches 600 feet, the aircraft at the higher altitude levels off and maintains its altitude. Next, when the time-to-collision reaches $\tau = 0$ seconds and the vertical separation is 600 feet then a collision is avoided. Additionally, when time-to-collision becomes negative $\tau < 0$, the aircraft has have passed one another then the conflict is cleared. Finally, both aircraft will switch back to Normal Flight mode. In the Normal Flight mode, both aircraft re-activate their normal flight altitude controllers. The aircraft that performed the avoidance manoeuvre returns to its nominal altitude exhibiting an exponential transient response.

4.3.1.3 Pairwise Overtaking Flight Scenario

The conflict scenario in Figure 4.15 shows the simulation results for an overtaking conflict scenario, with both aircraft travelling from left to right, but with the one aircraft overtaking the other. The simulation results illustrate the behaviour of the rules-based collision avoidance system in an extended conflict scenario with low horizontal closure rates, where the time to collision parameter does not detect the conflict. Both aircraft are flying from left to right at the same altitude of 5000 feet, with the faster aircraft approaching the slower aircraft from the rear.

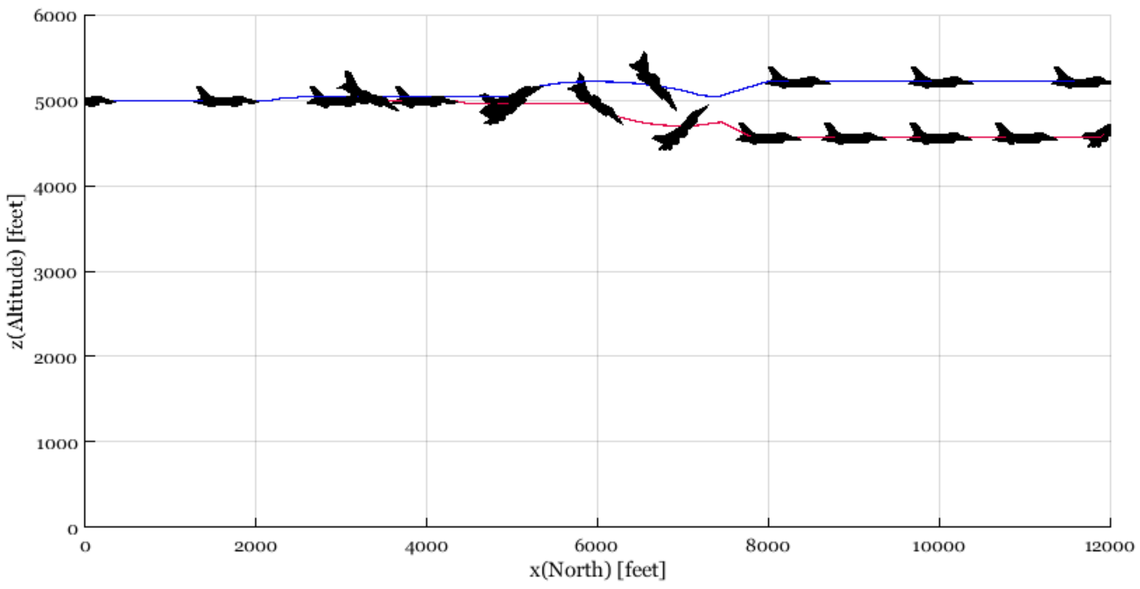


Figure 4.15: Two aircraft in a overtaking conflict

The aircraft start the simulation with an initial horizontal separation of 3000 feet. The aircraft are travelling at a closure rate of 15 feet per second (constant free of conflict).

Both aircraft are flying well above the minimum altitude of 1000 feet for terrain avoidance. Note that since both aircraft are flying in the same direction, their closure rate is close to zero, and the time to collision remains relatively large, even for relatively small horizontal separation distances. When the aircraft reach a horizontal range of $\Delta x = 2126$ feet, a conflict is detected due to the fact that the both the minimum horizontal separation threshold DMOD and the minimum vertical separation threshold ZTHR are violated. Then the algorithm switches to Collision Avoidance mode, and their conflict resolution modules are activated to issue collision avoidance commands. Due to fact that the vertical separation between the aircraft is initially less than 300 feet, the faster aircraft (which is at the higher altitude) performs a steep climb, and the slower aircraft (which is at the

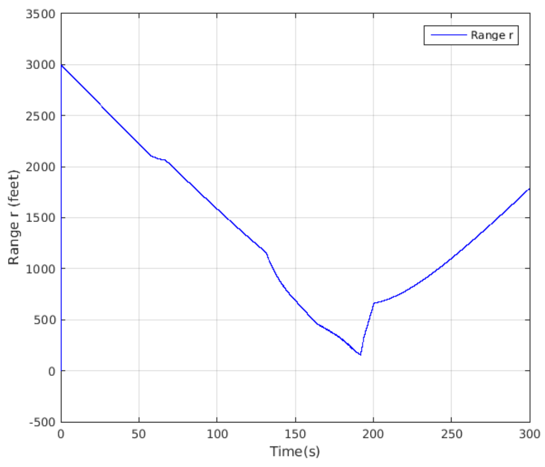


Figure 4.16: Range distance (r)

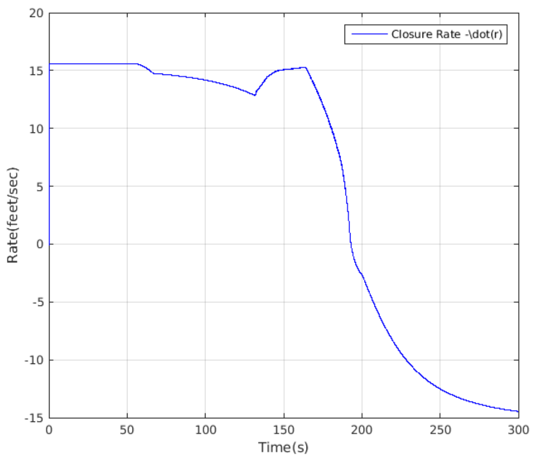


Figure 4.17: Closure rate ($-\dot{r}$)

lower altitude) performs a steep descent.

By the time that the vertical separation reaches 300 feet, the faster aircraft at the higher altitude switches to a normal climb, and the slower aircraft at the lower altitude switches to a normal descent. The moment that the vertical separation reaches 600 feet, both aircraft level off and maintain their respective altitudes. When the time to collision reaches $\tau = 0$ seconds, the vertical separation is 600 feet, and the collision is avoided. The faster aircraft then gradually overtakes the slower aircraft. During the overtaking, the horizontal separation remains below the minimum safe horizontal separation threshold, and the conditions for switching back to Normal Flight mode are not reached. Therefore, the conflict remain in Collision Avoidance mode until the faster aircraft has overtaken the slower aircraft sufficiently so that the horizontal separation satisfies the minimum safe horizontal separation threshold. Finally, when the horizontal separation satisfies the minimum horizontal separation threshold again $r > 600$ feet, the conflict has passed, and both aircraft switch back to Normal Flight mode. In Normal Flight mode, both aircraft use their altitude controllers to return to their nominal altitudes exhibiting an exponential transient response.

4.3.1.4 Pairwise Parallel Flight

The conflict scenario in Figure 4.18 shows the simulation results for two aircraft flying in parallel at the same speed in close proximity. Both aircraft are travelling from left to right, and start the simulation at the same horizontal position with relative altitudes that violate the minimum safe vertical separation. The simulation results illustrate the behaviour of the rules-based collision avoidance system in a persistent pairwise conflict scenario with low closure rates, where the time to collision parameter does not detect the conflict.

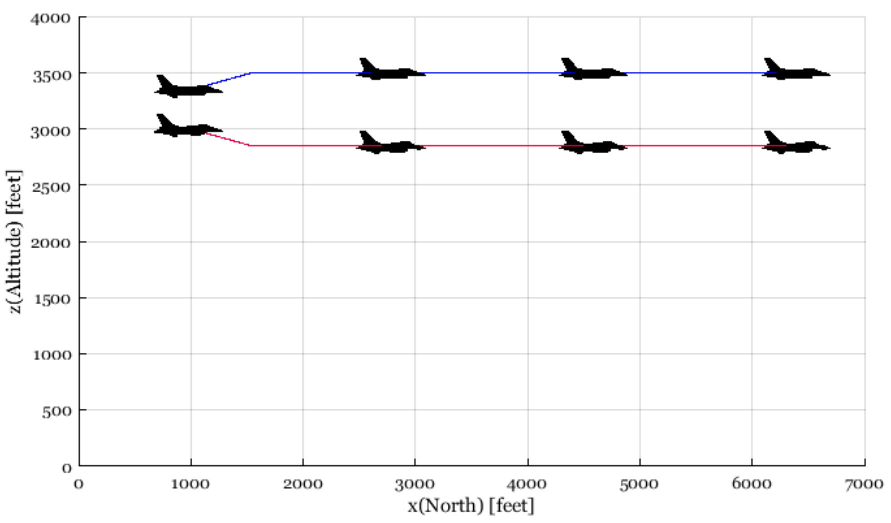


Figure 4.18: Parallel flight pairwise aircraft in conflict

The pairwise two in Figure 4.18 are flying from left to right at a speed of 90.52 feet per second, and both start the simulation at a horizontal position of 0 meters. One aircraft is flying at an altitude of 3000 feet and the other is flying at an altitude of 3300 feet, and both aircraft are flying well above the minimum altitude of 1000 feet for terrain avoidance. The two aircraft are maintaining level flight, the relative closure rate is zero, thus time-to-collision is infinite (which we assign as $\tau = -1$ for simplicity). However, their horizontal separation is 0 feet, and their relative altitude is only 300 feet, which violates the minimum safe horizontal separation of 2126.6 feet and the minimum vertical separation of 650 feet as illustrated in Figure 4.19.

At the start of simulation, a conflict is immediately detected due to the fact that both the minimum horizontal separation threshold DMOD and the minimum vertical separation threshold ZTHR are violated. Then the two aircraft switch to Collision Avoidance mode, and conflict resolution modules are therefore activated to issue collision avoidance commands. Because the vertical separation between the aircraft is initially greater than 300 feet, the aircraft at the higher altitude performs a normal climb, while the aircraft at the lower altitude performs a normal descent. When vertical separation reaches 600 feet, both aircraft level off and maintain their respective altitudes. Since the two aircraft are flying in parallel at the same speed, their horizontal separation remains below the minimum safe horizontal separation threshold, and the conditions for switching back to Normal Flight mode are never reached. Therefore, the aircraft remain in Collision Avoidance mode, and keep on flying in parallel, but at a safe minimum vertical separation. The commands to achieve this are shown in Figure 4.20 and 4.21.

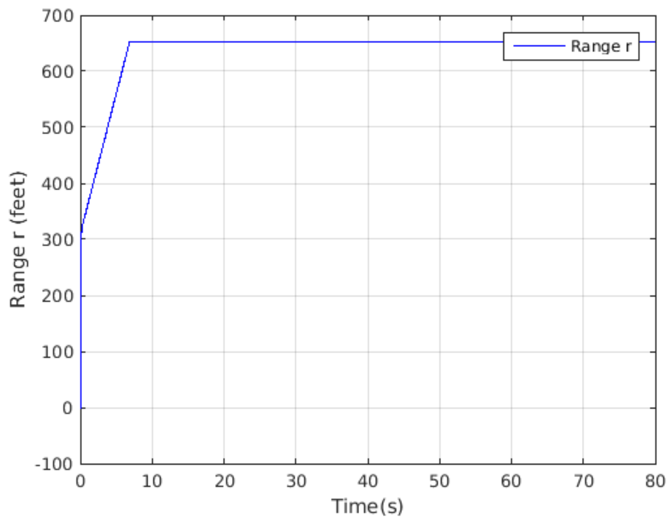


Figure 4.19: Pairwise aircraft separation distance of parallel flight

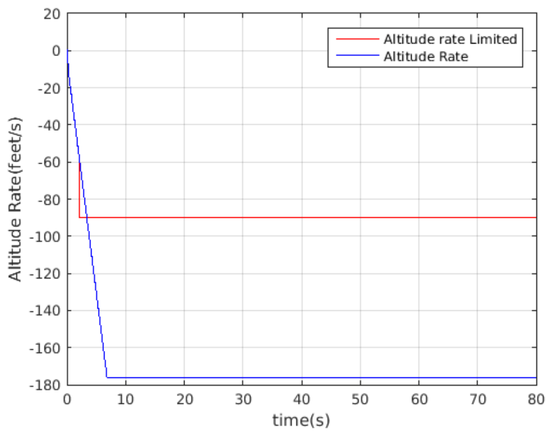


Figure 4.20: Descend rate commands

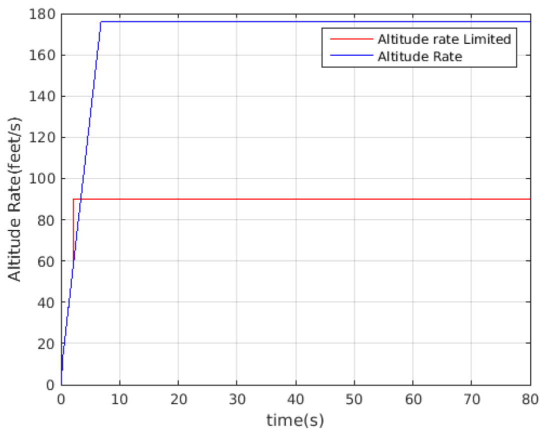


Figure 4.21: Climb rate commands

4.4 Multi-UAV Collision Avoidance

The conflict resolution scenarios presented in the previous section where for a pairs aircraft. However, the rules-based system will need to handle different conflict scenarios involving multiple aircraft. Therefore, two implementations of conflict arbitration algorithms are presented, namely Resolution Action Superposition (RAS) and Closest Intruder First (CIF), inspired by multi-aircraft conflict resolution rules found in literature [12]. The data structure for the conflict of multiple aircraft will be presented. We present an implementation based on the existing TCAS approach and other systems in the literature.

4.4.1 Resolution Action Superposition (RAS)

The following multi-aircraft conflict resolution algorithm is an extension of rules for pairwise conflict resolution which was presented in Section 4.3. The RAS procedure shown in Algorithm 6 extends pairwise conflict resolution to multiple aircraft conflict resolution. In the RAS approach, the host aircraft first determines the pairwise resolution actions for each intruder aircraft with which a conflict is detected. The resultant multi-aircraft resolution action is then determined by superimposing all of the individual pairwise resolution actions, using the following rules:

1. If all of the non-zero individual pairwise resolution actions have the same direction, then the multi-aircraft resolution action equals the individual pairwise resolution action with the largest magnitude.

    - For the set of pairwise resolution actions equals {level off, climb, climb, steep climb, climb}, the resultant multi-aircraft resolution action is steep climb.
    - For the set of pairwise resolution actions equals {level off, steep descend, level off, descend, level off}, the resultant multi-aircraft resolution action is steep descend.
    - For the set of pairwise resolution actions equals {level off, climb, level off, level off, level off}, the resultant multi-aircraft resolution action is climb.

2. In a case of multiple UAVs where individual pairwise resolution actions have different signs, the resultant multi-aircraft resolution action equals the sum of the individual resolution actions. However, the multi-aircraft resolution action "saturates" at a minimum of steep descend and a maximum of steep climb.

    - A multiple of pairwise resolution actions equals {descend, level off, climb}, the resultant multi-aircraft resolution action is level off.
    - For the set of pairwise resolution actions equals {descend, level off, steep climb}, the resultant multi-aircraft resolution action is climb.
    - For the set of pairwise resolution actions equals {descend, steep climb, steep climb, steep climb}, the resultant multi-aircraft resolution action is steep climb.

3. Aircraft that are at their minimum altitude may not descend.

Algorithm 6 RAS Conflict Resolution Scheme

```
Require: UAV conflict vector indicator $C = \{c_1, c_2, \dots c_n\} \in \mathbb{R}^n$ with $c_{ij} = 0$ for host
    uav $i$, multi-UAVs current altitudes $H = \{h_1, h_2, \dots h_n\}$
 1: $R \leftarrow \{\}$                                    ▷ Empty resolution recommendations
 2: conflict_uavs ← ConflictQuery($C$,resolution)
 3: $resolve \leftarrow 2$                                 ▷ resolution status
 4: for uav_id in conflict_uavs do
 5:     if $C[uav\_id] == resolve$ then
 6:         $R \leftarrow R \cup \{AltitudeDirection(h_i, h_{uav\_id})\}$
 7:     end if
 8: end for
 9: $R_{min} \leftarrow min(R)$
10: $R_{max} \leftarrow max(R)$
11: if $sign(R_{min}) == sign(R_{max})$ then
12:     $direction \leftarrow sign(R_{min})$
13:     $r_{min} \leftarrow direction \cdot R_{min}$
14:     $r_{max} \leftarrow direction \cdot R_{max}$
15:     Resolution ← $direction \cdot maximum(r_{min}, r_{max})$
16: else
17:     Resolution ← $R_{min} + R_{max}$
18: end if
19: return Resolution
```

The first input of the algorithm is a conflict indicator matrix $C \in \mathbb{R}^n$ produced after execution of the Conflict Detection algorithm. Then following input is the host UAV conflict indicator index $i = \{1, 2, \dots N\}$. The final argument given to the algorithm is $H \in \mathbb{R}^n$ as the current altitude of all UAVs. Thereafter, in line 2 of the algorithm the UAVs indexes are returned by the search procedure for a list of intruder UAVs. When the query returns "2" then there is iteration of conflict aircraft in line 4 until 8. In line 4-8 the algorithm generates a resolution manoeuvre using the algorithm shown in Algorithm 4. Finally, the algorithm will move to line 9 and 10 to obtain both the minimum and maximum resolutions manoeuvre magnitudes, including the signs and summation, are computed in lines 11 to 16.

4.4.2 Closest Intruder First (CIF)

The CIF approach was proposed in the decomposition of conflict resolution advisories for multiple aircraft. The decomposition of the multi-aircraft conflict into pairwise conflicts is to divide the conflict into pairwise decompositions and resolve them linearly [12], thereby dividing N number of aircraft into $N/2$ pairwise intruders. In the CIF approach, the host aircraft performs only the pairwise conflict resolution action for the closest intruder aircraft with which a conflict is predicted. The closest intruder is determined by first sorting the intruder aircraft in the order of increasing time to collision tau, and then in the order of increasing range distance r, as calculate below, where x_i, x_j are the horizontal positions and y_i, y_j are the vertical positions of host UAV i and intruder j.

$$r = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{4.17}$$

Our assumption during simulation is that the horizontal rates of all UAVs will be equal, therefore there is no need to consider high closure rates and close intruder competing priorities. The implementation of this approach is shown in Algorithm 7. The same conflict resolution direction and magnitude rules for pairwise in Section 4.3 are used for CIF. Additionally, this approach will introduce a distance metric R. The purpose of the metric is to identify the single intruder with the closest distance from the host UAV. Therefore, the conflict resolution advisories objectives for this approach, even with multiple aircraft, is to prioritise resolution advisories of an intruder aircraft that is closest in distance.

Algorithm 7 CIF Conflict Resolution Scheme

```
Require: UAV conflict vector indicator $C = \{c_1, c_2, \dots c_n\} \in \mathbb{R}^n$ with $c_{ij} = 0$ for host
    uav $i$, altitudes of host $h_h$, Distance matrix $R = \{r_1, r_2, \dots r_n\}$ between host and other
    $n - 1$ aircraft
1: $r_{min} \leftarrow min(R)$
2: $resolution \leftarrow 2$
3: magnitude $\leftarrow$ 0                                        ▷ feet/minutes
4: for uav_id from 1 until n do
5:     if $c_{uav\_id} == resolution$ and R[uav_id] equals to $r_{min}$ then
6:         magnitude $\leftarrow Magnitude(h_i, h_{uav\_id})$
7:     end if
8: end for
9: return magnitude
```

The CIF distance metric for the CIF is given in the first line in Algorithm 7. At each time step, the algorithm finds the minimum distance value between each pairwise aircraft that are in conflict. The minimum distance between all pairwise UAVs is used to select a UAV identity from the conflict metric C for the conflict resolution selection process in line 4. At the end of the for-loop in line 8, a single resolution is selected from a UAV with the same minimum distance on line 1. If there are ties, the if statement in line 4 will select the last UAV in the list of all UAVs in conflict.

4.4.3 Illustrative Simulation Results

In this section we present some illustrative simulation results for our rules-based collision avoidance system in different multi-aircraft conflict scenarios. Simulations are performed using both the RAS and the CIF approach, and the results are compared. The effect of equal altitude advisories with simulation noise on the altitude manoeuvre direction will be given in the first subsection for three UAVs. Thereafter, we shall compare the performance of the two algorithms (given in the previous section) on three distinctive tests; Symmetric Altitude, Horizontally Staggered, Head-on Intruders. Finally, the simulation results for a multi-aircraft conflict scenario involving a group of five UAVs is given to illustrate high traffic decision making using both the Resolution Action Superposition and the Closest Intruder First approaches.

4.4.3.1 Three UAVs with Equal Altitude Separation

The conflict scenario of three UAVs in Figure 4.22 and 4.23 shows a multiple aircraft scenario where a single UAV makes an attempt to pass between two other UAVs from an opposite direction. The UAV from the left has two simultaneous intruders at once, at equal altitude difference $\Delta h = 300$ feet. However, the parallel UAVs each has a single intruder. The parallel UAVs approach from the right each calculate a single pairwise resolution action to avoid the single UAV approaching from the left. The single UAV approach from the left calculates two pairwise resolution actions, one for each of the parallel UAVs approaching from the right. As results, the middle UAV maintains its altitude due to balancing the two pairwise conflict resolution action "climb" and "descend" from the altitude difference rules shown in Algorithm 3 in Section 4.2.2.1.

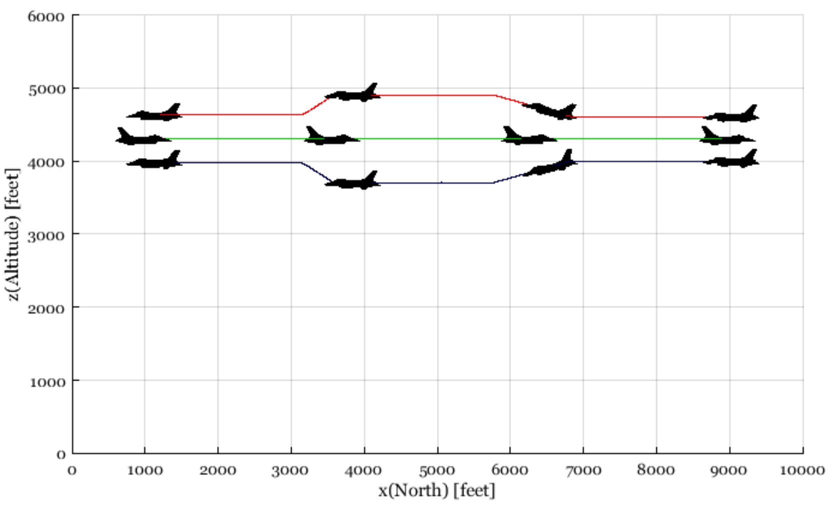


Figure 4.22: RAS conflict resolution for one UAV from the left and two intruders from the right

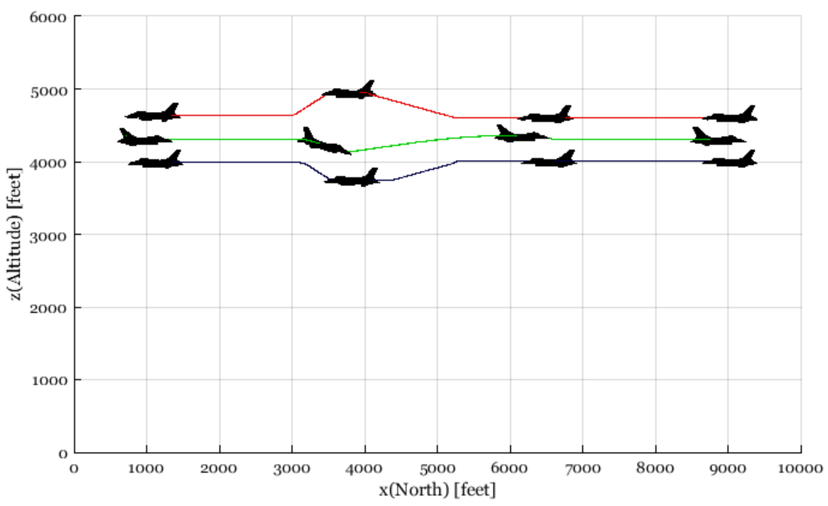


Figure 4.23: CIF conflict resolution for one UAV from the left and two intruders from the right

The time history of the time-to-collision for the parallel UAVs are the same for both the RAS and CIF methods. The time to collision calculated for each of the UAVs is shown in Figure 4.24. This illustrates, when the parallel intruder UAVs simultaneously begin conflict resolution about 45.50 seconds of the simulation.

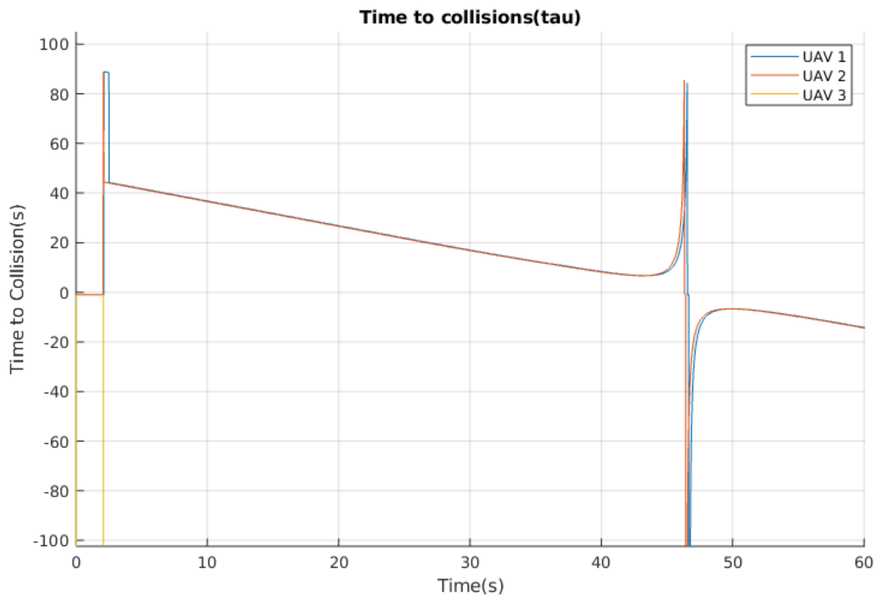


Figure 4.24: Conflict detection with time-to-collision of all three UAVs for both CIF and RAS

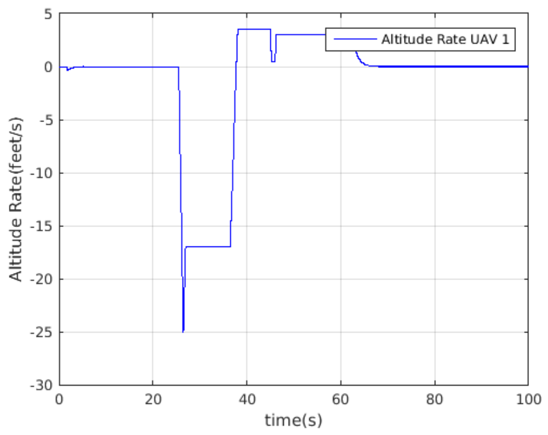


Figure 4.25: RAS middle UAV altitude commands

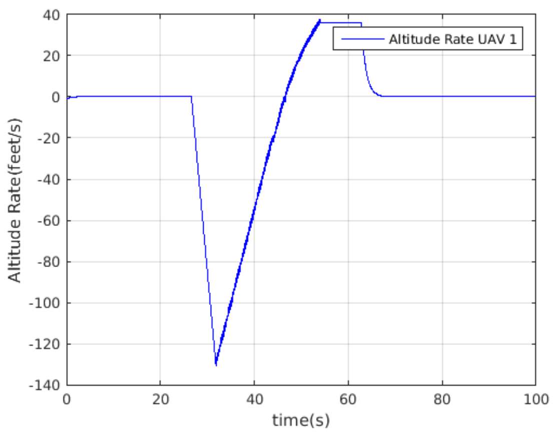


Figure 4.26: CIF middle UAV altitude commands

The altitude commands for the middle UAV illustrates how each of the conflict resolution algorithms attempt to deviate from nominal altitude with simultaneous equal resolutions. The RAS method illustrates short period resolutions which are balanced by other intruder UAVs, therefore the middle UAV maintains altitude most of the conflict period. However, the CIF method illustrates how the priority of UAVs switches based on the closest (in distance metric r) intruder, thus the lower altitude UAV is first because the middle UAV begins descend after 20 seconds. However, after a short period another intruder is detected and the UAV begins to climb, therefore switching between the approaching single intruder.

4.4.3.2 Three Aircraft with one passing through the middle (Near Minimum Altitude)

The following conflict scenario in Figure 4.27 and 4.28 shows the simulation results for a three UAVs conflict scenario where one of the aircraft which is already operating at its minimum altitude for terrain avoidance. Two UAVs from right direction near the minimum altitude of 1000 feet, one at 1100 feet and another at 1750 feet have an approaching intruder from the opposite direction at altitude of 1450 feet. The single approaching in at a equidistant altitude between the two UAVs. These simulation results illustrate how cooperative systems RAS and CIF are able to detect the minimum altitude constraint of collision avoidance.

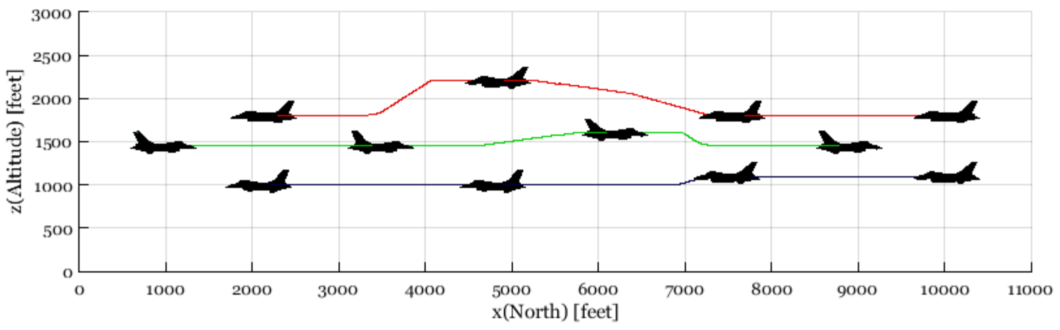


Figure 4.27: Three UAVs with one intruder at minimum altitude conflict resolution with RAS

The RAS approach in Figure 4.27 illustrates how the UAVs share avoidance effort with one of the UAV in a path constrained position (minimum altitude). Since the single UAV from the left cannot descend the two higher UAVs, and the lower UAV from the right also cannot descend further below the 1000 feet, the two higher UAVs cooperate to climb. Furthermore, when the lowest UAV reaches below 1000 feet the UAV cannot descend further and then it maintains its altitude. This behaviour of the UAVs demonstrates the benefits of cooperation and how RAS shares efforts between UAVs. Similar behaviour for the UAVs in CIF is seen in Figure 4.28. A UAV from the left direction detects that the intruder UAV near the minimum altitude cannot further descend, therefore it adjusts its altitude rate. However, with the CIF method the middle UAV and higher altitude UAV can be seen beginning to manoeuvre earlier than in the RAS approach, but with less effort.

4.4.3.3 Three Aircraft at Equal Altitude

A conflict simulation of three UAVs at an equal altitude of 4000 feet is shown in Figure 4.29. One UAV is from an opposite direction (left), whilst two UAVs are from the same direction (right) at different horizontal positions. However, the scenario is set up such that all three UAVs would be involved in a simultaneous conflict. As discussed in Section 4.2.3, some additive noise ($h_h = h_h + w_k$) provides the tie-breaker to determine the directions of the collision avoidance actions for the three UAVs.

The response to the conflict of three UAVs at the same altitude by the RAS method is illustrated in Figure 4.29. A similar conflict response by the CIF method is shown in Figure 4.30. The altitude deviations of each UAV illustrates how each method responds when all UAVs have been disturbed from their nominal altitudes and how the direction of a UAV influences future decisions.

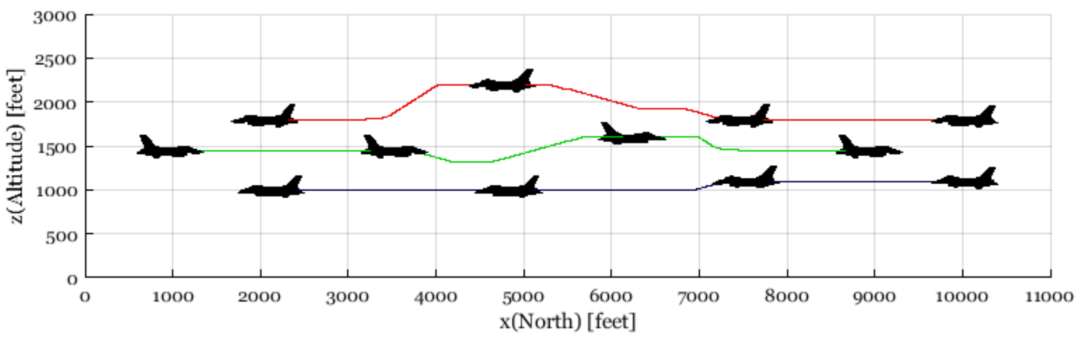


Figure 4.28: Three UAVs with one intruder at minimum altitude conflict resolution with CIF

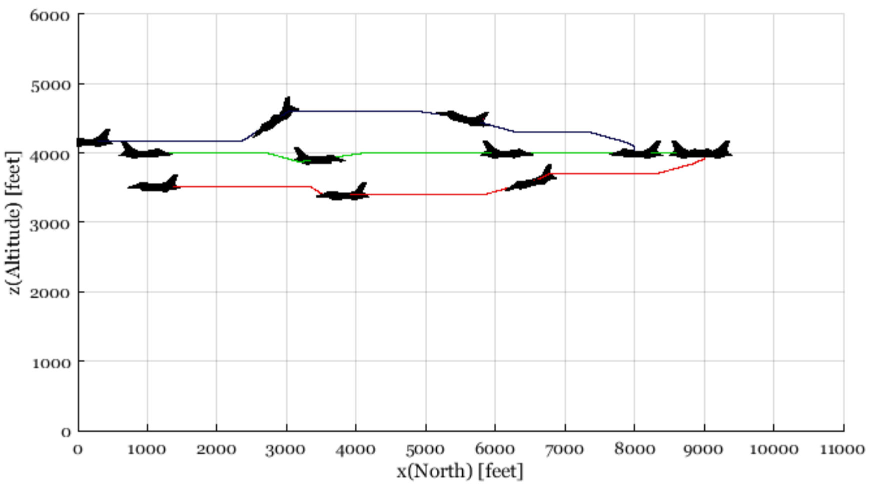


Figure 4.29: RAS multi-aircraft collision avoidance with simulated altitude noise for three UAVs at equal altitude

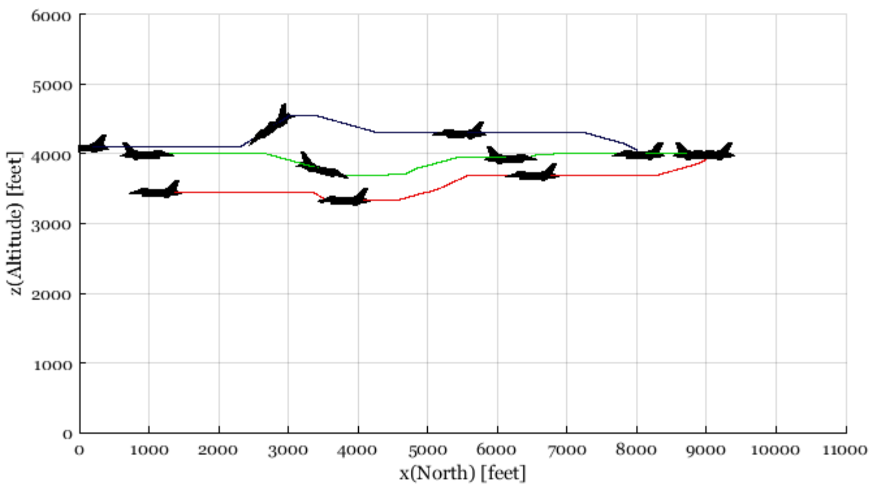


Figure 4.30: CIF multi-aircraft collision avoidance with simulated altitude noise for three UAVs at equal altitude

4.4.3.4 Five Aircraft : Parallel and Staggered Approach

The simulation results for a predicted conflict involving three UAVs in Section 4.4.3.3 has illustrated that the RAS and CIF methods produce different collision avoidance paths because avoidance actions taken at the beginning of the conflict resolution influence the actions that will be taken later. The conflict scenarios in Figure 4.31 and 4.32 are simulations examples which consider five UAVs with conflicts occurring at different times where there are encounters which are similar to the middle altitude and the head-on conflict which were simulated for three UAVs.

The conflict resolution results for the five UAVs scenario with the RAS method are shown in Figure 4.31. In the scenario, the UAVs that have the most altitude change, are situated at the end margin of the conflict, the UAVs at 4500 feet and 6200 feet. This phenomenon occurs because the resolutions are combined for the UAVs.

The conflict resolutions results for the five UAVs scenario with CIF method are shown in Figure 4.32. The CIF solutions have smoother paths compared to the RAS because CIF is a globally influenced method and the CIF is locally influenced method. Due to the fact that conflict resolution advisories of CIF will only respond to the closest intruder by distance priority, the UAVs at the middle altitude does not influence decision making of other UAVs not directly in the conflict.

The altitude deviation means ($\sigma_{TCAS}(h)$ and $\sigma_{CIF}(h)$) between the RAS and CIF methods are showns in Table 4.4. The results shows that UAVs at the edge part of altitude separation will issue higher altitude deviation commands when the manoeuvre space is limited for other intruder UAVs. However, we find that RAS can be utilized for five

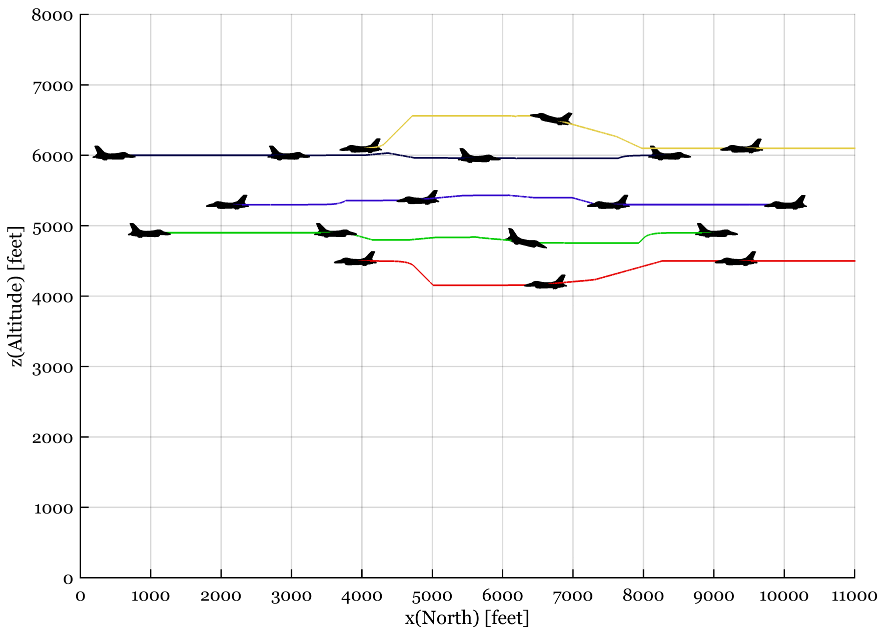


Figure 4.31: Five UAVs simulation of traffic with the Multi-resolution TCAS algorithm

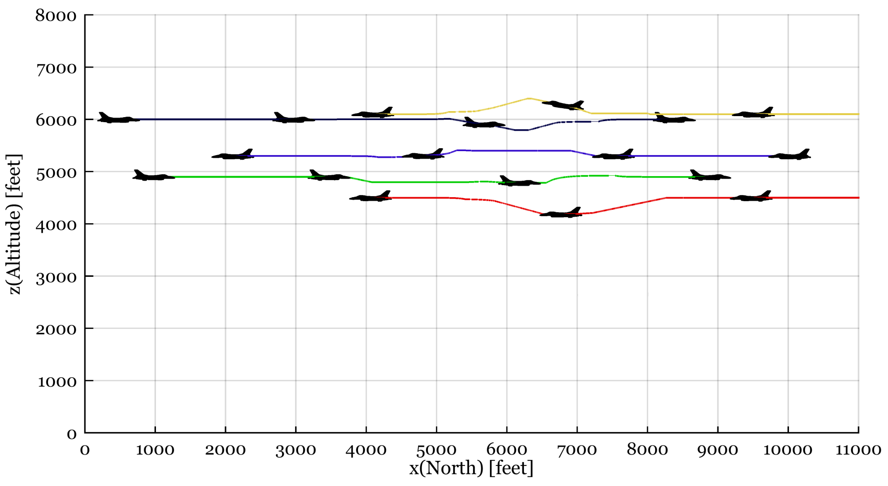


Figure 4.32: Five UAVs simulation of traffic with the CIF algorithm

UAVs to create solutions for other UAVs which are more constrained. Then, UAVs with more manoeuvring space can explore available conflict-free space for the benefit of other UAVs. Moreover, the results in Table 4.4 also illustrate that the CIF method is able to minimize altitude deviations more than the RAS resolution magnitude. Therefore, CIF is a better altitude deviation minimization technique for the problem of the given five UAVs.

UAV	**h(feet)**	$\sigma_{RAS}(h)$**(feet)**	$\sigma_{CIF}(h)$**(feet)**
Green	4900	51.0532	32.180
Black	4000	14.8391	23.9301
Red	4500	104.9475	56.0278
Blue	5300	34.508	24.0209
Yellow	6100	146.5897	36.1208
Total	NA	351.9375	172.2804

Table 4.4: Simulation results for altitude deviations of five UAVs for the RAS and CIF methods

4.5 Summary and Conclusions

The foundation of rules-based collision avoidance began with a study of the foundations of TCAS, an analytical time-to-collision concept of estimating a horizontal closest point approach time used for conflict detection. The analytical time-to-collision concept is a conflict prediction to enable manoeuvres 20 seconds before a collision occurs, the derivation is responsive to the instantaneous horizontal rate of closure and DMOD threshold of a pairwise encounter, as was discussed in detail in Section 4.1. Therefore, the rules-based collision avoidance approach in this chapter is a short-term safe separation responsive algorithm which also returns UAVs to their nominal altitude after conflict resolution. Throughout this chapter the algorithms were simulated for conflict not beyond 20 seconds. However, the benefit of a rules-based system which is modelled after TCAS is that it uses a simple analytical state prediction of time-to-collision which is nominal and computationally inexpensive. Furthermore, TCAS uses vertical altitude manoeuvre conflict resolutions which is a widely understood system used on commercial airliners.

The rules of decision making behind the vertical direction and magnitude of the conflict resolutions advisories were proposed in Section 4.2. Thereafter, simulations of benchmark conflict scenarios involving pairs of UAVs, including head-on conflict, head-on conflict at Minimum altitude, parallel flight and overtaking flight were analysed in Section 4.3. Next, two rules-based algorithms for conflict resolution of multiple UAVs were presented and implemented, namely RAS and pairwise CIF. The conflict resolution with RAS and CIF algorithms were analysed in Section 4.4. The two rules-based approaches a have demonstrated abilities to use cooperation to resolve multi-aircraft conflict scenarios, including three UAVs at equal altitude separation, three UAVs at equal altitude separations near the minimum altitude, and three UAVs at equal altitude. Finally, simulations were performed for a conflict scenario involving five UAVs with a head-on staggered approach pattern. However, we have been able to generated conflict results of the rules-based collision avoidance systems without optimization of control effort. This is due to the fact that the time-to-collision approach to conflict resolutions is a responsive technique. This observation was made from the difference in cost of altitude deviations of multiple UAVs during conflict resolutions.

Chapter 5

Path Planning Based Collision Avoidance

This chapter will discuss a strategic path planning approach for conflict prediction and resolution for an unmanned aerial vehicle (UAV). In our previous rules-based approach to conflict detection and resolution in Chapter 4, each UAV is responsive to time-to-collision, however, the time-to-collision is a short term solution. Thus, the proposed path planning in Section 5.2 will propagate the state dynamics of an intruder UAV for 60 seconds ahead of time for collision prediction. The state propagation and collision prediction methods in Section 5.3 will assist path planning algorithm to predict a collision 60 seconds before it occurs. Thereafter, in Section 5.4 we shall use the iterative sampling algorithm to construct a search-tree data structure which uses the TCAS vertical manoeuvre action space to propagate the states of the UAV. Furthermore, the search-tree divides the conflict prediction of 60 seconds into 10 second time steps. At each time step a single TCAS altitude rate control is held constant which iteratively creates conflict-free trajectories. In addition, path planning algorithm will optimise the conflict resolution path to minimise the deviation of the UAV from its nominal altitude in Section 5.4.1, where a cost function is used to measure the deviation of the UAV for optimisation and then a backward search algorithm finds the optimal path and the optimal sequence of actions in the search tree. Finally, in Section 5.5 illustrative simulations of UAVs in pairwise conflict scenarios are used to demonstrate how the path planning can effectively solve conflicts predicted 60 seconds into the future.

5.1 Overview

The unmanned aircraft path planning design process includes the planning of a flight path for 60 seconds into the future as shown in Figure 5.1. The inner loops control are faster modes of the UAV. The local motion planning as part of the flight path control is responsible for the planning of motion primitives for the UAV in an environment without entering the conflict regions of other UAVs. Therefore, collision avoidance as part of the constraints for the path planning will be discussed in Section 5.4. The UAV state propagation will be performed using a point mass representation for the UAV translational dynamics, and commanded altitude rates from a finite action space. This means that the vehicle does not have a geometric description, and is denoted as the configuration space $\mathcal{C}$ of the planning.

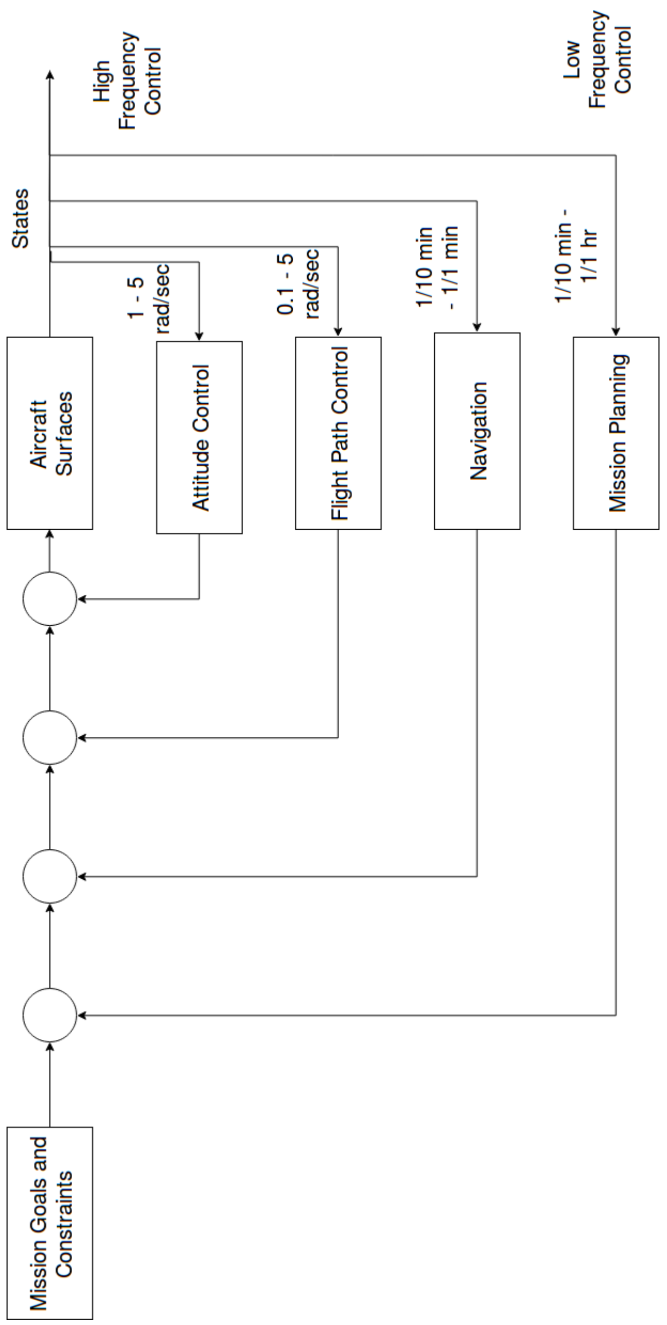


Figure 5.1: Hierarchical decomposition of path planning optimization of an aircraft control discussed in [37]

Historically, only a robot moves in a configuration space and the obstacles would remain static in the coordinate system [34]. However, in the input sampling method which we shall present below, we will have more than one UAV in motion in the next chapter. Therefore, the path planning will have to consider states sampling for an intruder aircraft as the dynamic obstacles which will be presented in the next section. We shall have the following design factors to consider:

- Coordination using communication
- Intruders states propagation
- Conflict prediction
- Configuration sampling
- Manoeuvre formulation
- Path planning optimization
- Path execution

The path planning-based collision avoidance with state propagation is shown in Figure 5.2. The cooperation framework is an abstraction of a Communication Network Hub which distributes the states of all UAVs in the network. The Path Simulator module propagates the states of each UAV at each time step from path plan inputs. The Token Allocator generates a token list for the execution of path plans, implementing a cooperative path planning approach which will be discussed in Section 5.4 and will be further elaborated in Chapter 6 for multiple UAVs.

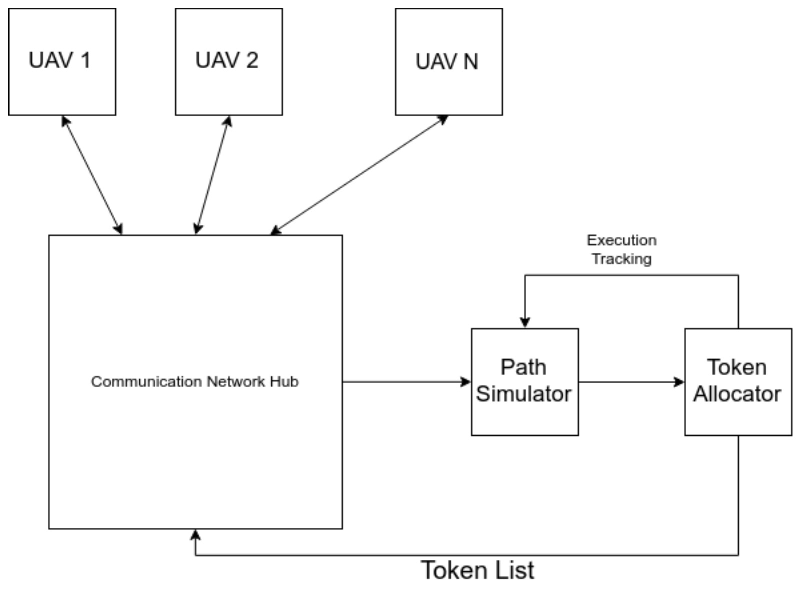


Figure 5.2: Path planning framework for cooperative collision avoidance

Path planning based approach to conflict resolution will implement a search-tree data structure for each UAV as part of the algorithm which will explore the state space. The search-tree algorithm is designed to generate conflict resolution manoeuvres that optimise a path cost function. Therefore, the dynamic states of each UAV will be sampled during the search-tree expansion by an iterative path planning algorithm. In literature there are two vehicle state sampling techniques known as the input-sampling and space-sampling [5]. The input-based sampling is the suitable approach for problems with a finite number of known actions in the vehicle input set. Additionally, in this thesis we will sample discrete inputs from the TCAS vertical input set for conflict resolution advisories. Consequently, the input-based sampling approach will be used to sample the states of the UAVs in the form of climbing/descend rates for the conflict prediction and path construction as illustrated in Figure 3.8.

5.2 Aircraft State Propagation

The state propagation component uses the discrete-time state equations and the expected climb rate actions of the host UAV and the intruder UAVs to propagate their states forward in time from their initial states. While a UAV is operating in Normal Flight mode, its climb rate actions are assumed to be supplied by the altitude controller. While a UAV is operating in Collision Avoidance mode, its climb rate actions are assumed to be supplied by the collision avoidance path planner.

The state $\mathbf{x}_i(k)$ of a UAV is defined as:

$$\mathbf{x}_i(k) = \begin{bmatrix} x_i(k) & h_i(k) \end{bmatrix}^T \tag{5.1}$$

where $x_i(k)$ is its horizontal position, $h_i(k)$ is its altitude, and k is the discrete-time sample index.

The state $\mathbf{x}_i(k)$ of a UAV is propagated forward in time using the following discrete-time state equations:

$$\begin{aligned} x_i(k+1) &= x_i(k) + \dot{x}_i(k)\Delta t \\ h_i(k+1) &= h_i(k) + \dot{h}_i(k)\Delta t \end{aligned} \tag{5.2}$$

The horizontal position $x_i(k)$ of a UAV is propagated forward in time from its initial horizontal position $x_i(0)$, assuming a constant forward velocity $\dot{x}$. The altitude $h_i(k)$ of a UAV is propagated forward in time from its initial altitude $h_i(0)$, using the expected climb rate actions $u_i(k)$ for $k = 0, 1, ...N-1$. While a UAV is operating in Normal Flight mode, its climb rate actions are assumed to be calculated by the following control law for the altitude controller:

$$\dot{h}_i(k) = K \cdot (h_{i,\text{nom}} - h_i(k)) \tag{5.3}$$

While a UAV is operating in Collision Avoidance mode, its climb rate actions are assumed to follow a collision avoidance action plan that was generated by the collision avoidance path planner. The planned collision avoidance actions for all of the UAVs are published in a table P, which has the following structure:

$$P = \begin{bmatrix} u_1(k_1 = 0) & u_1(k_1 = 1) & \dots & u_1(k_1 = N-1) \\ u_2(k_2 = 0) & u_2(k_2 = 1) & \dots & u_2(k_2 = N-1) \\ \vdots & \vdots & \ddots & \vdots \\ u_i(k_i = 0) & u_i(k_i = 1) & \dots & u_i(k_i = N-1) \\ \vdots & \vdots & \ddots & \vdots \\ u_m(k_m = 0) & u_m(k_m = 1) & \dots & u_m(k_m = N-1) \end{bmatrix} \tag{5.4}$$

Each row of the collision avoidance table P contains the planned collision avoidance actions $u_i(k_i)$ for each of the UAVs for a fixed number of time steps N into the future. For example, the i'th row contains the planned collision avoidance actions for the i'th UAV for time steps $k_i = 1$ to N. The element $u_i(k_i)$ is the climb rate action to be executed by the i'th UAV at the k_i'th time step of its collision avoidance plan. The climb rate actions are constrained to the following finite set of climb rate actions:

$$u_i(k_i) = \dot{h}_i(k) \in \{\dot{h}_{\text{steep descend}}, \dot{h}_{\text{descend}}, 0, \dot{h}_{\text{climb}}, \dot{h}_{\text{steep climb}}\} \tag{5.5}$$

Note that each row of the collision avoidance table uses its own relative time step index k_i. The relative time step index k_i tracks the specific UAV's progress through its collision avoidance plan relative to the absolute time step when it started executing its plan. At any given absolute time instant, different UAVs may have progressed to different relative time steps in the execution of their own collision avoidance plans. Some UAVs may only be starting their collision avoidance plan, while others may be halfway through, while others may be finishing their plan. While the UAV is in Normal Flight mode, the relative time step index is set to $k_i = -1$. When the UAV starts a collision avoidance manoeuvre, its relative time step index is set to $k_i = 0$. While the UAV is in Collision Avoidance mode, the relative time step index k_i is incremented every discrete-time sampling instant. When the relative time step index reaches $k_i = N$, the collision avoidance manoeuvre is completed, the UAV returns to Normal Flight mode, and the time step index is set to $k_i = -1$.

5.3 Collision Prediction

The collision prediction component propagates the host and intruder aircraft states forward in time (using the state propagation component) and checks whether the future states of the host and intruder aircraft result in a violation of the host aircraft's protected zone, or whether future states of the host aircraft violate the minimum terrain avoidance altitude.

5.3.1 UAV State Node

The state node of a UAV in the construction of a search tree data structure is given by the Algorithm 8. The node data structure contains the simulated position $(x_k, h) \in \mathbb{R}^2$ as the range and altitude. The tree is populated through iterative sampling from a single parent node. Each node has a single parent node from previous step $k-1$ as an index p in the tree data structure, used during the searching stage. A discrete time to conflict tracking variable τ is subtracted from a parent to a new value $\tau - T_i$. In our approach, we use the value of τ as a notation indicating the depth level of the node in the path planning tree T. During state propagation for a UAV, the Algorithm 8 is called several times. This step will create each of the five nodes in Figure 5.4. Furthermore, the procedure in Algorithm 9 takes as input the parent node position, and the input-space set U. At this point, all nodes at different altitudes are created and stored in a contained data structure which is returned.

Algorithm 8 UAV State Node $node(x_k, u_{k-1}, p, \tau)$

```
Require: Position $xk_k \in \mathbb{R}^3$, Previous input $u_{k-1}$, Parent Node index $p > 0$, time step t
1: $Node \leftarrow \{\}$
2: $Node.x_k \leftarrow x_k$
3: $Node.u_k \leftarrow u_{k-1}$
4: $Node.parent \leftarrow p$
5: $Node.tau \leftarrow \tau - t$
6: $Node.TerminalCost \leftarrow |x_k(3) - Tree(p).x_k(3)|$
7: return $Node$
```

Algorithm 9 Create Nodes $CreateNodes(X, U, p, \tau, \dot{x}, t)$

```
Require: UAV position $X \in \mathbb{R}^3$, input set $U = \{u_1, u_2, \dots u_n\}$, Parent tree node index
    $p$, time step t
1: $NodeSet \leftarrow \{\}$
2: $X.x \leftarrow X.x + \dot{x} \cdot t$
3: for i from 1 until n do
4:     $NodeSet \leftarrow NodeSet \cup node(X, U[i], parent, \tau)$
5: end for
6: return $NodeSet$
```

The tree population algorithm uses several design conventions due to the complex nature of multiple aircraft state propagation. The input for the algorithm is the current time position, and a square matrix X containing the positions of all UAVs current within communication range. Therefore, as part of the input, a path plan matrix P contains a sequence of altitude control inputs for the states propagation of other aircraft.

5.3.2 Collision Prediction Steps

The conflict prediction simulates the state nodes of intruder UAVs for conflict detection at each step. The state propagation is performed by executing the existing path plan for conflict resolution or by simulating the altitude controller in nominal flight mode for a UAV. Additionally, the inputs for state propagation of each UAV is shared over a communication. Therefore, simulation of each UAV that is cooperative is made possible. The data structure which each UAV will use for the path simulation is shown in Algorithm 10. The algorithm is simplified to take the state position X, path plan P, path execution variable *track* and x-axis velocity constant v as input.

The state propagation of the path planning is tracked using the variable *track* which should be greater than 0, but less than the path-input set P length. This mechanism is also used to check whether the path planning search-tree module has been able to sample a full length path plan. This is due to the iterative nature of the sampling structure as discussed in Section 5.4.2. Therefore, the state simulation algorithm from line 1 until line 4 will simulate a path plan to reconstruct the intruder aircraft trajectory. If the path execution tracking variable *track* indicates that path planning needs execution in line 1, no plan is executed. In line 3 the algorithm the uses the plan tracking mechanism to execute an altitude input to manoeuvre a UAV to a new altitude.

Algorithm 10 Pairwise UAVs Conflict Simulation $SimpleSimulate(X, P, track, v)$

```
Require: UAV position $X \in \mathbb{R}^2$, Path Plan $P = \{p_1, p_2, \ldots p_m\}$, path tracking value
    $track$, range velocity v
1: if track < m and track = 0 then
2:     X[1] ← X[1] + v · t
3:     X[2] ← X[2] + P[track] · t
4: end if
5: return X
```

The path following module of a UAV will monitor the execution of path planning for the multiple UAVs at every time step for continuous conflict detection with dynamic obstacles. The length of each path plan and conflict prediction are shown in Figure 5.3 (to be discussed in Chapter 6 in more detail). This is known as a "token" which enables a UAV to simulate trajectories of an intruder UAV from the path input sequence. In addition, a UAV with time reference t from the time a conflict has started, will require a mapping from time-space T for the path-input index during path execution. The function for mapping the input-set for state propagation from time-space T is a hashing function shown in Equation 5.6.

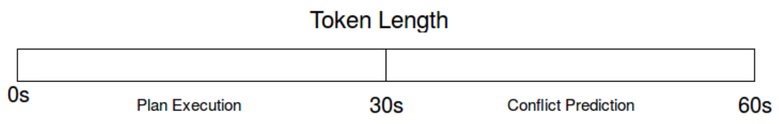


Figure 5.3: Path execution tracking with token allocation

The horizon T is divided into two equal regions, that is $\frac{T}{2}$ during positive values while approaching and negative values after the aircraft have passed each other. Positive time values indicate that the UAVs have passed each other, and negative time values indicate that UAVs are still approaching each other. We assume a constant forward velocity to propagate the horizontal position. Simulation of the tree nodes in two dimensions into the future use a linear equation at equal intervals steps $\Delta t = 10s$. This approach enables conflict detection with dynamic obstacles. The altitude-axis sampling is shown in Algorithm 9 and is discussed in depth in Section 5.3.1. At each time step the state-space sampling each of the nodes in the tree is checked for conflict detection.

$$Hash(t, \tau, \mathrm{T}) = \begin{cases} \lceil \frac{\tau}{t} \rceil + \lfloor \frac{\mathrm{T}}{2} \rfloor & \text{if } \tau < 0 \\ \lfloor \frac{\tau}{t} \rfloor + \lceil \frac{\mathrm{T}}{2} \rceil & \text{if } \tau > 0 \\ \lceil \frac{\mathrm{T}}{2} \rceil + 1 & \text{if } \tau = 0 \end{cases} \tag{5.6}$$

The conflict prediction for a pair of UAVs with a communication link is given in Algorithm 11 to enable cooperative path planning. This conflict prediction algorithm simulates the states of a pair of UAVs at once iteratively from a multiple of conflicts. Therefore, a multi-aircraft conflict involving N number of UAVs will be simulated into N(N-1)/2 number of conflicts. As input, the algorithm takes the state position X of a pair of UAVs, path plans P and path execution metric T. Therefore, the algorithm creates a look-ahead time for the state propagation of the UAVs in line 8. As was previously illustrated in Figure 5.3, the look-ahead time is half the path planning horizon. For this reason, a UAV can make path plans even after an intruder has passed.

Algorithm 11 Conflict Prediction Algorithm

```
Require: Position $X \in \mathbb{R}^{2\times 2}$ for a pair UAVs, Path plans $Plan = \{P_1, P_2\}$, host path
    track $T = \{t_1, t_2\}$
 1: H ← $[h_h, h_i]$                                          ▷ Two uavs nominal altitudes
 2: $p_h \leftarrow$ Plan[1]                                          ▷ host uav plans
 3: $p_i \leftarrow$ Plan[2]                                       ▷ intruder uav plans
 4: $X_h \leftarrow$ X[1]                                          ▷ host uav states
 5: $X_i \leftarrow$ X[2]                                      ▷ intruder uav states
 6: plan_length = length(Plan[1])
 7: conflict ← 0                                          ▷ Default conflict state
 8: for index from 1 until ceil(l/2) do
 9:     if track < len and track > 0 then
10:         $X_h[2] \leftarrow X_h[2]$ + Plan[track] $\cdot t_h$
11:         $t_h \leftarrow t_h + 1$
12:     else
13:         $X_h[2] \leftarrow X_h[2] + AltitudeController(X_h[2], H[1]) \cdot t_h$
14:     end if
15:     $X_h[1] \leftarrow X_h[1] + V_h \cdot t_h$
16:     $X_i = SimpleSimulate(X_i, P_i, t_i, H[2])$
17:     if $ConflictDetect(x_h, x_i) == 1$ then
18:         conflict ← 1
19:     end if
20: end for
21: return conflict
```

5.4 Path Planning

The path planning component grows a search tree of admissible conflict resolution paths, and searches the tree to find the conflict-free path with the lowest cost. Each node of the search tree represents the state of the aircraft at a discrete time instant. The search tree is created with a single root node that is initialised with the state of the host aircraft when the path planning starts. Nodes are added to the tree by iterating through the finite set of actions at discrete time instants, propagating the aircraft state, checking for predicted collisions, and only adding conflict-free nodes to the tree.

The optimization of the path for each UAV is local because there are multiple paths that the iterative algorithm has constructed with the equations of motions given in Section 5.4.1. The iterative sampling procedure for creating a search tree is presented in Section 5.4.2. Therefore, the path planner of a UAV will optimize a hierarchical cost function J that minimizes the altitude deviation from the nominal altitude as the first priority, and then minimises the avoidance effort for the UAV as a second priority. The search algorithm for the optimal path from the root of the node to an optimal leaf node will be discussed in Section 5.4.3. Thereafter, the complexity of the possible paths that a

UAV can take are discussed based on the optimization of the cost function of manoeuvres.

5.4.1 Problem Formulation

The collision avoidance path planning problem for the host UAV is formulated as an optimal control problem with the objective of finding the optimal state trajectory. The optimal input signal minimizes a cost function, while avoiding collisions between the host UAV and intruder UAVs. Furthermore, the planned path must remain above the minimum altitude for terrain avoidance and must also obey the differential constraints of the host UAV. In the following subsection, we discuss the formulation of the conflict resolution task as an optimal control problem.

5.4.1.1 Optimal Control Problem

Given the initial state $x_i(k = 0)$ and nominal altitude $h_{i,\text{nom}}$ of the host UAV, and the propagated future states of all intruder aircraft $x_j(k)$, $k = 0, 1, ..., N - 1$ and $j = 1, 2, ...n$ with $j \neq i$, the path planning algorithm must find the optimal collision-free state trajectory $\mathbf{x}_i^*(k)$ and the associated optimal climb rate actions $\mathbf{u}^*(k)$. The optimal climb rate actions are for a host UAV for discrete time steps $k = 0, 1, ...N - 1$ that will avoid all collisions with intruder UAVs. Additionally, the host UAV will make path plans which must remain above the minimum terrain avoidance altitude, furthermore the controls will minimise the difference between the final altitude and the nominal altitude. Finally, the conflict resolution actions must minimise the action cost, while obeying the differential constraints of the host UAV.

5.4.1.2 State Vector

The state vector $\mathbf{x}$ is defined as:

$$\mathbf{x}_i(k) = \begin{bmatrix} x_i(k) & h_i(k) \end{bmatrix}^T \tag{5.7}$$

where $\mathbf{x}_i$ is the two-dimensional position of the host UAV.

5.4.1.3 State Propagation

The state of the host UAV is propagated using the following discrete-time state equations:

$$x_i(k+1) = x_i(k) + \dot{x}_i(k)\Delta t \tag{5.8}$$

$$h_i(k+1) = h_i(k) + \dot{h}_i(k)\Delta t \tag{5.9}$$

where x_i is the horizontal position, h_i is the altitude, $\dot{x}_i$ is the horizontal forward velocity, and $\dot{h}_i$ is the climb rate of the host UAV. k is the discrete-time sample index and Δt is the discrete-time sampling period. The sampling technique will attempt to generate aircraft nodes whilst holding the horizontal rate constant. A node is successfully sampled if the state propagated for one sampling period, using the discrete-time equations of motion, is free of conflict. The motions created by the sampling method for a single time step which describe the equation of motion are shown in Figure 5.4.

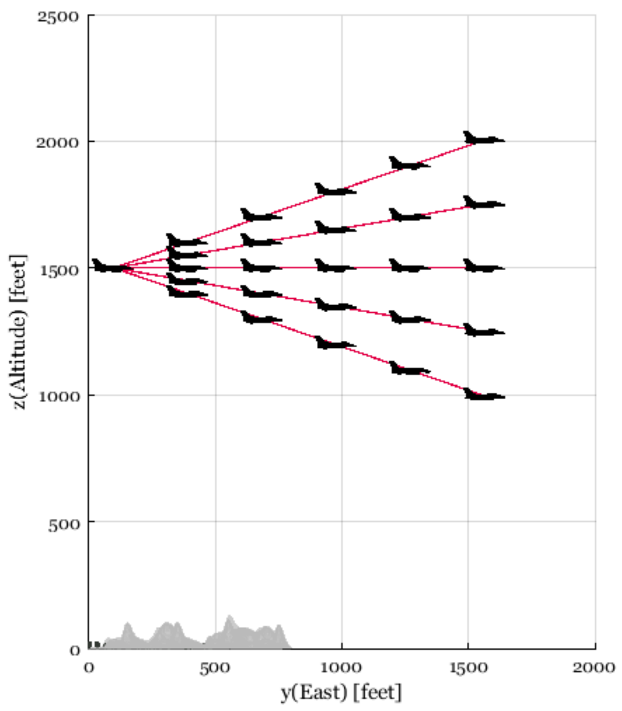


Figure 5.4: State propagation with constant horizontal rate and five altitude inputs

5.4.1.4 Control Input

The control input $\mathbf{u}$ is defined as :

$$\mathbf{u}_i(k) = \begin{bmatrix} \dot{x}_i(k) & \dot{h}_i(k) \end{bmatrix}^T \tag{5.10}$$

where $\mathbf{u}_1$ to $\mathbf{u}_n$ are the velocity actions of the n cooperative aircraft.

5.4.1.5 State Constraints

The state constraints for the path planning are formulated in such a way that the propagated states of the host UAV and the intruder UAVs may not result in a violation of the host aircraft's protected zone, and the altitude of the host UAV may not be below the minimum altitude for terrain avoidance, at any discrete time step along the planned path of the host UAV. The state constraint on the host UAV's protected zone is expressed mathematically as:

$$\begin{aligned} \mathbf{x}_j(k) \notin \mathcal{C}_i(\mathbf{x}_i(k) \quad \forall k &= 0, 1, ..., N-1 \\ \forall j &= 1, 2, ..., m \text{ (i} \neq \text{j)} \end{aligned} \tag{5.11}$$

where $\mathbf{x}_j$ is the position of the j'th intruder UAV, $\mathbf{x}_i$ is the position of the host UAV, and $\mathcal{C}_i(\mathbf{x}_i(t))$ is the protected zone of the host UAV. The host UAV's protected zone is defined as:

$$\mathcal{C}_{\mathrm{i}}(\mathrm{x},\mathrm{h}) = \{\mathrm{x},\mathrm{h} : \|\mathrm{x}-\mathrm{x_i}\| \leq \Delta\mathrm{x} \cap \|\mathrm{h}-\mathrm{h_i}\| \leq \Delta\mathrm{h}\} \tag{5.12}$$

The state constraint on the minimum terrain avoidance altitude is expressed mathematically as:

$$h_i(k) \geq h_{\min} \quad \forall k \quad = 0, 1, ..., N-1 \tag{5.13}$$

where $h_{\min}$ is the minimum terrain avoidance altitude.

5.4.1.6 Action Space

The action space for the host UAV is constrained to the following finite set of climb rate actions:

$$\dot{h}_i(k) \in \{\dot{h}_{\text{steep descend}}, \dot{h}_{\text{descend}}, 0, \dot{h}_{\text{climb}}, \dot{h}_{\text{steep climb}}\} \tag{5.14}$$

The horizontal forward velocity $\dot{x}_i$ of the host UAV is assumed to be constant and constrained to the following range:

$$v_{i,\min} \leq \dot{x}_{i,\text{hor}}(k) \leq v_{i,\max} \tag{5.15}$$

where $v_{i,\min}$ and $v_{i,\max}$ are the minimum and maximum horizontal forward velocities. Under these constraints a UAV navigates a search-tree from a root parent node at an altitude level h. The UAV altitude controller shall apply altitude rate input u_k to reach new altitude levels. An example of a level flight manoeuvre for a UAV is shown in Figure 5.5. Additionally, an example of climb manoeuvres to two higher altitude nodes from a parent node are shown in Figure 5.6. Similarly, descend manoeuvres at two lower altitude from parent node are shown in Figure 5.7.

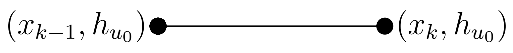


Figure 5.5: Level path transition

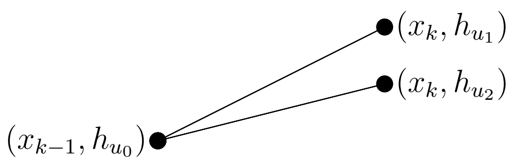


Figure 5.6: Climbing tree transition

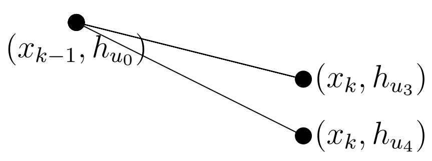


Figure 5.7: Descending tree transition

5.4.1.7 Terminal State Constraints

The path planning algorithm must provide a collision-free path plan for a minimum number of discrete time steps into the future. This requirement is translated into the following terminal state constraint:

$$k_{\text{final}} \geq N \tag{5.16}$$

where the k_{final} is the discrete time step index of a final state and N is the required minimum number of time steps for the path plan. Note that no further constraints are placed on the terminal state at the final time step, and that the host UAV is not required to return to its nominal altitude by the final time step. This is to provide for the possibility that the space at the nominal altitude may be occupied by an intruder UAV at the final planning time step.

5.4.1.8 Cost Function

A hierarchical multi-objective cost function is formulated to represent a primary objective to minimise the deviation of the aircraft from its nominal altitude at the final time step, whilst a secondary objective is formulated to minimise the avoidance effort.

Terminal Cost: The primary cost function is chosen as the host UAV's altitude deviation at the final time step of its planned path, and is formulated as a terminal cost. The terminal cost H_i is defined as the absolute difference between the host UAV's nominal altitude and its actual altitude at the final step of the collision avoidance plan, given as:

$$H_i = ||h_{i,\text{nom}} - h_i(k = N)|| \tag{5.17}$$

where h_{nom} is the host UAV's nominal altitude, and $h_i(k = N)$ is the host UAV's altitude at the final time step of the path plan.

Transition Cost: The secondary cost function is chosen as the host UAV's action cost to execute the path plan, and is formulated as a transition cost. The transition cost G_i is the sum of the action costs from the host UAV's initial state to its final state, give by:

$$G_i = \sum_{k=1}^{N-1} g(u(k)) \tag{5.18}$$

where G_i is the total transition cost, and $g(u_i(k))$ is the action cost associated with climb or descend rate action $u_i(k)$. The action costs are defined as follows:

$$g(u(k)) = \begin{cases} 2 & \text{if } |u(k)| = 1500 \text{ ft/min} \\ 3 & \text{if } |u(k)| = 3000 \text{ ft/min} \\ 0 & \text{otherwise} \end{cases} \tag{5.19}$$

Complete Cost Function: The hierarchical cost function is minimised as follows: First, the primary terminal altitude deviation cost function H_i is minimised without considering the secondary action cost function. Next, if more than one solution minimises the primary altitude deviation cost function, then the solution that also minimises the secondary action cost function G_i is selected from among the solutions that already minimise the primary cost function. The hierarchical cost function allows the primary objective to be prioritised without making any compromises to the secondary objective. (In contrast, a single multi-objective cost function would typically lead to trade-offs between the primary and secondary objectives to minimise the total cost based on some weighting scheme.)

5.4.2 Creating the Search Tree

The search tree shown in Algorithm 12 is created from a single root node is initialised with the state of the host UAV when the path planning starts. Nodes are added to the tree by iterating through the finite set of actions at discrete time instants, propagating the aircraft state, checking for predicted collisions, and only adding conflict-free nodes to the tree. Each node in the search tree is a data structure which contains the time step index k and the host UAV state $\mathbf{x}_i(k)$ at the node, the climb rate action $u_i(k-1)$ that was applied at the previous time step to transition from the parent node to this node, the cumulative action cost G_i and the terminal cost H_i (if applicable) at this node, and a pointer to this node's parent node.

The search tree is seeded with a single root node at time step index $k = 0$. The root node is initialised with the state of the host UAV when the path planning starts, and with the terminal cost and transition costs set to zero. Since the root node does not have a parent node, its climb rate action at the previous time step is undefined, and the pointer to its parent node is set to null. The children of the root node are then created by starting at the root node, iterating through the finite set of climb rate actions. The iteration is the propagation state of the host UAV forward one time step from each possible action, this step will include state propagation of all the intruder UAVs forward one time step. Furthermore, the state propagation for each UAV will check for UAV-to-UAV collisions and minimum terrain avoidance altitude, and will only create nodes for propagated states that do not result in collisions. Therefore, all admissible, conflict-free child nodes are added to the search tree, and their time step index, state, previous climb rate action, cumulative action cost, terminal cost, and parent node pointer are populated. The process is then repeated, with every child node becoming a parent node that spawns its own child nodes.

The procedure for population of a search-tree is given in Algorithm 12. The initialization of the tree is a few constants and collection data structures between line 1 until 6. Therefore, the depth of the tree N is the ratio of the total horizon time T to the time steps T_i in line 3 of the algorithm.

$$N = \frac{T}{T_i} \tag{5.20}$$

Algorithm 12 Iterative Altitude Sampling

```
Require: $X_{intruders} \in \mathbb{R}^2 \times \mathbb{R}^m$, $X_i \in \mathbb{R}^3$, Path plans $P = \{P_1, P_2, \dots P_n\}$, Unique UAV
    identity $i \in N$, Simulation Period $T = 60s$, path segment length $T_i = 10s$
 1: $\dot{X} = C_x$                                          ▷ Horizontal velocities of all UAVs
 2: $depth \leftarrow \frac{T}{T_i}$
 3: $root \leftarrow Node(X_i, 0, 1, T_i)$                   ▷ Create root of the tree
 4: $Tree \leftarrow \{root\}$
 5: $X_{intruders} \leftarrow PlanSimulate(X_{intruders}, P, Trackers, T_i)$
 6: $U = [-\frac{3000}{60}; -\frac{1500}{60}; \frac{0}{60}; \frac{1500}{60}; \frac{3000}{60}]$    ▷ feet/min = feet/60 sec
 7: children = root
 8: $parent = 1$                                             ▷ Parent Index
 9: for length from 1 until $depth$ do
10:     $X_{intruder} = X_{intruders}[length]$
11:     counter $\leftarrow ||children||$
12:     for j from 1 until counter do
13:         $child \leftarrow children[j]$                   ▷ Propagated state
14:         $X_k = child.X_k$
15:         $u_k \leftarrow child.u$                         ▷ Input to apply to a parent node
16:         $X_k \leftarrow Node(X_k,\ u_k,\ parent,\ T_i)$  ▷ New node
17:         if $ConflitDetectionTree(X_k,\ X_{intruder}) == 0$ then
18:             $Tree \leftarrow Tree \cup \{X_k\}$
19:             $parent \leftarrow parent + 1$
20:             $NewChildren \leftarrow CreateNodes(X_k,\ U,\ parent)$
21:             $children \leftarrow children \cup NewChildren$
22:         end if
23:     end for
24: end for
25: return $Tree$
```

The search-tree algorithm creates child nodes for the next depth by going through the current tree depth nodes from line 10 until 21. If all the nodes at a particular depth are in conflict as checked in line 16 there will be no nodes children for the next time step to propagate. The nodes of the tree at each level refers to the conflict detection algorithm in line 16, this is to ensure that the nodes are not in conflict with other UAVs states in X. The result of the sampling iteration process for the search tree at each altitude level is given by the density function as shown in Figure 5.8.

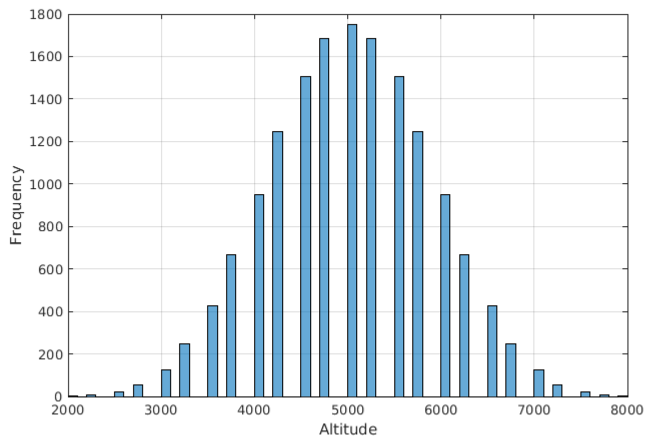


Figure 5.8: Search tree Nodes sampling distribution density of a UAV from the nominal altitude of a 6000 feet

The density function of the nodes population at different altitude levels in a tree data structure T of the path planning algorithm is depicted in Figure 5.8. The planning algorithm described above populates the data structures for path planning from the root of the tree at (x_{k-1}, h_{u_0}) for the number of conflict-free nodes based on the depth of the tree, as calculated in Equation 5.20. Therefore, the data structure is a tree with a single root node. The root of the tree is the position where conflict is detected. An example of a completed search tree with no conflict nodes is shown in Figure 5.9. Another example of a completed search tree with some conflict nodes is shown in Figure 5.10.

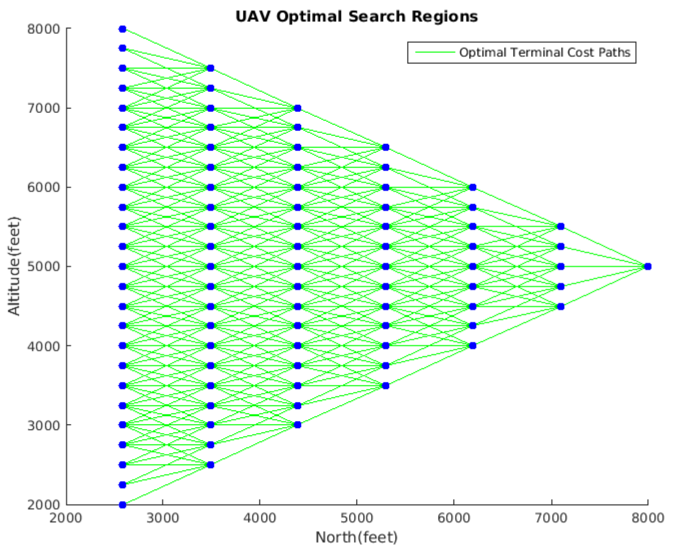


Figure 5.9: Example search tree without conflict nodes

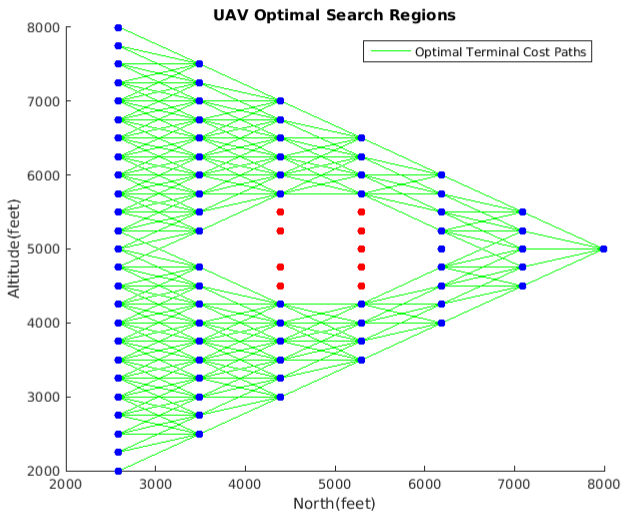


Figure 5.10: Example search tree with some conflict nodes

5.4.2.1 Maximum Number of Nodes

The maximum number of nodes in the search tree occurs when none of the propagated states results in conflict and all child nodes are added to the tree. Therefore, number nodes in a search tree which does not have any conflict nodes is given as :

$$\begin{aligned}\text{Max Number of Nodes} &= a^0 + a^1 + a^2 + \cdots + a^N \\ &= \sum_{k=0}^{N} a^k \qquad (5.21)\end{aligned}$$

where a is the number of discrete actions in the action space, and N is the number of time steps in a collision avoidance path plan. A search tree with five possible actions (a = 5) and six time steps (N = 6) therefore has a maximum number of 78124 nodes.

5.4.2.2 Distribution of Path Costs

The distribution of of the number of nodes per terminal altitude deviation cost and the distribution of action costs for all possible paths are shown in Figure 5.12 for search tree with no conflict nodes, and in Figure 5.11 for a search tree with some conflict nodes.

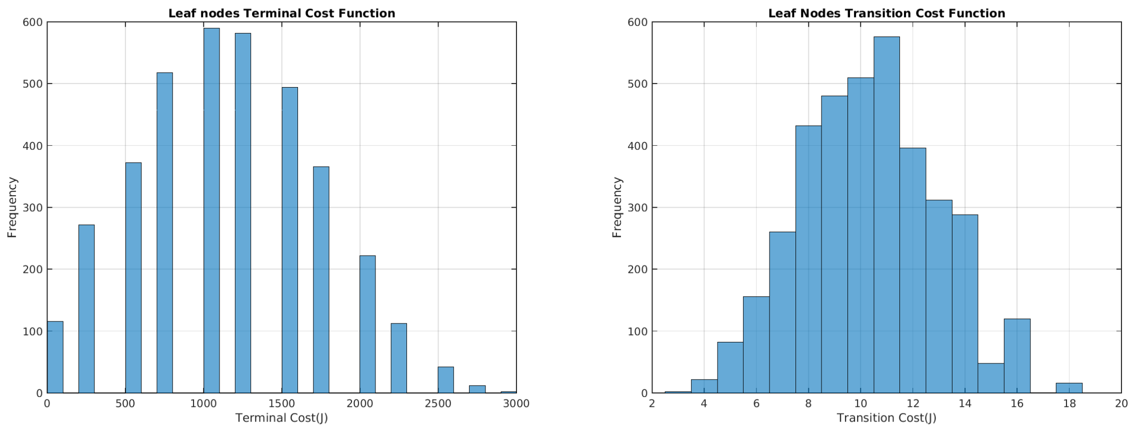


Figure 5.12: Distributions of altitude terminal costs and actions cost for all possible paths in a search tree with some conflict nodes

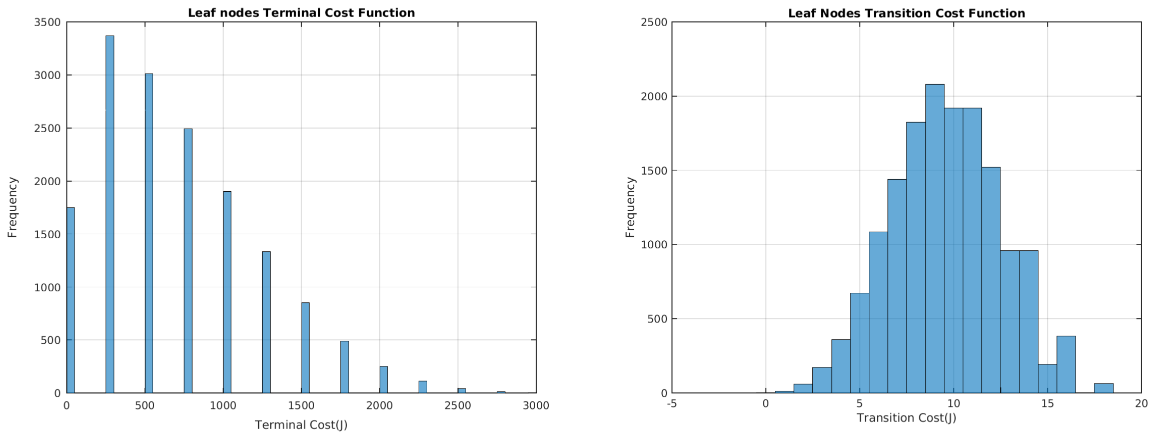


Figure 5.11: Distributions of altitude terminal costs and actions cost for all possible paths in a search tree without conflict nodes

5.4.3 Finding the Optimal Path

The nodes in the final layer at $k = N$ are called the leaf nodes. Each leaf node represents the endpoint of an admissible path from the root node to a final node. In other words, each leaf node represents the final position of a conflict-free path from the host UAV's initial position. The relationship between a leaf node and multiple nodes in a tree is shown in Figure 5.13, where every node contains a pointer to it's parent node. The path that ends in a specific leaf node can be reconstructed by starting at the leaf node and iterating backwards through each parent node until the root node is reached. In addition, each node contains both the state $\mathbf{x}_i(k)$ at its current time step and the climb rate action $u(k)$ that was applied at the previous time step. Both the path state trajectory and the climb rate action sequence can be reconstructed in this way.

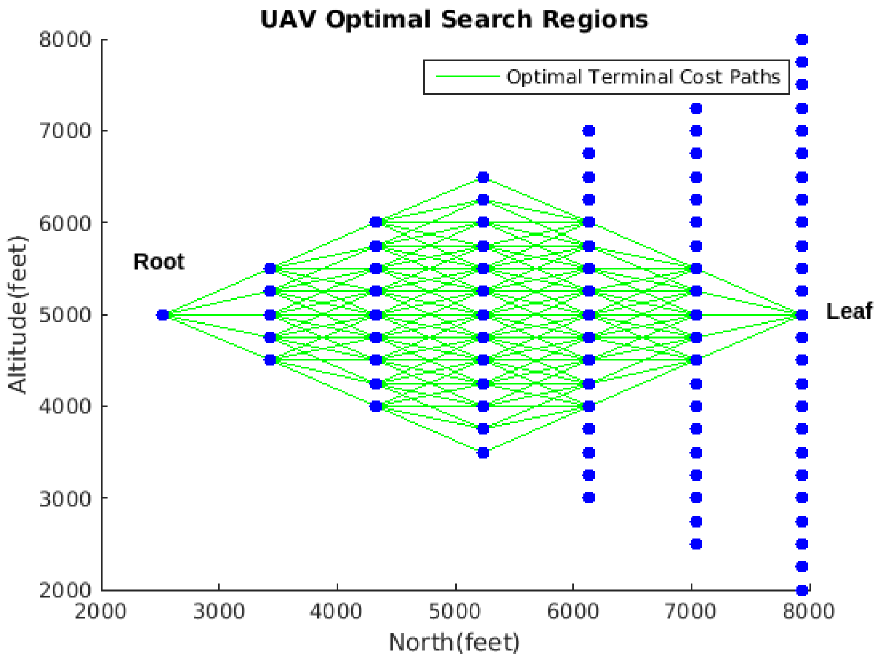


Figure 5.13: Collection of paths which maps to optimal nominal altitude of a UAV with a search tree

If there are no leaf nodes in the final layer, then its means that no admissible, conflict-free path exists. If there are leaf nodes in the final layer, then a number of conflict-free paths from the initial position to an admissible final position do exist, and the optimal path is the path with the lowest total path cost. Since each leaf node contains the total path cost of the specific path that is used to reach it, the optimal path can be determined by finding the leaf node with the lowest total path cost.

A search for the optimal cost node is a backward-search algorithm, presented in Algorithm 13. The algorithm iterates through all path planning tree leaf nodes linearly until it meets a non-leaf node then it terminates. The search algorithm is designed to locate a single optimal cost node in the tree, a leaf node with minimum terminal and transition cost. The optimal cost leaf node of the path plan tree then becomes the goal node for the UAV from the initial conflict position. An optimal node has minimum terminal cost as the first priority, thereafter a search for a minimum transition costs node at the minimum terminal-cost as the second priority.

Algorithm 13 Backwards Optimal Altitude Search

```
Require: A list of tree leaf nodes $leafs$,
 1: terminal_cost $\leftarrow$ inf
 2: control_cost $\leftarrow$ inf
 3: length $\leftarrow$ size($leafs$)
 4: transition_cost $\leftarrow$ leafs[1].transition - control_cost
 5: for index from 1 until length do
 6:     if leafs[index].terminal_cost $<$ terminal_cost or ( altitude_cost $<$ epsilon and
        transition_cost $<$ epsilon)  then
 7:         cheap_leaf $\leftarrow$ leafs[index]
 8:         terminal_cost $\leftarrow$ cheap_leaf.terminal
 9:         control_cost $\leftarrow$ cheap_leaf.transition
10:         if index $+ 1 <$ length then
11:             altitude_cost $\leftarrow$ leafs[index $+$ 1].terminal - terminal_cost
12:             transition_cost $\leftarrow$ leafs[index $+$ 1].transition - control_cost
13:         end if
14:     end if
15: end for
```

In line 1 and 2, the algorithm initialises both the terminal cost and transition cost to infinity for minimization. These initialization step is important for minimizing the costs function. In line 6 all nodes are iterated, then line 9 performs a check for optimality. The firstly priority is the terminal cost of a leaf node, this ensures that an aircraft will end up at an altitude that is closer to its nominal altitude. The second optimal cost check is the statement in line 6 which choose a node with best transition cost. The convergence of the cost function is shown in Figure 5.14 with the switching in line 7 of the algorithm given in Algorithm 13.

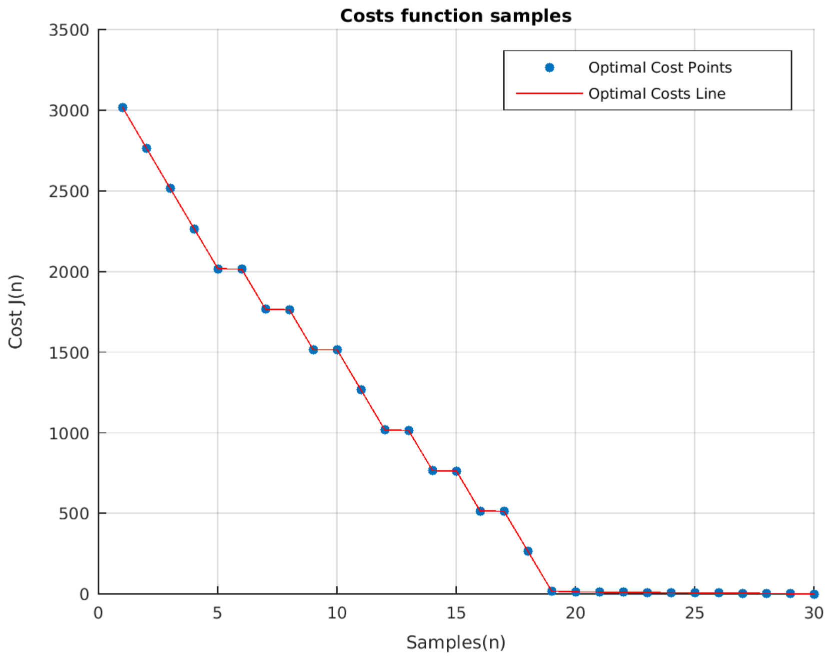


Figure 5.14: Leaf nodes cost-function (J) convergence plot during optimization

5.4.4 Search Algorithm Complexity

The search complexity of the paths will be constructed in Section 5.4.4 with the input as a list of $leafs$ for the Algorithm 13. The data structure for the nodes in the tree $\mathcal{T}$ is an array with the nodes linearly populated and identified by their index $p \in V$, where V is a set vertices or nodes in the tree. The array data structure for the tree structure T is illustrated below, to facilitate the discussion of the search algorithm complexity.

$$\mathcal{T} = \left[\ \underbrace{A_0}_{\text{root}}\ \ \underbrace{A_1\quad A_2\quad A_3 \ldots\quad A_n}_{\text{Level 2}}\ \ \underbrace{A_{1,1} \ldots\quad A_{2,1} \ldots\quad A_{3,1}}_{\text{Level 3}}\ \ \underbrace{\ldots\quad A_{n,n}}_{\text{leafs}}\ \right]$$

The search algorithm complexity for finding the optimal path is bounded by the maximum number of leaf nodes through which a loop must iterate to find the leaf node with the lowest cost. The maximum number of leaf nodes equals a^N where a is the number of discrete climb rate actions in the action space, and N is the number of time steps in the path plan. The algorithm complexity for finding the optimal path is therefore $\mathcal{O}(a^N)$. The following illustrations below are different scenarios which would be the best, average and worst cases for the search method. Hence, we make an illustration of the path plans which the method can generate when a UAV replans a path, but does not necessarily choose for execution. The analysis of the computational complexity is performed using the search-tree which we presented in Section 5.4.2. The comparison of best, average, and worst-case costs for the UAV paths are grouped based on the on the value of the altitude termination costs J.

- Best Case

  The best case for the optimal cost paths of a UAV in both conflicted and non-conflicted search trees can be considered to be from the region described in the figures shown in Figure 5.15 and 5.16. The terminal cost of these paths should be $J \leq 250$.

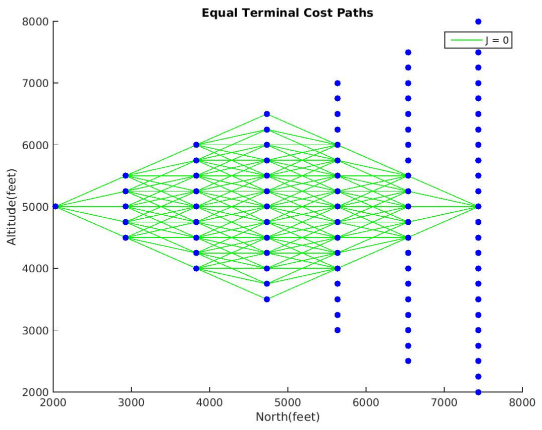


Figure 5.15: UAV 1 cost $J = 0$

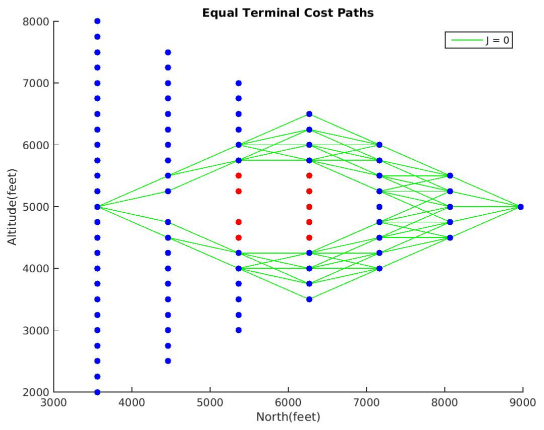


Figure 5.16: UAV 2 cost $J = 0$

- Average Case

  The average case for the optimal cost paths of a UAV in both conflicted and non-conflicted search trees can be considered to be from the region described in the figures shown in Figure 5.17 and 5.18. The terminal cost of these paths should be between $250 \leq J \leq 1750$.

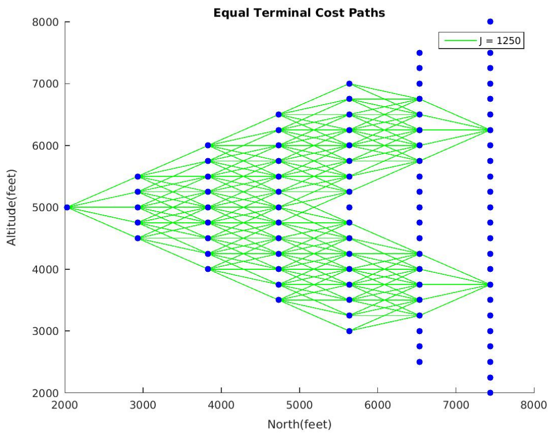


Figure 5.17: UAV 1 cost $J = 1250$

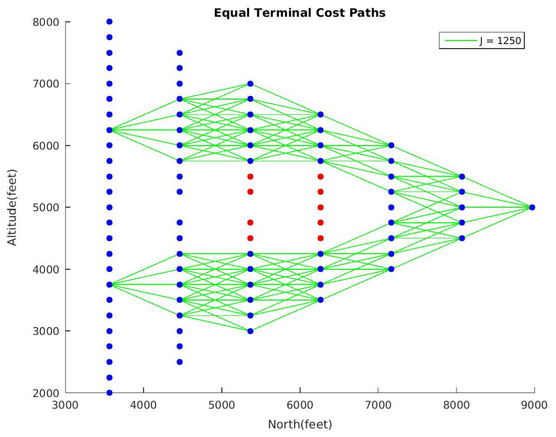


Figure 5.18: UAV 2 cost $J = 1250$

- Worst Case

  The worst-case for the optimal cost paths of a UAV in both conflicted and non-conflicted search trees can be considered to be from the region described in the figures shown in Figure 5.19 and 5.20. The terminal cost of these paths should be between $1750 \leq J \leq 3000$.

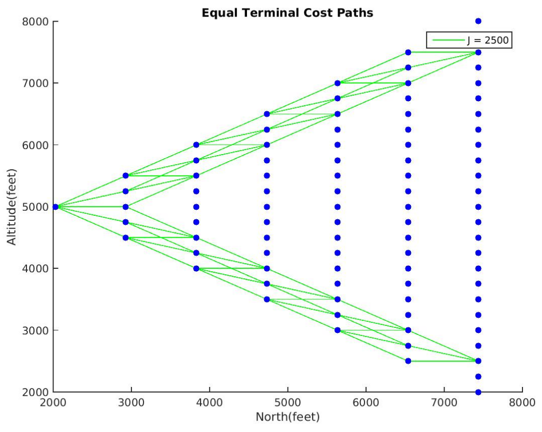


Figure 5.19: UAV 1 cost $J = 2500$

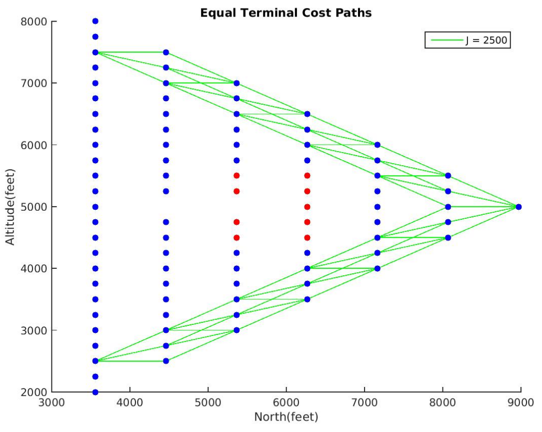


Figure 5.20: UAV 2 cost $J = 2500$

5.5 Illustrative Simulation Results

In this section we shall present illustrative simulation results for pairwise conflict resolution using the path planning-based algorithm. The aim is to show how the algorithm solves the benchmark examples such as Head-On Collision, Head-On Conflict at Minimum Altitude, Overtaking Scenario, and Parallel Flight. This shall enable us to compare the behaviour of the rules-based and path planning-based collision avoidance algorithms. Thus different examples of optimal search-tree are shown graphically.

5.5.1 Head-On Collision

The conflict scenario in Figure 5.21 below shows the path planning simulation results for a head-on conflict scenario. Both UAVs are at the same altitude of 5000 feet heading in opposite horizontal direction. The simulation results illustrate that the path planning based algorithm produces a solution where only one of the UAVs performs an avoidance manoeuvre. This is different from rules-based collision avoidance where both UAVs perform vertical manoeuvre.

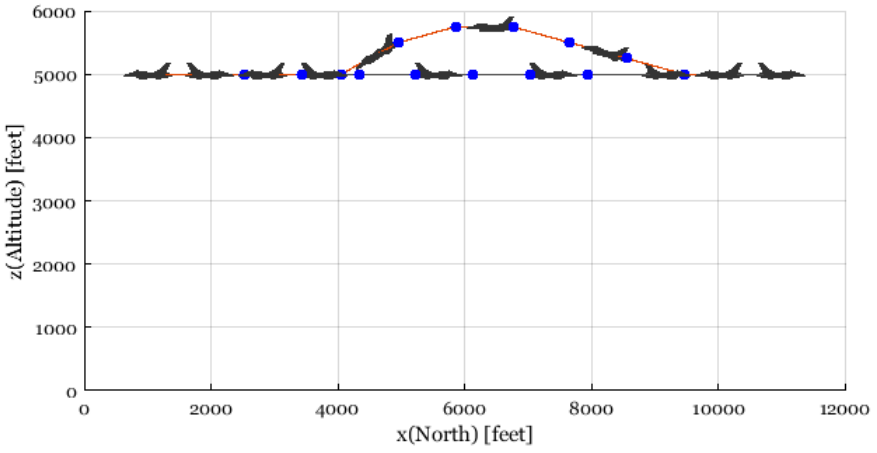


Figure 5.21: Pairwise UAVs head-on conflict avoidance

The altitude difference between the UAVs is zero, therefore manoeuvres made by one of the UAVs applies maximum altitude commands. Furthermore, these manoeuvres represents the maximum altitude deviations for any pairwise conflict resolutions because one of the UAVs maintains its altitude. However, in Figure 5.22 there are conflict nodes from the state propagation of the other intruder UAV and the paths which the host UAV is allowed to take are shown in green. Since the objective is to minimise altitude rate, the UAV begins with a shallow climb and thereafter returns to nominal altitude with a steep descend.

The altitude inputs to the UAV during path execution can be seen over time. The path planning commands a climb rate input of 25 ft/sec which is held constant from 0 seconds until 30 seconds. Thereafter, the UAV maintains its altitude for a single time step using an input of 0 ft/sec; this is because the two UAVs are horizontally crossing each other. Thereafter, this is followed by 10 seconds of descend manoeuvres with a rate of -50 ft/sec. The descend manoeuvre is applied as part of conflict resolution to return the UAV to the nominal altitude as soon as possible. This occurs after the intruder is further away and no longer a threat.

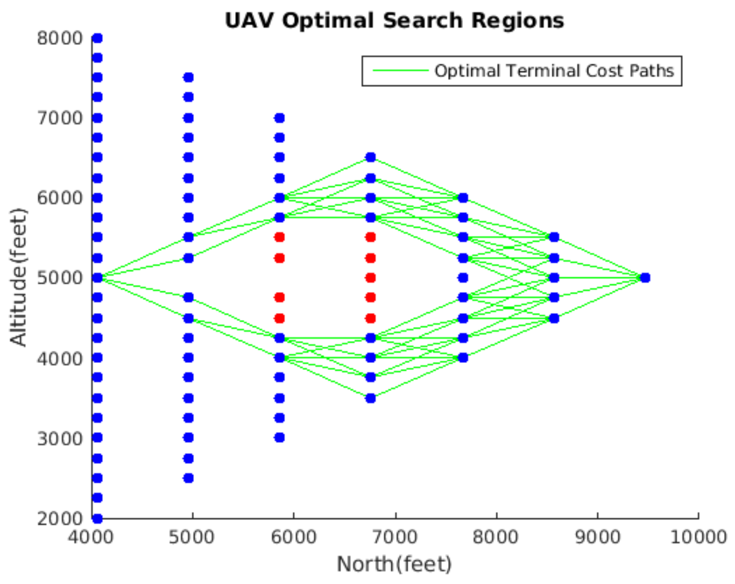


Figure 5.22: Search-tree of a manoeuvring UAV

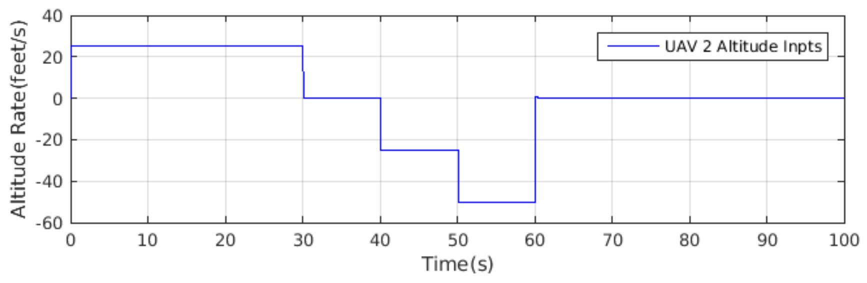


Figure 5.23: Timeline of altitude rate commands during path execution

The timeline of altitude change of the UAV which deviates from its nominal altitude is shown in Figure 5.24. The timeline shows that the duration of the conflict is 60 seconds, hence, the path plan execution is exhausted when the path plan horizon has ended and there is no new conflict. However, if the conflict still existed, then a new path plan would be created for the next 60 seconds.

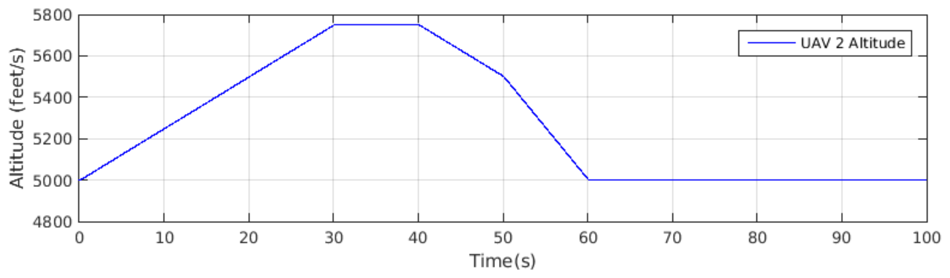


Figure 5.24: Timeline of the altitude changes for the manoeuvring UAV

5.5.2 Head-On Conflict at Minimum Altitude

The following conflict scenario in Figure 5.25 shows the path planning simulation results for a head-on conflict scenario. However, the one UAV is at an altitude of 990 feet which is below the minimum altitude of operation for the collision avoidance system, while the other approaching UAV is above the minimum altitude at 1100 feet, heading in the opposite horizontal direction. The simulation results illustrate that the path planning algorithm can detect a non-cooperative dynamic obstacle.

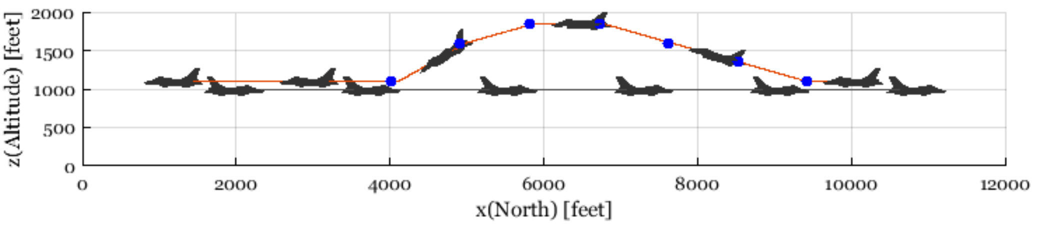


Figure 5.25: Pairwise conflict resolution with one of the UAVs at minimum altitude

The search-tree in Figure 5.26 illustrates that the algorithm will not be able to add nodes below the altitude of 1000 feet. However, the tree is able to populate nodes at higher altitudes. Therefore, the higher UAV chooses to climb to allow the approaching lower UAV to pass below.

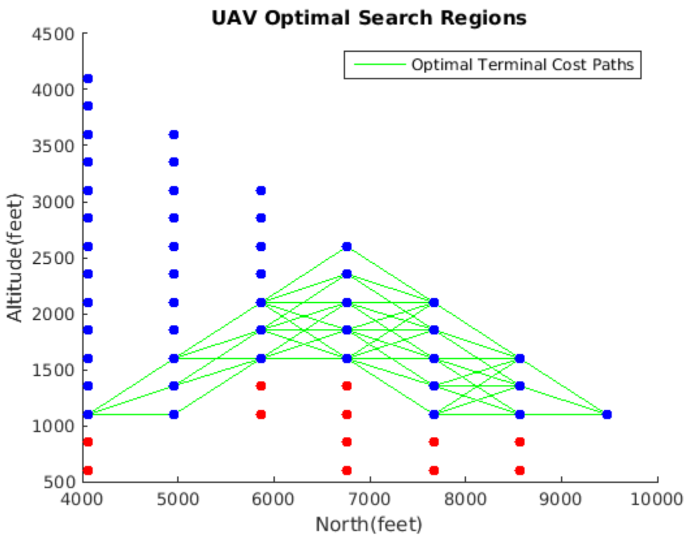


Figure 5.26: Manoeuvre UAV path planning tree

The tree nodes distribution at altitude level is shown in Figure 5.27. The conflict nodes at terminal costs below the nominal altitude of 1100 feet are excluded from the tree because they are below 1000 feet as propagated from nominal altitude of 1100 feet. This is seen at the density depth level of 1500 feet, 2500 feet and 3500 feet in the density function.

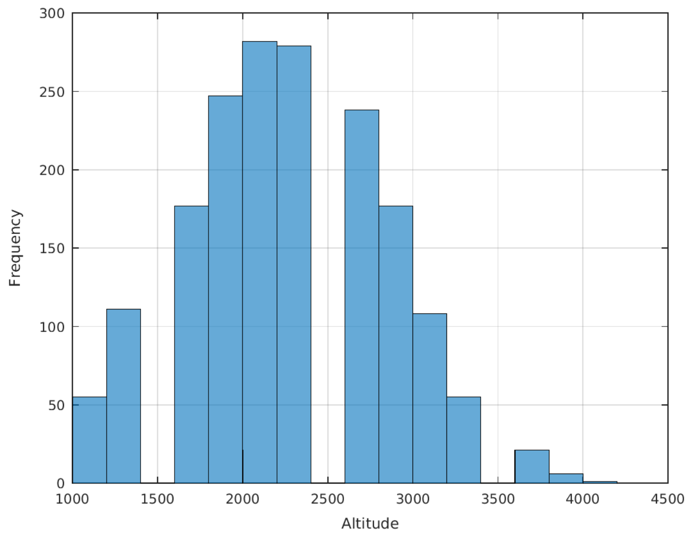


Figure 5.27: Distribution of leaf nodes from nominal altitude 1100 feet

5.5.3 Parallel Flight

The following conflict scenario in Figure 5.28 shows collision avoidance for two UAVs with parallel flight and the same horizontal rates. The simulation is set up in such a way that both UAVs are in conflict from $t = 0$. The initial vertical separation of the simulation was set to 500 feet, which is below the conflict vertical threshold of our conflict separation of 650 feet.

The two UAVs shown in Figure 5.28 will remain in conflict for time infinity, because there is no change in the relative horizontal positions of the UAVs. Furthermore, the nominal altitude of the UAVs are vertically in a conflict. The search-tree which was used for conflict prediction is shown in Figure 5.29 and it illustrates the parallel conflict nodes between the UAVs. However, one of the UAVs makes an effort to maintain the vertical separation distance. In addition, the deviation from an altitude of 4500 is maintained at about 4250 feet, leading to an altitude difference of $|5000 - 4250| = 750$ feet between the two UAVs as shown in Figure 5.30.

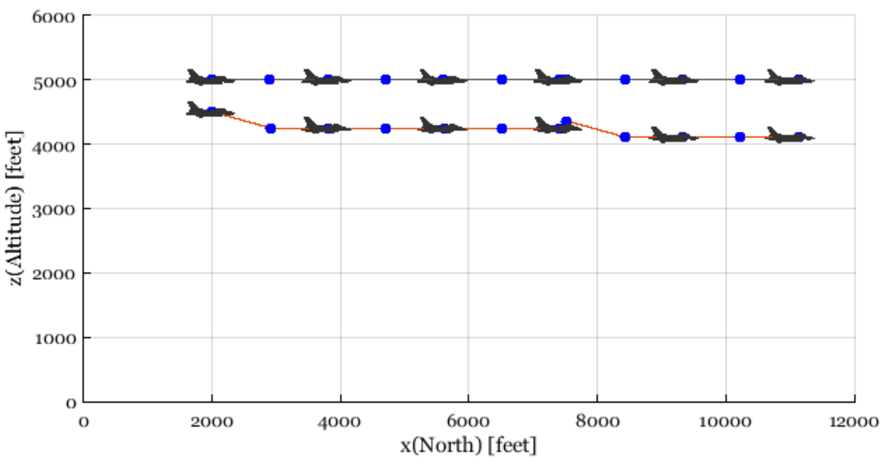


Figure 5.28: Pairwise conflict resolution for pairwise UAVs in parallel flight

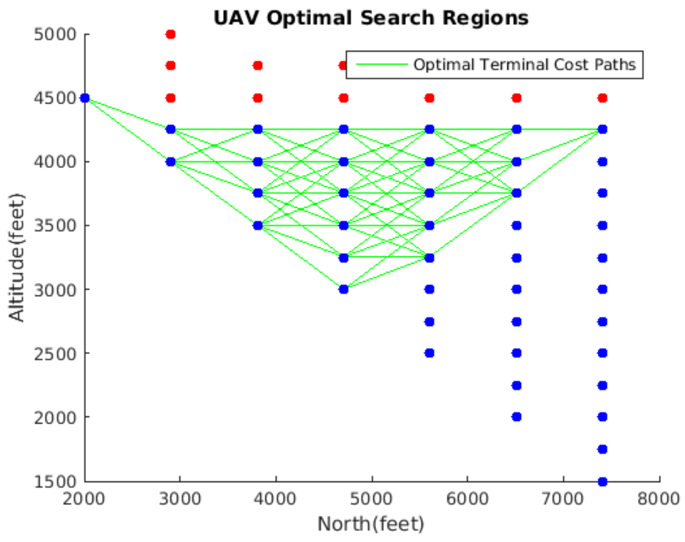


Figure 5.29: Parallel descend manoeuvre UAV path planning tree

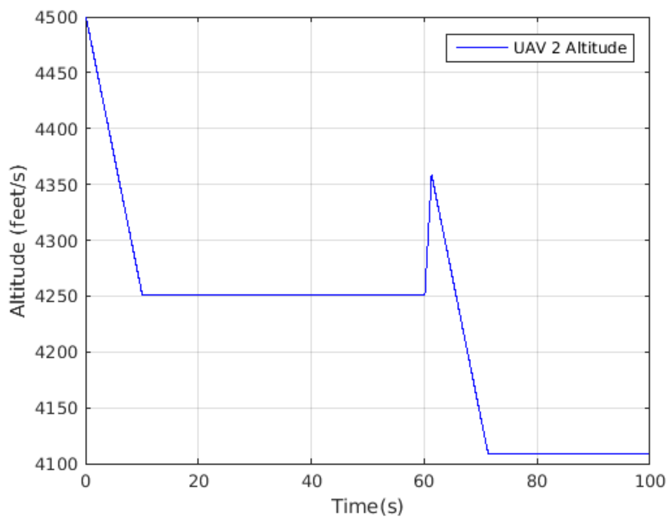


Figure 5.30: Altitude path of UAV with conflict resolution descend commands

A similar manoeuvre of conflict resolution for the higher altitude UAV are shown in Figure 5.31. The path planning search tree is illustrated in Figure 5.32. The altitude responses of the UAV that deviates from its nominal altitude is shown in Figure 5.33, with the first response to the parallel flight a climbs from a nominal altitude of 5000 feet to 5250 feet. This deviates the UAV to a new altitude of 5250 feet, thus an altitude difference of $|4500 - 5250| = 750$ feet relative to the intruder UAV. In both descend and climb manoeuvres an advisory magnitude $u_k = |25|$ feet per seconds is used. Thus, a UAV can either choose to climb or descend during parallel flight.

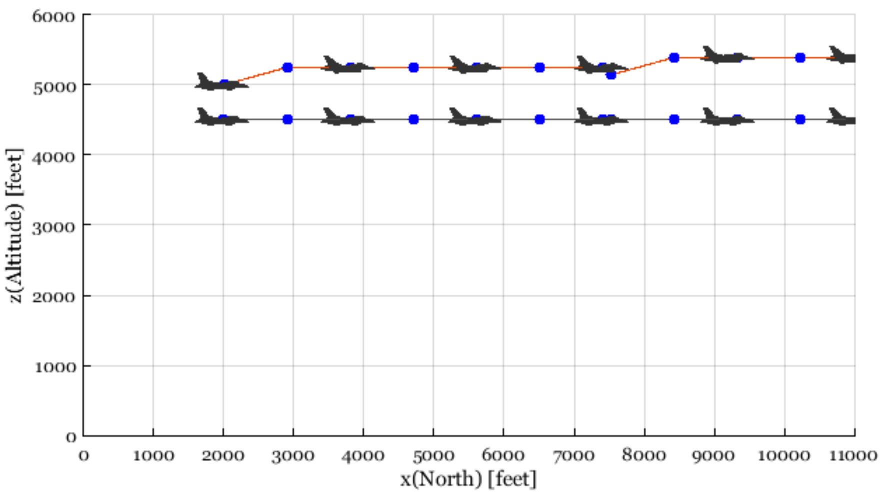


Figure 5.31: Pairwise UAVs parallel flight with climb conflict resolution

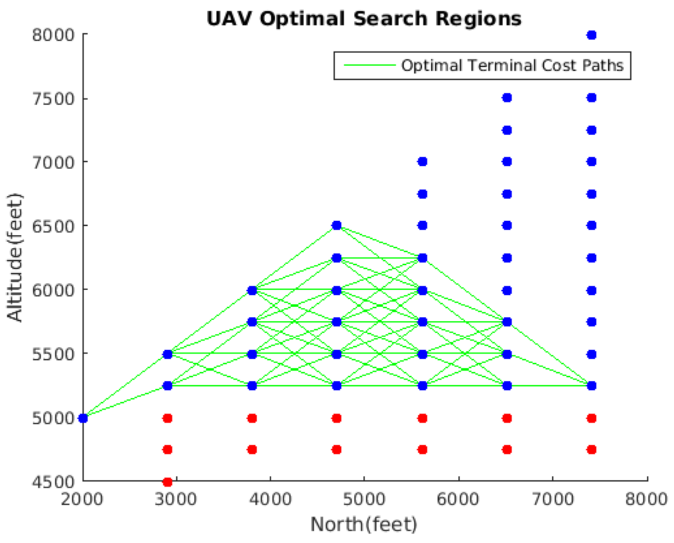


Figure 5.32: Parallel climb manoeuvre UAV path planning tree

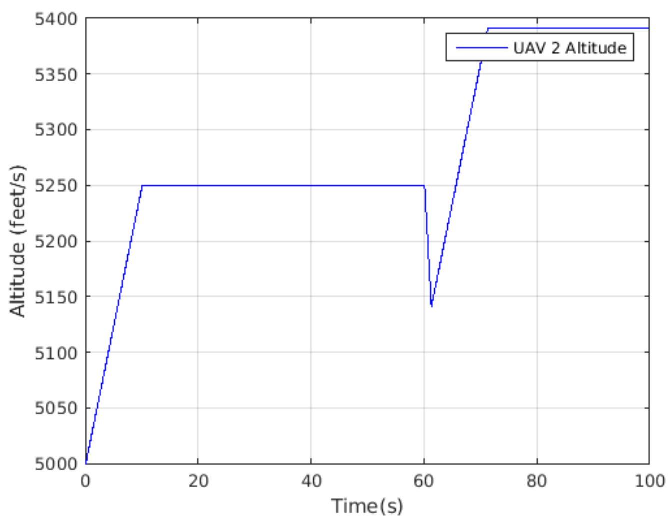


Figure 5.33: Altitude path of UAV with conflict resolution climb commands

5.5.4 Overtaking Scenario

The simulation results shown in Figure 5.34 represents a pairwise conflict resolution between two UAVs in an overtaking scenario. The two UAVs begin at two different horizontal positions, with same altitude of 600 feet. However, the UAVs have different horizontal velocities. The two UAVs i and j have different horizontal velocities with a given inequality $\dot{x}_i < \dot{x}_j$. The horizontal velocity $\dot{x}_j$ is for the leading UAV, and $\dot{x}_i$ is for the overtaking UAV. Since the horizontally lagging UAV has a significantly larger $\dot{x}_j$ in magnitude than the leading UAV with $\dot{x}_i$, the overtaking lasts for a short time. There are two search-trees which were created by the overtaking UAV which are shown in Figure 5.35 and 5.36. These results illustrate an example where a single path planning search-tree was not able to resolve the conflict therefore a second search-tree was created by the same UAV.

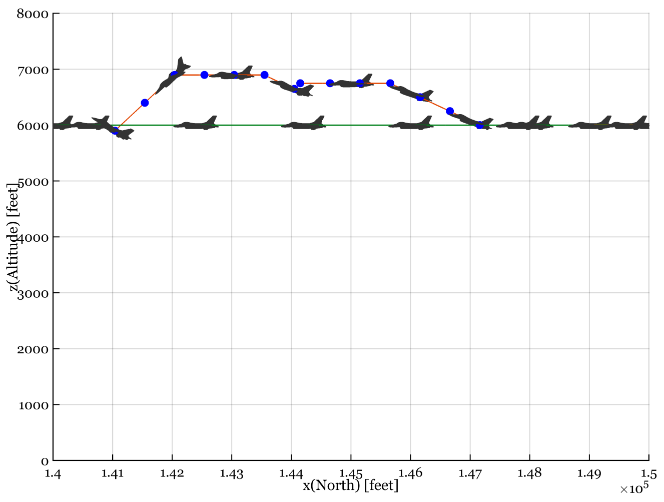


Figure 5.34: Conflict resolution for a UAV overtaking another UAV at a relative horizontal velocity of 40 feet/s

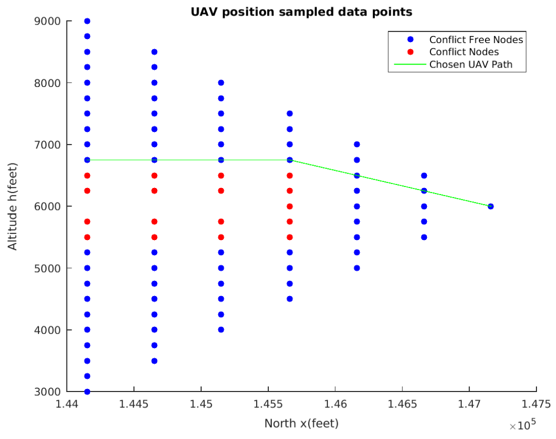


Figure 5.35: First plan ($\Delta \dot{x} = 40$ ft/sec)

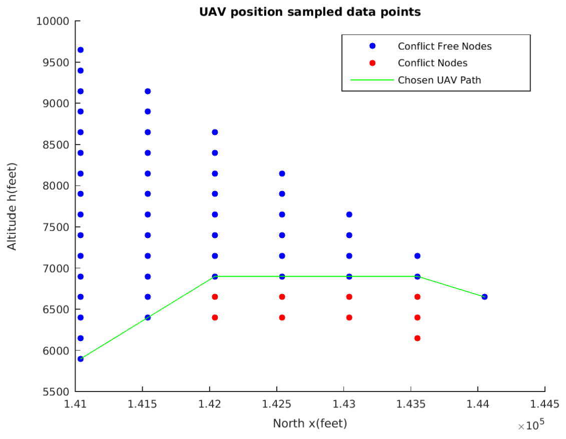


Figure 5.36: Second plan ($\Delta \dot{x} = 40$ ft/sec)

The two UAVs shown in Figure 5.34 have a high horizontal rate difference $|\dot{x}_j - \dot{x}_i|$, therefore a single path planning tree allows the overtaking UAV to return to its nominal altitude. However, if the conflict overtaking takes longer to resolve at low relative horizontal rates since overtaking will need to create multiple search-trees. These scenarios are illustrated in Figure 5.37 where the path planning will create multiple trees for the overtaking UAV as shown in Figure 5.38 and 5.39.

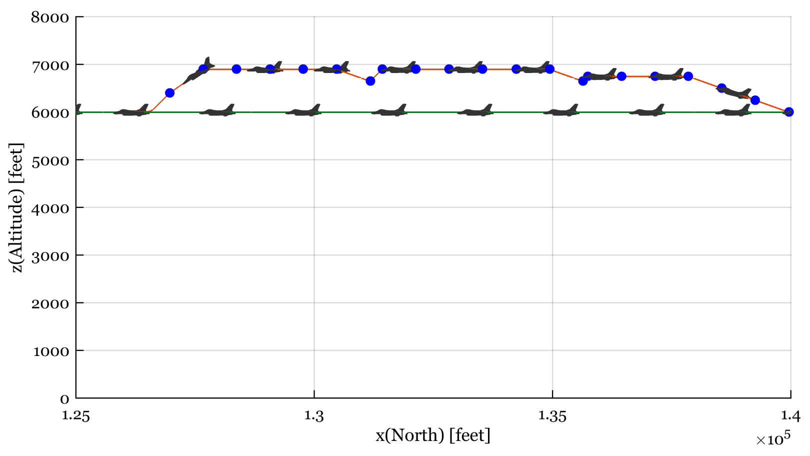
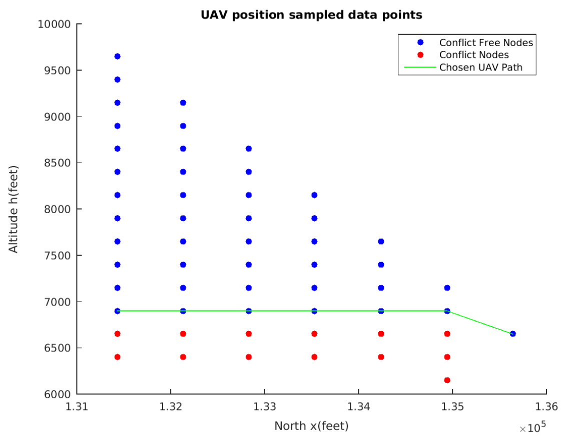


Figure 5.37: Collision avoidance in an overtaking scenario at a relative horizontal velocity of 20 feet/s



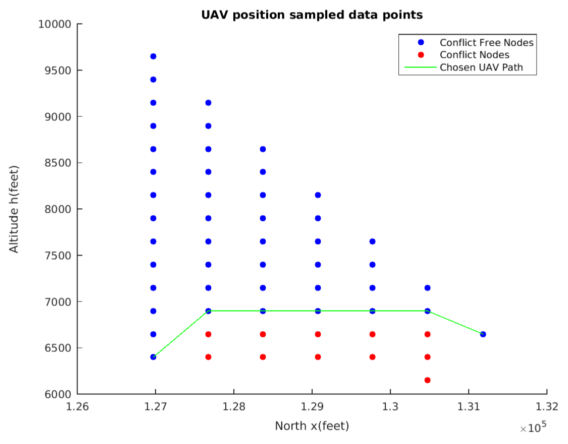


Figure 5.38: First plan with ($\Delta\dot{x} = 20$ ft/sec) Figure 5.39: Second plan ($\Delta\dot{x} = 20$ ft/sec)

5.6 Summary and Conclusions

This chapter presented an implementation of a search-based path planning algorithm for strategic conflict prediction and resolution for pairwise UAVs. The search-based path planning used the nominal state propagation of the intruder UAV the state propagation of the host aircraft performed using a finite vertical manoeuvre input set that uses the same actions as TCAS. In addition, the path planning approach shares its intended path plan with other UAVs for propagation of trajectories in path segments of 10 seconds as it was conceptualised in Chapter 3 in Section 3.2.4. State propagation is used for collision prediction to a time horizon of 60 seconds into the future. Furthermore, the collision prediction step enables the path planning based algorithm to use conflict detection and TCAS altitude input sampling to simulate the trajectory of an intruder UAV. Therefore, the path planning algorithm is able to create a search-tree from vertical climb and descend motion primitives which enables searching for conflict-free trajectories. Each trajectory in the search-tree is a collection of state node data structures from the root of the tree to a leaf node with 30 seconds conflict approach and 30 seconds of nominal altitude return. Furthermore, the path planning formulation includes a cost function for hierarchical optimal control; firstly it minimises the altitude deviation of the leaf nodes from the nominal altitude, then it minimises the control for the collision avoidance path. Next, each path plan in the search-tree is searchable for the optimal conflict resolution altitude level which minimises the cost function of the manoeuvres. Finally, the simulation results illustrated the execution of the path planning based conflict detection and resolution algorithm for pairwise conflict scenarios. The path planning based approach presented in this chapter will be extended to enable cooperative conflict resolution for multiple UAVs in the next chapter.

Chapter 6

Cooperative Collision Avoidance

In this chapter we shall present a cooperative path planning based collision avoidance algorithm for multiple UAVs, as an extension of the work presented in Chapter 5. The cooperative path planning will use token allocation as the communication framework for the decision making for each UAV to perform path planning for a group of UAVs. In Section 6.1 we provide an overview discussion of a cooperative path planning algorithm for multiple UAVs which implements token allocation. Thereafter, cooperative collision prediction for multiple UAVs is discussed in Section 6.2. Next, in Section 6.3, we deal with the token allocation priority policy that controls the order of the sequential path planning for the individual UAVs. Finally, in Section 6.4, illustrative simulation results for conflict scenarios involving groups of three and five UAVs, where the UAVs utilise path planning, token allocation, and the communication network to cooperatively resolve conflicts in cluttered environments

6.1 Cooperative Collision Avoidance System

The system shown in Figure 6.1 has a central node that performs the collision prediction and coordinates the sequential, prioritised path planning. The UAVs transmit their own states and intended flight plans to the Communication Hub central node, and receive the published flight plans of the other UAVs from the central node. The central node and Position Simulator propagate the states of all UAVs forward in time and check for predicted collisions. For each UAV-to-UAV collision that is predicted, the corresponding elements in the Conflict Indicator matrix are set to indicate that there is a conflict between the two UAVs that must be resolved. If one or more collisions are predicted, then the central node coordinates the cooperative collision avoidance using a Token Allocator strategy. The Path Planner for each UAV proposes its new collision avoidance path sequentially according to a priority order. Therefore, the Communication Hub central node determines the priority order and transmits the path planning tokens to the individual UAVs.

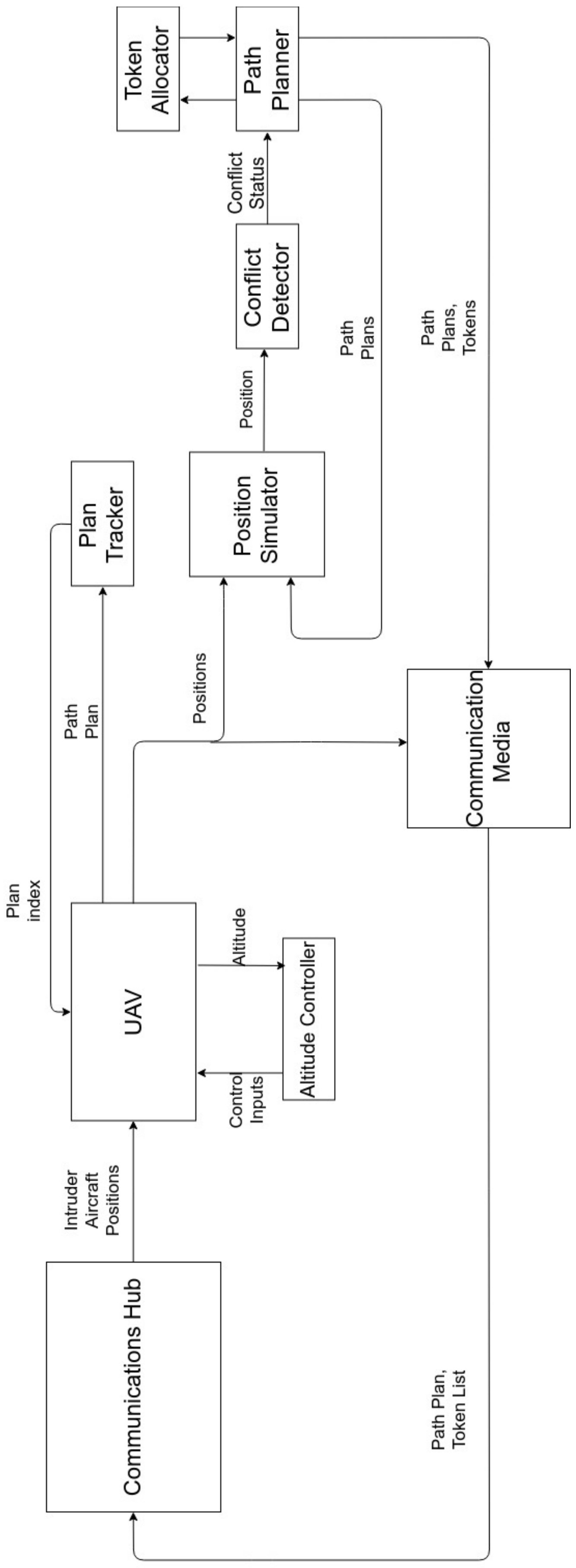


Figure 6.1: Cooperative path planning collision avoidance overview system

A use case for the token allocation cooperative architecture is when multiple UAVs generate path plans for a single conflict. These plans cannot be executed all at once, therefore the architecture uses the Communication Hub network to share flight data and to help coordinate the path planning. The cooperative planning is useful to coordinate conflict resolutions between multiple aircraft in a short amount of time. The proposed co-operative algorithm is the Token Allocation subsystem which will allow conflict resolution for multiple UAVs. The Token Allocation will have multiple path plan proposals from each conflict. Therefore, the implementation of Plan Tracker will monitor the path plan execution of each UAV during conflict detection by other UAVs. The Plan Tracker method estimates the future states of intruder UAVs to enable cooperative conflict prediction.

6.2 Cooperative Collision Prediction

To perform cooperative collision prediction, all UAVs communicate their current positions and intended flight paths to the central node. The central node uses the state propagation component and the collision prediction component described in the previous chapter to propagate the states of all UAVs forward in time and to predict whether any of the future states of the UAVs result in collisions. For each UAV-to-UAV collision that is predicted, the corresponding elements in the Conflict Indicator matrix C are set to $c_{ij} = 1$ to indicate that there is a conflict between the two UAVs that must be resolved.

$$C = \begin{bmatrix} 0 & c_{12} & c_{13} & \dots & c_{1n} \\ c_{21} & 0 & c_{23} & \dots & c_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{n1} & c_{n2} & c_{n3} & \dots & 0 \end{bmatrix} \in \mathbb{R}^{n\times n} \tag{6.1}$$

Let us suppose that a conflict detection algorithm such as Algorithm 3 is being executed, then as a result, a conflict indicator matrix C is produced. Each row of C will contain indicator values which give the a conflict status with all others UAVs except the host; that is $C_{ii} = 0$ for each UAV $i \in N$.

$$C_{ij} = \begin{cases} 0 & \text{no conflict or i = j} \\ 1 & \text{conflict resolution} \end{cases} \tag{6.2}$$

6.2.1 Concepts of Operation

In this section we explain the concept of operation of the multi-UAV cooperative collision avoidance system. We use an example of three UAVs in a potential multi-UAV conflict scenario. The example illustrates the cooperative collision prediction using state propagation, as well as the cooperative collision avoidance using sequential path planning for the individual UAVs according to a priority order. The initial states for the three UAVs in the example multi-UAV conflict scenario is shown in Figure 6.2. One UAV is approaching from the left, and two UAVs are approaching from the right in parallel flight.

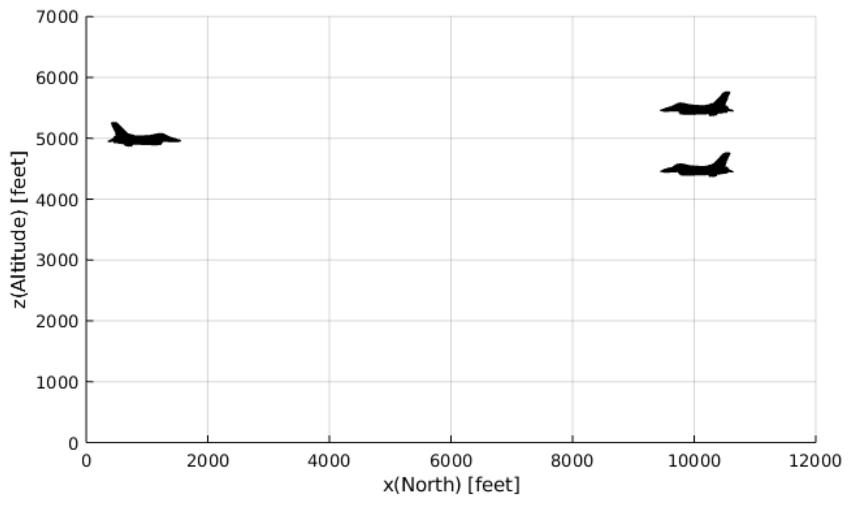


Figure 6.2: Example multi-UAV conflict scenario involving three UAVs

The cooperative collision prediction is illustrated in Figure 6.3. The states of all three UAVs are propagated forward in time, and two conflicts are predicted. The first conflict is between the left UAV and the right higher UAV; the second conflict is between the left UAV and the right lower UAV. The cooperative collision avoidance algorithm is then triggered, and collision avoidance paths are planned sequentially for the three UAVs, based on the priority order determined by the token allocation strategy. For this scenario, the right higher UAV will plan first, then the lower right UAV, and finally the left UAV.

The new path plan for the right higher UAV is shown in Figure 6.4. The right higher UAV will perform a climbing manoeuvre to avoid the conflict with the left UAV and will then return to its nominal altitude. The UAV publishes its new planned path, and since one unresolved conflict still remains, the token is allocated to the right lower UAV.

The new path plan for the right lower UAV is shown in Figure 6.5. The right lower UAV will perform a descend manoeuvre to avoid conflict with the UAV from the left and will then return to its nominal altitude. The UAV publishes its new planned path, and since no unresolved conflicts remain, the left UAV does not need to replan its path. No further tokens are allocated, and the three UAVs execute the planned collision avoidance manoeuvres.

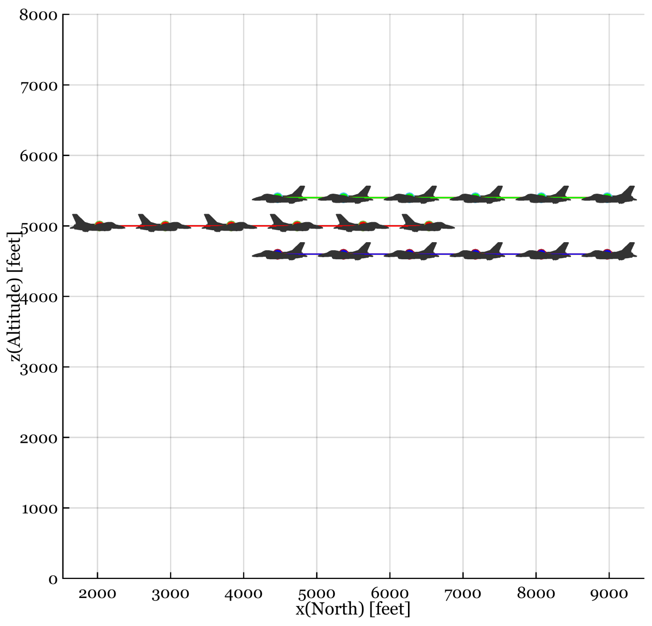


Figure 6.3: Cooperative conflict prediction using state propagation (two conflicts predicted)

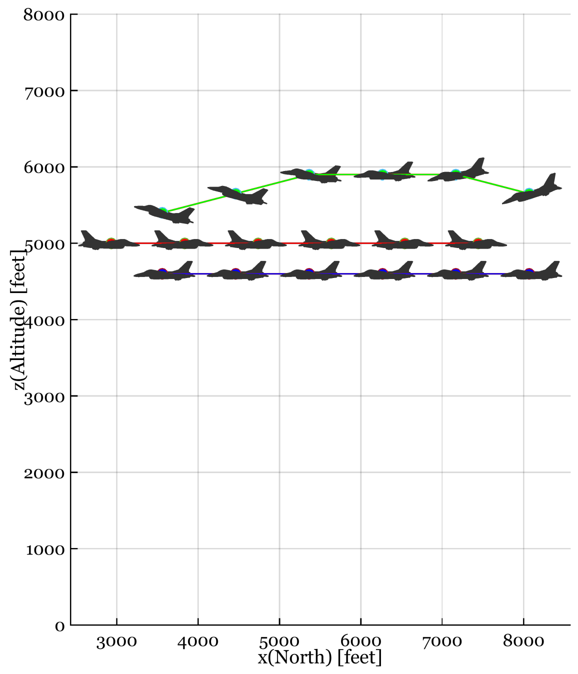


Figure 6.4: Cooperative, sequential path planning for first UAV (one conflict remains)

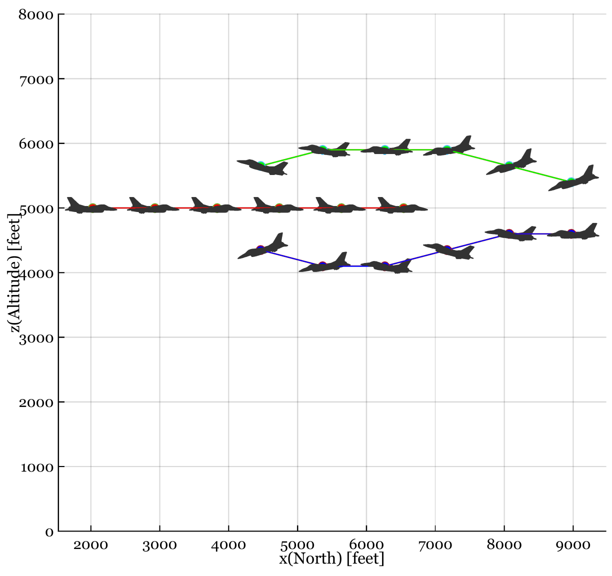


Figure 6.5: Cooperative, sequential path planning for second UAV (no more conflicts)

6.3 Cooperative Collision Avoidance

This section describes the implementation of the token allocation strategy to perform sequential path planning for multiple UAVs. The state machine that controls the sequential path planning for multiple UAVs with predicted conflicts is described in Section 6.3.1. In Section 6.3.2, the token allocation rules that govern the priority order for the sequential path planning are outlined. Finally, the token allocation algorithm that determines which UAV has the least space to manoeuvre is explained.

6.3.1 Sequential Path Planning

Token allocation rules are applicable after a conflict prediction has simulated the existing path plans of the UAVs. The the token allocation algorithm that determines which UAV has the least space to manoeuvre is explained to control the sequence of multi-aircraft conflict resolutions where each UAV makes its own path plans and each wishes to execute all plans at once. The rules are designed to control the order in which the UAVs perform their sequential path planning so that complex conflicts are resolved. A token is granted to a single UAV at each time step. The UAV that received the token then plans and publishes its new conflict avoidance path. After the UAV is selected to execute its own path plan, a new conflict prediction is done to monitor the state of the multi-aircraft conflict. When no conflicts remains, then no further planning is required. However, token allocation does not know how a UAV executes path plans. Therefore, the detail of the execution of the path plan remains the function of a local path planner. When the conflict persist after token allocation, a new UAV is selected until there is no further conflict.

To facilitate cooperative collision prediction and avoidance, all UAVs publish their path plans and their token statuses over the communication network. The path plans are published in a path plan table P, and the token statuses are published in a token tracking arrary T. The token status T[i] of a given UAV indicates whether the UAV has received a token and is therefore executing a planned collision avoidance manoeuvre. The token status also indicates the UAV's current time index within its collision avoidance manoeuvre, so that its progress can be tracked. The token status is used by other UAVs to determine the UAVs intent so that its state can be propagated for collision prediction purposes. Finally, the token status is also used to coordinate the sequential path planning, since a UAV that has already received a token cannot be allocated a new token until it has completed its current collision avoidance manoeuvre. Due to the fact that UAVs can be at different stages of path plan execution, a UAV will use a locking mechanism and continue to publish its token status T. Therefore, a UAV will not be available for new path planning until its previous path plan is exhausted. This is to avoid having UAVs stop execution of previous optimal path plans.

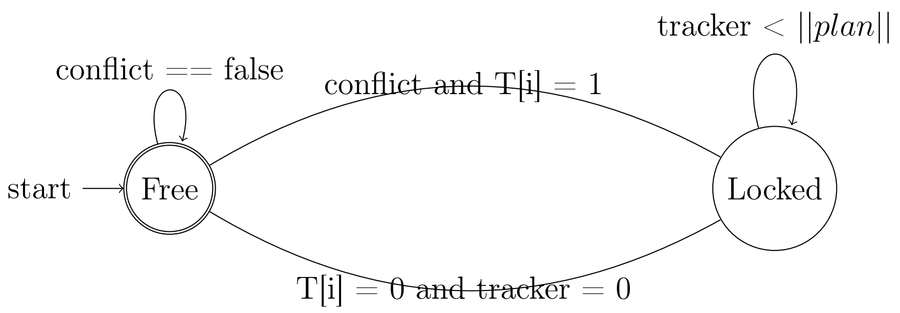


Figure 6.6: Token allocation communication State Machine

The state-machine shown in Figure 6.6 is a communication model for the token execution status of a single UAV i to a multiple of cooperative UAVs. The default state of the state machine is "Free". In this state a planned collision avoidance manoeuvre UAV is conflict-free and is not executing any path planning. The state "Free" indicates that a UAV is free for the token allocation stage and it is available to receive a token and then plan and execute a collision avoidance manoeuvre, if selected. The token allocation state machine is in a "Locked" state if a UAV is currently executing a planned collision avoidance manoeuvre. This mechanism is established to ensure UAVs are able to complete previously planned paths. The "Locked" state of the state machine indicates that the UAV is currently not available for sequential path planning, and is not available to participate in cooperative conflict resolution. The current state of the Token Allocation state machine is published via token status variable, to be discussed further in Algorithm 14 below.

6.3.2 Priority Order for Token Allocation

The central node allocates a token to a single UAV at a time. The central node allocates the tokens in the following priority order (from highest to lowest priority):

1. The UAV that is above the minimum altitude for terrain avoidance
2. The UAV that is not currently executing a collision avoidance
3. The UAV which has the least space to manoeuvre
4. The UAV with the smallest deviation from its nominal trajectories
5. The UAV with the lowest identity number

6.3.3 Token Allocation Algorithm

Algorithm 14 implements the token allocation rules that determine the priority order for the sequential planning. The rules are implemented according to the priority in order to obtain the index or UAV identity from the path planning *Planned* variable. The *Planned* variable contains a list of all UAVs that have a predicted conflict but are not executing collision avoidance manoeuvres, and are therefore eligible for token allocation. From the Planned list of eligible UAVs, the UAV with the least space to manoeuvre (with the lowest Manoeuvrability metric) is chosen. If one or more UAVs are tied for manoeuvrability, then the UAV with the minimum number of predicted conflicts (lowest intruder count) and with the lowest path deviation cost, is chosen.

Algorithm 14 Token allocation with cooperation rules

```
Require: UAVs list identities $Planned$, Intruders metric $Intruder\_Count$, Path costs
    $Path\_Cost$, Manoeuvrability metric $Manoeuvre$
 1: chosen_uav ← Planned[1]                                        ▷ Default ID
 2: min_manoeuvre ← minimum($Manoeuvre$)
 3: length = count(Manoeuvre == min_manoeuvre);
 4: if length > 1 then
 5:     min_intruders ← minimum($Intruder\_Count$)
 6:     if count($Intruder\_Count$) > 2 then
 7:         path_min ← min($Path\_Cost$)
 8:         for uav_id each in $Planned$ do
 9:             if $Intruder\_Count$[uav_id] == min_intruders and path_min then
10:                 chosen_uav ← uav_id
11:                 break
12:             end if
13:         end for
14:     else
15:         for uav_id each in $Planned$ do
16:             if $Manoeuvre$[uav_id] == min_manoeuvre then
17:                 chosen_uav ← uav_id
18:                 break
19:             end if
20:         end for
21:     end if
22: end if
23: T[chosen_uav] ← 1
24: return chosen_uav
```

The procedure given in Algorithm 15 is a checking mechanism for a UAV path execution. The algorithm checks the conflict status of each of the UAV in line 3, and if a UAV has predicted conflict but is already executing a collision avoidance manoeuvre, then it is not listed in the Unplanned list of UAVs identities. The collection Unplanned is a container used by Algorithm 14 to indicate whether a UAV is available for token allocation. A UAV is available when line 4 passes, meaning the UAV with identity i is in conflict $c_i == 1$ and its path execution tracker $T[i] == 0$ is not in the middle of path plan execution.

Algorithm 15 UAV Path Plan locking $PathCheck(C, T)$

```
Require: Conflict Indicator $C = \{c_1, c_2, \dots c_n\}$, Path tracking T
1: Unplanned $\leftarrow \{\}$
2: length = size(T)
3: for $c_i$ in C do
4:     if $c_i == 1$ and T[$i$] $== 0$ then
5:         Unplanned $\leftarrow$ Unplanned $\cup\{i\}$
6:     end if
7: end for
8: return Unplanned
```

6.3.4 Path Execution

The token status variable is used to control the execution of the UAV's collision avoidance manoeuvre. If the token status variable is zero, then the UAV is in normal flight mode, and the altitude controller is executed. If the token status variable is non-zero, then the UAV is in collision avoidance mode, and the planned collision avoidance manoeuvre is executed. While in collision avoidance mode, the execution time index is advanced at every time step. When the execution time index reaches the end of the collision avoidance path plan, the token status is set to zero, and the UAV returns to normal flight mode. The path execution is implemented by Algorithm 14.

At each time step a time-step, the input *time* is given as a time reference since the plan execution started. This value is hashed in line 3 to obtain the input to execute. In line 4 the tracking variable is checked if it is within indexes of path plan P. An altitude command rate of a UAV from a path plan is given as a reference, thereafter the token status T is updated. In line 7 the UAV token status is set to zero and to indicate that the UAV has returned to normal flight mode and that the UAV is available for token allocation. The altitude rates executed during a "Free" token state in Algorithm 16 are control inputs during nominal flight of a UAV.

Algorithm 16 $PathTrack(P, time, T, h_n, h, T)$

```
Require: UAV Path Plans $P = \{p_1, p_2, \dots p_m\}$, Simulation timer value $time$, Path tracking metric T, Nominal Altitude $h_n$, Current Altitude h
 1: timestep ← time                                                ▷ seconds
 2: time_horizon ← T                                               ▷ seconds
 3: tracker ← $PathHashing(timestep, time\_horizon, time)$
 4: if tracker $\leq$ m and tracker $>$ 0 then
 5:     $\dot{h}$ ← P[tracker]
 6:     T[$UAV_{id}$] ← tracker
 7: else
 8:     $\dot{h}$ ← $AltitudeControl(h, h_n)$
 9:     T[$UAV_{id}$] ← 0
10: end if
11: return $\dot{h}$
```

6.4 Illustrative Simulation Results

This section presents illustrative simulation results for cooperative multi-UAV collision avoidance using the path planning based algorithm and the token allocation strategy. The simulations include the following multi-UAV conflict scenarios: two UAVs in head-on conflict affecting a third, three UAVs in head-on conflict, three UAVs in head-on conflict near minimum altitude, and five UAVs in a staggered, head-on conflict.

6.4.1 Three UAVs: Head-on Conflict with effect on third UAV

The discussion below (Figure 6.7) presents a head-on conflict of three UAVs where there is a predicted conflict between two UAVs at the same altitude of 5000 feet. Additionally there is a third UAV at a higher altitude of 6000 feet, for which a conflict is not predicted. The third UAV is flying in parallel with UAV approaching from the right direction. The purpose of the simulation below is to illustrates how the token allocation sequence affects the future path planning of other UAVs. The conflict scenario below is applicable where there is a possibility of new conflict with other UAVs where the new UAV, even though it was not involved in the original predicted conflict, is affected by the past conflict resolution decisions.

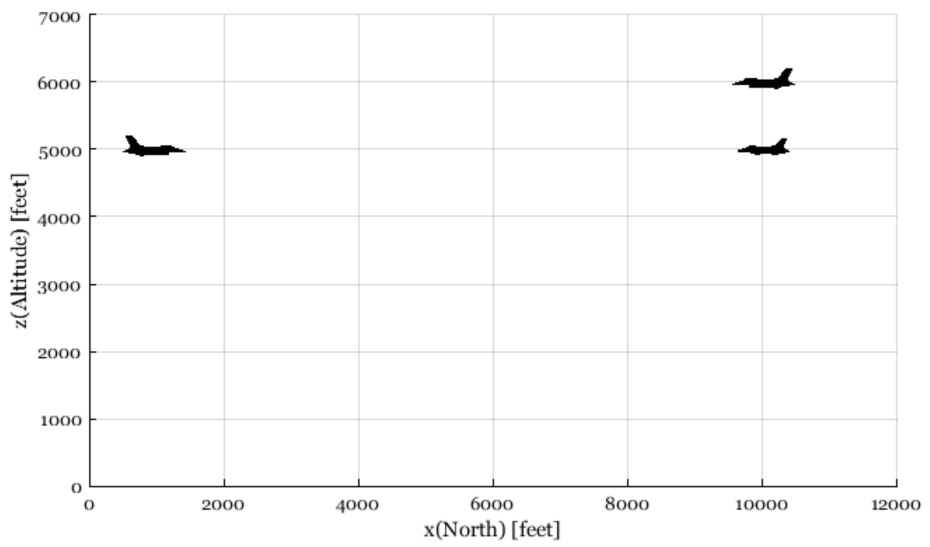


Figure 6.7: An initial conflict scenario of a three UAVs with influence to higher altitude neighbouring UAV

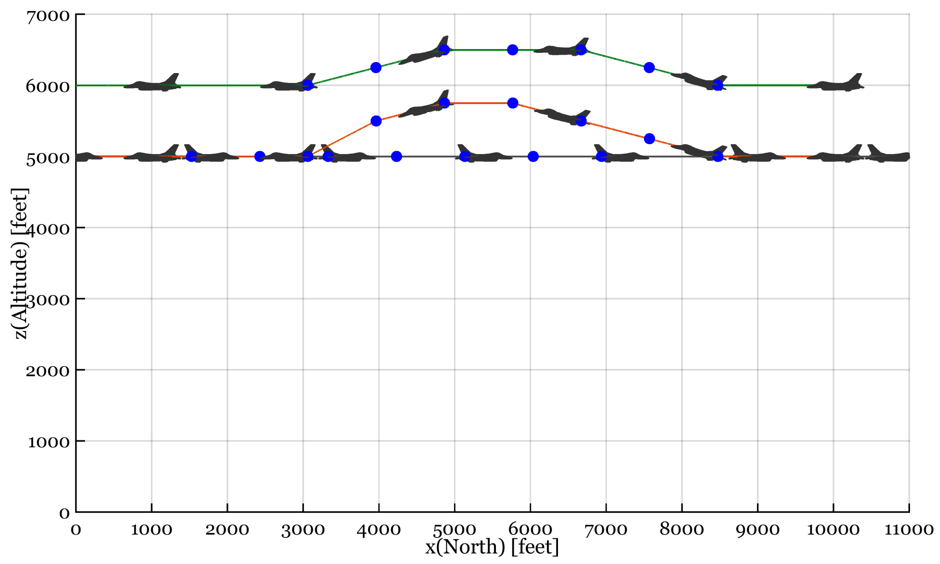


Figure 6.8: Cooperative conflict avoidance paths for the three UAVs with influence to higher altitude neighbouring UAV

The results of the path planning for conflict resolution are shown in Figure 6.8. The first token allocation maintains the altitude of the approaching UAV, however the conflict still exists. Thereafter, the head-on intruder with a parallel intruder UAV chooses to climb because the higher altitude UAV is not holding any token and was not part of the initial predicted conflict. Next, a new conflict between the higher altitude and the UAV which wishes to climb is predicted through state propagation and collision prediction. The search trees which the parallel UAVs construct are shown in Figure 6.9 and 6.11.

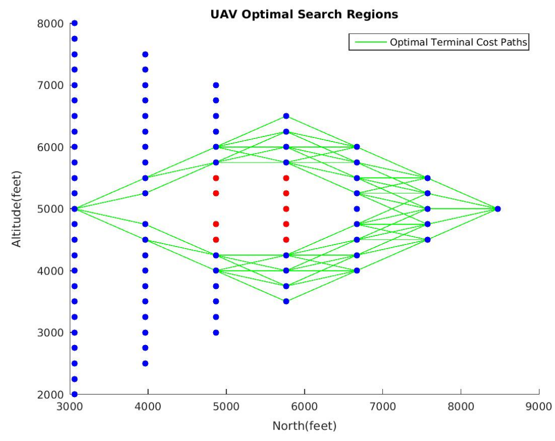


Figure 6.9: UAV 2 Search Tree

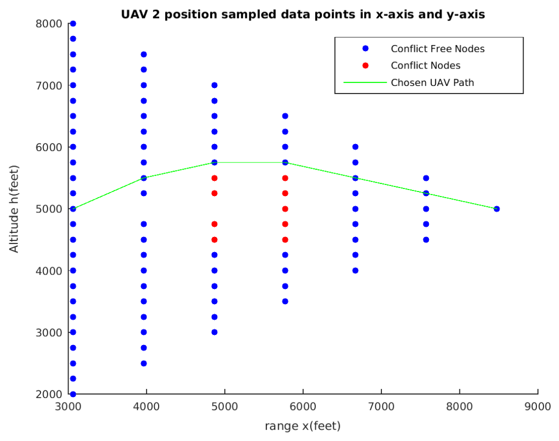


Figure 6.10: UAV 2 Optimal Path

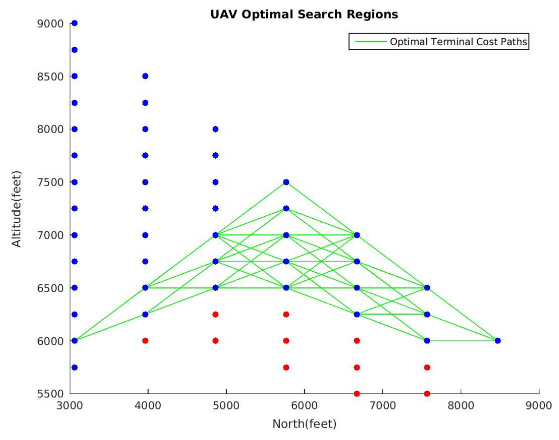


Figure 6.11: UAV 3 Search Tree

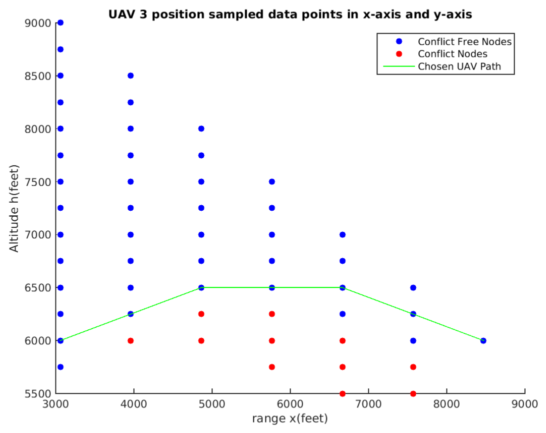


Figure 6.12: UAV 3 Optimal Path

The chosen path plans demonstrate how token allocation determines the priority between a UAV which is already holding a token and executing a collision avoidance manoeuvre and another UAV which is not holding a token. This is illustrated by the UAV approaching from the left receiving the first token, while the other two UAVs approaching from the right have no tokens. The UAV approaching from the left therefore ignores both UAVs approaching from the right, and maintains its altitude. The lower UAV approaching from the right is allocated the second token, because it has a predicted conflict, while the higher UAV approaching from the right does not have a predicted conflict yet. The lower UAV cannot ignore the UAV approaching from the left, because it is already holding a token, but it ignores the UAV above it, because it is not holding a token. The lower right UAV therefore performs a climbing manoeuvre to avoid the left UAV, as shown in Figure 6.10. The conflict avoidance action by the lower right UAV now causes a conflict to be predicted for the third UAV (the higher right UAV). The higher right UAV is then allocated the third token and must perform a collision avoidance manoeuvre. The higher right UAV cannot ignore the left UAV or the lower right UAV, because they are both already holding tokens. The higher right UAV therefore also performs a climbing manoeuvre to avoid the climbing lower right UAV below it, as shown in Figure 6.12. The altitude control inputs which were used by the parallel UAVs to avoid collisions are shown in Figure 6.13 and 6.14.

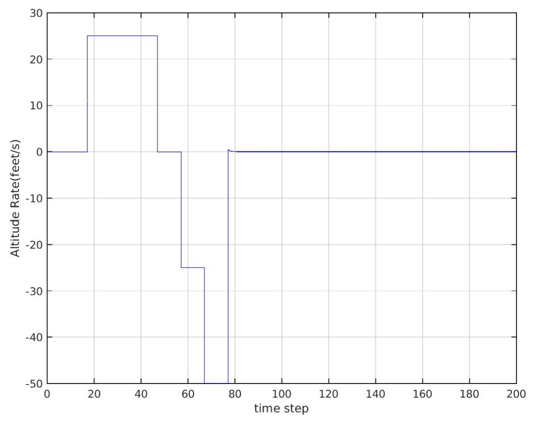


Figure 6.13: UAV 2 Altitude Inputs

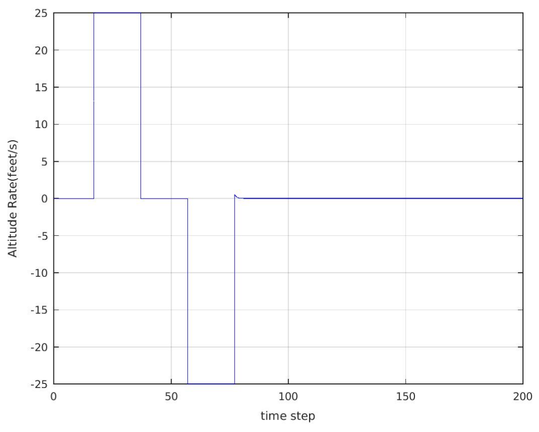


Figure 6.14: UAV 3 Altitude Inputs

6.4.2 Three UAVs : Equal separation altitude head-on conflict

The conflict scenario shown in Figure 6.15 illustrates three UAVs where two UAVs are flying in parallel but at a greater separation distance than the conflict detection thresholds. The two UAVs at parallel altitude paths are at an equal altitude separation with an approaching single UAV. Therefore, the single UAV which is approaching the two parallel UAVs from opposite direction wishes to pass between the parallel UAVs.

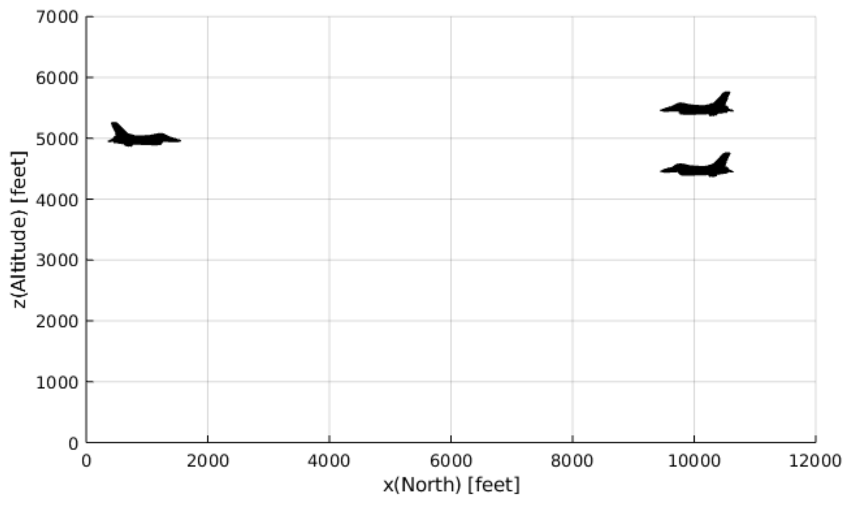


Figure 6.15: Initial conflict of three UAVs with equal altitude separation between two intruders

The conflict resolution for the three UAVs as shown in Figure 6.16 uses a token allocation to control the order of the sequential path planning. Since the single UAV is approaching other parallel UAVs at a equidistant altitude, the two parallel UAVs both have the same manoeuvre domain. Moreover, altitude deviation for the single approaching UAV is greater than that which is proposed by the parallel UAVs. Therefore, token allocation begins with maintaining the altitude of the single approaching UAV, because it is the first proposed optimal solution and because none of the parallel UAVs have planned a path yet.

The first token allocation chooses to maintain altitude, but it does not solve the multi-aircraft conflict, because the parallel UAVs are still in conflict with the approaching single UAV which is holding a token. Hence, the search-trees for the optimal path planning for the two parallel UAVs are shown in Figure 6.17 and 6.19. The second token is allocated to the tree in Figure 6.17, and the third token is allocated to the search tree Figure 6.19. Both search trees show the positions of the single UAV which passes between the parallel UAVs. The red nodes indicates the positions of the single UAV currently holding a token.

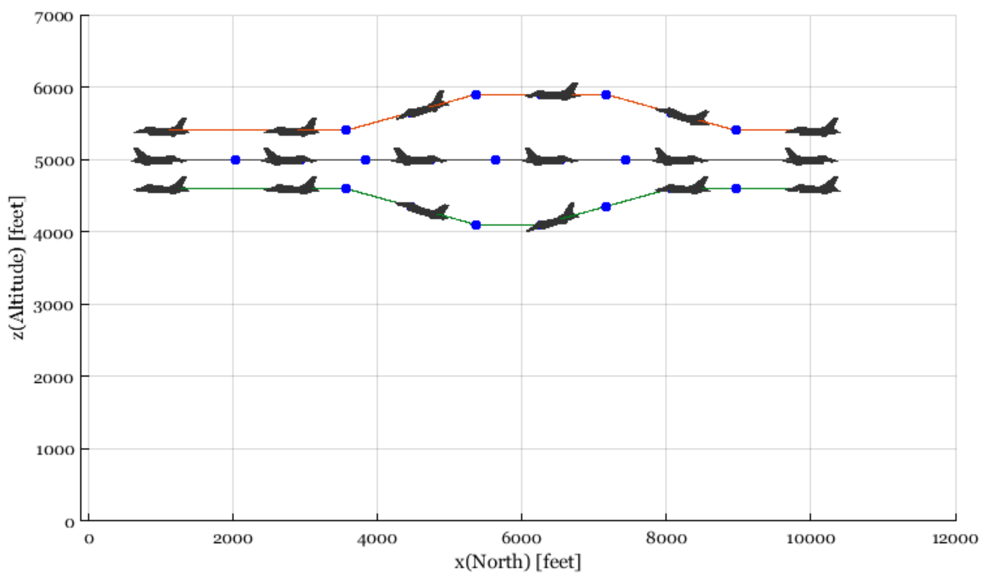


Figure 6.16: Final conflict resolution wih equal altitude separation between two UAVs and a host UAV

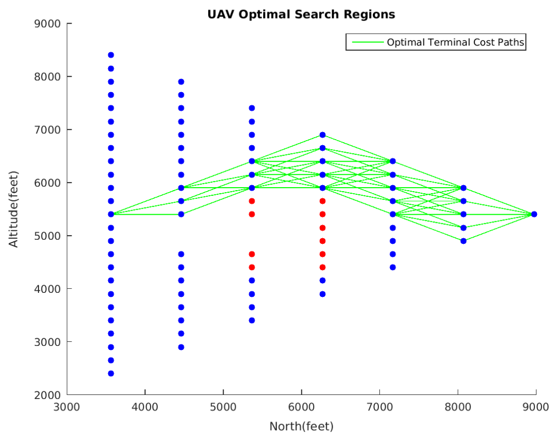


Figure 6.17: UAV 2 Search Tree

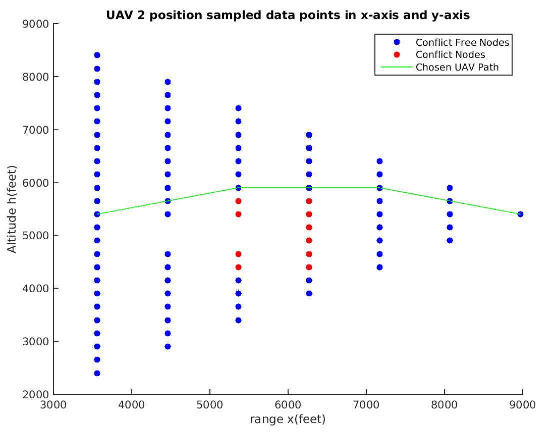


Figure 6.18: UAV 3 Optimal Path

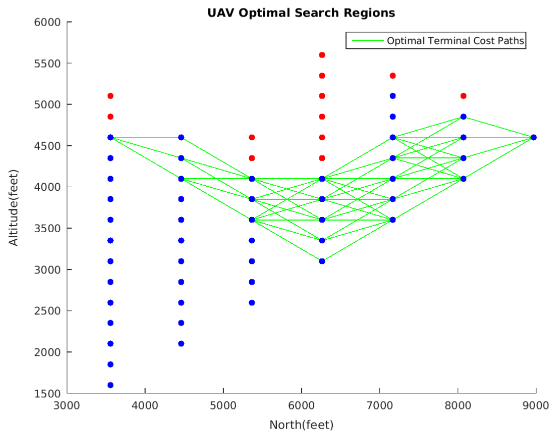


Figure 6.19: UAV 3 Search Tree

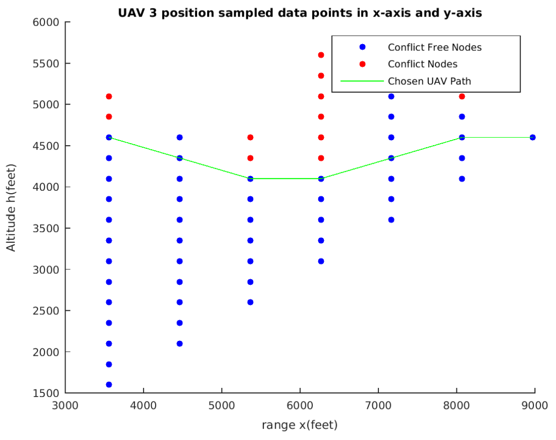


Figure 6.20: UAV Optimal Path

The chosen optimal collision avoidance paths for the two parallel UAVs are shown in Figure 6.18 and 6.20 respectively. The higher UAV from the right chooses a climbing manoeuvre, while the lower UAV from the right chooses a descending manoeuvre.

The cost of executing the climb and descend are equal for conflict resolution since the intruder UAV is at a equidistant altitude separation. The time execution of the altitude control commands of the pairwise UAVs are shown in Figure 6.21 and 6.22 respectively for the climbing and descending UAVs.

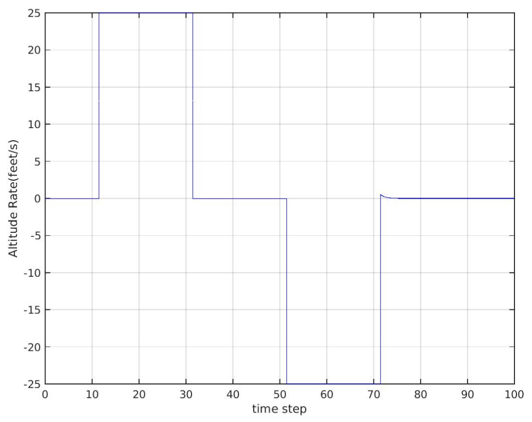


Figure 6.21: UAV 2 Altitude Commands

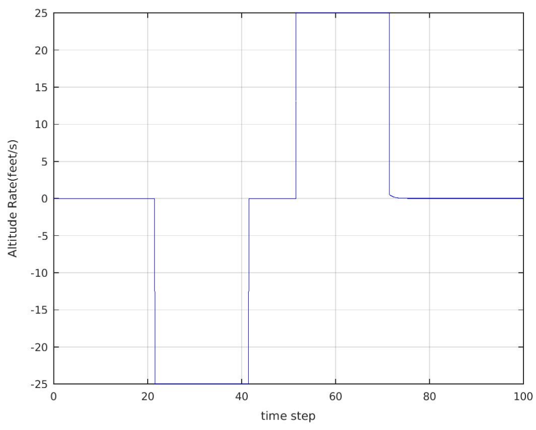


Figure 6.22: UAV 3 Altitude Commands

6.4.3 Three UAVs : Near Terrain Altitude

The following simulation shown in Figure 6.23 illustrates an initial conflict scenario that involves of three UAV where two are on a head-on collision path. All three UAVs are near the minimum terrain altitude of 1000 feet. The two UAVs approaching from the left UAVs are staggered horizontally, and a single UAV is approaching from the right direction. All three UAVs are near the 1000 feet minimum altitude which is near terrain obstacles. Therefore, a descending manoeuvres for collision avoidance will not be allowed and will therefore not be added to the search tree.

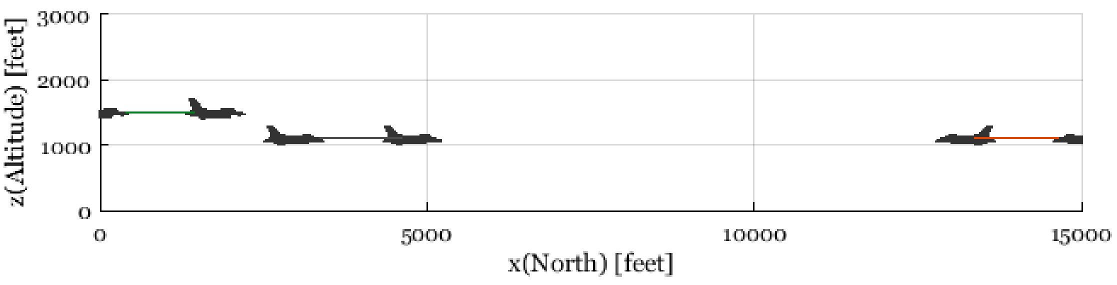


Figure 6.23: Initial conflict scenario for three UAVs near the minimum altitude

The results of path planning for the three UAVs are shown in Figure 6.24 with none of the UAVs allowed to descend. The first conflict is detected between the two head-on UAVs at 1100 feet. The head-on UAVs have limited manoeuvre space, and also neither of them are allowed to descend for conflict resolution. Therefore, the first token is allocated to a UAV which maintains its nominal altitude because when the conflict started neither of the head-on UAVs had received a token for path planning. The search tree created for the head-on UAV approaching from the left is shown in Figure 6.25 and the chosen optimal collision avoidance path is shown in Figure 6.26.

After the token allocation of the first token which maintains altitude of the receiving UAV the original head-on conflict still exists. Thereafter, a second token is allocated to the UAV from the right direction; it chooses to climb as shown in Figure 6.27. The action of the path shown in Figure 6.28 after the second token leads to a new conflict with the third UAV at 1500 feet travelling from the left to the right. A third token is then allocated to the UAV at 1500 feet which creates the search tree shown in Figure 6.29 for a UAV which chooses to climb. Finally, the climb action by the third token which is shown in Figure 6.30 is executed to resolve the conflict.

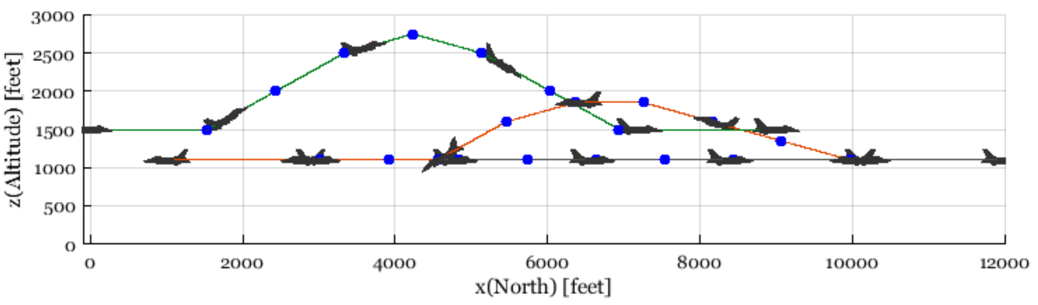


Figure 6.24: Final steps conflict scenario of Three UAVs conflict manoeuvre near minimum altitude

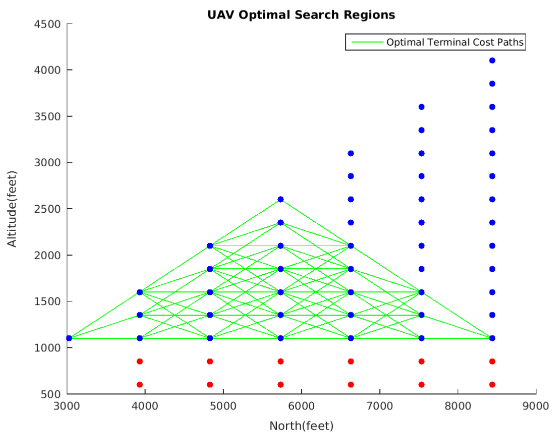


Figure 6.25: UAV 1 Search Tree

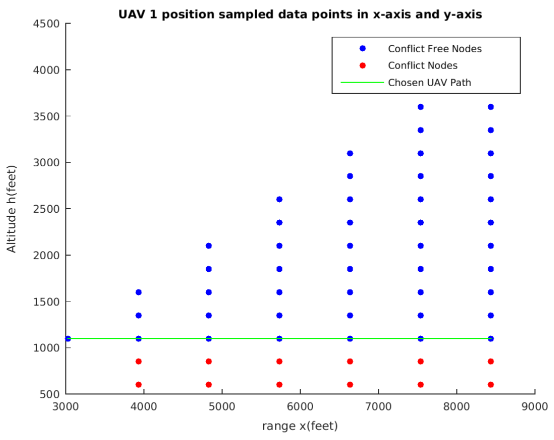


Figure 6.26: UAV 1 Optimal Path

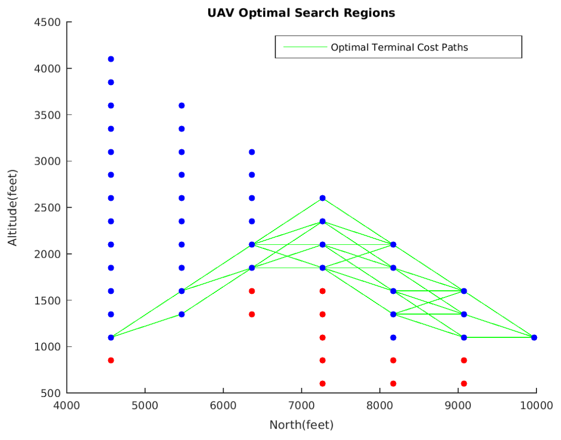


Figure 6.27: UAV 2 Search Tree

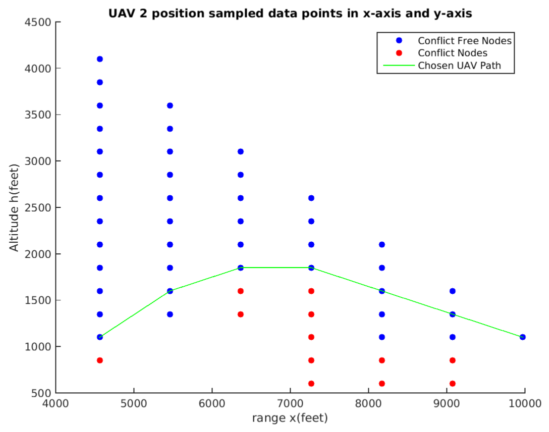


Figure 6.28: UAV 2 Optimal Path

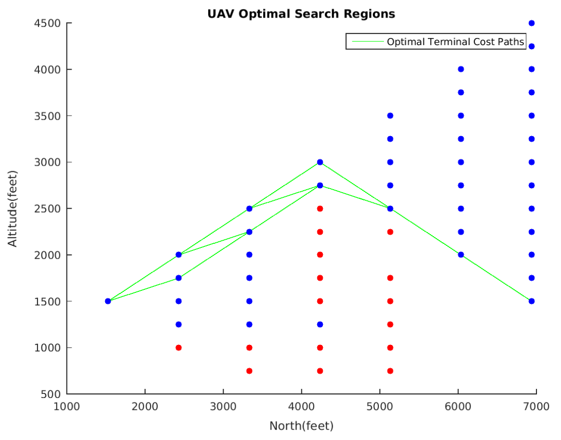


Figure 6.29: UAV 3 Search Tree

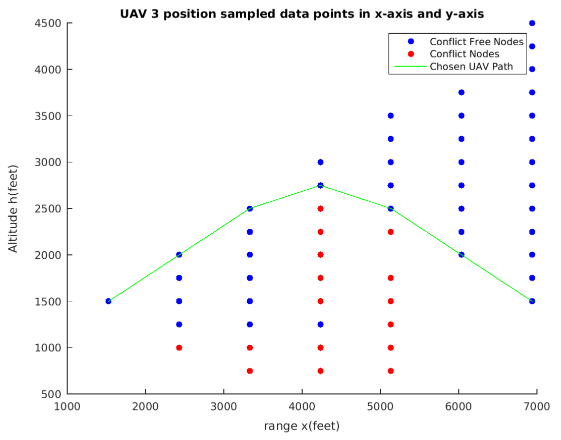


Figure 6.30: UAV 3 Optimal Path

The optimal altitude commands to the UAVs which deviates from their nominal altitude are shown in Figure 6.31 and Figure 6.32.

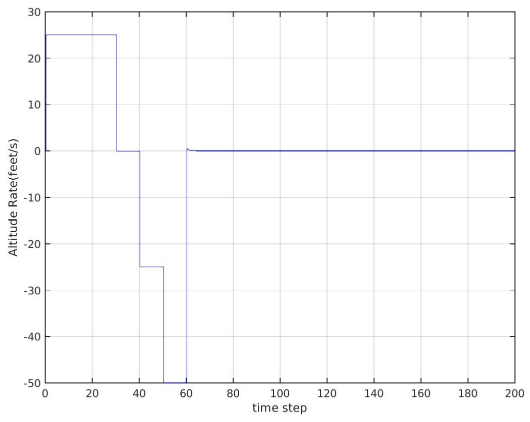


Figure 6.31: UAV 2 Altitude Inputs

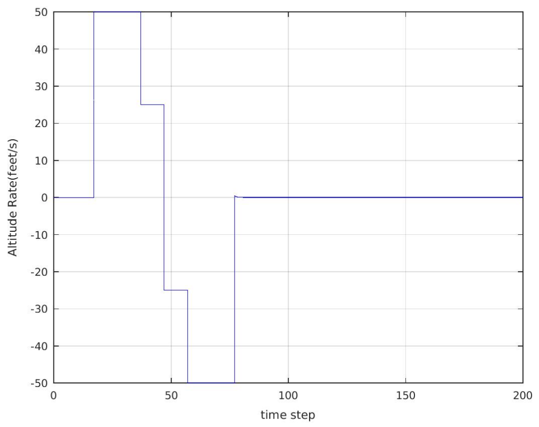


Figure 6.32: UAV 3 Altitude Inputs

6.4.4 Five UAVs : Staggered head-on conflict

The following simulation is an example of a conflict scenario representing a five UAVs representing a cluttered airspace, with two pairs of UAVs flying in Figure 6.33. The UAVs in the conflict scenario are horizontally staggered, with two pairs of UAVs flying parallel in altitude and the three closes UAVs already having predicted conflicts. In this conflict resolution scenario, there are examples where the token allocation algorithm determines different priorities for the sequential planning of the UAVs in a cluttered environment.

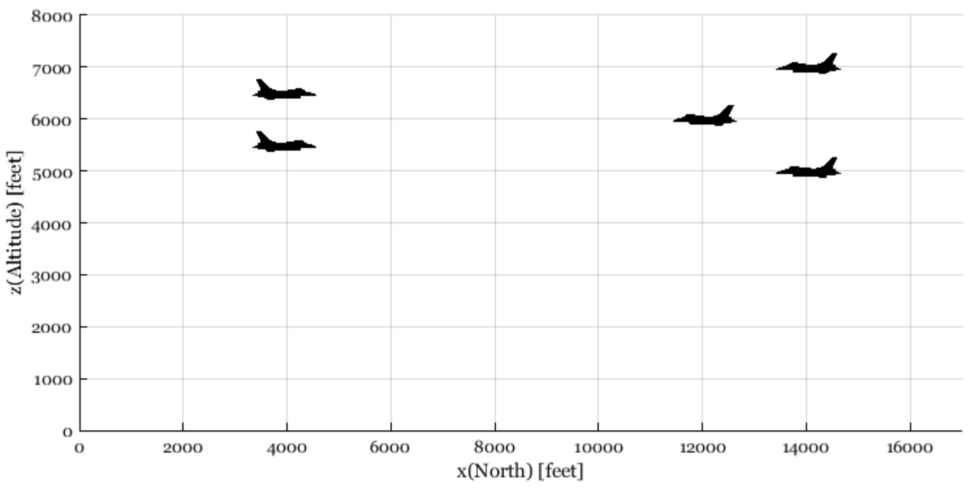


Figure 6.33: Initial states of five UAVs conflict scenario with staggered horizontal positions

The results of path planning and token allocation for the sequential planning of these multiple plans are shown in Figure 6.34. The token allocation begins with the UAV with most conflict nodes in its search-tree , which means that it has the least space to manoeuvre. The UAV with the most conflict nodes is the one initialised at $(6000ft, 12000ft)$ because it is approaching the two parallel UAVs initialised at $(4000ft, 5500ft)$ and $(64000ft, 5000ft)$.

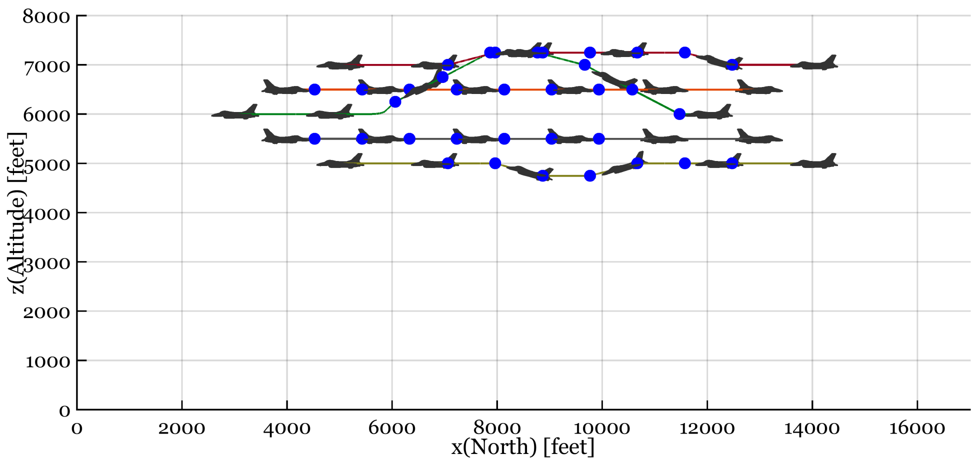


Figure 6.34: Conflict resolution of five UAVs with staggered horizontal positions

The two parallel UAVs present a similar conflict scenario as in Section 6.4.2 where a single UAV is at an altitude equidistant between two UAVs. However, in this case the two parallel UAVs approaching from the left were found to have more limited manoeuvre space than the two parallel UAVs approaching from the right, therefore both UAVs approaching from the left receive the first two tokens and both maintain their nominal altitudes. The third token for the multiple UAVs conflict is to the UAV initialised at (6000 feet,12000 feet). Its search tree which is shown in Figure 6.35 and its chosen optimal path is shown in Figure 6.36. The last two tokens are allocated to the parallel UAVs initialised at (14000 feet, feet) and (14000 feet, 7000 feet) which deviate for the approaching parallel UAVs. The search trees for their path plans are shown in Figure 6.37 and 6.39 with their respective chosen optimal paths are shown in Figure 6.38 and 6.40.

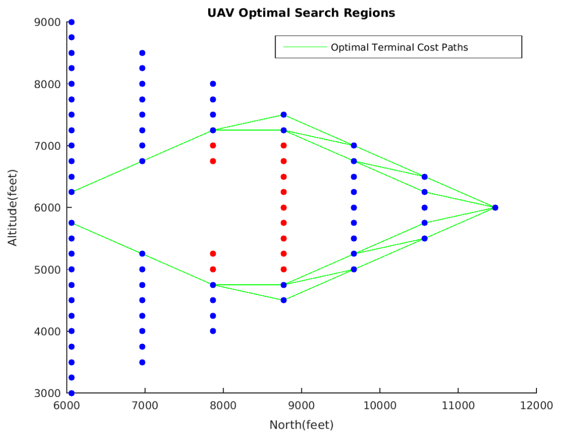


Figure 6.35: UAV 3 Search Tree

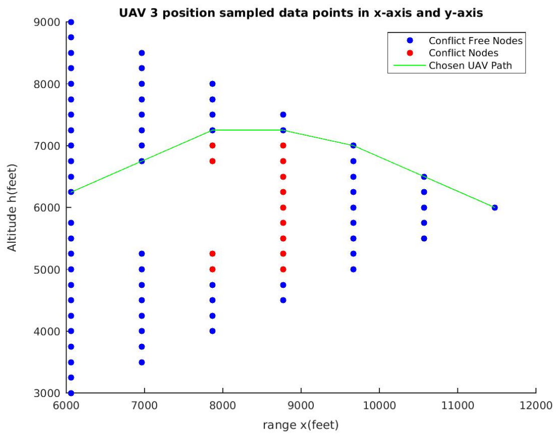


Figure 6.36: UAV 3 Optimal Path

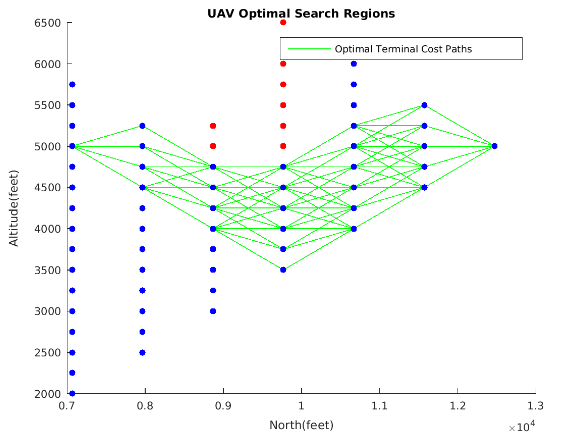


Figure 6.37: UAV 4 Search Tree

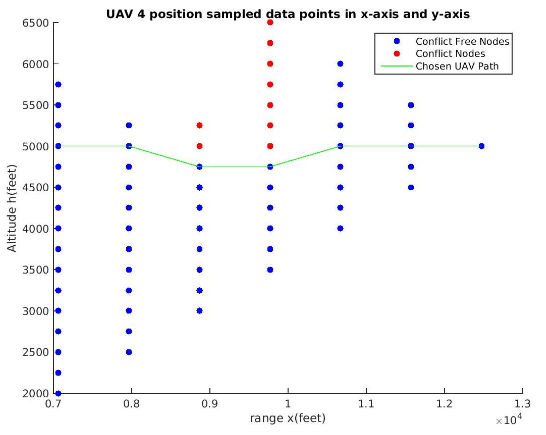


Figure 6.38: UAV 4 Optimal Path

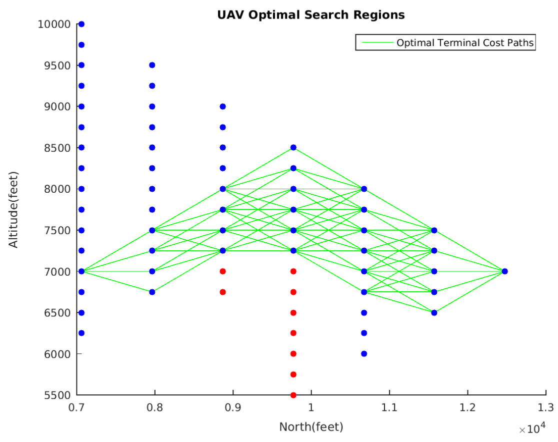


Figure 6.39: UAV 5 Search Tree

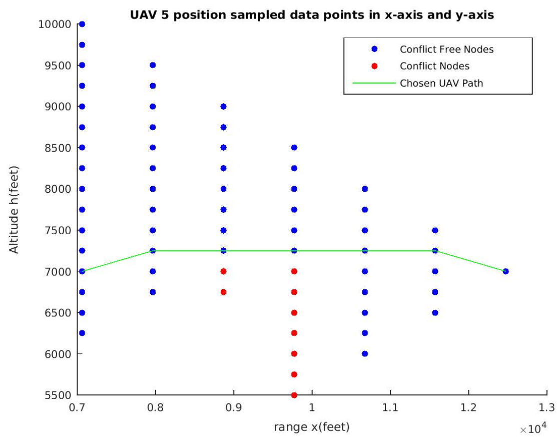


Figure 6.40: UAV 5 Optimal Path

6.5 Summary and Conclusion

In this Chapter a token allocation algorithm for cooperative path planning of multiple UAVs was presented for analysis. The path planning algorithm proposed in Chapter 5 for searching of collision-free trajectories was extended here to include multiple UAVs. A token allocation algorithm was used to determine the priority order for the sequential path planning for a group of UAVs. To achieve the priority order for the sequential path planning, simulations of path planning inputs for collision prediction was presented in Section 6.2. Thereafter, a token allocation is used as a global path planner for the maximization of the separation safety for multiple UAVs. The path planning executed is from an individual UAV which guarantees that a UAV will execute a path plan for 60 seconds. This reduces uncertainty of planning for other UAVs during conflict prediction. Moreover, any token receiving UAV will furthermore optimize other global objectives as implemented in Section 6.3. Thereafter, a UAV is selected for token allocation and next, the algorithm given in Section 6.3.3 is executed until completion for fixed length full path plans. Our simulations involving conflict encounters of multiple UAVs have analysed search-tree construction in Section 6.4 for the optimal path and token allocation for execution sequence of the path plans. Additionally, with this approach we were able to combine path planning and token allocation for execution order of collision risk priority through UAVs with least manoeuvre space in its search tree. Furthermore, we have learned that token allocation is useful for multiple UAVs collision avoidance through our path planning metrics. Therefore, the metrics used can order UAVs with the most manoeuvrable space to find conflict resolutions in the near future to minimise the cost of collision avoidance. Finally, it also became clear that the limitations of token allocation were inherited from the path planning algorithm which proposes conflict-free trajectories. However these techniques cannot be seen as complete collision avoidance algorithms on their own.

Chapter 7

Monte Carlo Simulations

In this Chapter we present the statistical Monte-Carlo simulation results for the Rules-Based and Cooperative Path Planning algorithms which we developed in Chapter 4 and 6 respectively. Monte Carlo simulations are useful for the estimation of the conflict performance metrics of both the rules-based and the path planning algorithms for conflict resolution of multiple UAVs. In Section 7.1 we describe the setup for the Monte Carlo simulations of multiple UAVs with nominal altitudes which are randomly sampled from a uniform distribution function. The altitude distributions have a constant nominal altitude as its mean; additionally the horizontal rates are initialised at the beginning of each simulation. Thereafter in Section 7.2, a list of performance metrics are outlined for long-term conflicts such as the collision avoidance success rate, minimum separation, trajectory deviation cost, and control effort cost. The statistical results of the Monte Carlo simulations for the distribution estimation of the performance metrics are discussed in Section 7.3. In summary, this chapter will analyse the conflict resolution performance of the methods given in Chapter 4 and Chapter 5 respectively to illustrate how our system performs in random complex conflict scenarios.

7.1 Monte Carlo Setup

For the Monte Carlo simulations, the UAV positions are initialised with nominal fixed altitudes $h_1, h_2, \dots h_n$ which can be equal but must be below the vertical threshold. The altitude separations between UAVs are initialised in such a way that the vertical separation between two or more $h_1, h_2, \dots h_n$ are less than 650 feet to guarantee a conflict during simulation in the near future. The altitude of each UAV will be randomly sampled from a uniform distribution $\mathcal{U}(h_{min}, h_{max})$ with $h_{min} = h_{nominal} - 650ft$ and $h_{max} = h_{nominal} + 650ft$ as demonstrated in Figure 7.1. The simulation setup will enable measurement performance metrics analysis for the rules-based and path planning based collision avoidance algorithms. The rules-based Monte Carlo simulation will the evaluate statistical data of the collision avoidance success rate, the minimum separation at closest point of approach, the altitude deviations from nominal altitude, and the altitude rate as the control effort.

The distribution of the altitude sampling space for each UAV is stated as follows:

$$f(h) = \begin{cases} \frac{1}{h_{max}-h_{min}} & \text{if } h_{min} \leq h \leq h_{max} \\ 0 & \text{Otherwise} \end{cases} \tag{7.1}$$

$$\bar{h} = h_{nominal} = \frac{1}{2}(h_{min} + h_{max}) \tag{7.2}$$

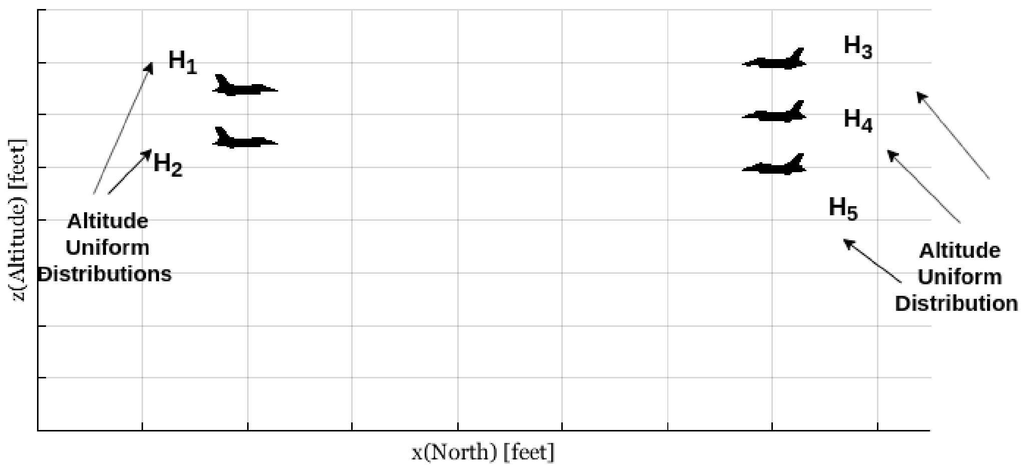


Figure 7.1: Monte-Carlo simulations setup of five UAVs for the rules-based collision avoidance algorithm

The setup of each simulation for a cooperative path planning algorithm is illustrated in Figure 7.2. The Monte Carlo simulation will provide statistical results for a multi-conflict resolution using the path planning algorithm in where the latter implements the token allocation strategy discussed in Section 6.3. The number of UAVs in the simulation environment will be ten with a uniform altitude distributed from the mean nominal altitude. Therefore, the UAVs will be simulated from altitudes $h_1, h_2, \ldots h_{10}$ which can be equal, but will be distributed below a vertical threshold of 650 feet from each other to guarantee a conflict in the near future.

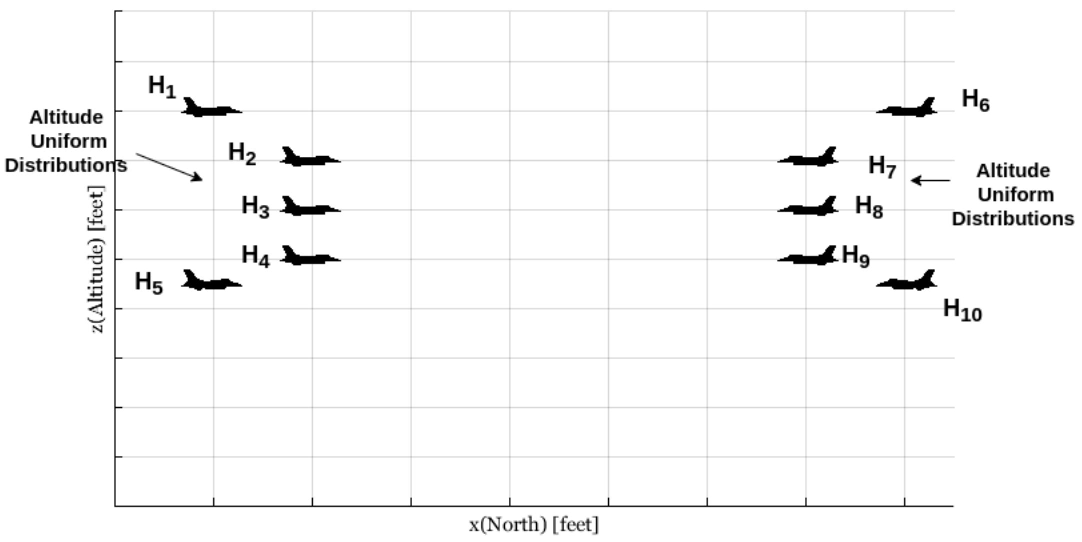

Altitude Uniform Distributions
H1
H2
H3
H4
H5
H6
H7
H8
H9
H10
Altitude Uniform Distributions
z(Altitude) [feet]
x(North) [feet]

Figure 7.2: Monte-Carlo simulations setup of ten UAVs for the path planning collision avoidance algorithm

7.2 Performance Metrics

The following section discusses the performance metrics that will be used for the evaluation of the rules-based and path planning based collision avoidance algorithms. The performance metrics include collision avoidance success rate, the minimum separation at closest point of approach, the altitude deviations from nominal altitude, and the altitude rate as the control effort.

7.2.1 Collision Avoidance Success Rate

The failure rates of the two rules-based algorithms are determined by checking the total number of times that the minimum safe separation distance of 1000 feet is not maintained, and expressing the total number of failures as a percentage of the total number of simulations. The "number of near encounters" is the minimum separation distances between UAVs. The success rate formula of the CIF and RAS is calculated as follows:

$$\text{Success rate} = \frac{\text{number of near encounters - number of NMAC encounters}}{\text{number of near encounters}} \times 100 \quad (7.3)$$

The failure rate of the path planning based algorithm is determined by checking the number of times that conflict-free paths could not be planned for all of the UAVs, and expressing the total number of failures as a percentage of the total number of simulations. The success rate formula of path planning of multi-UAV conflict resolution is given as follows:

$$\text{Success rate} = \frac{\text{number of planned paths - number of non-flyable paths}}{\text{number of planned paths}} \times 100 \qquad (7.4)$$

7.2.2 Minimum Separation

The performance of the collision avoidance algorithms to maintain safe separation will be evaluated by analysing the statistical distribution of the minimum separation distances at the closest point of approach, the minimum vertical separation when there is insufficient horizontal separation, and the minimum horizontal separation when there is insufficient vertical separation for all three algorithms (RAS, CIF, and path planning).

For all three these variables (minimum distance, minimum time to collision, and minimum vertical separation) the statistical distributions will be plotted using histograms and the box and whiskers plots will be compared.

For all three these variables (minimum distance, minimum time to collision, and minimum vertical separation) the statistical distributions will be plotted using histograms and the box and whiskers plots will be compared.

7.2.3 Maximum Altitude Deviation

The performance of the collision avoidance algorithms to minimise the deviations of the UAVs from their original flight path will be evaluated by analysing the statistical distribution of the maximum altitude deviations of the individual UAVs for all three algorithms (RAS, CIF, and path planning). The statistical distributions of the maximum altitude deviation will be plotted using histograms for all three algorithms. The box and whiskers plots of the maximum altitude deviations will also be plotted for all three algorithms and compared.

7.2.4 Altitude Rate Commands

The control effort used by the collision avoidance algorithms will be evaluated by analysing the statistical distribution of the altitude rate commands of the individual UAVs for all three algorithms (RAS, CIF, and path planning). The statistical distributions of the altitude rate commands will be plotted using histograms for all three algorithms. The box and whiskers plots of the altitude rate commands will also be plotted for all three algorithms and compared.

7.3 Monte Carlo Results

The following section presents simulation results for multiple UAVs where the altitude of each UAV will be randomly normal distributed between $h \in (h_{nominal} - 650, h_{nominal} + 650)$

feet from their initial altitude. There will be an analysis of Monte Carlo simulation success rates for maintaining the safe separation between UAVs. The results will be shown for five UAVs for rules-based algorithm, and ten UAVs for the path planning algorithm. Thereafter, the simulation results for the minimum separation metrics will be presented and analysed. The time to collision estimation provides an indication of the responses to multiple conflicts. Furthermore, the simulation results will analyse the statistical distribution of the trajectory deviations of all UAVs for conflict resolutions. Finally, the evaluation will include an analysis of the control efforts (altitude rates) of the altitude deviations during conflict resolution.

7.3.1 Collision Avoidance Success Rate

The path planning evaluation of multiple UAVs is performed after each token allocation. Below in Table 7.1 the distribution of the number of successful and non-complete path planning iterations of UAVs in a conflict for the simulation of Monte Carlo simulations. The median of replanning success for a given number of UAVs is about 33 percent, equal to 25 percent of all data. The observation from the path planning simulation results is that the minimum success rate is 20 %, and median of 33.5 % of all UAVs were able to generate path plans. Additionally, the simulation results of rules-based are also shown in Table 7.1. The success rates of the two rules-based algorithms are the ability to maintain the separation distance for encounters, with NMAC distance less than 1000 feet.

Algorithm	**Samples #**	**Minimum**	**Median**	**Success Rate**
RAS	225444	531.32 feet	4302.45 feet	87.97 %
CIF	530395	515.12 feet	2542.62 feet	95.13%
Path Planning	511	20 %	33.5 %	40.25 %

Table 7.1: Success rate of conflict resolution for search-tree path planning and rules-based algorithms (NMAC = 1000 feet)

7.3.2 Minimum Separation

The following includes the analysis of the minimum separation metrics for the two rules-based methods, known as RAS and pairwise CIF, and the path planning algorithm. Because all algorithms were simulated with the same conflict resolution objectives, the separation distances, altitude separation and horizontal separation distance during conflict resolution are analysed.

This will be followed with is a discussion of the control efforts of five UAVs for rules based algorithm and ten UAVs for path planning, used during conflict resolution. The control efforts applied during conflict resolution leads to different separation distance and time to collisions. The summary (box and whiskers) and density distribution plots are analysed below.

7.3.2.1 Minimum Separation Distances

The separation distances of the rules-based conflict resolutions are not allowed to get below both the horizontal and vertical conflict thresholds at the same time. A summary of the least encountered separation distances of the rules-based algorithms is found in Figure 7.3. The CIF method responds to nearest distance intruders first, thus the closest encounter separation distance between pairwise UAVs were determined to be a minimum of 691 feet with a median of 2550 feet. However, the response of the RAS separation distance had a minimum of 607 feet with a median of 4296 feet. A summary of the separation distances for path the planning algorithm are given in Figure 7.4.

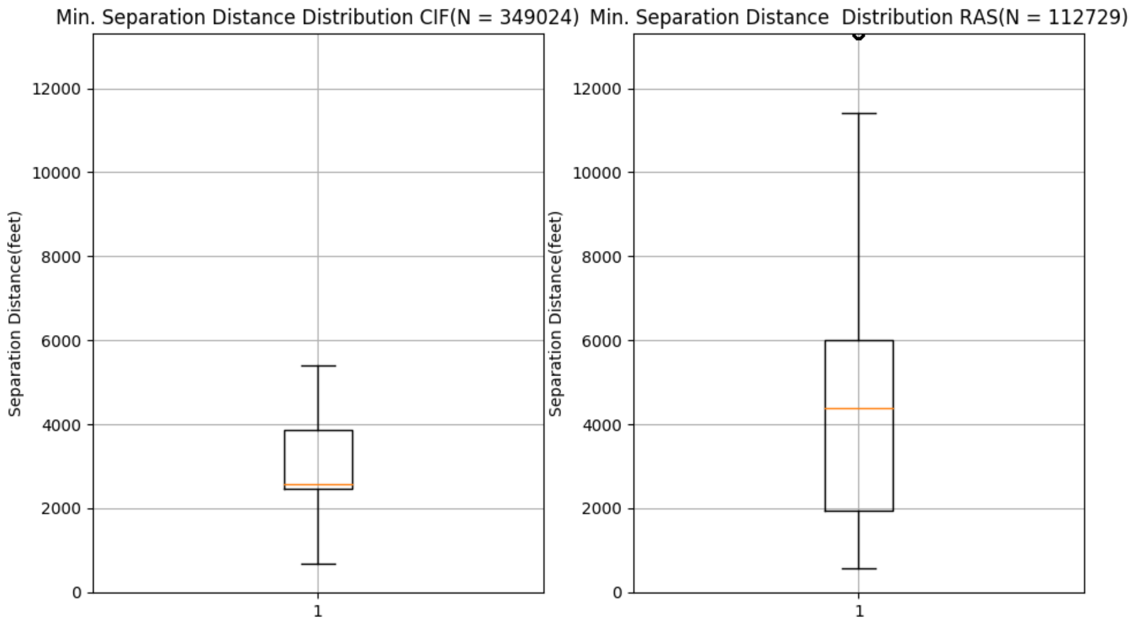


Figure 7.3: Summary of separation distances of rules-based algorithm for five UAVs

The density distribution of the separation distances commonly encountered during conflict for rules-based algorithms are shown in Figure 7.5 and 7.6. The RAS minimises the separation distances to 0 feet, however the distributions show that the CIF algorithms respond early to intruders between 2000 feet and 3000 feet. The density of the minimum separation distance is shown in Figure 7.7.

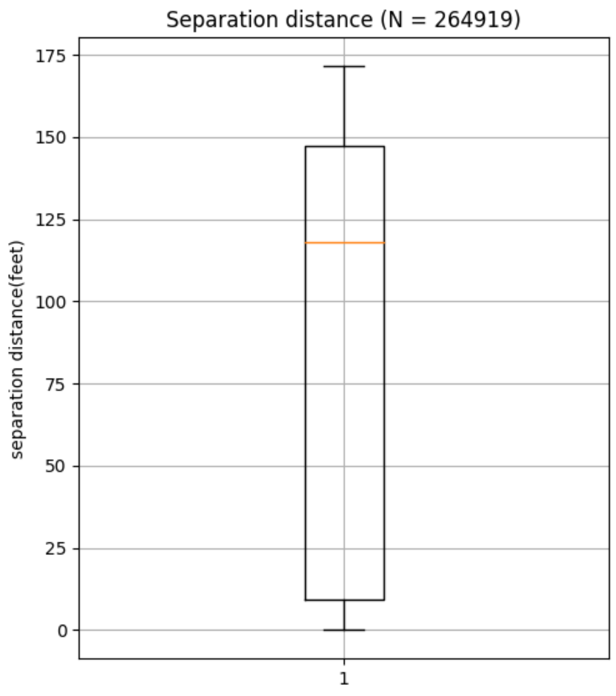


Figure 7.4: Summary of separation distances of path planning algorithm for ten UAVs

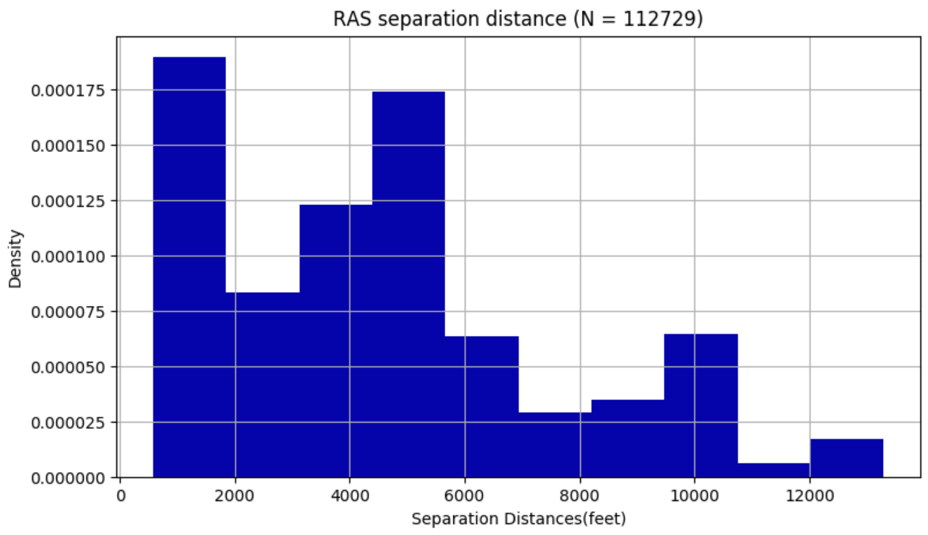


Figure 7.5: Distribution of separation distances for RAS algorithm

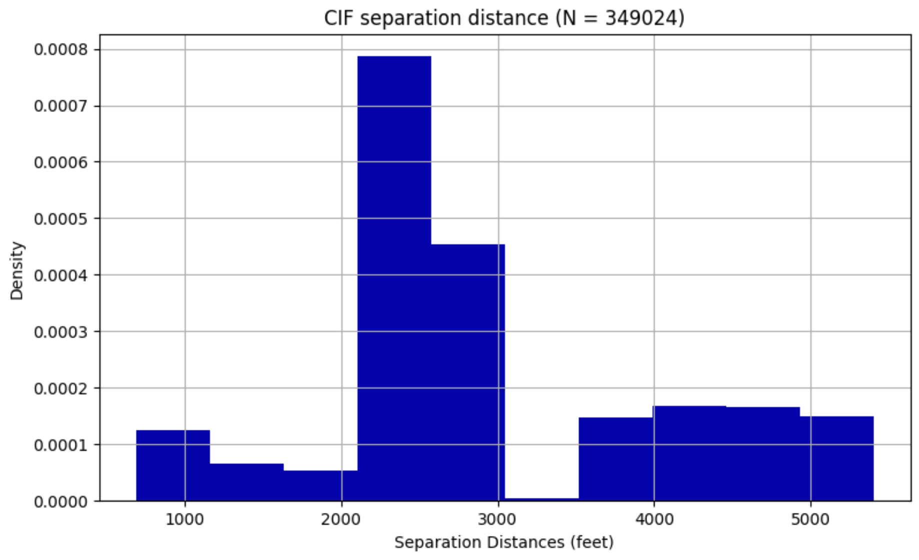


Figure 7.6: Distribution of minimum separation distances for CIF algorithm

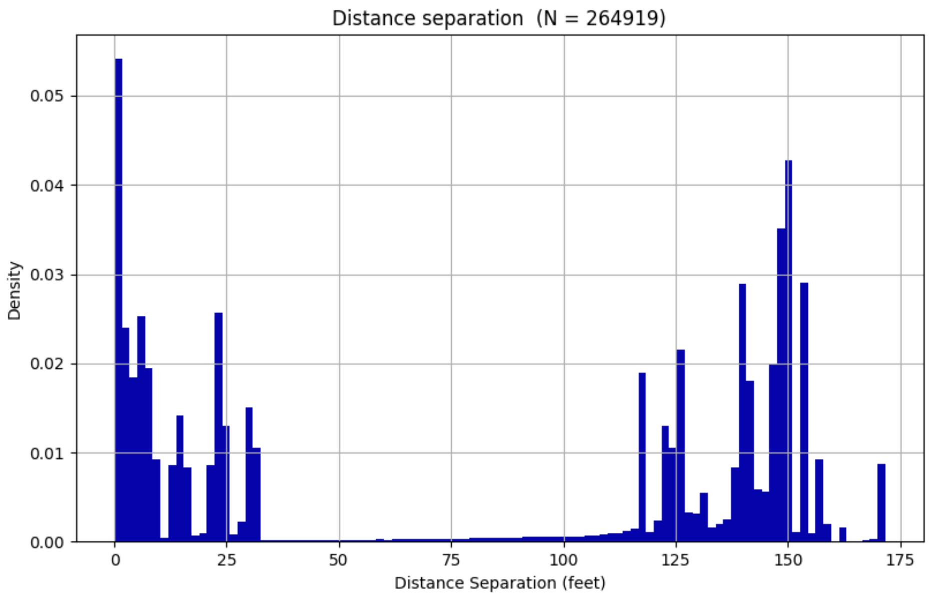


Figure 7.7: Distribution of minimum separation distances for path planning algorithm

The summaries of the distributions of the minimum vertical separation distances between the simulated conflict encounters are presented in Figure 7.8 and 7.9 for rules-based and path planning respectively.

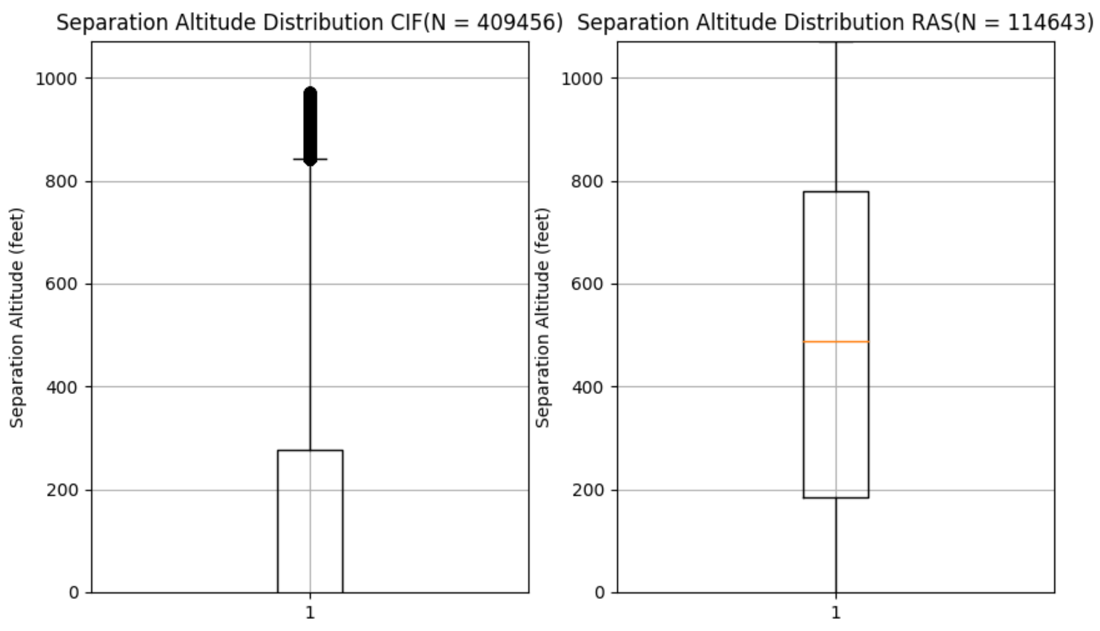


Figure 7.8: Summary of minimum altitude separation distances for rules-based algorithms

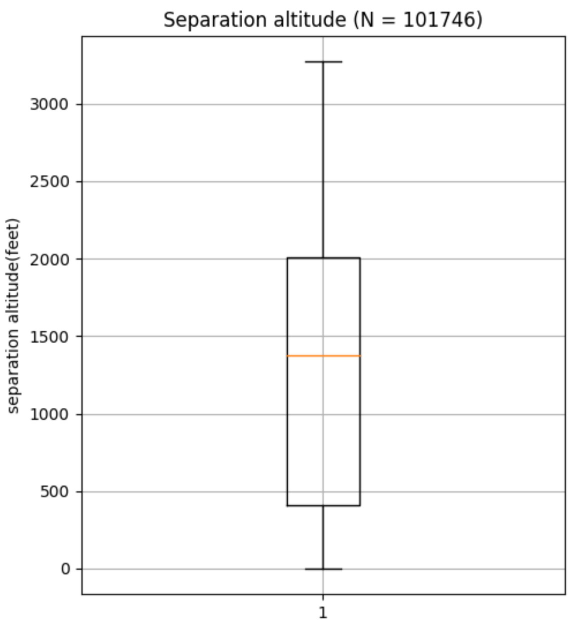


Figure 7.9: Summary of minimum altitude separation distances for path planning algorithm

The distribution densities of the minimum vertical separation distances of all UAVs during conflict resolution are shown in Figure 7.10 and 7.11 for the RAS and CIF algorithms. The path planning algorithm distribution density as shown in Figure 7.12.

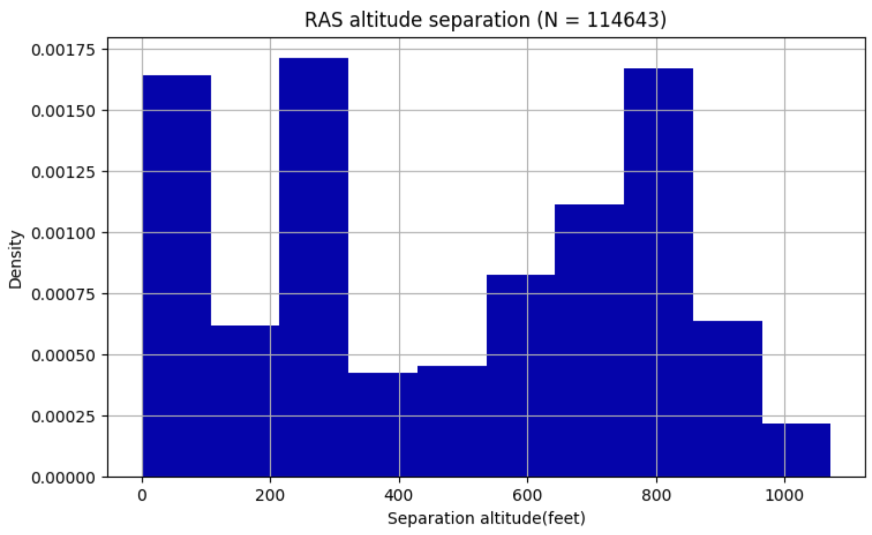


Figure 7.10: Distribution of minimum altitude separation distance RAS algorithm

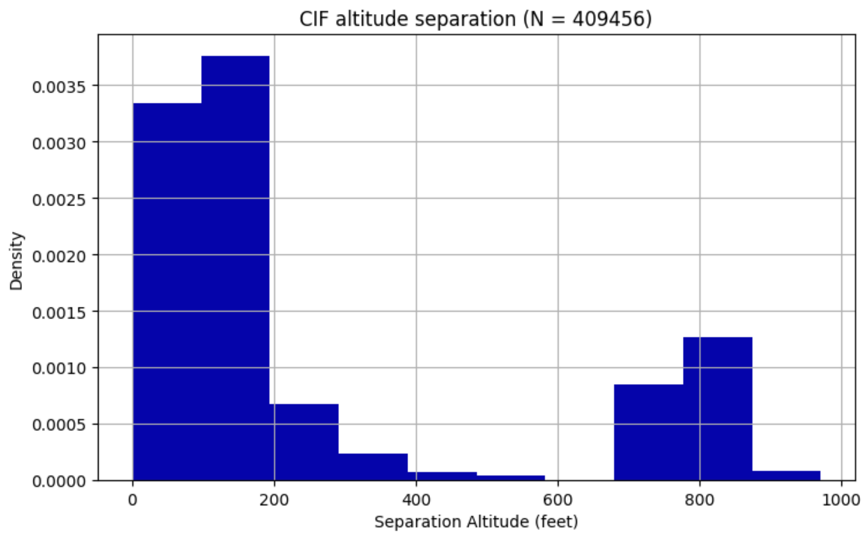


Figure 7.11: Distribution of minimum altitude separation distance CIF algorithm

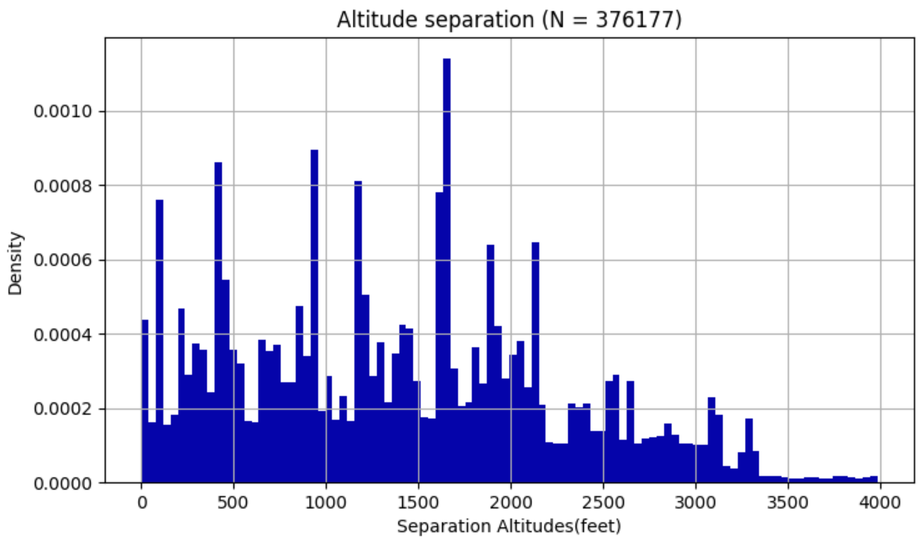


Figure 7.12: Distribution of minimum altitude separation distance path planning algorithm

The summaries of the minimum horizontal separation distances of the rules-based approach are presented in Figure 7.13 and the summary for the path planning algorithm in Figure 7.14. In all encountered horizontal separation distances of CIF approach, none are greater than 6000 feet. However, the 75th percentile of the minimum horizontal separation distance is approximately 6000 feet. The median of the minimum separation distance for the path planning is between 2000 and 3000 feet.

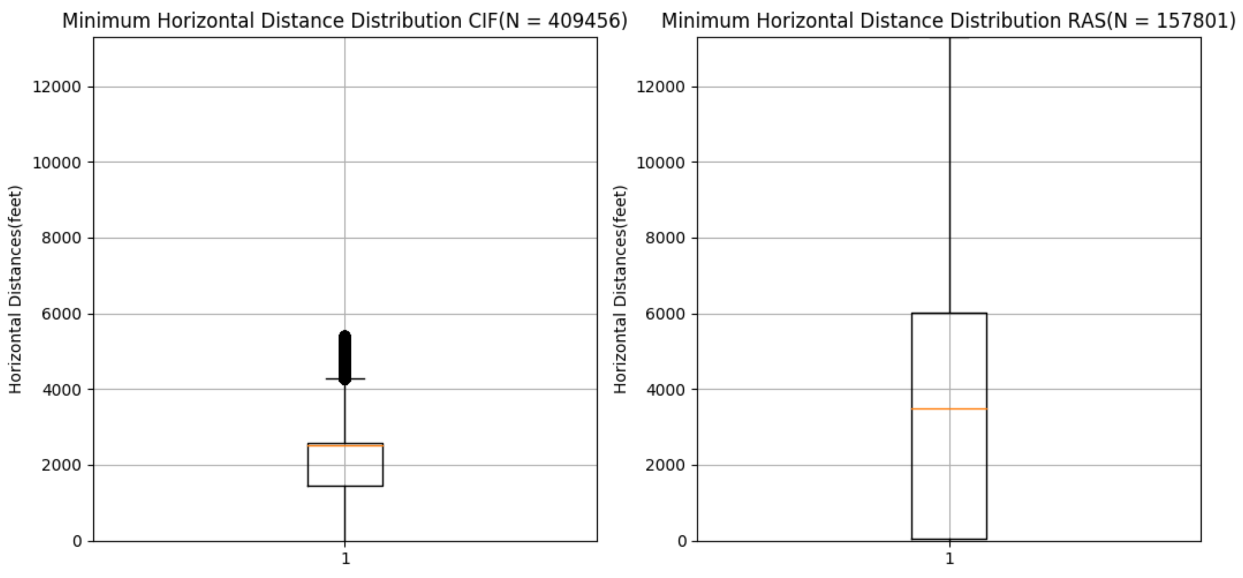


Figure 7.13: Summary of minimum horizontal separation distances for rules-based algorithms

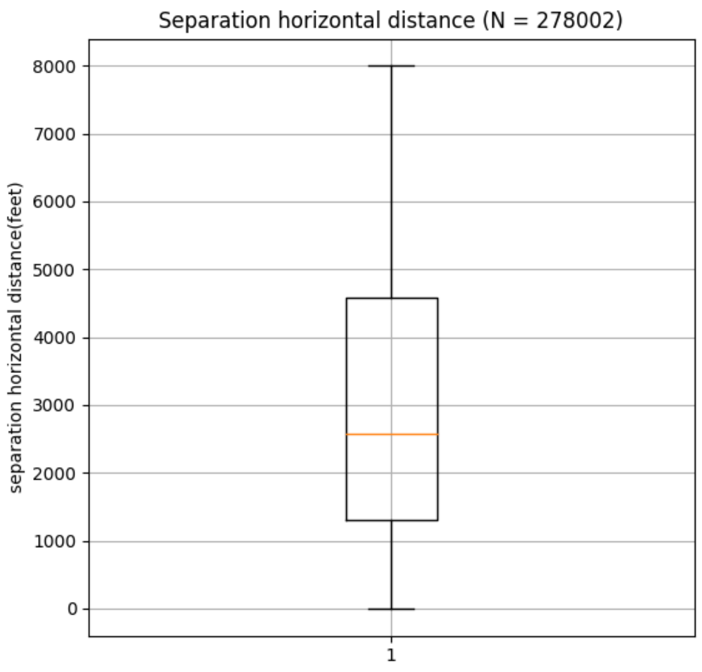


Figure 7.14: Summary of minimum horizontal separation distances for path planning algorithm

The density distribution of the minimum horizontal separation distances for rules-based are presented in Figure 7.15 and 7.16. The results of the two rules-based algorithms illustrates the horizontal threshold, with RAS near 5000 feet, and CIF near 2500 feet. The path planning distribution of the UAVs minimum horizontal separation distances is shown in Figure 7.17.

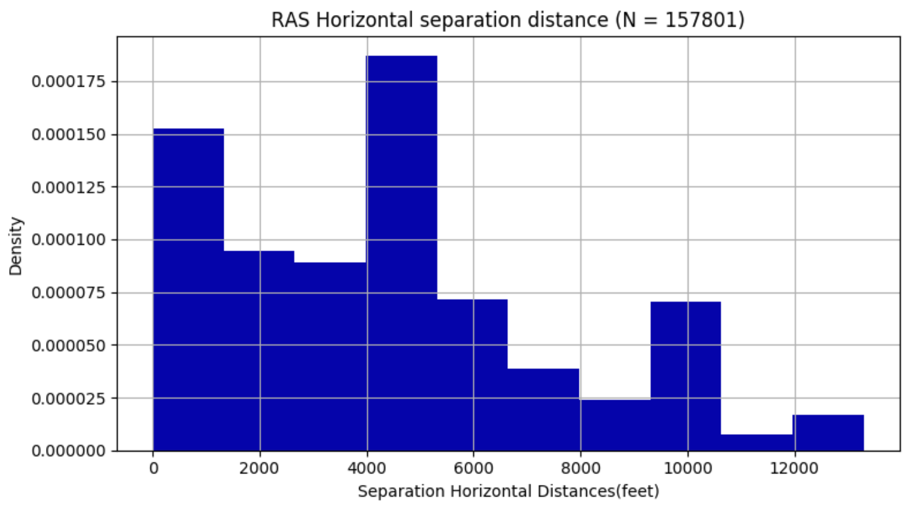


Figure 7.15: Distribution of minimum horizontal separation distances of RAS algorithm

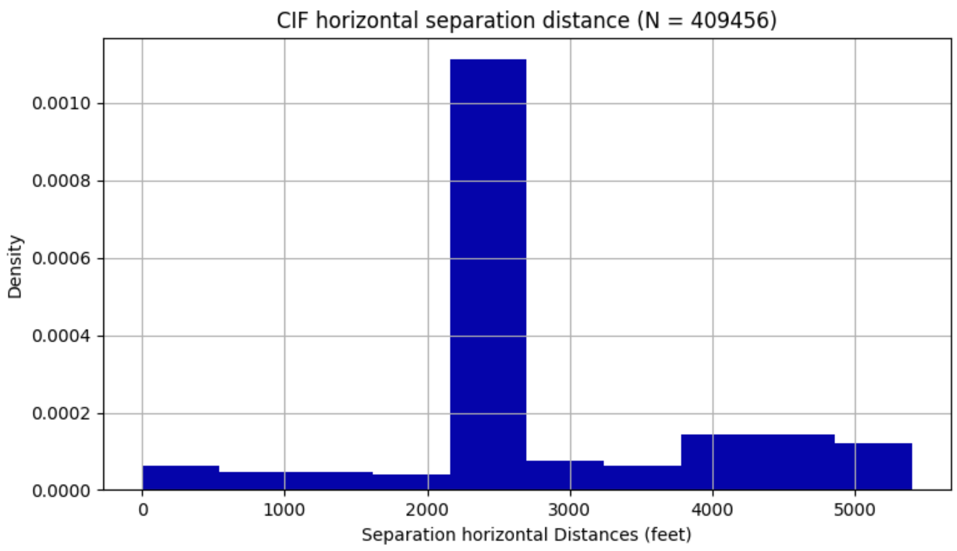


Figure 7.16: Distribution of minimum horizontal separation distances of CIF algorithm

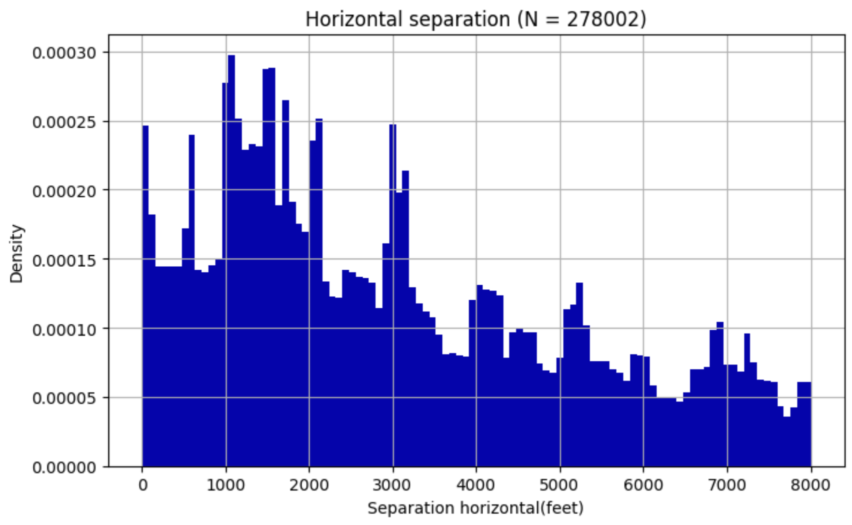


Figure 7.17: Distribution of minimum horizontal separation distances of path planning algorithm

7.3.2.2 Minimum Time-to-Collision

Figure 7.18 shows the summary of the minimum time to collision for the rules-based algorithms. The summaries show the distributions of the minimum time to collision for the CIF and RAS approaches while there is insufficient vertical separation distances during conflict resolutions.

The density distributions of the minimum time to collision when there is insufficient vertical separations between the UAVs are shown in Figure 7.19 and 7.20 for RAS and CIF respectively. The distributions show the thresholds that UAVs are allowed to get at close range before collisions. The RAS prioritises the time to collision between UAVs before to reach a minimum of 20 seconds before collisions. However, the CIF approach only prioritises the time to collision proprieties when they fall below 20 seconds.

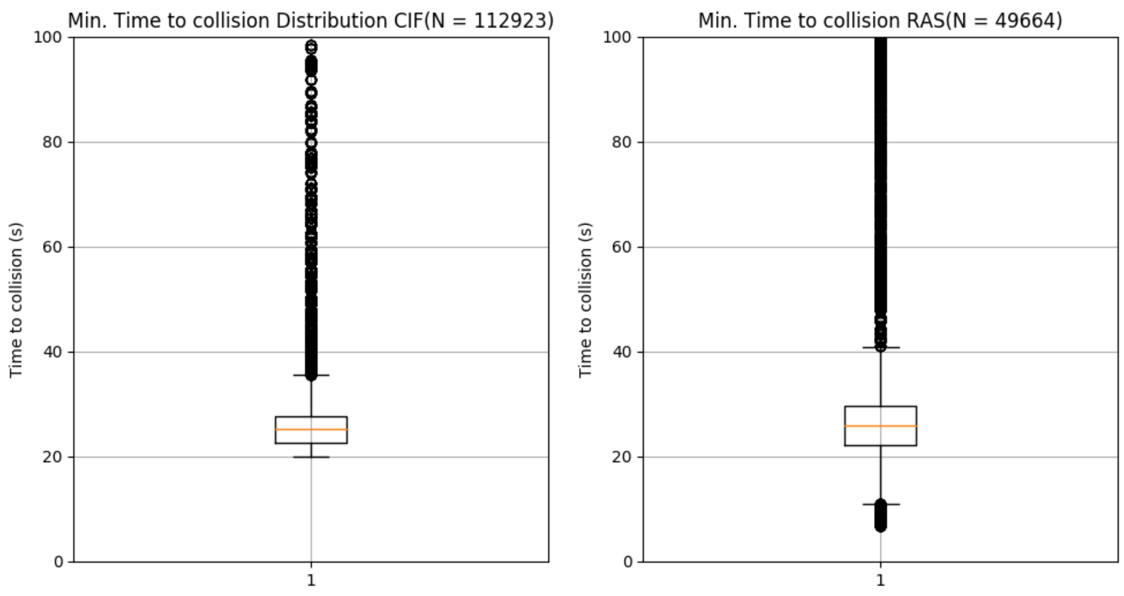


Figure 7.18: Rules-based algorithms time-to-collision (τ) summary

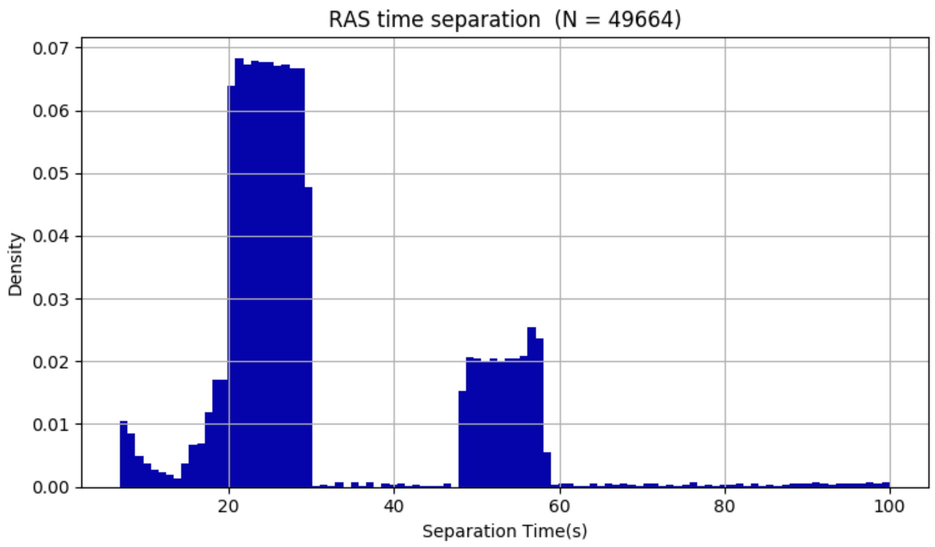


Figure 7.19: Distribution of minimum time-to-collision for RAS when vertical separation distance in minimum

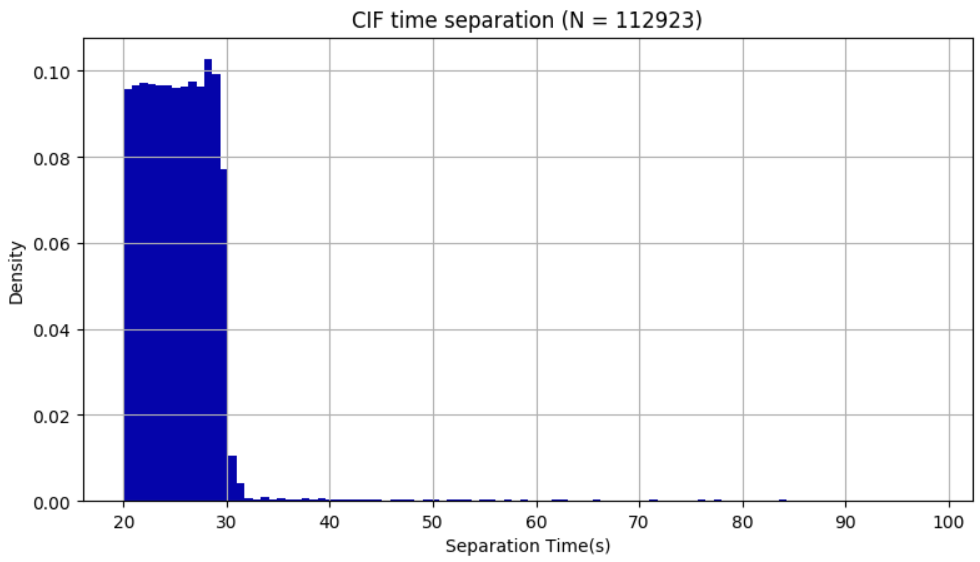


Figure 7.20: Distribution of minimum time-to-collision for CIF when vertical separation distance in minimum

7.3.3 Trajectory Deviation Cost

The summaries of the altitude deviations of the rules-based algorithms are illustrated in Figure 7.21 and 7.22 . We can see that the pairwise CIF method performs better than the RAS approach in terms of nominal altitude deviations.

The density distributions of the maximum altitude deviations during conflict resolution are given in Figure 7.23, 7.24 and 7.25 respectively for the RA, CIF and path planning algorithms. It can also be noted in the distributions that the altitude deviations of the RAS advisory are minimised with the median near 50 feet, compared to the most likely value near 150 feet for RAS. This indicates that RAS distributes the altitude deviations required for conflict resolution between multiple aircraft. This will however, for RAS, lead to the he under-actuated multiple UAVs which do not explore the altitude deviation space. In Figure 7.25, the distribution of altitude deviations for path planning has a median of near 500 feet, and 25 percent of the values below 250 feet.

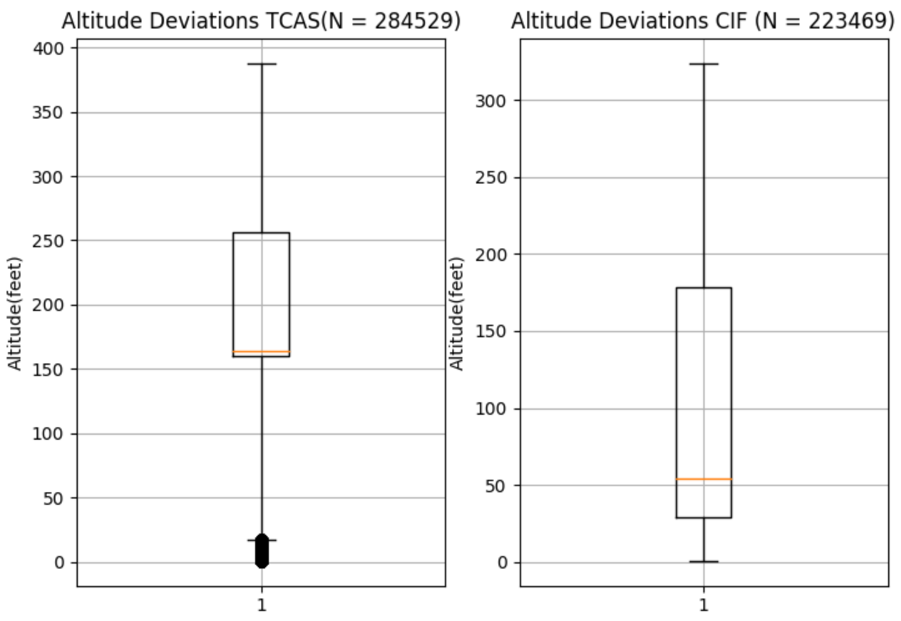


Figure 7.21: Summaries of altitude deviations for five UAVs using the RAS and CIF rules-based algorithms

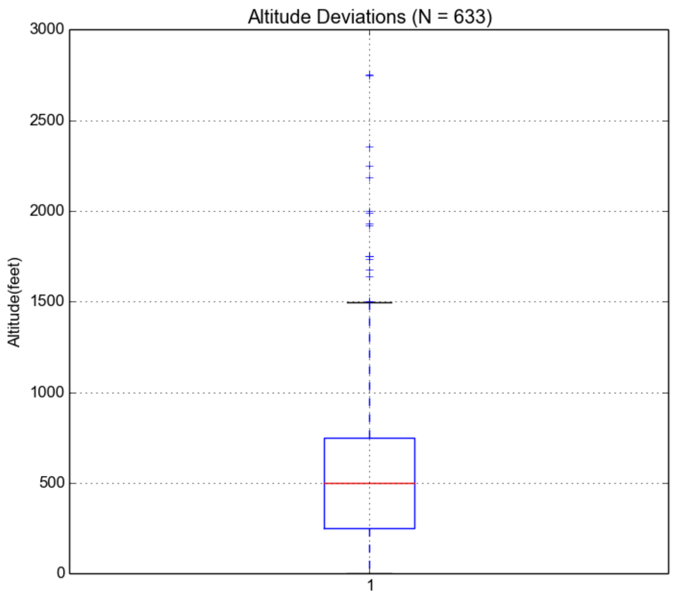


Figure 7.22: Distribution of altitude deviations for path planning algorithms

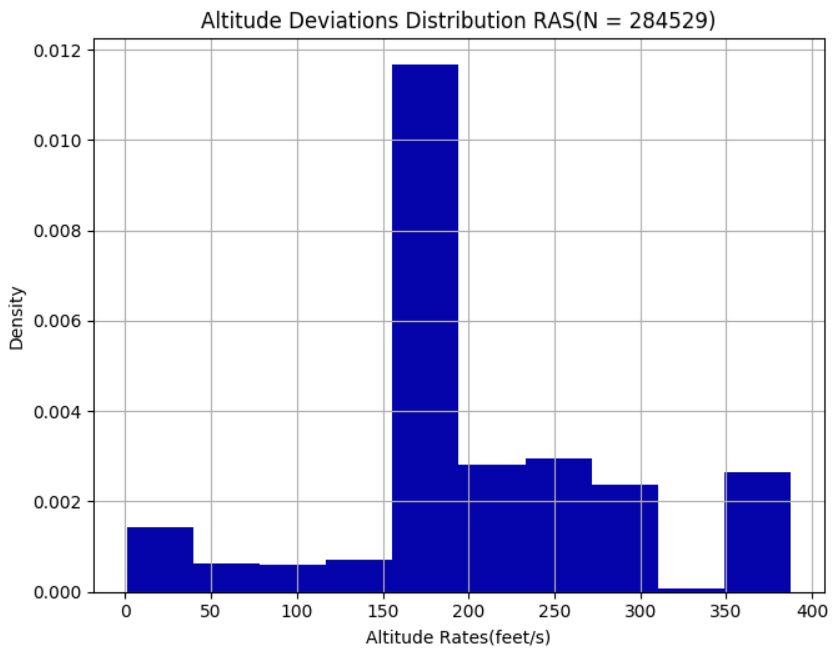


Figure 7.23: Distribution of the altitude deviations for the RAS algorithm

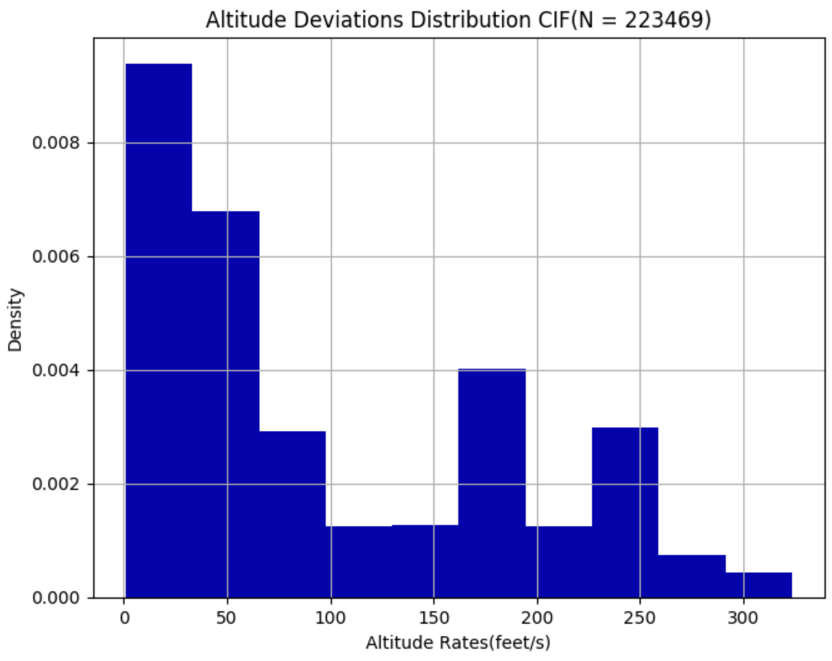


Figure 7.24: Distribution of the altitude deviations for CIF algorithm

The altitude deviations of each UAV was formally defined in Section 5.3 as the cumulative altitude changes from the root node to any other node within a search-tree. The altitude deviation of each node is the transition cost which is local to each node. The altitude deviations of a UAV are minimised as transition cost for the search of least cost manoeuvre. Figure 7.25 shows the statistical distribution of the maximum altitude deviations for each of the UAVs selected by the token allocation. This distribution is related to the control efforts required by each UAV as discussed in Section 5.4.1.

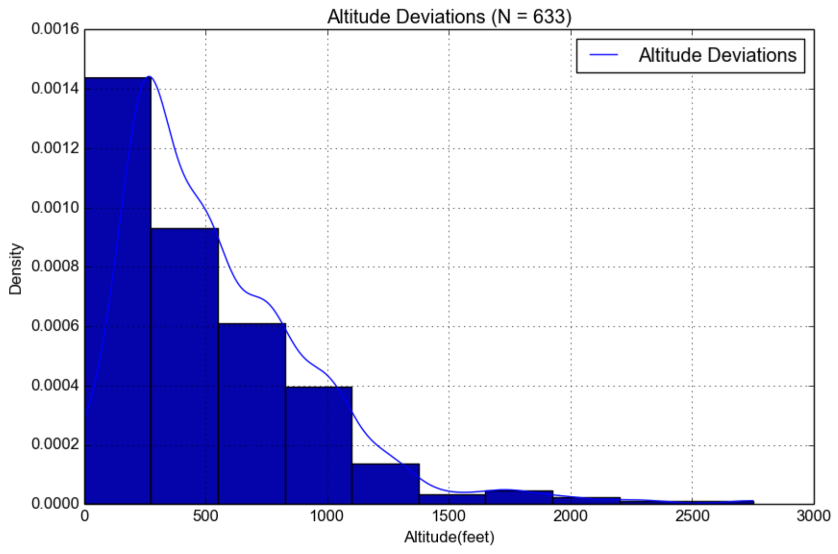


Figure 7.25: Distribution of altitude deviations of all UAVs which received tokens

The following are simulation results of the altitude deviations of the path planning algorithm during conflict resolution for each of ten UAVs are illustrated in Figure 7.26 to Figure 7.35.

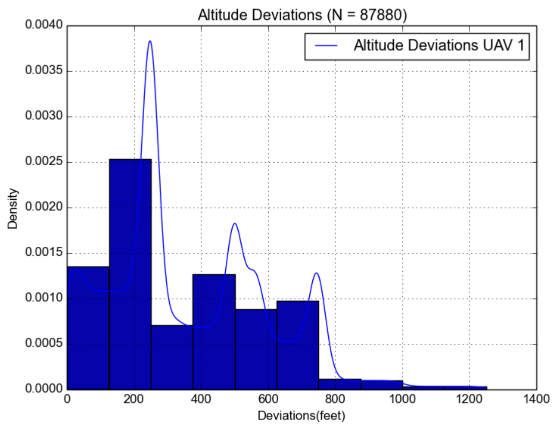


Figure 7.26: UAV 1: Altitude Deviations

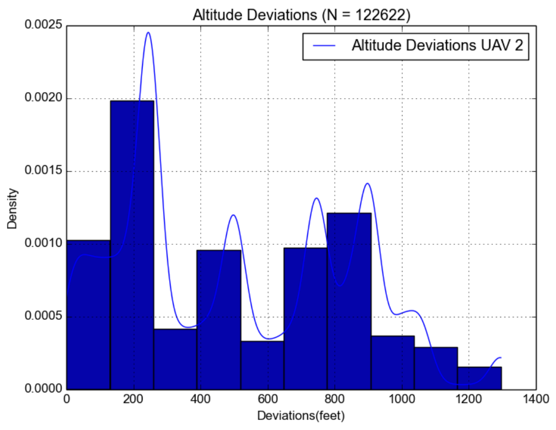


Figure 7.27: UAV 2: Altitude Deviations

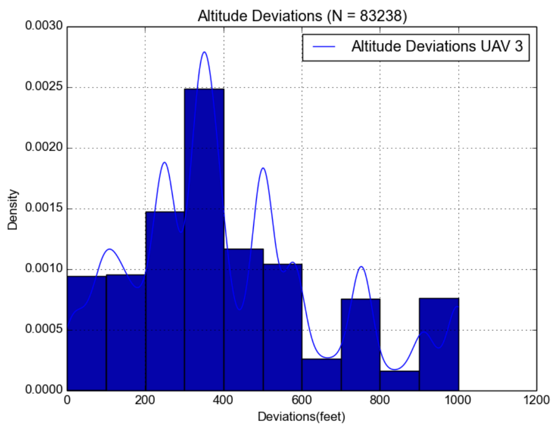


Figure 7.28: UAV 3: Altitude Deviations

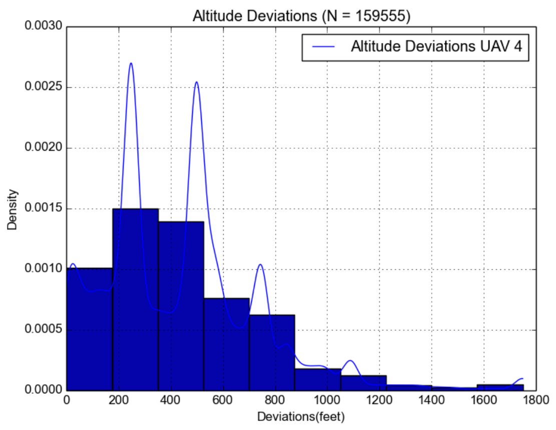


Figure 7.29: UAV 4: Altitude Deviations

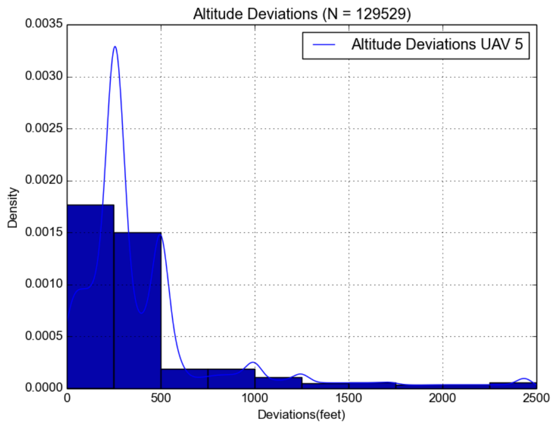


Figure 7.30: UAV 5: Altitude Deviations

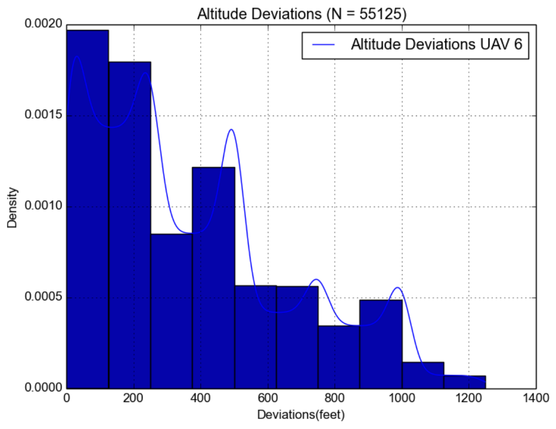


Figure 7.31: UAV 6: Altitude Deviations

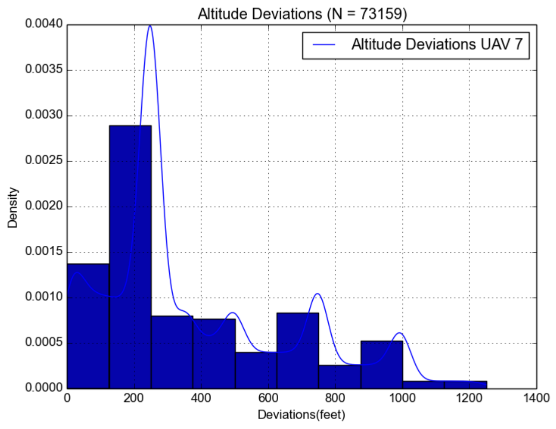


Figure 7.32: UAV 7: Altitude Deviations

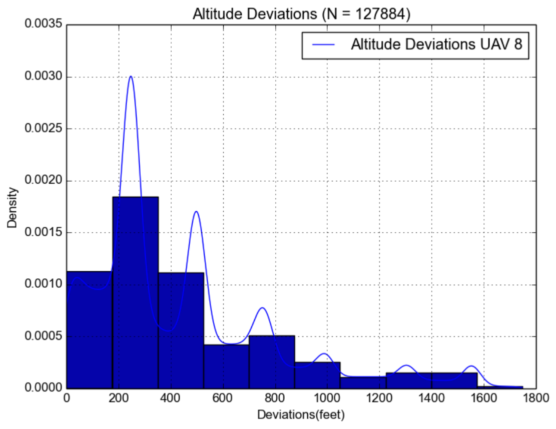


Figure 7.33: UAV 8: Altitude Deviations

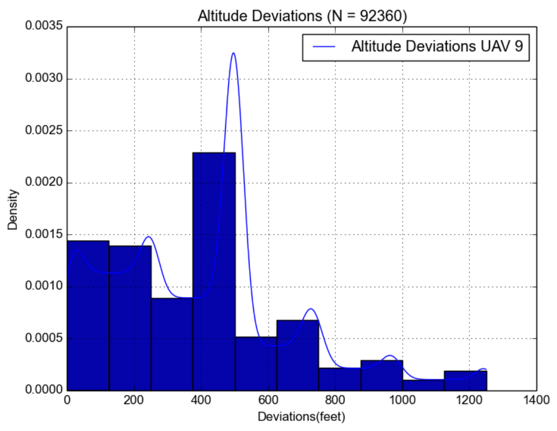


Figure 7.34: UAV 9: Altitude Deviations

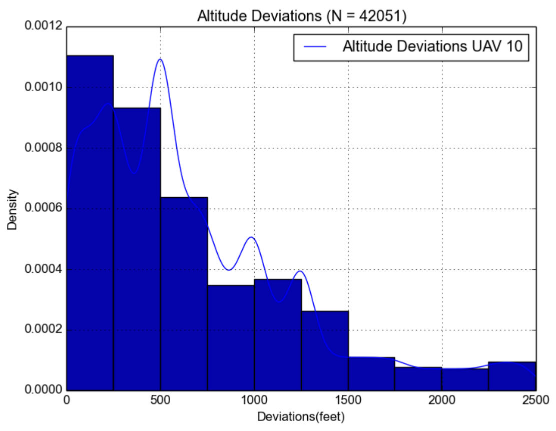


Figure 7.35: UAV 10: Altitude Deviations

The terminal cost function is defined as the magnitude of the altitude deviation at the end of the planned collision avoidance manoeuvrer. This is also minimised by each UAV selected by the token allocation. The terminal cost of the Monte Carlo simulation is shown in Figure 7.36. There are the two most frequent terminal costs in the distribution of the function. The most likely terminal cost is the level flight and the second likely cost is the vertical threshold that leads to parallel flight between UAVs (discussed in the next subsection on distribution of used altitude rates). The distribution of the altitude rates used throughout the simulation is shown in Figure 7.37. This illustrates that the UAVs were not under-actuated for the climb and descend manoeuvres. However, we can observe that climbing was favoured due to the minimum altitude threshold of 1000 feet above ground level.

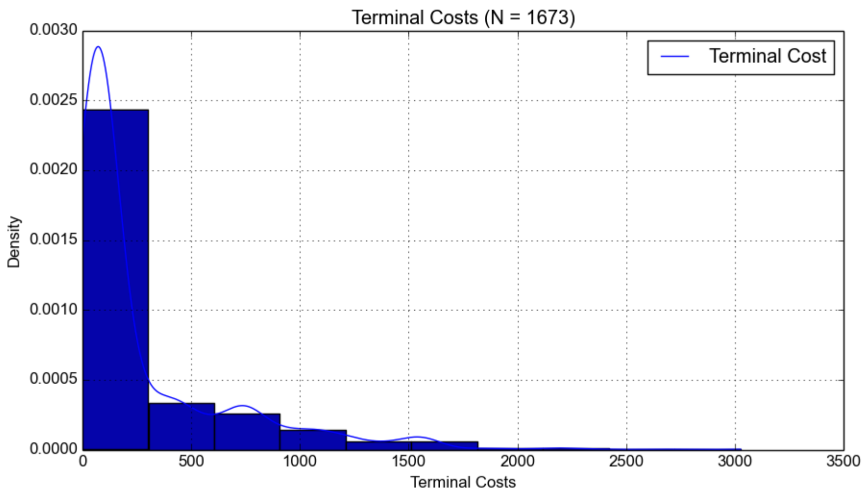


Figure 7.36: Distribution of terminal cost per UAV at each planning stage

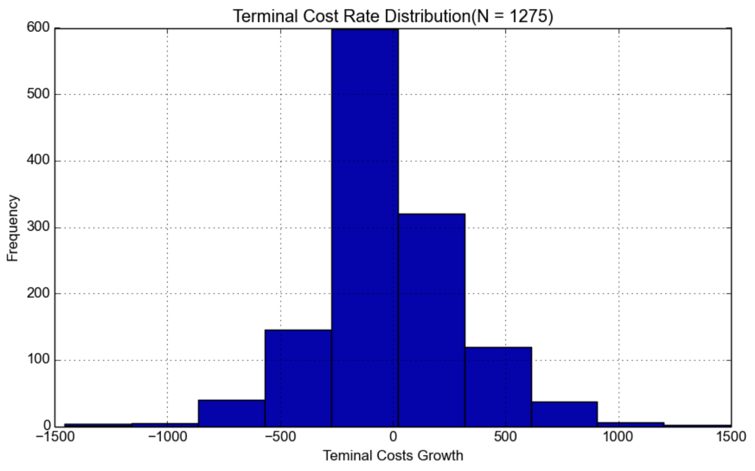


Figure 7.37: Altitude rates distribution function of terminal cost function

7.3.4 Control Effort Cost

The summaries of altitude rates distributions used by the rules-based algorithms to effect the altitude deviations are given in Figure 7.38. The results for the CIF and RAS have median altitude rates of control effort approximately 50 feet per second. There is a large skewness in the 75th upper quantiles between CIF and RAS, which indicates that the CIF conflict resolutions are not easily scalable. Additionally, only 25 percent of the altitude commands of the CIF were more than 130.48 feet/sec. The altitude rate commands used by CIF are minimised, however RAS is a recommendable method for multiple UAVs conflict resolution because the method will weigh altitude advisories according to the number of intruders encountered.

The altitude rates density distribution of the RAS in Figure 7.39 are evenly distributed between between 0 feet per second and 400 feet per second, however 0 feet per second conflict resolution advisories for level flight are the most likely. The density distribution of the CIF technique in Figure 7.40 prioritises conflict advisories between pairwise UAVs, therefore, the method will consequently ignore the number of intruders in a conflict neighbourhood. Thus, the distribution of altitude rates of CIF are demonstrated better in Figure 7.40 to illustrates that pairwise UAV conflict advisories are at 50 feet per second and 350 feet per second.

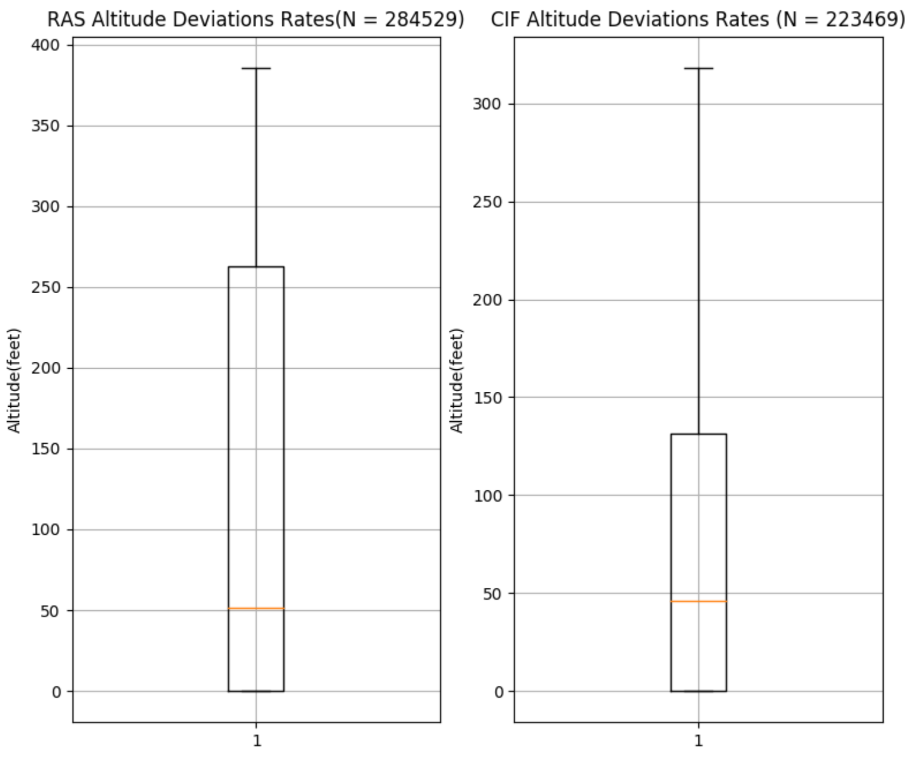


Figure 7.38: Summaries of the altitude rates used by the rules-based algorithms

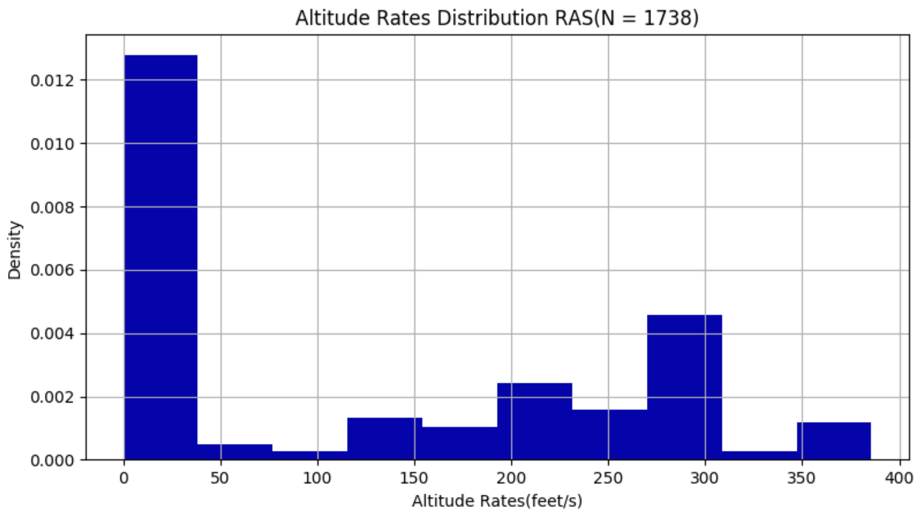


Figure 7.39: Altitude rate distribution for the RAS algorithm

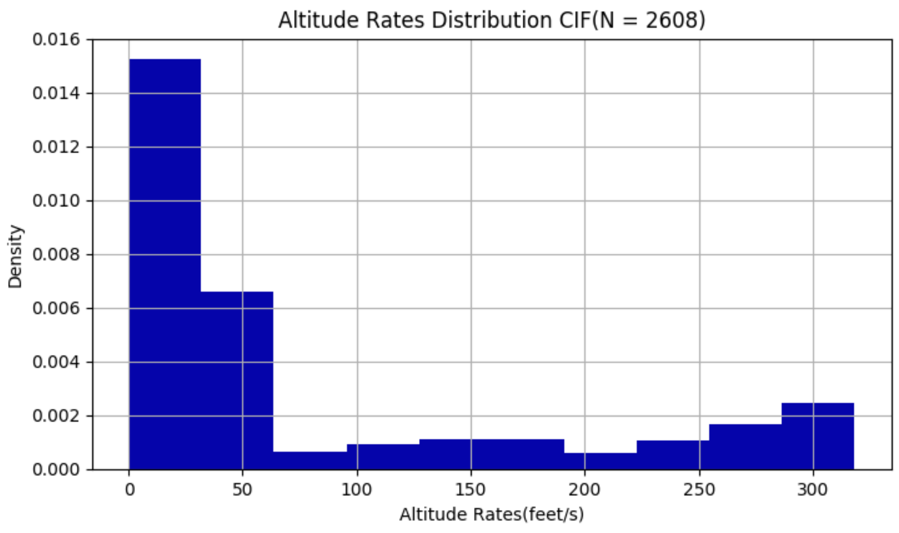


Figure 7.40: Altitude rate distribution for the CIF algorithm

The unit cost of altitude deviations rates/costs (|25| feet per second) of the path planning for ten UAVs is summarised in Figure 7.41. The transition cost distribution represents the unit deviations of climb and descend of the UAVs selected by the token allocation. It is observable that the transition costs are minimised to level flight (0 feet per second) by the search-tree algorithm. The most likely value of the transition cost for conflict resolution is the level flight.

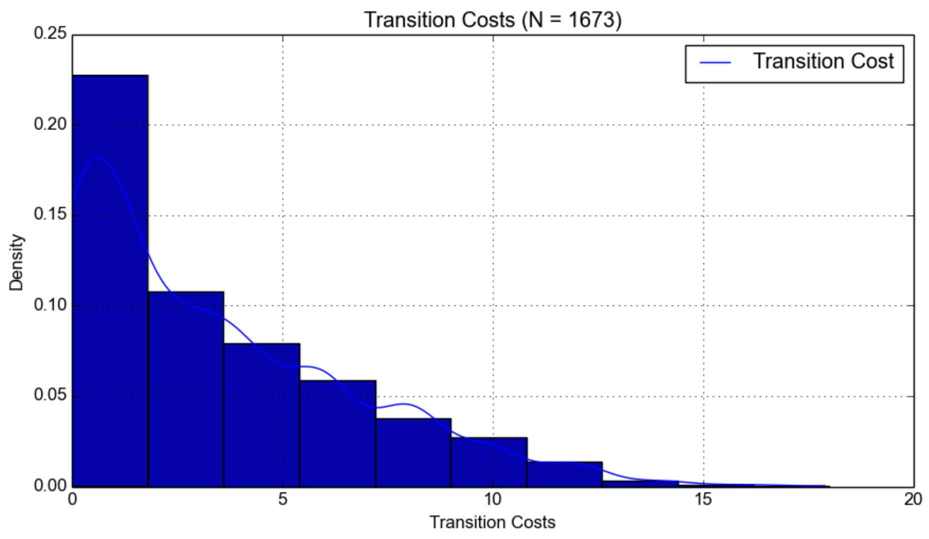


Figure 7.41: Distribution of transitional cost per UAV for path planning algorithm

7.4 Summary and Conclusion

The conflict resolution performance metrics for collision avoidance algorithms presented in this chapter to analyse their ability to avoid collisions. The rules-based simulation distributions of the minimum time-to-collisions and separation distances were also analysed. The success rate of maintaining the separation distance above 1000 feet are 87.97 % and 95.13 % respectively for the RAS and CIF rules-based algorithms. The RAS was able to respond in time 20 seconds before a collision occurs, while CIF allowed time to collision below 20 seconds. Additionally, we have observed from the simulation results that neither RAS nor pairwise CIF optimises the time-to-collision and altitude deviation control inputs. However, the RAS algorithm minimises the vertical separation distance and keeps time-to-collision at the minimum of 20 seconds before collision. The CIF was able to minimise the altitude deviations from the nominal flight path. The success rate of the path planning with token allocation was achieved with 40%. The simulations also provided density estimations of the terminal and transitional cost functions of the path planning. The altitude deviations, terminal and transition cost functions were exponentially minimised closer to zero for each token allocation. Furthermore, the simulation results illustrated that the cost of trajectories of path planning search-trees are minimised by the token allocation algorithm.

Chapter 8

Conclusions and Recommendations

This thesis presented a study on rules-based and path planning algorithms for collision avoidance of multiple unmanned aeriel vehicles with deterministic states modelling. The rules-based algorithms are modelled after TCAS as a benchmark study and simulation of tactical vertical manoeuvre conflict resolution. Thereafter, an iterative path planning algorithm was proposed that uses TCAS vertical inputs for strategic optimisation collision avoidance. In the summary of work done (Section 8.1), an overview of multiple UAVs conflict resolution with rules-based and path planning algorithms are given in detail. The conclusions made in Section 8.2 elaborate on the achievements of the work done and reflects on the research objectives and goals. Thereafter, the limitations on the execution of the proposed cooperative rules-based and search-based path planning in terms of conflict modelling, prediction and resolution, are considered in Section 8.3. Finally, recommendations on the improvement of cooperative path planning are given in Section 8.4, concluding with Future Work.

8.1 Summary of Work Done

The goal of this research was to develop and evaluate cooperative collision avoidance algorithms for unmanned aerial vehicles in multi-aircraft conflict scenarios. We wished to investigate two types of collision avoidance algorithms: a rules-based algorithm and a cooperative path planning based algorithm. It was assumed that all vehicles share their state and intent information with one another and possibly with a central node. The research goal was achieved using the following methodology: a literature study was performed on existing collision avoidance systems for manned aircraft, previous research on collision avoidance for unmanned aerial vehicles, and cooperative path planning for autonomous vehicles. The dynamics of the multi-aircraft system with terrain was conceptualised and modelled. An abstract communications framework for the unmanned aerial vehicles to share their state and intent information with one another and with a central node was conceptualised. Two types of collision avoidance algorithms were developed: a rules-based algorithm and a cooperative path planning based algorithm. The rules-based collision avoidance algorithm was modelled after the tactical Traffic Collision Avoidance System (TCAS) that is used on commercial passenger airliners. To enable multi-aircraft collision avoidance, two methods for combining the pairwise rules-based collision avoidance actions were proposed, namely Resolution Action Superposition (RAS) and pairwise Closest-Intruder-First (CIF). The path planning based collision avoidance algorithm grows a search tree of admissible conflict resolution paths, and searches the tree to find the conflict-free path with the lowest cost. To enable cooperative collision avoidance, all aircraft communicate their current positions and intended flight paths to all other aircraft. A token allocation strategy is used so that the individual aircraft plans its new collision avoidance paths sequentially, according to a predetermined priority order. A simulation environment was created that simulated the vehicles, the terrain, the communication interface, and the cooperative collision avoidance algorithms. The rules-based and path planning based collision avoidance algorithms were implemented and verified in simulation. Set-piece conflict avoidance scenarios were performed to produce illustrative results. Monte Carlo simulations were presented to produce statistical results and evaluate the performance of both algorithms in random conflict scenarios. The illustrative simulation results were analysed to investigate and compare the qualitative behaviour of the rules-based and path planning based collision avoidance algorithms. The Monte Carlo simulation results were analysed to evaluate and compare the quantitative performance of the rules-based and path planning based collision avoidance algorithms. The simulation results show that both the rules-based and path planning based solutions are able to successfully resolve collision scenarios involving multiple unmanned aerial vehicles. However, the rules-bases solution requires less computational effort but does not optimise the collision avoidance plans the path planning based solution requires much more computational effort, but provides optimal solutions that minimises the deviation from the original flights, and minimises the control effort of the avoidance actions.

8.2 Conclusions

In the presentation of rules-based collision avoidance systems in Chapter 4 we have seen that continuous intent communication abstraction enables the conflict detection algorithm to propagate the states of pairwise aircraft. The time-to-collision and horizontal threshold were derived for TCAS conflict detection alert system with a goal for short-term reactive time. However, the state propagation conflict resolution for TCAS can only react to instantaneous dynamics states of pairwise UAVs in conflict, which can lead to a new conflict in few seconds. Furthermore, multiple conflict resolution advisories presented in the RAS and pairwise CIF are also reactive without any objective function of optimising of conflict resolution decisions. Thus far, we conclude that TCAS simulation of its conflict detection algorithm provided us with enough modelling for multiple aircraft conflict resolution procedures for **Objective 1**.

The path planning algorithm was able to propagate the states of intruder aircraft in sixty seconds with the conflict detection region within a trajectory. Furthermore, the same conflict detection used in the rules-base conflict avoidance simulations was reused in the path planning algorithm towards **Objective 2**. A conflict prediction was able to be performed in long term with control input optimisation within a search-tree data structure for **Objective 3**. Therefore, the path planning algorithm design was able to reuse the TCAS vertical input set to simulate the trajectories of other aircraft. In addition, the search-tree data structure for path planning based collision avoidance was implemented with a search algorithm which found an optimal altitude level for a UAV during conflict resolution. Therefore, the manoeuvre which each UAV made had a cost which was made up of the climb and descend inputs. Conflict resolution with cooperative path planning was able to simulate both the nominal and path plans states of multiple intruder UAVs as dynamic obstacles.

The Monte Carlo simulations of both rules-based and path planning algorithms were presented in Chapter 7 in line with **Objective 5**. The safe separation distance between UAVs was analysed as the main safety factor. The conflict resolution results of the rules-based algorithms were able to react and resolve majority of conflicts involving five UAVs. However, CIF and RAS rules-based algorithms were only able to maximize the separation distances between aircraft. Thus, in respect to **Objective 6**, we conclude that altitude deviations were not minimised by the rules-based algorithms. Meanwhile, the cooperative path planning with token allocation was able to consider utilised a cost function for minimization of altitude deviations and manoeuvre space. This was achieved with exponential growth of a search-tree from TCAS input sets. Finally, the path replanning time surveillance with token allocation showed that UAVs in a multiple conflict can take decisions in linear time. The statistical terminal and transitional costs of trajectories which were selected for execution were minimised as set out as the objective of our cost function.

8.3 Limitations

The main limitations of the research are:

- The state propagation and conflict detection modules did not consider any uncertainty in the measurement of the UAV positions and velocities, or any uncertainty in the execution of the planned flight paths.
- Only vertical collision avoidance actions were considered. Horizontal avoidance actions, such as heading changes and speed adjustments, were not considered in this research.
- The rules-based collision avoidance algorithm did not include manoeuvre reversal. Once a UAV has committed to a vertical action direction, it cannot reverse its action.
- UAVs were only allowed to proposed a full length trajectories of 60 seconds forward in time for low uncertainty conflict resolution.
- The path planning based collision avoidance algorithm does not allow an avoidance manoeuvre to be interrupted while it is being executed. Once a UAV enters collision avoidance mode, it is "locked in" until it has executed the entire collision avoidance

plan. If new collisions are detected, then any of the UAVs which are already executing collision avoidance plans are not allowed to replan their collision avoidance paths.

8.4 Recommendations and Future Work

The following recommendation and future work references are made on the cooperative path planning for multiple UAVs over a wireless reliable communication network.

- The two-dimensional collision prediction and avoidance should be extended to three dimensions.
- Horizontal actions, such as heading changes and speed adjustments, should be investigated as possible alternative collision avoidance actions.
- Other token allocation strategies should be investigated.
- Sampling-based path planning techniques, that use random state or input sampling, should be investigated as an alternative to the deterministic actions sampling that was employed in this research.

Appendices

Appendix A

Rules-Based Collision Avoidance

A.1 Random Noise in Altitude Manoeuvres

The noise w_k is a deciding factor for the climb or descend if the relative altitude Δh is zero without the noise. The effect of the additive noise are shown in the simulations of three UAVs conflict simulations in Figure A.1 and Figure A.2 for climb and descend manoeuvre respectively.

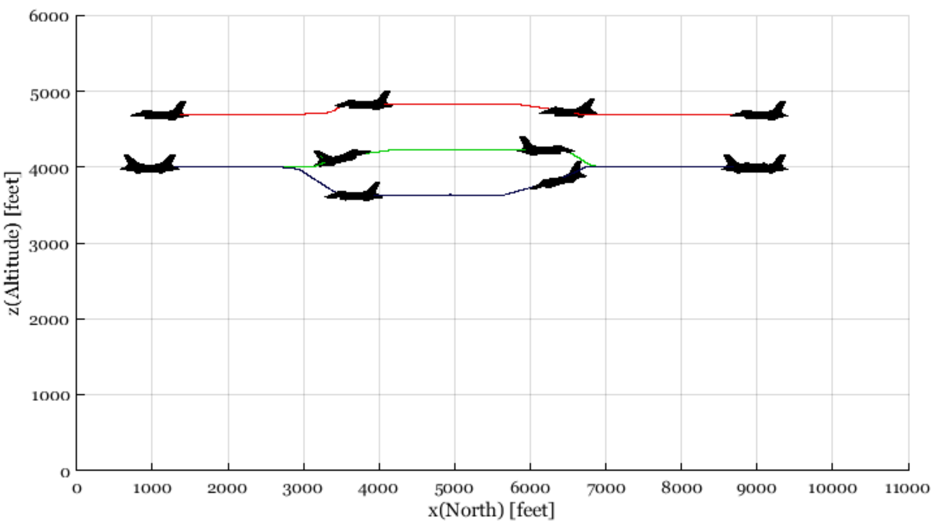


Figure A.1: Climbing manoeuvre influenced by w_k with two UAVs at equal altitude

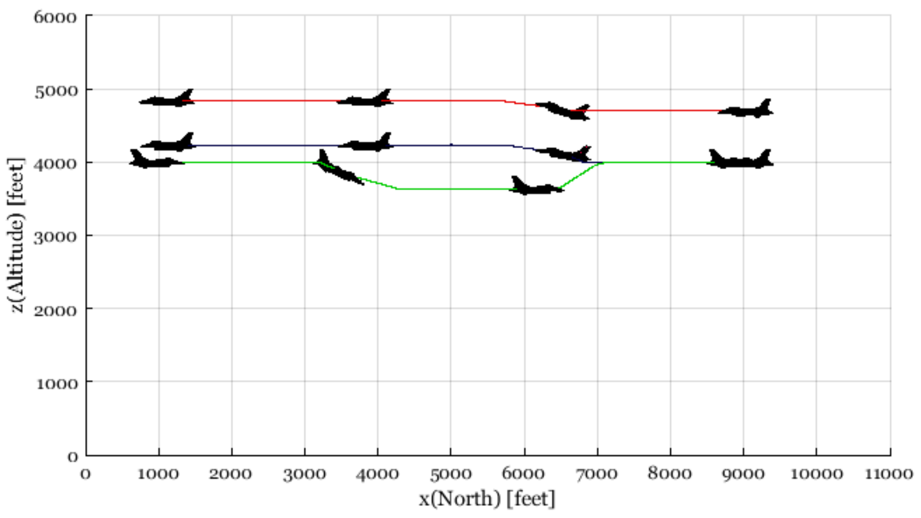


Figure A.2: Descend manoeuvre influenced by w_k with with two UAVs at equal altitude

A.2 Time-to-Collision(τ) of Five UAVs

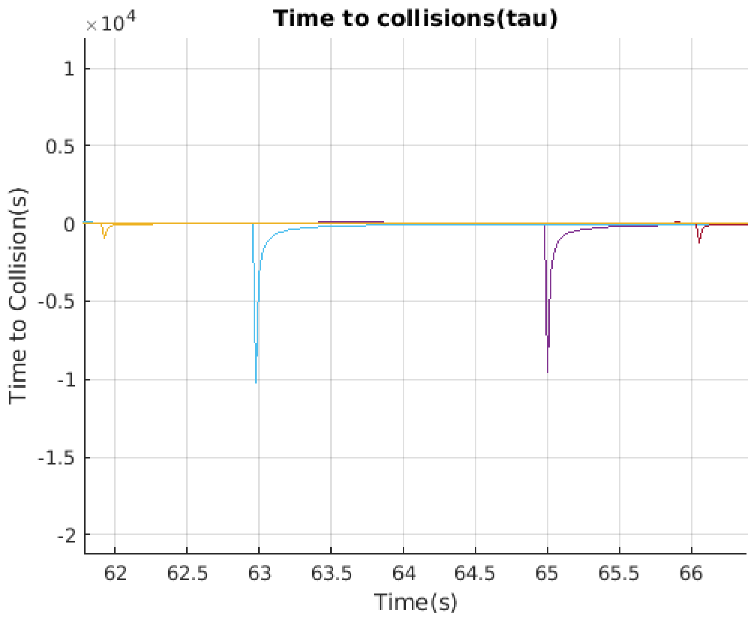


Figure A.3: Time-to-Collision(τ) of the Resolution Action Superposition for Five UAVs

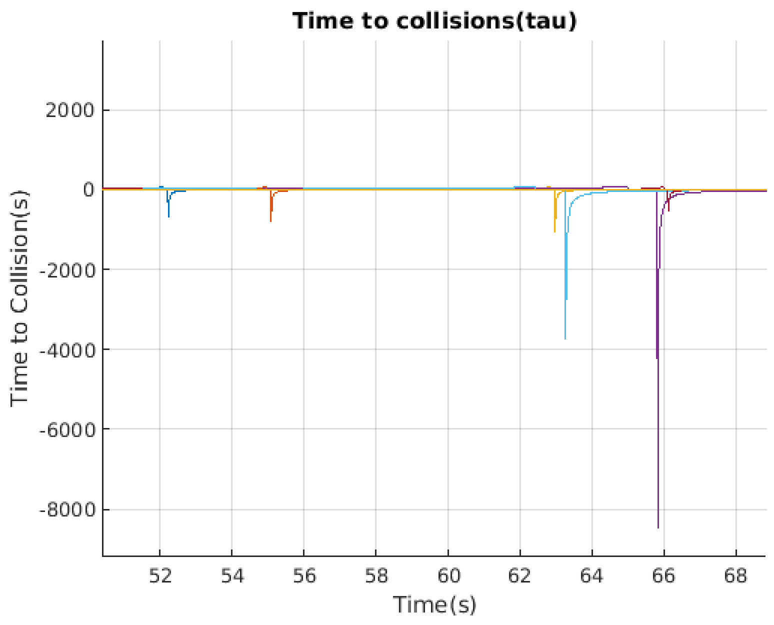


Figure A.4: Time-to-Collision(τ) of the Closest-Intruder-First for Five UAVs

Appendix B

Search-Tree Data Structures

B.1 Time Execution of a Path

Algorithm 17 Path time-indexing hashing $PathHashing(t, T, \tau)$

```
Require: Time steps t, simulation period T, Execution step depth tau
 1: if tau $\geq$ 0 then
 2:     index $\leftarrow$ ceil($\tau$/t) + floor(T * 0.5)
 3:     if tau == 0 then
 4:         index $\leftarrow$ ceil(T*0.5) + 1
 5:     else
 6:         index $\leftarrow$ ceil($\tau$/t) + floor(T*0.5)
 7:     end if
 8: else
 9:     index $\leftarrow$ floor($\tau$/t) + ceil(T*0.5)
10: end if
11: return index
```

B.2 Path Completeness

Algorithm 18 Path Length $PathLength(leaf, time)$

```
Require: Planning Tree Leaf node $leaf$, Horizon time length $time$
 1: tau $\leftarrow$ leaf.tau
 2: if tau not ewuals to time then
 3:     complete $\leftarrow$ 0
 4: else
 5:     complete $\leftarrow$ 1
 6: end if
 7: return complete
```

B.3 Simulation of Path Plans

Algorithm 19 Plan States Simulation $PlanSimulate(X, A, U, P, T_i)$

```
Require: UAVs position vector $S \in \mathbb{R}^3 \times \mathbb{R}^m$, nominal altitudes $A = \{A_1, A_2, A_3, \dots A_n\}$,
    Path Plan $U = \{U_1, U_2, \dots U_n\}$, Plan Tracker $T = \{t_1, t_2, \dots t_n\}$, Time step length
    $t \in \mathbb{R}$
1:  $l \leftarrow length(U_1)$                                ▷ Each UAV path plan length
2:  $h_{thr} \leftarrow C_h$
3:  for i from 1 to n do                                      ▷ i is a uav set iterator
4:      $x_i \leftarrow X[i]$
5:      $h_i \leftarrow s_i[3]$                                ▷ UAV altitude
6:      $u_i \leftarrow \{\}$
7:      if $T[i] > 0$ then
8:          track ← T[i]
9:          $u_i \leftarrow$ U[i]
10:         NewT ← U[i]
11:         $u_i \leftarrow u_i$[from track until $l$]
12:         $r \leftarrow l - length(u_i)$
13:         for iterator from track until l do
14:             $x_i[3] \leftarrow x_i[3] + NewT[iterator] \cdot T_i$
15:         end for
16:         for index from 1 intil r do
17:             $\dot{z} \leftarrow AltitudeControl(h_i, A_i, h_{thr})$
18:             $x_i[3] \leftarrow x_i[3] + \dot{z} \cdot T_i$
19:             $u_i[j] \leftarrow \dot{z}$
20:         end for
21:     else
22:         for j from 1 to l do
23:             $z\_dot \leftarrow AltitudeControl(h_i, A_i, h_{thr})$
24:             $x_i[3] \leftarrow x_i[3] + \dot{z} \cdot T_i$
25:             $u\_i[j] \leftarrow \dot{z}$
26:         end for
27:     end if
28:     $U[i] \leftarrow u_i$
29: end for
30: return $X$
```

Appendix C

Path Planning Algorithms

C.1 C-Space Notation

The C-space is a superset of other spaces which are useful for path planning of unmanned aircraft motions. Let us get familiar with them notationally. An unmanned aircraft is described as a geometric rigid body $\mathcal{A}$ in a working space $\mathcal{W} \in \mathbb{R}^m$. The manoeuvre space which unmanned aircraft can execute are limited by unreachable obstacle regions $\mathcal{O} \subseteq \mathcal{W}$. If the obstacle region is known to the path planning algorithm it is $\mathcal{C}_{obs}$, a subset of C-space $\mathcal{C}_{obs} \subseteq \mathcal{C}$ can be dynamic or static obstacles. The free configurations space C_{free} is the vehicle free planning space, a subset of C-Space $C_{free} \subseteq \mathcal{C}$.

$$\mathcal{C}_{obs} = \{q \in \mathcal{C} | \mathcal{A}(q) \cap \mathcal{O} \neq \emptyset\} \tag{C.1}$$

$$\mathcal{C}_{free} = \{q \in \mathcal{C} | \mathcal{A}(q) \cap \mathcal{O} = \emptyset\} \tag{C.2}$$

C.2 Conflict Resolutions Direction

The manoeuvre direction of a UAV are sensitive to its intruder's altitude. Each of the UAV should sense its environment and take altitude direction that require more effort and is highly distributed. In Figure C.2, two UAVs with a predicted conflict with a negative relative altitude, the black-path UAVs descents as a resolution action. The manoeuvre is taken in the direction which resolve the conflict but also demonstrate altitude threshold existence. Similarly shown in Figure C.1 the same UAV chooses to climb because the altitude change is positive with relative to the intruder.

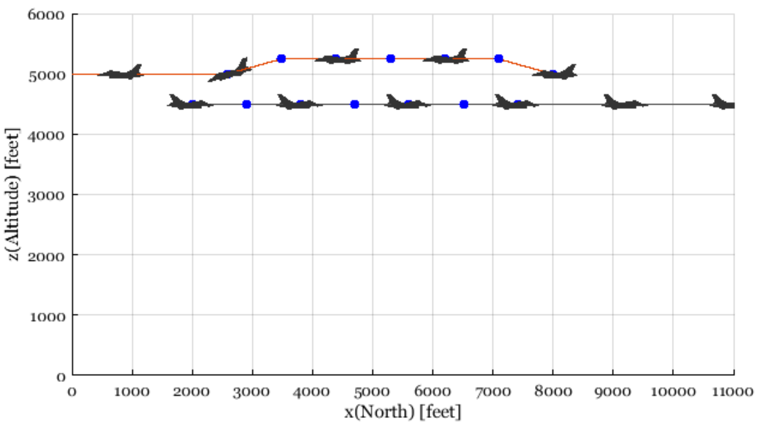


Figure C.1: Climbing resolution due to a positive relative altitude of pair UAVs

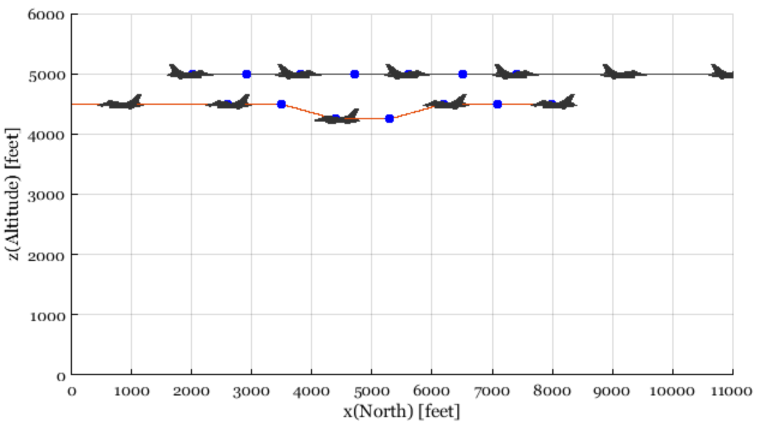


Figure C.2: Descending resolution due to a negative relative altitude of pair UAVs

C.3 Path Sampling Examples

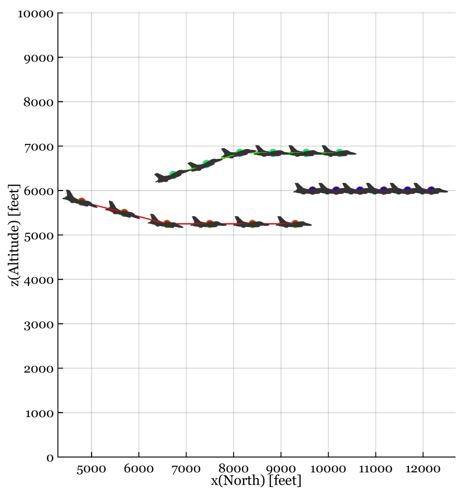


Figure C.3: Example 1

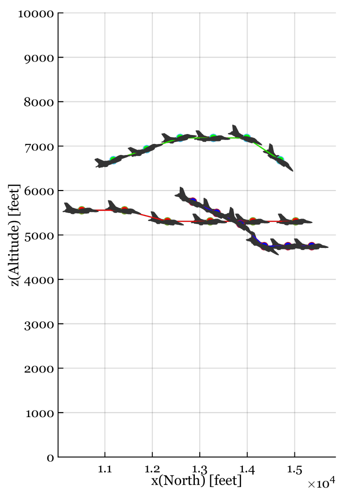


Figure C.4: Example 2

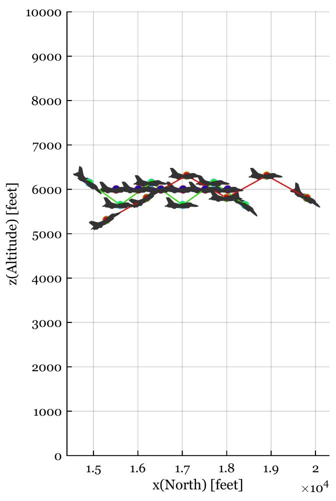


Figure C.5: Example 3

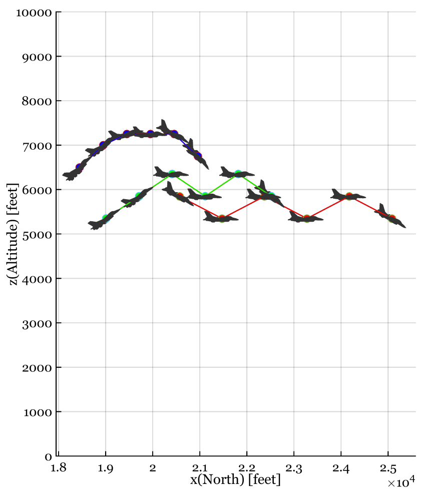


Figure C.6: Example 4

List of References

[1] Federal Aviation Administration: UAS Sightings Report. 2017.
Available at: `https://www.faa.gov/uas/resources/uas_sightings_report/`

[2] FAA: Integration of Civil Unmanned Aircraft Systems (UAS) in the National Airspace System (NAS) Roadmap. p. 74, 2013.

[3] Kochenderfer, M.J., Holland, J.E. and Chryssanthacopoulos, J.P.: Next-Generation Airborne Collision Avoidance System. *Lincoln Laboratory Journal*, vol. 19, no. 1, pp. 17–33, 2013.

[4] Kuchar, J.K. and Yang, L.C.: Survey of conflict detection and resolution modeling methods. *American Institute of Aeronautics and Astronautics, Inc.*, pp. 1388–1397, 1997.

[5] van Daalen, C.E.: Conflict Detection and Resolution for Autonomous Vehicles. *Proceedings of the IEEE*, , no. March, 2010.
Available at: `http://hdl.handle.net/10019.1/45120`

[6] Shanmugavel, M., Tsourdos, A., White, B. and Zbikowski, R.: Co-operative path planning of multiple UAVs using Dubins paths with clothoid arcs. *Control Engineering Practice*, vol. 18, no. 9, pp. 1084–1092, 2010. ISSN 09670661.

[7] Alliot, J.-m. and Durand, N.: FACES : a Free flight Autonomous and Coordinated Embarked Solver. 2014.

[8] Williamson, T. and Spencer, N.A.: Development and Operation of the Traffic Alert and Collision Avoidance System (TCAS). *Proceedings of the IEEE*, vol. 77, no. 11, pp. 1735–1744, 1989. ISSN 15582256.

[9] Espindle, L.P., Billingsley, T. and Griffith, J.: TCAS multiple threat encounter analysis. *Massachusetts Institute of Technology, Lincoln Laboratory, Project Report ATC-359*, 2009.
Available at: `http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:TCAS+Multiple+Th`

[10] Ground, E., Warning, P., Evolve, S. and Safety, P.G.: Enhanced Ground Proximity. , no. july, pp. 38–40, 2007.

[11] Fan, T.P., Hyams, D.S. and Kuchar, J.K.: Study of In-Flight Replanning Decision Aids. *American Institute of Aeronautics and Astronautics*, pp. 980–988, 1998.

[12] Chryssanthacopoulos, J.P. and Kochenderfer, M.J.: Decomposition methods for optimized collision avoidance with multiple threats. *AIAA/IEEE Digital Avionics Systems Conference - Proceedings*, 2011. ISSN 2155-7195.

[13] Pienaar, L.J.: Probabilistic Conflict Detection for Commercial Aircraft near Airports. *University of Stellenbosch*, , no. October, 2014.

[14] Kuchar, J.K. and Yang, L.C.: A Review of Conflict Detection and Resolution Modeling Methods. *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, no. 4, pp. 179–189, 2000. ISSN 15249050.
Available at: `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=898217`

[15] Kelly, W.: Conflict detection and alerting for separation assurance systems. *Gateway to the New Millennium. 18th Digital Avionics Systems Conference. Proceedings (Cat. No.99CH37033)*, vol. B.6-6 vol., pp. 6.D.1–1–6.D.1–8, 1999.
Available at: `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=821978`

[16] John P. Wangermann and Stengel, R.F.: Principled negotiation between intelligent agents: a model for air traffic management. 1997.

[17] Pachter, M. and Chandler, P.R.: Challenges Of Autonomous Control - IEEE Control Systems Magazine. pp. 92–97.

[18] Kuchar, J.K. and Drumm, a.C.: The traffic alert and collision avoidance system. *Lincoln Laboratory Journal*, vol. 16, no. 2, pp. 277–296, 2007.

[19] U.S Department of Transportation Federation Aviation Administration: Introduction to TCAS II, 2011. `arXiv:1011.1669v3`.

[20] Pritchett, A.R., Haga, R.A. and Li, H.: Attempting to Automate Compliance to Aircraft Collision Avoidance Advisories. vol. 13, no. 1, pp. 18–25, 2016.

[21] Dowek, G., Muñoz, C. and Geser, A.: Tactical Airspace Conflict Detection and Resolution in a 3-D Airspace. *Contract*, p. 20, 2001.

[22] Krozel, J. and Peters, M.: Strategic Conflict Detection and Resolution for Free Flight. , no. December, pp. 1822–1828, 1997.

[23] Geser, A.: A Geometric Approach to Strategic Conflict Detection and Resolu. pp. 1–11.

[24] Cockerill, B.-g.S.I.R.G.: Cooperative Collision Avoidance between Multiple Mobile Robots. vol. 17, 1885.

[25] De Wilde, B.: Cooperative Multi-Agent Path Planning. *Thesis Master*, 2012.

[26] Temizer, S., Kochenderfer, M.J., Kaelbling, L.P., Lozano-Perez, T. and Kuchar, J.K.: Collision Avoidance for Unmanned Aircraft using Markov Decision Processes. *AIAA Guidance, Navigation, and Control Conference (GNC)*, 2010.
Available at: `http://people.csail.mit.edu/lpk/papers/aiaa10.pdf`

[27] Boutilier, C.: Planning, learning and coordination in multiagent decision processes. *Proceedings of the 6th conference on Theoretical aspects of rationality and knowledge*, pp. 195–210, 1996.
Available at: `http://dl.acm.org/citation.cfm?id=1029710`

[28] Ong, H.Y. and Kochenderfer, M.J.: short-term conflict resolution for unmanned aircraft traffic management.

[29] Yu, X. and Zhang, Y.: Sense and avoid technologies with applications to unmanned aircraft systems: Review and prospects. *Progress in Aerospace Sciences*, vol. 74, pp. 152–166, 2015. ISSN 03760421.
Available at: `http://dx.doi.org/10.1016/j.paerosci.2015.01.001`

[30] Park, J.-W., Oh, H.-D. and Tahk, M.-J.: UAV Conflict Detection and Resolution Based on Geometric Approach. *International Journal of Aeronautical and Space Sciences*, vol. 10, no. 1, pp. 37–45, 2009. ISSN 1229-9626.
Available at: `http://koreascience.or.kr/journal/view.jsp?kj=HGJHC0&py=2009&vnc=v10n1&sp=37`

[31] Wilkie, D., Van Den Berg, J. and Manocha, D.: Generalized velocity obstacles. *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009*, pp. 5573–5578, 2009.

[32] Fox, D., Burgard, W. and Thrun, S.: The dynamic window approach to collision avoidance. *IEEE Robotics and Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997. ISSN 10709932. `arXiv:1011.1669v3`.

[33] Abe, Y. and Yoshiki, M.: Collision avoidance method for multiple autonomous mobile agents by implicit cooperation. *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No.01CH37180)*, vol. 3, pp. 1207–1212, 2001. ISSN 0780366123.
Available at: `http://ieeexplore.ieee.org/document/977147/`

[34] Reif, J.H.: Complexity of the mover's problem and generalizations. *20th Annual Symposium on Foundations of Computer Science (sfcs 1979)*, pp. 421–427, 1979.

[35] Kindel, R., Hsu, D., Latombe, J.C. and Rock, S.: Kinodynamic motion planning amidst moving obstacles. *Proceedings 2000 ICRA Millennium Conference IEEE International Conference on Robotics and Automation Symposia Proceedings Cat No00CH37065*, vol. 1, no. April, pp. 537–543, 2000. ISSN 10504729.
Available at: `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=844109`

[36] Tsourdos, A., White, B. and Shanmugavel, M.: *Cooperative Path Planning of Unmanned Vehicles*. 2010. ISBN 9780470741290.
Available at: `http://www.wiley.com/WileyCDA/WileyTitle/productCd-0470741295.html`

[37] Krozel, A. and Lafayette, W.: search problems in mission planning and navigation of autonomous aircraft. 1988.

[38] Burgard, W., Stachniss, C., Bennewitz, M. and Arras, K.: Introduction to Mobile Robotics Robot Motion Planning. , no. July, p. 151, 2011.

[39] Menon, P.K., Sweriduk, G.D. and Sridhar, B.: Optimal Strategies for Free-Flight Air Traffic Conflict Resolution. *Journal of Guidance, Control, and Dynamics*, vol. 22, no. 1, pp. 202–211, 1999. ISSN 0731-5090.

[40] Richards, A.G.: Trajectory Optimization using Mixed-Integer by. 2002.

[41] Kochenderfer, M.J. and Chryssanthacopoulos, J.P.: Robust Airborne Collision Avoidance through Dynamic Programming. pp. 1–118, 2011.

[42] Billingsley, T.B., Kochenderfer, M.J. and Chryssanthacopoulos, J.P.: Collision avoidance for general aviation. *IEEE Aerospace and Electronic Systems Magazine*, vol. 27, no. 7, pp. 4–12, 2012. ISSN 08858985.

[43] Kochenderfer, M.J. and Chryssanthacopoulos, J.P.: A decision-theoretic approach to developing robust collision avoidance logic. *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pp. 1837–1842, 2010. ISSN 2153-0009.

[44] Chryssanthacopoulos, J.P. and Kochenderfer, M.J.: Collision avoidance system optimization with probabilistic pilot response models. *Proceedings of the 2011 American Control Conference*, pp. 2765–2770, 2011. ISSN 07431619.

[45] Sahawneh, L.R. and Beard, R.W.: Path Planning in the Local-Level Frame for Small Unmanned Aircraft Systems. *Kinematics*, vol. i, no. tourism, p. 13, 2017.

[46] Kavraki, L.E., Švestka, P., Latombe, J.-C. and Overmars, M.H.: probability roadmaps for path planning in high dimensional configuration space. *CEUR Workshop Proceedings*, vol. 1542, pp. 33–36, 2015. ISSN 16130073. `arXiv:1011.1669v3`.

[47] LaValle, S.M.: Rapidly-Exploring Random Trees: A New Tool for Path Planning. *In*, vol. 129, pp. 98–11, 1998. ISSN 1098-6596. `arXiv:1011.1669v3`.
Available at: `http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Rapidly-explorin`

[48] Webb, D.J. and Berg, J.V.D.: Kinodynamic RRT*: Optimal Motion Planning for Systems with Linear Differential Constraints. *arXiv preprint arXiv:1205.5088*, p. 8, 2012. `1205.5088`.
Available at: `http://arxiv.org/abs/1205.5088`

[49] LaValle, S.M. and Kuffner, J.J.: Radomized Kinodynamic planning. *Journal of Chemical Information and Modeling*, vol. 53, no. 9, pp. 1689–1699, 2013. ISSN 1098-6596. `arXiv:1011.1669v3`.

[50] Perez, A., Platt, R., Konidaris, G., Kaelbling, L. and Lozano-Perez, T.: LQR-RRT*: Optimal sampling-based motion planning with automatically derived extension heuristics. *Proceedings - IEEE International Conference on Robotics and Automation*, , no. 0, pp. 2537–2542, 2012. ISSN 10504729.

[51] LaValle, S.M. and Kuffner, J.J.: Random Kinodynamic Planning.

[52] Latombe, J.-C.: Potential Field Methods. *Robot Motion Planning*, pp. 295–355, 1991.
Available at: `http://link.springer.com/10.1007/978-1-4615-4022-9_7`

[53] Eby, M.S.: A self-organizational approach for Resolving Air Traffic Conflicts. vol. 7, no. 2, pp. 239–254, 1994.

[54] Giese, A.: A Comprehensive , Step-by-Step Tutorial on Computing Dubin ' s Curves. 2012.
Available at: `http://gieseanw.files.wordpress.com/2012/10/dubins.pdf`

[55] Prandini, M., Hu, J., Lygeros, J. and Sastry, S.: A Probabilistic Approach to Aircraft Conflict Detection. *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, no. 4, pp. 199–219, 2000. ISSN 15249050.
Available at: `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=898224`

[56] Lacher, A.R., Maroney, D.R. and Zeitlin, a.D.: Unmanned aircraft collision avoidance-technology assessment and evaluation methods. *7th Air Traffic Managemtent Research & Development Seminar*, pp. 1–10, 2007.

[57] Bakker, G.J., Kremer, H.J. and Blom, H.A.P.: Geometric and probabilistic approaches towards conflict prediction. *3 rd USA / Europe Air Traffic Management R & D Seminar*, , no. June, pp. 13–16, 2000.

[58] Nguyen-Duc, M., Briot, J.-P. and Drogoul, A.: An application of multi-agent coordination techniques in air traffic management. *IEEE/WIC International Conference on Intelligent Agent Technology, 2003. IAT 2003.*, pp. 6–9, 2003.

[59] Yang, L.C. and Kuchar, J.: Using intent information in probabilistic conflict analysis. *AIAA Guidance, Navigation, and Control Conf*, pp. 797–860, 1998.

[60] Paielli, R.A. and Erzberger, H.: Conflict Probability for Free Flight Estimation. , no. October 1996, 2017.

[61] Erzberger, H., Paielli, R.A., Isaacson, D.R. and Eshow, M.M.: Conflict Detection and Resolution In the Presence of Prediction Error. 2000.

[62] Jones, T. and Appleby, B.: Real-Time Probabilistic Collision Avoidance for Autonomous Vehicles, Using Order Reductive Conflict Metrics. p. 146, 1999.
Available at: `http://dspace.mit.edu/handle/1721.1/29747`

[63] Švestka, P. and Overmars, M.H.: Coordinated path planning for multiple robots. *Robotics and Autonomous Systems*, vol. 23, no. 3, pp. 125–152, 1998. ISSN 09218890.

[64] Ding, X.C., Rahmani, A.R. and Egerstedt, M.: Multi-UAV convoy protection: An optimal approach to path planning and coordination. *IEEE Transactions on Robotics*, vol. 26, no. 2, pp. 256–268, 2010. ISSN 15523098.

[65] Versteegt, H. and Visser, H.: Traffic complexity based conflict resolution. *Air Traffic Control Quarterly*, pp. 1–11, 2003.
Available at: `http://arc.aiaa.org/doi/pdf/10.2514/6.2002-4443`

[66] Cap, M., Novak, P., Kleiner, A. and Selecky, M.: Prioritized Planning Algorithms for Trajectory Coordination of Multiple Mobile Robots. *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 3, pp. 835–849, 2015. ISSN 15455955. `arXiv:1409.2399v1`.

[67] van den Aardweg, W.: Robust Sampling-Based Conflict Resolution for Commercial Aircraft in Airport Environments. *University of Stellenbosch*, , no. March, 2015.

[68] Chiddarwar, S.S. and Babu, N.R.: Coordination Strategy for Path Planning of Multiple Manipulators in Workcell Environment. *Automation and Robotics in Construction &#8213; Proceedings of the 24th International Symposium on Automation and Robotics in Construction*, , no. iii, 2017.

[69] Bennewitz, M., Burgard, W. and Thrun, S.: Optimizing schedules for prioritized path planning of multi-robot systems. *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 1, pp. 271–276, 2001. ISSN 10504729.

[70] Jamoom, M.B., Joerger, M. and Pervan, B.: Unmanned Aircraft System Sense and Avoid Integrity and Continuity : Uncertainty in Aircraft Dynamics. vol. 39, no. January, pp. 1–10, 2016. ISSN 0731-5090.

[71] Weitz, L.A.: Derivation of a Point-Mass Aircraft Model used for Fast-Time Simulation MITRE Technical Report. vol. 7013, no. April 2015, pp. 1–27, 2015.

[72] Kim, K.-Y., Park, J.-W. and Tahk, M.-j.: UAV collision avoidance using probabilistic method in 3-D. *2007 International Conference on Control, Automation and Systems*, pp. 826–829, 2007.
Available at: `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4407015`

[73] ZHUANG, H.-z. and DU, S.-x.: On-line real-time path planning of mobile robots in dynamic uncertain environment. *Science*, vol. 7, no. 4, pp. 516–524, 2006. ISSN 1009-3095.

[74] Munoz, C., Narkawicz, A. and Chamberlain, J.: A TCAS-II Resolution Advisory Detection Algorithm. *AIAA Guidance, Navigation, and Control (GNC) Conference*, , no. 16, pp. 1–12, 2013.
Available at: `http://arc.aiaa.org/doi/abs/10.2514/6.2013-4622`

[75] Honeywell International Inc.: TCAS I Collision Avoidance System Pilot's guide. Tech. Rep., Honeywell, 2006.