Author:
Bonnell, William D

Title:
Development of Flight Rules and Distributed Traffic Management for Autonomous UAVs

Development of Flight Rules and Distributed Traffic Management for Autonomous UAVs

By

William Bonnell

T-BPHASE
FARSCOPE CDT
University of Bristol & University of West England

A dissertation submitted to the University of Bristol
in accordance with the requirements of the degree of
Doctor of Philosophy in the Faculty of Engineering.

July 2023

Word count: 37,175

Abstract

This thesis investigates how to ensure safe, efficient, and fair operations for autonomous unmanned aerial vehicles (UAVs) in the future civilian use case of point-to-point goods delivery. If UAV fleet sizes grow to predicted levels, then centralised traffic control, similar to traditional aviation, would likely be infeasible. As such we first develop a tactical-level conflict management method that uses velocity obstacles and a 'right hand' rule to produce accelerations that steer UAVs such that two UAVs can pass each other with some safe separation and ensures scalability. We then show in simulation that, while this method is effective for many initial conditions, when the angle of approach between two UAVs is small, one of the UAVs experiences a much longer delay than the other, despite applying the same manoeuvre. From this result we develop a hybrid avoidance method where the UAV can choose from a set of avoidance behaviours based on the relative position and velocity of its neighbour. We then explore how this tactical conflict management performs in a large-scale setup defined by a set of origins and destinations that form streams of UAV traffic. These streams form crossings, around which the UAVs will need to engage in avoidance manoeuvres and thus incur delay compared to a straight-line path. We use this delay to characterise the performance of these setups for various demand levels and show that, as the demand increases toward some maximum, the delay increases rapidly and non-linearly. From this we also show that, at these high demand levels, when we split a traffic stream into two parallel streams we can decrease the average delay, despite having increased the number of crossings. Finally we show how we can improve these large-scale traffic setups by designing three high-level traffic management methods. These methods are *waypoint-defined routes*, *floor field zones*, and *platoon formation*.

Dedication and Acknowledgements

“It takes a village to raise a child”, and to produce a thesis, it would seem. I would like to thank all those who have supported me in this endevour. To my parents, for providing me with the space and support to thrive, and develop in to the person I am today. To my grandmother, who instilled in me a curiosity about the world around me and the joys of learning. To my friends, old, new, and ‘as yet unmet’ for challenging me and offering a wide variety of perspectives. To my supervisor Eddie, for his invaluable input, limitless patience, and spurring me to always do better.

Above all though, to my partner Elena, for talking to me when everything got too much, for always telling me how proud of me she was, and for generally putting up with my nonsense.

This work was funded and delivered in partnership between the Thales Group and the University of Bristol, and with the support of the UK Engineering and Physical Sciences Research Council Grant Award EP/R004757/1 entitled ‘Thales-Bristol Partnership in Hybrid Autonomous Systems Engineering (T-B PHASE)’.

Author's declaration

I declare that the work in this dissertation was carried out in accordance with the requirements of the University's *Regulations and Code of Practice for Research Degree Programmes* and that it has not been submitted for any other academic award. Except where indicated by specific reference in the text, the work is the candidate's own work. Work done in collaboration with, or with the assistance of, others, is indicated as such. Any views expressed in the dissertation are those of the author.

SIGNED: William Bonnell DATE: 08/09/2022

TABLE OF CONTENTS

List of Tables

List of Figures

Chapter 1

Introduction

Unmanned aerial vehicles (UAVs) were developed throughout the 20th century alongside manned aviation. For much of their history UAVs have been primarily used for military missions ranging from simple target drones through to highly sophisticated surveillance platforms. The recently retired MQ-1 Predator [1] is an example of the latter and perhaps the quintessential UAV for a certain generation which grew up seeing these represented in popular culture, e.g., the video game Call of Duty [2], and who closely associates them with both the Iraq and Afghanistan wars. This UAV was capable of flying hundreds of kilometres while being piloted remotely using satellite communications and live camera feeds. However, as is often the case, as this military technology matured it became cheaper and smaller until it developed into a commercial product. In 2010 the Parrot AR UAV [3] was released with a retail price of $300 and was the first UAV capable of being piloted with a smartphone or tablet computer connected by Wi-Fi. Marketed as a toy, the Parrot AR and similar products can be flown by amateurs with little to no prior experience, a feat only made possible by the electronic flight controller and a range of onboard sensors.

There are ongoing research efforts to further add to a UAV's sensing and thus autonomous capabilities. For example, computer vision and machine learning, active areas of academic research in their own right, have already contributed greatly to the nascent civilian UAV markets. By 2016, the Chinese commercial drone manufacturer DJI had released UAVs with the ability to sense and avoid obstacles in five directions and with "active tracking" [4]. The latter feature allows a UAV to follow a designated person or object using its cameras. Another group, from the Flying Machine Arena in Zurich, demonstrated UAVs working collaboratively to throw and catch an inverted pendulum [5]. Here machine learning techniques were used to perfect the catching manoeuvre. Another example is work being done at the Bristol Robotic Laboratory to provide

UAVs with tactile sensing appendages for navigation in cramped, Global Navigation Satellite Systems (GNSS) denied environments such as wrecked buildings [6].

As these and other technologies continue to extend UAVs' capabilities so too do the potential use cases for UAVs. In particular, while UAVs are used in several industrial sectors already (see Section 1.1), they are usually piloted remotely and within visual line of sight (VLOS) of an operator in charge of a single aircraft. In order to unlock the true potential of UAVs it will be necessary to facilitate large numbers of autonomous UAVs capable of flying beyond-VLOS (BVLOS) on a regular basis. For that to happen we need to develop a robust set of traffic management methods that can handle the high number of UAVs that will be sharing an airspace that some use cases will require.

1.1 Background

UAVs are poised to become an integral part of the global economy with the global UAV service market predicted to be worth $40.7 billion by 2026 [7]. PWC predicted in 2018 [8] that the widespread adoption of UAVs has the potential to boost the Gross Domestic Product (GDP) of the UK by £42 billion and produce net savings of £16 billion across the economy by 2030. A more recent report from PWC [9], published in July 2022, has revised this prediction to £45 billion, though this is using a "best case drone adoption" assumption. In the more recent report they state that for these benefits to be realised there are many challenges still to be overcome including issues around technology, legislation, and perception.

This economic potential is due in part to several technologies maturing together including AI, machine learning, computer vision etc. This is called the 4th industrial revolution [10] and is marked by greater integration of systems and autonomous capabilities. Without these supporting technologies, UAVs would likely achieve a much smaller impact. All of this is to say that the UAVs themselves do not represent any great leap forward. Instead it is the way in which the UAVs are used and integrated with other technology that will make UAVs game changing to so many industries.

The idea of using UAVs outside of their original military context is not a new one. A Wall Street journal article from 2006 [11] highlights the possibility of using UAVs for domestic surveillance and disaster relief. The Yamaha R-MAX remote controlled helicopter was designed in the 1990's to be used by Japanese farmers for tasks such as precise crop dusting [12]. However, the range of civilian uses for UAVs has exploded as the supporting technologies have matured. A survey from Shakhatreh *et al.* [13] provides a review of potential applications for UAVs along with challenges and research trends. Some of these application include; search and rescue, precision agriculture, remote sensing, construction and infrastructure inspections, providing wireless coverage, and delivery of goods. Some of those applications which have been used commercially already will be explored in more detail below.

It is important to note that not all uses of UAVs enjoy universal public support. For example, Nesta, a UK based innovation charity, completed the first phase of its “Flying High Challenge” (FHC) in 2018. In an accompanying report [14], they discuss the the outcomes and learning as a result of testing five use cases with five different partner cities, with each city trialling one use case each. What sets the FHC report apart from many other similar initiatives is that the focus was not only on the technical use case itself. The report also explicitly considers the nature of the geographical areas where these trials took place and the stakeholders involved, including municipal authorities and the general public. The report outlines that while there is appetite for the use of commercial UAVs in cities, both the general public and municipal authorities seem to be more interested in use cases that present a clear social benefit: for example, the use of UAVs in medical logistic networks, an area which has been explored by recent work from the University of Southampton [15, 16].

Despite the marketing from some sectors of the UAV industry, the participating regions all seemed sceptical of more speculative uses of UAVs such as ‘urban mobility’ (flying taxis). The FHC report highlights how these have the potential to be introduced as premium services, excluding those who might be most heavily impacted by the associated drawbacks, for example by living next to a noisy UAV depot. This problem could however be mitigated in the long term through ensuring both municipal authorities and the public are engaged in the development of UAV regulations.

Regardless of public opinion, the benefits of UAVs are already being seen in several industrial sectors. A report published by PWC [8] explores how the use of UAVs for the inspection of oil and gas rigs has become common place in the energy industry. One of the major benefits of using UAVs, as in their original military settings, is the removal of personnel from dangerous situations. However, the benefits go beyond an increased level of safety as the inspection process itself is simplified. The traditional method for the inspection of an oil rig’s underdecking, with the associated scaffolding and people in harnesses, could take up to 8 weeks. A UAV can complete the same task in 5 days without exposing the operator to any risk. The use of UAVs for this kind of survey or inspection role can be applied to many other industries such as mining or agriculture. One of the main areas for improvement of the use of UAVs in these contexts is in developing “drone in a box” solutions. These represent the next stage for this sort of UAV operation where a UAV can be given an objective or set of tasks, plan how to accomplish its mission, take off from its “box”, complete the mission and return. This sort of solution could potentially eliminate the need to maintain the UAV itself with repairs and recharging being done autonomously, while the UAV is docked.

Another industrial sector where the use of UAVs is now commonplace is in media with well established companies such as the New York Times, which in 2016 published a list of five high profile stories that were enabled through the use of UAVs, and the network CNN, which established a team dedicated to the operation of UAVs [17]. This new tool for journalists has not been universally welcomed however. During civil unrest in the US, both in 2014 in the city of Ferguson [18] and at Standing Rock [17], the government had imposed no fly zones in order to limit the use of UAVs to capture footage of events. In the UK, the British Broadcasting Company (BBC) has published its own set of editorial guidelines [19] over the use of UAVs. This recognises the responsibility an organisation like the BBC has in terms of establishing best practice and ensuring the safe operation of UAVs. To that end any use of UAVs to gather material is subject to "senior editorial approval" and where necessary a "privacy impact assessment" will be carried out, reflecting the unprecedented ability UAVs have to breach people's right to privacy. The guidelines also recognise the ability the BBC has to influence behaviour as the use of any third party footage where the UAV is in breach of Civil Aviation Authority (CAA) guidelines must be justified "in the public interest". Finally, any use of UAVs in-house or by independent production companies must be carried out by a certified pilot with a Permission for Commercial Operations (PfCO) provided by the CAA. The BBC guidelines are indicative of current UAV use across sectors in the UK. UAV operations are strictly monitored, scrutinised and subject to approval by the relevant authorities. In October 2019 the UK CAA published "The Drone and Model Aircraft Code" [20] which outlines the rules for flying a UAV. Before taking off, a remote pilot must pass a theory test in order to be assigned a flyer ID and then register their UAV for a separate operator ID which must be displayed like a licence plate.

Both of these use cases have some important factors in common. When UAVs have been used for inspection or filming they are typically constrained to a specific area which is either controlled in some way, as with oil rigs and other heavy industry sites, or the operation has been given prior consent. Not only this but the operations are usually conducted within VLOS of a trained pilot who is in direct control of one or a small number of UAVs. Many of the future use cases being developed will not conform to these restrictive mission parameters.

One such use case, and perhaps the best known at least among the general public, is the use of UAVs as delivery robots, something PWC suggests could be "business as usual" by 2030 [8]. Ever since Amazon founded "prime air" in 2016 the idea of having a UAV deliver small packages, or even food, direct to your door, has captured the public's imagination. One ambitious potential future is for an entirely automated logistic chain. The idea suggested in [8] is that autonomous trucks could be loaded by purpose built robots at warehouses. These trucks would then drive around and release autonomous UAVs to deliver the packages and then return to the truck for resupply and recharging before making another delivery. UPS demonstrated a similar, if less ambitious, concept in 2017 [21]. In the demo a UPS truck is driven by a human who can then deploy a UAV to deliver a small parcel to a property that would otherwise require a lengthy detour. While the UAV delivers its package the human can complete other deliveries nearby until the UAV returns. This sort of UAV-truck partnership takes advantage of the complementary features of the road vehicle and the UAV: for example, the much higher carrying capacity of the road vehicle and the UAVs' ability to reach customers who are otherwise inaccessible. In operations research it has also led to a new variant of the travelling salesman problem (TSP), namely the TSP with drone (TSP-D) [22–24]. Note that this sort of approach to UAV goods delivery is unlikely to produce high density UAV traffic. The road vehicles are often considered to only carry one UAV and this delivery scheme is best suited to sparsely populated rural or suburban areas which UAVs could not reach from a warehouse due to their limited range. Combined with the added complexity of modelling the road vehicle side of this problem, we will not consider this particular approach to UAV goods delivery in this thesis.

While a fully automated supply chain may represent a long term goal, UAVs have already been used for delivery on a routine basis. Zipline, a California based UAV start-up, was founded in 2014 and signed an agreement with the Rwandan government in 2016 to begin delivering medical supplies to remote hospitals [25]. Despite an impressive programme of modernisation over the past 30 years, leading to 95% of the country being covered by a 4G wireless network, the challenging terrain makes ground based delivery a time consuming process.

Delivering blood via the road network can take up to 5 hours and lead to around 7% of blood packs in Rwanda expiring. Zipline's solution is for drones to deliver blood packs to hospitals throughout the country from two distribution centres, see Figure 1.1. These distribution centres, or "Nests" as Zipline has dubbed them, can then receive requests for supplies from partner hospitals as and when they are needed. Once an order has been placed, a modular UAV is assembled and loaded with supplies before being launched via an electric catapult. Once in the air the UAV will fly autonomously to its destination where it delivers its payload via parachute before returning to its "Nest".

In order to provide this service Zipline has had to ensure that it can fulfil requests safely. In order to do this the UAVs have a number of safety features including redundancy in major systems, such as a back up motor, and an emergency landing system that incorporates a self-deploying parachute. In addition to these mechanical features, the UAVs are equipped with Automatic Dependent Surveillance–Broadcast (ADS-B) receivers. ADS-B is used widely in manned aviation as a method for aircraft to transmit their positional data for use by ground-based traffic control or for self-separation with other aircraft. The receivers on Zipline's UAVs allow them to detect other manned aircraft and adjust their flight path if needed. We could suppose a future then where all UAVs are equipped with ADS-B transceivers and are treated the same as traditional aviation. This is unlikely to be the case, however, as ADS-B has been designed to operate over hundreds of kilometres with high transmission powers. As such the rate of ADS-B message loss has been shown to increase along with the number of aircraft [26], see Figure 1.2. Also Strohmeier [26] has shown that proximity to the receiver can actually result in a higher percentage loss of messages as other nearby aircraft can effectively "blind" the receiver. The trade off between transmission power and traffic density is further explored by Guterres [27] who explored how both parameters need to be balanced to ensure ADS-B receivers are not saturated by traffic that poses no real threat to each other. Due to these challenges, other technologies are being

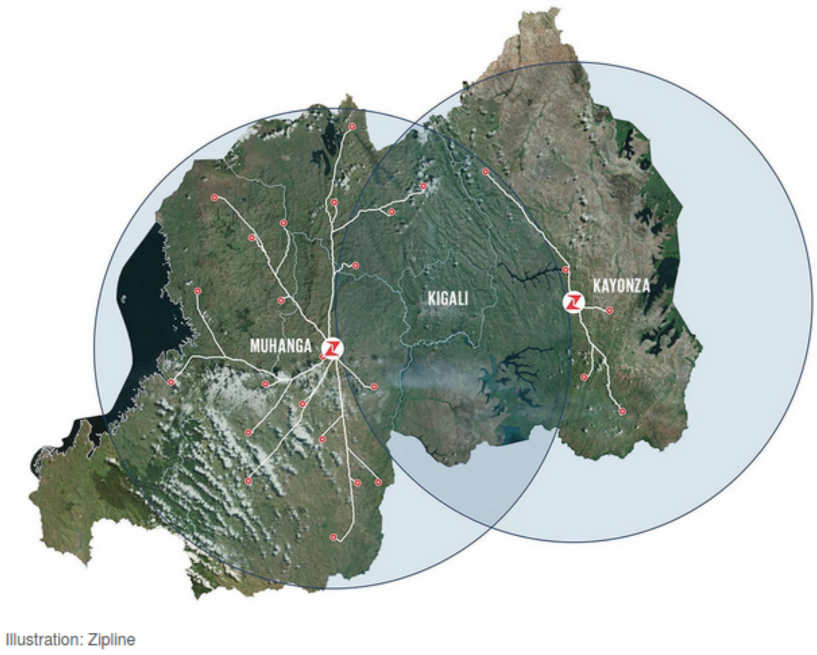


Figure 1.1: Zipline distribution hubs in Rwanda and the areas they serve. Reproduced from Ackerman, 2019 [25].

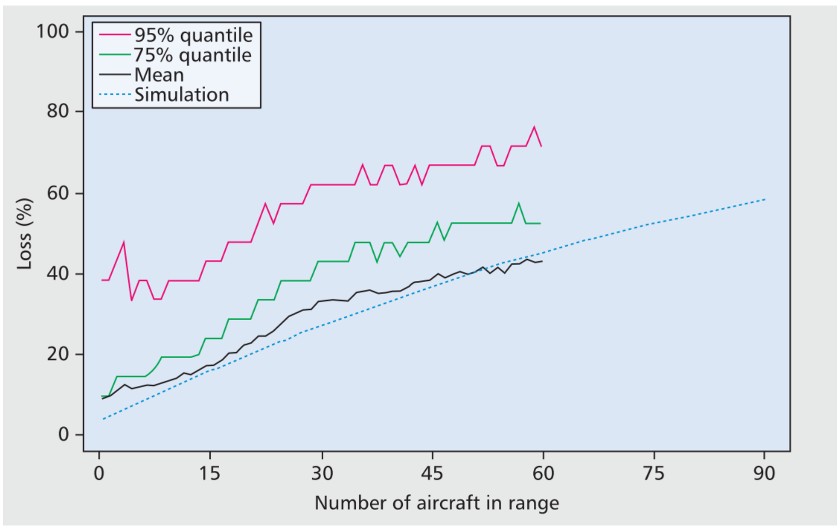


Figure 1.2: The percentage loss of ADS-B broadcasts increases as the number of other nearby senders increases. Reproduced from Strohmeier *et al.*, 2014 [26].

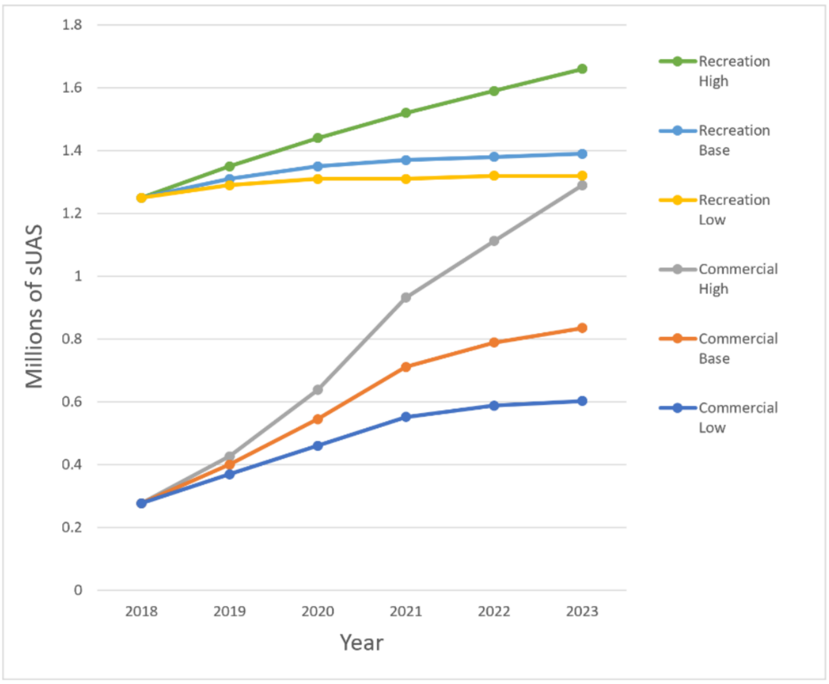


Figure 1.3: Predicted numbers of recreational and commercial UAVs in the USA. Reproduced from FAA, 2019 [32]. The low end of these predictions is in keeping with the most recent data that suggests there are 0.6 million commercial UAVs in the USA in 2021 [33].

investigated for enabling communication between UAVs such as Wi-Fi [28] which, along with cellular communications technology such as Long-Term Evolution (LTE), are also being considered for autonomous cars [29]. However, even if the problem of providing UAVs, and other airspace users, with localisation data in a timely manner can be solved it is still necessary to develop the rules that will dictate how this information is used.

Zipline works then by effectively treating its UAVs as traditional manned aircraft and does not involve some of the other challenges faced by UAV delivery operations envisaged for the future. The most stark difference between Zipline's operation and other potential UAV delivery services is scale in terms of fleet size. Predictions of future demand are highly uncertain, but some authors predict dramatic growth in the sector, for example [30] suggests 87,000 concurrent UAVs above Paris (France) by 2035 and [31] forecasts that 32,887 deliveries a day in Sendai (Japan) will be targeted for UAV delivery. In 2019 the FAA predicted that the overall fleet of commercial UAVs would increase from about 0.3 million to between 0.6 and 1.2 million by 2023 [32]. The most recent data [33] shows that in 2021 there were around 0.6 million commercial UAVs in the USA, in line with the lower end of the predicted fleet size. It should be noted though that between the predictions in [32] and the latest data in [33] the global Covid-19 pandemic occurred which has negatively impacted many industries and slowed economic activity globally.

Not only does this represent a significant increase in the number of UAVs (Zipline Nests having flown around 20-30 missions per day) but also a much denser traffic scenario. This is partly due to the restrictions on how an operator can fly their UAV in many countries. In the UK [20], EU [34], and US [35], this includes a flight ceiling of about 120 m above ground level, which ensures safe separation from most traditional aviation. There is also a desire to maximise separation between UAVs and people or property, $50-150$ m in the UK depending on the circumstances. This is done not only for safety reasons but also for concerns around privacy and noise pollution. Therefore we might expect that in such a scenario the UAVs would be required to fly close to their respective ceilings. In [36] Jiang discusses a layered urban airspace, proposed by Amazon, in which UAVs operate in two layers, a low-speed layer and a high-speed layer above it. The maximum flight ceiling in this scenario for the high-speed layer is the same as the current regulatory limits with the low-speed layer having a flight ceiling equal to half of this. In such an airspace scenario we expect that we could model the UAVs in a pair of planes that may interact as UAVs transition between layers.

The case studies used in [30, 31] also consider a much more urbanised airspace compared with the airspace Zipline currently operates in. Operating any UAV in such an airspace, let alone in large numbers, comes with a new set of challenges. One such challenge is ensuring the integrity of Global Navigation Satelite Systems (GNSS) in an urban environment, see [37] for a review. GNSS has become a cheap and ubiquitous localisation technology which UAVs are already using today. As such it will be important that the data provided by on-board GNSS receivers is accurate despite errors introduced by effects such as multi-path interference and non-line of sight (NLOS) reception. Both of these phenomena are the result of how the structures in an urban environment are more likely to reflect GNSS signals. In the case of NLOS reception, tall buildings can block the direct path of a GNSS satellite to the receiver on a UAV. However, the signal still reaches the UAV by taking a longer path, where it is reflected off other buildings, and thus the UAV will report that it is further away from the satellite than it actually is. Multi-path interference occurs when it receives the same GNSS signal several times as some signals take reflected paths. Both phenomena are discussed in more detail by Groves *et al.* [38].

Another potential issue with operating in an urban environment is the potential for the airspace to be more restrictive. For example, in [14] one of the case studies explored was delivering medical supplies between London hospitals. Three potential routes were identified. The shortest straight line path fully realises the benefits of a UAV based delivery system, avoiding the congested London streets, but involves flying directly over people, vehicles and property. The second option was to follow train tracks and the third was to follow the Thames. Both of the latter two reduce the risk to the general public at the cost of longer flight paths and further increasing the traffic density. There is also the issue that an urban airspace has the potential to be much more dynamic than traditional airspace. For example, [14] considers a case study where UAVs are used for accident response by emergency services which might involve setting up temporary 'no-fly' zones for other UAV traffic. This, or other features of an urban environment such as construction work, could further limit the usable airspace and exacerbate issues around traffic density.

Zipline and their operation in Rwanda also benefited from an effective monopoly on UAV delivery. This is unlikely to be achieved in a busy urban environment where many of the established parcel delivery companies and new start-ups alike will be competing not just for customers but for access to the airspace. Having many different UAV operators like this raises questions around how they will interact. One way to address this problem is to develop traffic control for UAVs. Highlighted as one of the main challenges for the deployment of UAV delivery services in [13], the development of traffic control, or management as we will call it, is essential as it will provide the framework for deciding how UAV operators, UAVs and other stakeholders will interact with each other. This will cover not only how access to airspace is gained but also how low level avoidance is handled and other safety considerations that need to be addressed for policy makers to allow UAV operations on the scale suggested by papers [30, 31] and the report [33].

Given the potential economic impact of UAVs there has been much discussion in the policy space around the development of regulation and legislation that will allow the use of UAVs to be accelerated. In [39] the World Economic Forum (WEF) defines “advanced drone operations” as any operation which takes place close to people, is carried out by autonomous UAVs, requires routine BVLOS operation or some combination of these. Note that UAV goods delivery as discussed earlier likely requires all of these. The report then goes on to present the “performance based regulation” model which focuses on risk rather than certification. By considering mission elements such as environment, goals, flight systems etc. regulators can engage in conversation with potential UAV operators about what risks are associated with a given mission and how best to mitigate them. It is thus necessary to have a framework to discuss and quantify risk and this is where something like the Specific Operations Risk Assessment (SORA) [40] can be used.

Frameworks for the traffic management of UAVs, or Unmanned Traffic Management (UTM), soon followed with authorities in the US [41], UK [42] and EU [43] each producing reports outlining their UTM vision in 2020. Common to all the proposals, the UTM system is expected to deliver a wide range of services to UAV operators and that these operations will take place in low level airspace in the near future, effectively segregating UAVs from traditional airspace. These services cover all aspects of a UAV operation including registering operators, formulation and submission of flight plans, providing information about weather and other environmental factors and ensuring safety. All of these functions will be provided by UTM service providers (UTM-SPs). The architecture for how these UTM-SPs interact is one of the main areas where the three proposals differ. The two main architectures proposed are *centralised* and *federated*. In the former architecture one of the UTM-SPs, possibly setup as a sub-division of a regulatory body, becomes the main arbiter and repository of information in the system. In the latter, UTM-SPs are expected to interact with each other on an ‘as-needed’ basis to resolve issues.

While many challenges have been identified in establishing UTM, such as cyber security for communication between UTM-SPs and the UAV operators, this thesis will focus on the problem of ensuring that UAV operations are conflict free. Specifically, we are interested in how UTM can ensure that UAVs maintain some minimum separation between themselves and other UAVs or obstacles. In [42] “conflict management services” are split into two levels, *strategic* and *tactical*. Strategic conflict management is achieved through the use of “flight notice boards” or similar technologies. Flight plans are submitted ahead of time to the UTM-SP and logged on an internal flight notice board and a public flight notice board if a centralised UTM architecture is used. The flight plan can then be checked if it intersects with an existing, approved flight plan. If this is the case then the flight plan can be altered or negotiations between the affected UAV operators can be facilitated by the UTM-SP in order to produce new flight plans that ensure UAVs remain safely separated. On the other hand, tactical conflict management includes any action taken by a UAV in flight to maintain safe separation. To facilitate this, UTM-SPs will be expected to monitor UAVs and provide possible updates to their flight plans, i.e., alternative routes, should the need arise. The report also explicitly calls for “rules of the air” so that consensus on how a flight plan should be changed can be reached in a timely manner.

This multi-level approach to conflict management is clearly inspired by traditional air traffic management (ATM). In manned aviation the International Civil Aviation Organisation (ICAO) includes in its definition of ATM the goal that it should ensure the safety of both the air traffic and airspace [44]. This is achieved through conflict management, that is ensuring that traffic maintains a minimum separation with other aircraft and hazards. In ATM the conflict management is comprised of three layers: *strategic conflict management*, *separation provision* and *collision avoidance*. Strategic conflict management is the highest level of conflict management and often takes place before an aircraft has even taken off by adjusting the flight plan, e.g., by delaying the aircraft. Separation provision is the means by which aircraft “keep away from hazards” while in flight and may consist of different separation modes. Finally, collision avoidance is any action taken to avoid collision when the “separation mode has been compromised”.

ICAO also provides a set of rules so that collision avoidance can be carried out by the pilots either in emergencies or in the absence of ATM. The collision avoidance is based on a set of right of way (RoW) rules for aircraft [45]. From [45], the aircraft with right of way “shall maintain its heading and speed”. Three potential collision scenarios are identified, “head-on”, “converging” and “overtaking”, along with which aircraft has the RoW in each, see Figure 1.4. In both “head-on” and “overtaking” the manoeuvre to be undertaken by the aircraft without RoW is prescribed such that it must “alter its heading to the right”. A set of similar rules for UAVs would be useful for facilitating autonomous UAVs since it would limit the need for negotiation.

In [41–43] strategic conflict management is explored in greater depth than tactical management. Perhaps in the near term, when traffic density is still small, a strategic level approach to conflict management may be sufficient on its own. An example of such a system is used in [47] in order to compare different principles for route design (free choice, over buildings, over

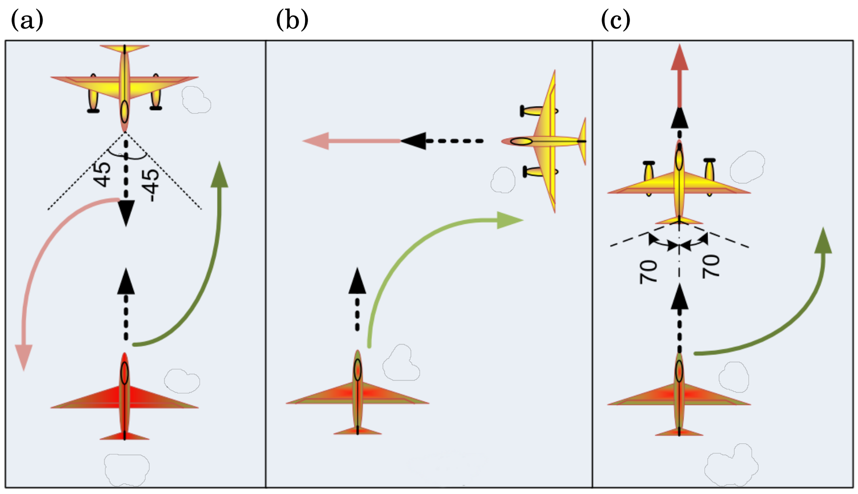


Figure 1.4: ICAO RoW scenarios [45]. (a) "Head-on": neither craft has RoW and therefore both must turn to their right. (b) "Converging": the craft with the neighbour on its left has RoW and the other craft must now turn to its right. (c) "Overtaking": the craft being overtaken has RoW and the overtaking craft must turn to the right and when completing the overtaking manoeuvre. Reproduced from Alturbeh *et al.*, 2020 [46].

roads). However, if the level of UAV traffic approaches the estimates from [30, 31] then we expect this centralised scheduling approach to become infeasible. The large number of UAVs operating within a confined airspace, again likely due to low flight ceilings and a requirement to maximise separation with people or property, will not only increase the likelihood of flight paths intersecting but also limits the ability for UTM-SPs to resolve the issue through re-routing. A decentralised approach also offers the potential for UAV airspace to operate on shorter time scales than traditional airspace. For example, in the case of consumer UAV delivery services, the UAV operators may desire quick approvals for flight plans submitted on an ad hoc basis. This reduced operational horizon will only compound issues introduced by high traffic density.

Even if we assume that it is possible to satisfy the scheduling problems of conflict free flight plans, it is still necessary to consider what UAVs should do in "off nominal" situations. In other words, what should a UAV do when it can no longer conform to its pre-approved flight plan? This could happen for any number of reasons, such as in response to dynamic flight restrictions imposed by emergency services or the detection of a malfunctioning UAV. In such a scenario the UAVs will require tactical conflict management in order to maintain safe flight. Whatever the reason for a UAV to deviate from its original route, it may also be beneficial to develop a response that can be applied in a distributed way. A centralised approach to handling this sort of low level conflict management requires a near real-time link to the UAV in order to have detailed situational awareness and then provide commands in a timely manner. This sort of link might be difficult to ensure as traffic levels rise in the near future, especially without large infrastructure investment in supporting wireless communications etc. One solution to this problem, and the one this work will focus on, is to have autonomous UAVs capable of carrying out conflict management through on-board sense and avoid (S&A) capabilities.

Whether a system uses strategic or tactical conflict management, or a mixture of both, there is also an open question about what the airspace UTM is responsible for should look like: in other words to what degree should it be structured. It has been shown that airspace capacity can be increased by enforcing some level of structure on the airspace [48, 49]. Other studies have explored different kinds of airspace structure such as lanes [50], or compared single-layer airspace with multi-layer [51]. These works show that the airspace structure and the method of tactical conflict management used by the UAVs are both important for the overall performance of a UTM system.

1.2 Research aims and assumptions

Based on the above discussion of future UAV operations and the need for a UTM system we summarise the overall aim of this thesis as:
Develop a set of traffic management methods for large scale, low flight ceiling UAV operations and test them in simulation. These methods should be computationally simple and scalable in order to facilitate large numbers of autonomous UAVs capable of avoiding one another while in flight.

We hope this can contribute to the ongoing discussion concerning the legislation that will govern this emerging transport technology. This thesis however will consider stylised traffic scenarios in order to remove some of the complexity associated with fine detail models of a real-world airspace. The idea then is to develop our understanding using parsimonious setups, and then try to apply this understanding to setups with increasing complexity. We therefore make the following assumptions about the UAVs, their mission and the environment in which they will fly, all of which will be used to construct the various models we use throughout this thesis.

A1 All UAVs will be engaged in delivering small packages between fixed locations.

As we have discussed in Section 1.1, there are many potential applications for UAVs in the future. We have chosen to focus on this sort of ‘point-to-point’ delivery use case for two main reasons. Firstly, this use case has the potential to result in high density traffic scenarios where traditional, centralised control (e.g., scheduling) will be infeasible. Secondly, this use case is easy to model as UAVs only need to take off from one point and arrive at another. Other use cases often require UAVs to interact with other objects that would also need to be modelled, e.g., the TSP-D which has delivery trucks as well as UAVs. We also specify “small packages” as we do not consider urban mobility. This is because urban mobility is likely to be subject to separate and more stringent regulation due to the significant risks associated with transporting humans.

A2 All UAVs will be small, lightweight, and use multi-rotor propulsion.

One of the many potential business models for UAV delivery is where each UAV delivers one package per trip. This is particularly suited to urban and suburban areas where UAVs can complete deliveries faster than conventional delivery methods over relatively short distances. This allows the UAVs to be small and lightweight which reduces the cost of each UAV and can improve fuel efficiency. This also means that the UAVs should be safer in case of failure.

We also make the assumption that all of the UAVs are of the multi-rotor type for two main reasons. Firstly, multi-rotor UAVs are capable of vertical takeoff and landings. This capability will be vital in urban settings where space for building runways will be limited. Secondly, multi-rotor UAVs are typically more manoeuvrable than fixed wing UAVs and as such will allow us to model the dynamics more simply, see Section 2.2 for more details.

A3 All UAVs will fly at some regulator imposed maximum flight ceiling but are otherwise allowed to move freely throughout the airspace which will be free of obstacles except for other UAVs.

As discussed in Section 1.1, the current rules that govern UAV use in many countries restrict UAVs to some maximum flight ceiling. This is likely to be true for future UTM as well, at least in the near term, because of the desire to separate UAVs from conventional aircraft. We assume then that UAVs will fly at this maximum altitude whenever possible as this will maximise separation between the UAVs and the ground. Maximising this vertical separation is likely to be desirable for many reasons including safety, privacy, noise pollution etc.

If UAVs are required to maintain a cruising altitude near this flight ceiling, then we suggest that modelling the UAV dynamics in a 2D plane is an appropriate first step. This is because if a UAV descends from the cruising altitude in order to avoid a neighbour it will then have to ascend back to the flight ceiling. It has been shown, for example by Liu *et al.* [52], that a multi-rotor UAV requires more power on average during ascent than while hovering. Therefore, we suggest that significant changes in the altitude should be reserved for landing at a destination or during an emergency situation in order to ensure the energy efficiency of the UAVs. We also make the assumption that this airspace will be free of other obstacles. This may not be the case in a busy urban airspace but it is both the simplest case to model and best realises the advantages of UAVs over traditional road-bound delivery methods.

A4 All UAVs will be identical in their capabilities, including their dynamics and communications.

We make this assumption to keep the conflict avoidance process simple. We can imagine many cases where UAVs might deviate from some basic avoidance scheme, e.g., to give way to emergency vehicle UAVs, however here we aim to develop a scheme which most UAVs will apply most of the time as a sort of base case. This assumption will also make simulation easier as only one type of UAV 'object' will need to be defined. Furthermore, again for simplicity, we assume that the dynamics and communication between UAVs is 'perfect'. That is, they are not subject to failure or noise.

The effect of these assumptions overall is to greatly simplify the problem of developing conflict avoidance for UTM in general. It does this by restricting the scope of this work to one particular use case and type of UAV while also simplifying the details associated with both. Also, due to our interest in large fleet sizes this thesis will be constrained to simulation which further limits the direct applicability of this work. However, we hope that this can form the basis for future work in which some of the complexities associated with a real-world setup can be added either in simulation, or through experiments with real fleets of UAVs.

1.2.1 Research questions

From the aim and assumptions, as stated above, we propose the following set of research questions:

RQ1 How can two UAVs be made to avoid each other safely while also ensuring efficiency and fairness?

As discussed in Section 1.1, UAVs will always need a tactical conflict management method that can be implemented to avoid other UAVs in the absence of input from their UTM-SP, at least in an emergency. Therefore, we will attempt to answer **RQ1** by developing a set of avoidance rules that can be applied by all UAVs operating within a given airspace. This decentralised approach, where each UAV is responsible for its own avoidance, should also ensure that the conflict avoidance is scalable, as stated by the overall aim above.

While it is crucial that UAVs avoid one another mid-flight, it is also important to consider how doing so affects the UAVs in other ways. When one UAV avoids another it will deviate from some original trajectory. When this happens the UAV will usually experience some delay, i.e., it will arrive at its destination later than it would have. The magnitude of this delay will provide the basis for our evaluation of the efficiency of an avoidance manoeuvre. We consider a smaller delay to be more efficient than a longer delay. Fairness can then be defined by comparing the delay between UAVs. Given the assumption that all UAVs are identical we consider a fair avoidance manoeuvre to be one where both UAVs experience the same delay.

RQ2 How does the performance of a UAV traffic scenario, where UAVs implement some tactical conflict avoidance, depend on traffic demand and other factors?

While **RQ1** focuses on a pair of UAVs, the use case that this work targets is likely to involve large fleets where many UAVs will need to avoid each other simultaneously. As such, to answer **RQ2** we will first develop a traffic scenario where two streams of UAVs intersect to form a crossing around which the UAVs will need to engage in avoidance manoeuvres. We will still use efficiency and fairness, based on an incurred delay, to quantify the performance for such a setup but we will now be interested in metrics such as the average delay and how the experience of UAVs in one stream compares to that of the other stream. We specify demand since we expect that system performance will deteriorate as demand is increased and if this is the case then exploring this relationship may give some indication as to what traffic levels can be supported by a UTM using only a decentralised avoidance scheme. We will also explore any emergent behaviour that may arise, for example, as a result of the avoidance rules and UAV dynamics.

RQ3 How can we supplement a tactical conflict avoidance scheme with other traffic management methods in order to improve the performance of a UAV traffic crossing?

Finally, in order to answer **RQ3**, we can explore whether adding different airspace structures or other, higher level, traffic management methods can improve the performance of the traffic setup used to answer **RQ2**. This is a similar concept to the work presented by Sunil *et al.* [48]. The idea is that these will be methods that the UTM can use to influence traffic without having to directly control any UAVs mid-flight and therefore preserve scalability.

1.3 Thesis structure

In order to address the research questions presented above, the rest of the thesis is structured into the following chapters. In Chapter 2 we aim to answer **RQ1** by first producing a simple dynamical model for multi-rotor UAVs. This model allows a UAV to both steer towards a desired direction and to avoid other UAVs by producing a number of acceleration components. Inspired by the mathematical models used in swarm robotics and pedestrian modelling, the model makes use of a simple summative scheme to combine the different acceleration components. With this model for UAVs in place, we then develop a tactical conflict management method based on the velocity obstacle method used in robotics which is further supplemented with a 'right hand' rule which is inspired by the RoW rules for traditional aircraft described in Section 1.1. We explore how this avoidance method performs for a pair of UAVs in simulation and then develop a hybrid avoidance method where a UAV can choose from three possible avoidance behaviours.

In Chapter 3 we aim to answer **RQ2** by first defining a model for UAV ports, i.e., the places where UAVs take off from and land at. These ports can then be used to define origin-destination networks which describe a particular UAV traffic setup. We then design simulations that use four of these ports to form a simple crossroads like setup where two streams of UAVs will cross at a point, around which the UAVs will need to avoid each other. For such a setup we define a performance metric in terms of a delay which the UAVs incur while avoiding each other compared with an ideal straight line path. We initially compare the simple and hybrid avoidance methods for a variety of demand levels but the demand experienced by each stream is the same. Next, we use only the hybrid avoidance method in setups where the demand on each stream is varied independently of the other in order to investigate how the demand experienced by one stream affects the delay experienced by the other stream. Finally, we explore how splitting a traffic stream into two parallel streams with the same total demand affects system performance.

In Chapter 4 we aim to answer **RQ3** by developing three distinct traffic management methods that can be applied at the port level to improve performance of the UAV crossing setup presented in Chapter 3. Crucially, these methods do not require the port to communicate with a UAV once it has departed. The first of these methods introduces the idea of intermediary waypoints that a port can use to define routes for a UAV such that they no longer take the straight line path between an origin and destination. The second method defines zones of the airspace which, when entered into by UAVs, will induce some collective motion. These zones will be defined at the port level and will have a velocity floor field associated with them which is given to UAVs on take off. The last method we develop stops UAVs from departing their origin until a small group of UAVs are ready to depart. This ensures that there are groups of UAVs with small gaps between each UAV in the group and then larger gaps between the different groups.

In Chapter 5 we begin to explore possible extensions to the work presented in previous chapters. This includes extensions to the dynamical models of the UAVs and experimental setups. This chapter is intended to provide a framework for how future work might be carried out but includes some preliminary results where possible.

Finally, in Chapter 6 we provide a brief summary of this thesis.

Chapter 2

Velocity Obstacle Based Avoidance Methods

As discussed in Chapter 1, one of the key components of an unmanned traffic management (UTM) system will be a tactical-level conflict management method. We suggest that the easiest way to provide this is for the UAVs themselves to carry out sense and avoid (S&A) manoeuvres based on a set of rules prescribed by the UTM system and assumed to be obeyed by all participants. The aim of this chapter then will be to attempt to answer **RQ1** as defined on page 14 and develop the rules that would be used in such a system.

In order to achieve this chapter's aim, we first need some idea of the UAV dynamics which will be controlled to carry out S&A manoeuvres as they interact. Therefore, a stylised model of UAV motion is presented. The model is based on each UAV experiencing various acceleration components generated by a set of rules that aim to either move a UAV towards a destination or maintain a minimum separation with other UAVs. With this model of UAVs in place, we will then design the conflict avoidance rules which we base on the velocity obstacle method and the current right-of-way (RoW) rules for manned aircraft.

Once the conflict avoidance rules have been developed, we will test their performance through simulations of two UAVs. In these simulations, the UAVs are set up such that if they followed a linear flight path, their minimum separation would be less than some predefined safe separation distance and thus they would have to subsequently avoid each other. The performance of the avoidance rule will then be judged on whether the minimum separation between the UAVs is maintained and the impact the rules have on each UAVs' journey in terms of the delay they cause, when compared to a setup where no avoidance manoeuvre would be needed.

In Section 2.1 we will put this chapter in to context and provide a summary of some of the techniques that inspired this work, such as the social force model from pedestrian flow theory, collision avoidance methods from robotics, and velocity obstacle methods. Then in Section 2.2 we will design the stylised dynamical model for UAVs that will be used throughout this thesis, which will include a method for steering UAVs towards a desired heading or destination and a simple method for combining acceleration components. The details of the conflict avoidance method we will design for a pairwise interaction between UAVs will be provided in Section 2.3. This section will explain how a UAV uses a velocity obstacle to determine if it and a neighbour will conflict with each other some time in the future, and then determine a new avoidance velocity to accelerate towards. We design the experimental setup for testing the conflict avoidance method in Section 2.4 and present simulation results in Section 2.5. Based on these results, Section 2.6 will explore extensions to the conflict avoidance method where the manoeuvre is altered by one of the UAVs deviating from the original default behaviour. This results in a hybrid conflict avoidance method which uses a simple heuristic to determine which version of the avoidance behaviour a UAV should adopt. Finally, in Section 2.7 we will provide a brief discussion and summarise the contributions of this chapter.

2.1 Background

Given the predictions for high levels of UAV traffic in the future [30, 31], we suggest that strategic conflict management, and in particular scheduling conflict-free flight plans, will become infeasible. Low flight ceilings and the desire to maximise separation between UAVs and pedestrians and/or property could lead to a highly restricted airspace which in turn will limit the ability of a UTM system to resolve conflicts prior to takeoff. If a conflict free schedule could be developed, then it is likely to be brittle and easily disrupted.

UAVs might deviate from the centralised schedule mid-flight due to uncontrollable factors. Anything that causes the UAVs to be delayed, and thus deviate from their original flight plan, could lead to a subsequent scheduling conflict that would need to be resolved. In order to resolve the new conflict other UAVs may be delayed, thus causing new scheduling conflicts which then propagate through the system. A ‘cascading failure’ effect such as this is also likely to be exacerbated in heavy traffic scenarios where the schedule will have few free slots. The mid-flight delays that lead to these failures could be caused by error or the environment. In the case of the former, this could be when the UAV experiences a malfunction with its propulsion or localisation systems, either of which could lead to the UAV being unable to adhere to its flight plan. Environmental factors that lead to delays could be as simple as inclement weather e.g., strong winds, or imposed by the UTM system itself in the form of dynamic flight restrictions (DFR). A DFR might be applied to areas where emergency service drones need to operate e.g., above a traffic incident, and other UAVs are forced to leave and/or avoid that area. We therefore propose that UAVs operating within such an airspace should be capable of S&A enabled by rules set by the UTM, similar to the RoW rules for manned aircraft developed by the ICAO [45].

Swarm robotics is an area that similarly must deal with large numbers of agents moving

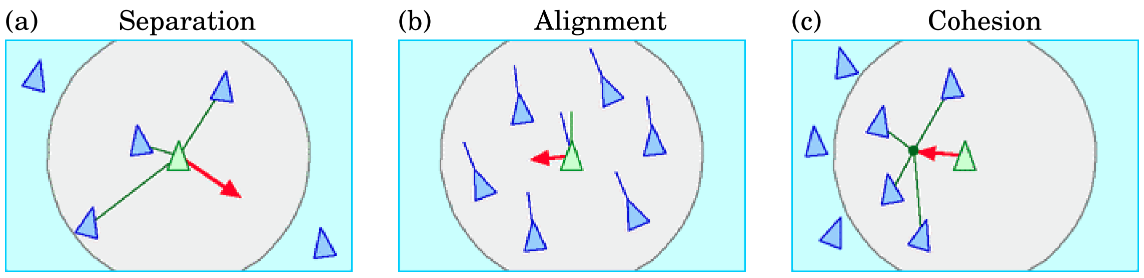


Figure 2.1: The three rules for simulating flocking behaviour developed by C. Reynolds. (a) Separation: steer to avoid crowding local flock mates. (b) Alignment: steer towards the average heading of local flock mates. (c) Cohesion: steer to move towards the average position of local flock mates. Reproduced from Reynolds, 1999 [55].

through an environment, often with limited processing power for each agent, from which we may learn some design principles. Reynolds [53] proposed three simple rules for generating complex collective behaviour that mimics large animal flocks. These rules are "separation", "alignment" and "cohesion", see Figure 2.1. Under these rules an agent will attempt to align its velocity with its neighbours while ensuring that it does not get too close to any particular neighbour but neither does it stray too far from the group. Reynolds' simple model has been very influential and undergone much development. For example, Vásárhelyi [54] tackles the problem of facilitating swarm behaviour in enclosed spaces. This is achieved through using evolutionary techniques to tune parameters and is also notable for being applied to a small, physical, swarm of UAVs. However, some of the assumptions made about agents in swarm robotics do not carry over to UAVs in a future airspace setting. The main difference is that in swarms the goal of each agent is usually the same, for example each agent is attempting to maintain the flock or achieve a desired formation. In a future airspace scenario many of the UAVs will have different, if not competing goals. For example delivery UAVs operated by different companies might compete for airspace, rather than collaborate to achieve smooth traffic flow.

Given this lack of shared goal, UAVs may be considered more similar to pedestrians. Generally pedestrians do not all share the same end goal and must move through the space and past each other, as is expected for UAVs. Helbing [56] presents the social force model (SFM) as a way to model pedestrians as being subject to different 'forces', see Figure 2.2. Some of these are repulsive, stopping individual pedestrians from getting too close to one another, while others are attractive, corresponding to goal destinations.

Both the social forces for pedestrians and the swarming rules are models that describe the behaviour of intelligent agents. In both cases these models are influenced by 'psychological' elements, for example, how close the humans or animals are willing to get to each other. While dynamical models for UAVs will not be subject to such considerations, these previous models still provide us with a useful mathematical framework for how we can combine different component behaviours. We will discuss this in more detail in Section 2.2 but essentially we can develop a dynamical model of UAVs in which steering toward some destination and avoiding their

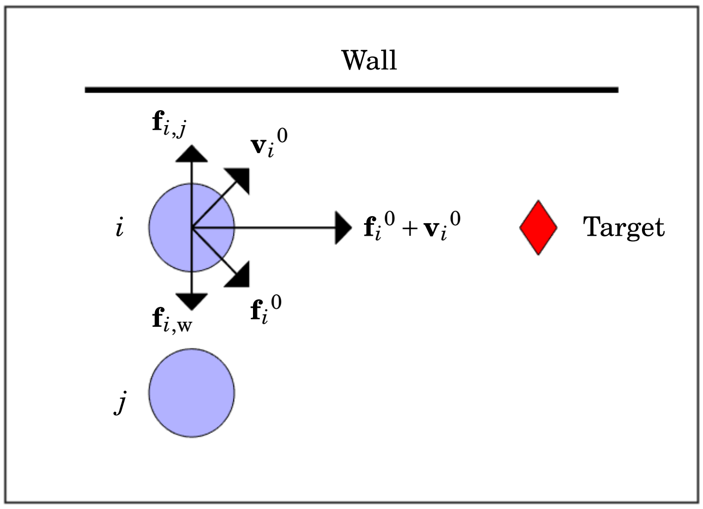


Figure 2.2: In the SFM [56], pedestrian i experiences an attractive force $\mathbf{f}_i{}^0$ that drives it towards its target. At the same time pedestrian i experiences the repulsive forces $\mathbf{f}_{i,j}$ and $\mathbf{f}_{i,\mathrm{w}}$ which drive it away from pedestrian j and the wall respectively. These different forces are then combined to produce an overall force that drives the motion of the pedestrians. Pedestrian j experiences its own set of forces which are usually different from i's: Newton's first law need not be obeyed.

neighbours are separated in to different processes, the results of which are then combined to produce some overall motion.

The problem of getting robots, agents, or aircraft to move through an environment without colliding with each other is by no means new. Albaker [57] identifies several types of collision avoidance for UAVs; “E-field methods”, “Optimised escape trajectory approaches”, and “Protocol-based decentralised collision avoidance”. The first two methods provide collision avoidance which can explicitly consider path planning as well so that the UAVs reach their end goal. For example, the E-field (also called the potential field) method was first proposed in [58, 59] where an agent constructs a map of its environment containing sinks and sources that correspond to things like waypoints and obstacles. This map can then be used to generate a potential field analogous to those produced by electrostatic charges, which the agent can then navigate through by gradient descent methods. Barraquand [60] describes a numerical potential field method which instead places a grid over the agent's map and assigns each cell a value depending on its proximity to the agent's target and whether it contains an obstacle. The simplest way to do this is to pick a cell as the goal and set its value to 0, then increase the value of each neighbouring cell by 1. Repeat this process for the whole grid and then any cell which contains an obstacle has its value set to some arbitrarily large number. This map can then also be navigated using a gradient descent method which both avoids obstacles and eventually leads to the end goal. One problem with simple implementations of a potential field method is the possibility for local minima to form which if entered into, an agent will be unable to escape from. Potential fields can also be easily applied to a distributed system, since each agent can construct its own potential field. The method has also been extended to be used in dynamic environments where the agent can sense changes in the environment in order to construct a new map in an iterative process. Another potential drawback, however, is the possibility that the paths resulting from these methods are infeasible. This can happen when the potential field is constructed without considering the kinematics of the agents.

Another class of collision avoidance methods that account for path planning are optimised escape trajectory approaches. These methods are grounded in model predictive control (MPC) and are abundant in robot navigation. At its core MPC is about producing a mathematical description of the system that can be used to predict what the future state of the system looks like depending on some initial conditions and control input. The input that produces a desired future state is then chosen. This allows the problem of collision avoidance to be framed as an optimisation problem with some constraints, i.e., design a path that leads to some goal while ensuring that the separation between the agent and other agents or obstacles is always greater than some minimum value. These methods can also be augmented with other constraints such as ensuring that the collision free path is also the most fuel efficient.

In order to produce the optimal path the problem of path planning and the accompanying constraints must be put into a suitable form. Typically this takes the form of a mixed-integer quadratic program (MIQP) where $\min(J) = \min \int_0^\infty (\mathbf{s}'\mathbf{Q}\mathbf{s} + \mathbf{u}'\mathbf{R}\mathbf{u})\mathrm{d}t$, subject to: $\dot{\mathbf{s}} = \mathbf{A}\mathbf{s} + \mathbf{B}\mathbf{u}$, which can then be approximated as a mixed-integer linear program [61–63]. The approximation is done to make solving the optimisation problem easier, since an MIQP scales quadratically with the number of avoidance partners. However some work has been done using MIQP for collision free path planning with multi-rotor UAVs since this leads to smoother trajectories [64].

Another method for simplifying the MIQP is to approximate it as a convex program. Sequential convex programming (SCP), as described by Schulman *et al.* [65], involves making the convex approximation over a short time window, solving the sub-problem and then repeating the process. Thus the agents involved must make many more computations compared to the classical MILP or MIQP approach but each computation is easier to solve. Morgan *et al.* [66] applied SCP to a swarm of small spacecraft. Here the SCP approach will produce a sequence of control inputs to achieve a desired end position with the optimal trajectory, where optimal refers to being collision free and with the least fuel expenditure.

The last class of collision avoidance identified by Albaker *et al.* [57] is “protocol-based decentralised collision avoidance”. For these methods some rules are developed offline that dictate how each agent in the system will act under certain conditions. This approach therefore does not require an optimisation problem to be solved during an agent’s mission, however the development of the protocols may involve an offline optimisation. Therefore, during its mission, an agent only needs to be able to determine it and its neighbours state and select the appropriate protocol. An agent will do this through the use of sensors and/or communication with other agents. An early version of this approach identifies safe sets of headings which ensure collision free trajectories regardless of the action of another agent [67]. To do this there is a limited set of actions that an agent can take in response to a potential conflict.

In some more recent work [46, 68] an MPC implementation of the ICAO RoW rules was applied to UAVs. This work uses a mix of protocol and optimisation approaches as the RoW rules provide a new component to a cost function which can then be optimised. These methods have been shown to provide collision free trajectories, in both single and multiple conflict scenarios. However, the level of complexity is high due to a number of factors such as the necessity to identify what type of collision scenario two UAVs are in and then to construct and optimise the cost function. The conclusion of [68] explicitly states that increasing the number of UAVs “consumes significant computation time” which may be unfeasible for some UAVs with limited on board computing power. Also, by not introducing RoW, the rules presented in this thesis aim to provide a more equal sharing of the responsibility and thus sharing of the disruption as a result of conflict avoidance between UAVs.

The MPC methods outlined here, and countless others in the literature, have several advantages. Often the models explicitly account for vehicle dynamics ensuring feasible solutions and, where optimisation is used, can provide near optimal solutions for a wide variety of optimisation criteria. The main drawback is the way in which these methods scale with the number of participants, particularly as the number of UAVs and their operations is expected to keep rising in the near future. Combined with the need for UAVs to use simple computational hardware, the need to limit computational complexity is also important. One way to deal with this is to take the computational complexity offline. Chen [69] used deep reinforcement learning to enable mobile robots to learn avoidance policies. This has been shown to produce policies that are at least as effective as other collision avoidance methods. However, in terms of implementing something like this for UAV airspace there are some barriers. Firstly there is the possible inability to react to novel situations that have not been planned for. This then contributes, along with the black box nature of machine learning, to challenges in verifying and certifying these sorts of methods for real world use.

In this thesis the conflict avoidance rule will be based on the velocity obstacle method [70, 71]. Velocity obstacle methods have been applied to the problem of robots navigating in a dynamic environment for some time. One of the main appeals of this method is that it only requires agents to know their neighbours’ positions and velocities. It does not require any further co-ordination.

Using this localisation information, a UAV can construct the velocity obstacle. The velocity obstacle itself is a set of relative velocities which will result in the separation between two UAVs at their closest point of approach being smaller than some minimum acceptable value set by the UTM system. The problem then becomes how a UAV should choose its new avoidance velocity so that it does not result in a relative velocity that belongs to the velocity obstacle.

A particular problem with the early implementations of velocity obstacles however was the presence of undesirable oscillatory motion. This is due to both agents picking avoidance velocities such that they are no longer on a conflict course and then subsequently returning to their original heading, resulting in another conflict course. The reciprocal velocity obstacle (RVO) was developed in order to resolve this problem [72]. The RVO method assumes that both agents will make the same manoeuvre and in doing so constructs a modified velocity obstacle based on the average velocity of the conflicting pair. If both agents choose a new velocity that is closest to their current velocity, i.e., the closest point on the edge of the RVO, then it is guaranteed that they will avoid each other on the same side and that their current velocity will belong to the subsequent RVO.

However, if an agent does not conform to the assumptions made in the RVO method, then it can no longer guarantee collision avoidance or freedom from oscillatory trajectories. An example of when this might happen is when an agent's preferred velocity, perhaps due to a waypoint or other goal, requires the agent to turn in the opposite direction to the RVO manoeuvre. One solution to this is to construct a velocity obstacle using a combination of the standard velocity obstacle method and the RVO method [73]. The method in [73] forms a hybrid reciprocal velocity obstacle (HRVO) such that if an agent chooses the "wrong" side to avoid on then it "has to give full priority" to the other.

In order to deal with multi-agent scenarios both [72] and [73] suggest combining the velocity obstacles induced by each other agent and then selecting a new velocity that is not inside the union of these velocity obstacles. The optimal reciprocal collision avoidance (ORCA) [74] uses the same idea but uses a method to turn the velocity obstacle into a half plane constraint. This half plane is tangential to the point on the truncated velocity obstacle that is closest to the current relative velocity, meaning that any velocity chosen inside the half plane ensures that two agents will be collision free for some time horizon. The half planes induced by several other neighbouring agents can then be combined to produce a convex set of velocities which can in turn be chosen from to be as close as possible to some preferred velocity.

There are two main reasons we do not use ORCA in this work. Firstly, [75] showed that ORCA can be more adversely impacted by the effects of noise on position and velocity than other VO methods. Secondly, though ORCA is computationally more efficient than RVO or HRVO methods for multi-agent scenarios it is still more complex than the rules presented here which do not need to combine the velocity obstacles at all. This simplicity comes at the cost of the collision avoidance guarantees provided by ORCA.

2.2 Baseline dynamical model for autonomous UAVs

In this section we will develop a stylised dynamical model of multi-rotor UAVs inspired by the SFM [56] for pedestrians. In this model a UAV i, modelled as a point mass in a continuous 2D space, will be prescribed various desired acceleration components $\mathbf{a}_i^k$ which will be combined to produce an overall acceleration $\mathbf{a}_i$ in order to drive the motion of the UAV. While in flight, at any time t, UAV i will also be described by a displacement $\mathbf{r}_i(t)$ and a velocity $\mathbf{v}_i(t)$ in the plane. We restrict the model to a 2D world based on the assumption that, at least in the short term, UAVs will be restricted to one airspace layer as we discussed in Section 1.2.

For our dynamical model we assume that, similar to pedestrians, UAVs in flight will make their way to some destination while avoiding obstacles in their way, including each other. Specifically we assume that the UAVs will need to maintain a minimum horizontal separation S with other UAVs. This will be achieved by the conflict avoidance method developed in Section 2.3. Our baseline dynamical model then, in the absence of other UAVs, will need to ensure that any UAV is able to steer itself in order to reach its destination. Therefore, UAV i will also have a desired heading described by a unit vector $\hat{\mathbf{t}}_i$. This desired heading can be a constant vector which describes a preferred direction of motion or be defined by a waypoint at position $\mathbf{r}_i^{\mathrm{WP}}$ such that

$$\hat{\mathbf{t}}_i = \frac{\mathbf{r}_i^{\mathrm{WP}} - \mathbf{r}_i}{|\mathbf{r}_i^{\mathrm{WP}} - \mathbf{r}_i|}, \tag{2.1}$$

which the UAV is trying to reach. A UAV may change its desired heading throughout a flight, but it will only have one at any instant and its waypoint might therefore be simply modelled as piece-wise constant in time. The UAV will also have a cruising speed v_{CS} which it will attempt to maintain. It is assumed that v_{CS} will be less than or equal to some regulatory maximum speed. In reality there are a wide variety of UAVs available in terms of characteristics such as size and speed that may be selected by an operator for different tasks. However, for simplicity we will model and simulate a base case where all UAVs are identical.

Models for a UAV's acceleration in the plane are different, depending on which type of craft is being considered. UAVs can be broadly broken down into two categories depending on their method of flight. Fixed wing UAVs fly like aeroplanes with lift generated by air flowing over a wing. Multi-rotor UAVs, such as the ubiquitous quadcopter, produce lift and thrust through spinning blades and adjust this thrust by accelerating different rotors at different rates [76, 77]. We will assume for all our experimental setups that all UAVs are of the multi-rotor type. This kind of UAV has a number of advantages over fixed wing when operating in a crowded airspace including the ability to hover in place, though it should be noted that fixed wing UAVs are often better suited to long range missions due to their better energy efficiency. Given the different strengths and weaknesses of the two main types of UAVs it is likely that they will have to share the airspace with each other. Therefore, while this thesis will focus on multi-rotor UAVs we will present a simple dynamical model for fixed wing UAVs in Section 5.1.1 which could be used in future work.

For multi-rotor UAVs, our stylised model assumes that each UAV is capable of a maximum magnitude of acceleration a_{max}. The acceleration that can be achieved is also assumed to be independent of the UAV's velocity and can be in any direction in the plane. In order to assume this independence we must further assume that the cruising speed v_{CS} set by the regulator is sufficiently smaller than the maximum speed a UAV is physically capable of. This simplification captures the multi-rotor UAV's ability to manoeuvre more sharply than fixed wing UAVs. As in the fixed wing case, this assumption ensures that the UAVs are able to change course as quickly as possible in response to a changing environment. In the future it will be up to regulators to decide exactly how to balance the system goals of short flight times (large values of v_{CS}) and manoeuvrable UAVs (small values of v_{CS}).

As we have discussed, we model the dynamics of a multi-rotor UAV as a combination of acceleration components, similar to the various social forces that act on pedestrians in the SFM. The different acceleration components a UAV is subject to are attempting to change $\mathbf{v}_i$ in order to steer the UAV. The first acceleration component will apply at all times and takes the form

$$\mathbf{a}_i^0 = \frac{1}{\tau}(v_{\mathrm{CS}}\hat{\mathbf{t}}_i - \mathbf{v}_i), \tag{2.2}$$

which models the first order relaxation towards the cruising speed with the desired heading, i.e., an *optimal velocity model* (OVM) [78]. The remainder of the acceleration components, $k = 1, 2, ...,$ will be produced by a pairwise S&A interaction with neighbouring UAVs, see Section 2.3. Here we set the rate parameter τ by

$$\tau = \frac{2v_{\mathrm{CS}}}{a_{\max}}, \tag{2.3}$$

so that a UAV flying at the cruising speed in exactly the opposite direction to its desired heading will initially retard itself with the maximum allowed acceleration. Thus τ also provides a natural timescale for the model.

We implemented Equation 2.2 in simulation in order to illustrate the resulting behaviour. We plot the flight path for a UAV both when $\hat{\mathbf{t}}_i$ is equal to a constant vector $\mathbf{c} = (1,0)^{\mathrm{T}}$ and using Equation 2.1 for a waypoint at $\mathbf{r}_i^{\mathrm{WP}}$, see Figure 2.3. In both cases we use the same initial condition of $\mathbf{v}_i(0) = (0, v_{\mathrm{CS}})^{\mathrm{T}}$ which is perpendicular to $\mathbf{c}$. As such the two flight paths are initially the same and eventually diverge from each other as $\mathbf{c}$ no longer points towards the waypoint. Therefore, when we set $\hat{\mathbf{t}}_i$ to a constant vector, we can effectively model a waypoint that is infinitely far away.

Given the acceleration components produced by the OVM and pairwise avoidance, a UAV needs to combine them to produce $\mathbf{a}_i$. The simplest way to combine the component accelerations is to add them together. The resulting acceleration is then scaled in order to respect the maximum acceleration

$$\mathbf{a}_i := \begin{cases} \sum_k \mathbf{a}_i^{(k)}, & \text{when } |\sum_k \mathbf{a}_i^{(k)}| \le a_{\max}, \text{ or} \\[2ex] \dfrac{\sum_k \mathbf{a}_i^{(k)}}{|\sum_k \mathbf{a}_i^{(k)}|} a_{\max}, & \text{when } |\sum_k \mathbf{a}_i^{(k)}| > a_{\max}. \end{cases} \tag{2.4}$$

This method of combining accelerations is used to provide a simple starting point but is an aspect of the model that could be developed in future work. There are obvious drawbacks to using this sort of capped summative approach. For example, consider a simple avoidance scheme where UAVs accelerate directly away from other UAVs, analogous to 'electrostatic' repulsion. We can

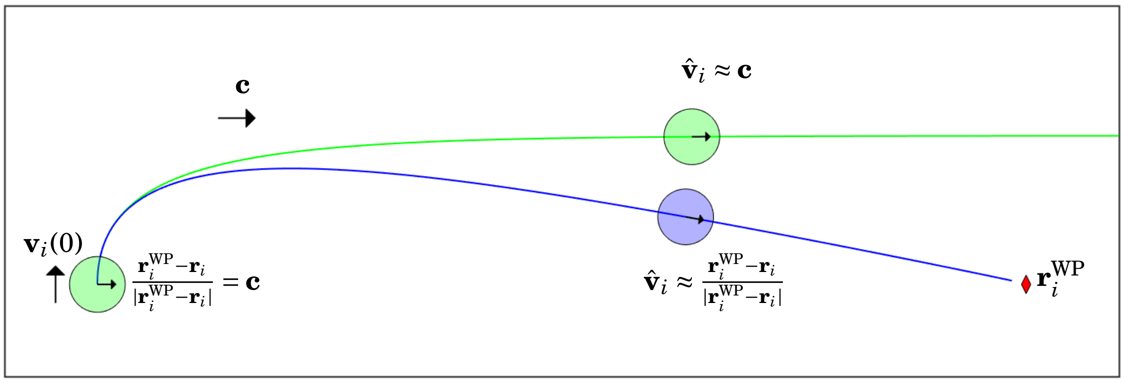

$\mathbf{c}$
$\hat{\mathbf{v}}_i \approx \mathbf{c}$
$\mathbf{v}_i(0)$
$\frac{\mathbf{r}_i^{\mathrm{WP}}-\mathbf{r}_i}{|\mathbf{r}_i^{\mathrm{WP}}-\mathbf{r}_i|} = \mathbf{c}$
$\hat{\mathbf{v}}_i \approx \frac{\mathbf{r}_i^{\mathrm{WP}}-\mathbf{r}_i}{|\mathbf{r}_i^{\mathrm{WP}}-\mathbf{r}_i|}$
$\mathbf{r}_i^{\mathrm{WP}}$

Figure 2.3: Two flight paths are produced using the OVM from Equation 2.2 where $\hat{\mathbf{t}}_i$ is a constant vector with $\mathbf{c} = (1,0)^{\mathrm{T}}$ (green curve) and using Equation 2.1 (blue curve) with a waypoint at $\mathbf{r}_i^{\mathrm{WP}}$ (red diamond). Note that the flight paths are initially the same since $\mathbf{c}$ points towards $\mathbf{r}_i^{\mathrm{WP}}$. We can therefore use a constant vector for $\hat{\mathbf{t}}_i$ to effectively model a waypoint that is infinitely far away.

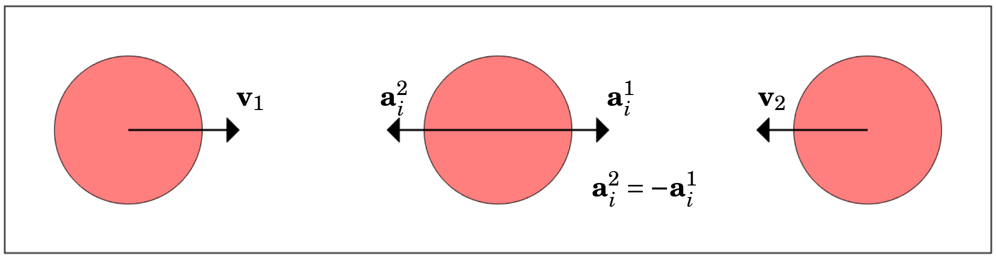


Figure 2.4: An example of where the summative method for combining acceleration components, like that in Equation 2.4, can produce undesirable outcomes. Here the UAVs employ a simple 'electrostatic' like avoidance. Two UAVs are flying towards a third UAV placed between them. As a result the third UAV produces two acceleration components such that $\mathbf{a}_i = \mathbf{a}_i^1 + \mathbf{a}_i^2 = 0\,\mathrm{ms}^{-2}$ and therefore does not undergo any avoidance manoeuvre.

then easily construct examples where two or more UAVs act on another UAV in such a way as to cancel each other out, see Figure 2.4. Here two UAVs approach a UAV i from either side. As such, UAV i produces two acceleration components that point in opposite directions. Assuming that the magnitude of both acceleration components is the same, which would depend on the implementation of such an avoidance scheme, then $\mathbf{a}_i^1 + \mathbf{a}_i^2 = 0\,\mathrm{ms}^{-2}$ which would result in no avoidance manoeuvre for UAV i.

We could alternatively resolve the problem of equal and opposite acceleration components through a weighting process. Consider a variation of Equation 2.4 where we modify each component by a constant $c^{(k)}$ which allows us to prioritise certain components. We can take this to the extreme where $c^{(k)} = 0$ or $c^{(k)} = \infty$. In the case of the former this effectively turns off the given interaction. In the case of the latter, due to the scaling process in Equation 2.4, this ensures that the UAV will accelerate with magnitude $a_{\max}$ in the direction of that particular acceleration component. This then begs the question of how to choose values for the constant $c^{(k)}$. Possible heuristics include distance between the UAVs, relative velocity, or how severe the potential collision will be. Designing the rules which would set $c^{(k)}$ then would add an extra layer of complexity in addition to designing the conflict avoidance method itself and will therefore be neglected in this work. Instead, as we develop the conflict avoidance method in Section 2.3 we will use a similar OVM to Equation 2.2 to produce $\mathbf{a}_i^{(k)}$ using a different rate parameter. This will provide a sort of simple weighting to the avoidance components.

2.3 Velocity-obstacle inspired pairwise conflict avoidance

We have shown that using the OVM from Equation 2.2, a UAV can steer towards a desired heading. We will now develop a conflict avoidance method which will enable a UAV i to steer themselves such that the separation with a neighbour j tends to achieve

$$|\mathbf{r}_j - \mathbf{r}_i| \geq S. \tag{2.5}$$

In a pairwise interaction our rule will always maintain this safe separation assuming the pair of UAVs do not begin too close to each other. If the inequality is violated then the UAVs are said to be in *conflict* with each other. Further, a UAV i is said to be on a *conflict course* if it will violate this inequality at some point in the future based on it and a neighbour's current velocities remaining fixed. The conflict avoidance method presented here falls under the general category of S&A and we therefore assume that the UAVs will be capable of localisation and of sharing their position and velocity with each other. The exact implementation of these capabilities is a matter of academic research in its own right, as discussed in Section 1.1. We will take an optimistic view in this work and further assume that the localisation data each UAV produces is accurate and that this can be broadcast to neighbours without significant loss or delay. In future work these issues could be modelled in more detail and their (probably detrimental) effect on the conflict avoidance explored, for example how S might need to be correspondingly increased.

Besides maintaining a separation of S or larger with other UAVs, it is still necessary that a UAV in flight reaches its destination in a timely manner in order to complete its mission. As such we will design the conflict avoidance with notions of efficiency and fairness in mind. In order to carry out an avoidance manoeuvre, a UAV will necessarily be deviated from its original flight path. This in turn will produce a cost which can be described in terms of an extension to flight time or flight path. Therefore any conflict avoidance manoeuvre should seek to minimise this cost for an individual UAV. Furthermore, we consider a fair conflict avoidance manoeuvre to be one where the cost is split more or less evenly between the UAVs involved. To this end we will assume that all UAVs implement the same conflict avoidance method and have the same priority. For simplicity we will neglect the possibility that some UAV missions may require higher priority in terms of avoidance, e.g., emergency services en route to an incident.

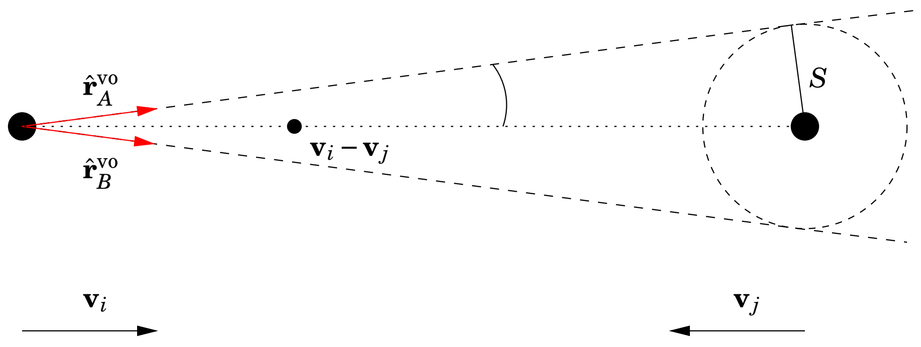

$\hat{\mathbf{r}}_A^{\mathrm{vo}}$
$\hat{\mathbf{r}}_B^{\mathrm{vo}}$
$\mathbf{v}_i - \mathbf{v}_j$
S
$\mathbf{v}_i$
$\mathbf{v}_j$

Figure 2.5: The velocity obstacle of two converging UAVs. A circle, of radius S, is centered on the neighbour j and thus the velocity obstacle is defined by the two lines that are tangential to the circle and intersect at $\mathbf{r}_i$. If the relative velocity of the two UAVs lies within the cone of the velocity obstacle then they are on a conflict course. The red arrows show the unit vectors, $\hat{\mathbf{r}}_A^{\mathrm{vo}}$ and $\hat{\mathbf{r}}_B^{\mathrm{vo}}$, that are parallel with the tangents and are used to find the avoidance velocity $\mathbf{v}_i^j$.

The conflict avoidance method we propose here is inspired by the velocity obstacle method [70, 71] and can be broken down into the following three steps; detecting a conflict, choosing an avoidance velocity, and producing the avoidance acceleration component. Using this method a UAV i will first check whether it and a neighbour j are on a conflict course by constructing a velocity obstacle. Then, if the two UAVs are on a conflict course, it will choose an avoidance velocity $\mathbf{v}_i^j$ that if adopted would ensure the (future) minimum separation between the two UAVs is equal to S. Finally, the UAV will produce an avoidance acceleration $\mathbf{a}_i^j$ in order to change its velocity towards $\mathbf{v}_i^j$. In the event that the inequality in Equation 2.5 is violated, or UAV i is unable to choose an avoidance velocity, then it will produce an emergency acceleration

$$\mathbf{a}_i^j = -\hat{\mathbf{r}}_{ij} a_{\max}, \tag{2.6}$$

in the opposite direction to its neighbour j.

UAV i checks whether it and j are on a conflict course by first constructing the velocity obstacle $\mathrm{VO}_{i|j}$ induced by j. To construct $\mathrm{VO}_{i|j}$, consider a circle with radius S centered on a neighbour j, see Figure 2.5. The velocity obstacle is then defined by the tangents to this circle that originate from $\mathbf{r}_i$, shown by the dashed lines, with unit vectors $\hat{\mathbf{r}}_A^{\mathrm{VO}}$ and $\hat{\mathbf{r}}_B^{\mathrm{VO}}$. The unit vectors are obtained by noting that the radius S is normal to the tangents, thus simple trigonometric rules can be applied. Two UAVs are on a conflict course if their relative velocity belongs to the velocity obstacle, shown as the small black circle in Figure 2.5. Note that if the relative velocity lies on either of the tangents, then the UAVs will have a separation of S at their closest point of approach if the relative velocity is maintained.

If two UAVs are on a conflict course, as in Figure 2.5, then UAV i must choose an avoidance velocity $\mathbf{v}_i^j$. To do so it assumes that j's velocity is constant. In other words, i does not assume that j will act cooperatively and therefore will will take full responsibility for the avoidance. This

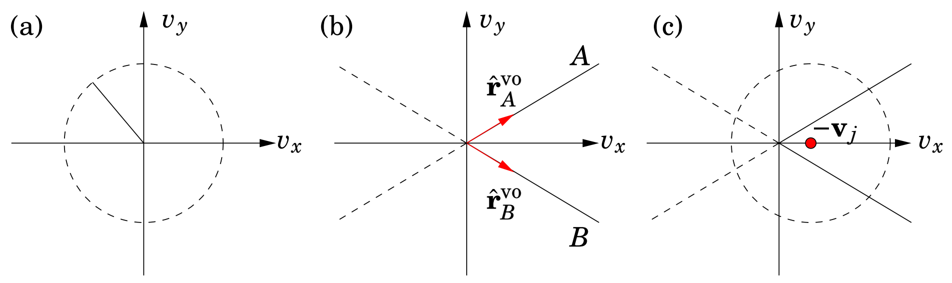

(a)
v_y
v_x
(b)
v_y
A
$\hat{\mathbf{r}}_A^{\text{vo}}$
v_x
$\hat{\mathbf{r}}_B^{\text{vo}}$
B
(c)
v_y
$-\mathbf{v}_j$
v_x

Figure 2.6: Geometric representation of the constraints from (a) Equation 2.7 and (b) Equation 2.9. In (c) the two constraints are combined, shifting the center of the circle from (a) by $-\mathbf{v}_j$. The roots from Equation 2.10 correspond to the two points where line A intersects with the dashed circle. An imaginary root is obtained if the dashed circle does not intersect with the corresponding line.

assumption is useful since it means that this avoidance method can be applied to static obstacles, can account for UAVs that are given priority during avoidance (e.g., emergency vehicles), and can deal with scenarios where j's ability to sense i has been compromised.

An even more pessimistic scenario would be one in which j attempts to collide with i, i.e., where j attempts to change velocity in order to negate i's attempts at avoidance. We call this the 'malicious UAV' scenario and consider it to be far outside the scope of this work.

The magnitude of the avoidance velocity is constrained to be

$$|\mathbf{v}_i^j| = v_{\text{CS}}, \tag{2.7}$$

so that the UAV maintains its cruising speed, see Figure 2.6(a). The new relative velocity

$$\mathbf{v}_{ij}^* = \mathbf{v}_i^j - \mathbf{v}_j, \tag{2.8}$$

is constrained to be parallel to either of the velocity obstacle tangents

$$\mathbf{v}_{ij}^* = \begin{cases} |\mathbf{v}_{ij}^*|\hat{\mathbf{r}}_A^{\text{vo}}, & \\ & \text{or,} \\ |\mathbf{v}_{ij}^*|\hat{\mathbf{r}}_B^{\text{vo}}, & \end{cases} \tag{2.9}$$

ensuring that the UAVs have a minimum separation of S and should result in a small deviation from their original flight path, see Figure 2.6(b). In fact, UAV j is applying the same logic so that $\mathbf{v}_j^i$ and $\mathbf{v}_i^j$ would, if adopted, cause the UAVs to approach at nearest distance $2S$. However, this does not happen in practice because the avoidance velocities are not instantly adopted. Instead they are used to generate acceleration components through an OVM which will be described later in this section. Therefore both UAVs will gradually change their velocities over time until their predicted separation at the closest point of approach is greater than or equal to S. Once this is the case then the two UAVs will cease their conflict avoidance manoeuvre.

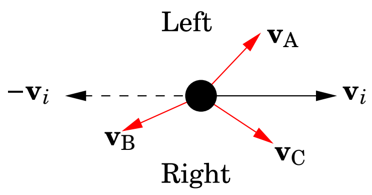


Figure 2.7: Consider a UAV with velocity $\mathbf{v}_i$ and three potential avoidance velocities $\mathbf{v}_\mathrm{A}$, $\mathbf{v}_\mathrm{B}$, and $\mathbf{v}_\mathrm{C}$ (red arrows). Any velocity in the same half plane as $\mathbf{v}_\mathrm{A}$ is considered to be a left hand turn. Any velocity in the same half plane as $\mathbf{v}_\mathrm{B}$ or $\mathbf{v}_\mathrm{C}$ is considered a right hand turn. The UAV would choose $\mathbf{v}_\mathrm{C}$ over $\mathbf{v}_\mathrm{B}$ because the angle between it and $\mathbf{v}_i$ is smaller.

Using these constraints we obtain the following quadratic equation for the magnitude of the new relative velocity

$$|\mathbf{v}^*_{ij}|^2 + 2|\mathbf{v}^*_{ij}|\hat{\mathbf{r}}^{\mathrm{vo}}_A \cdot \mathbf{v}_j + (|\mathbf{v}_j|^2 - (v_{\mathrm{CS}})^2) = 0, \tag{2.10}$$

and a similar one for $\hat{\mathbf{r}}^{\mathrm{VO}}_B$. This gives four possible roots, two associated with $\hat{\mathbf{r}}^{\mathrm{VO}}_A$ and two associated with $\hat{\mathbf{r}}^{\mathrm{VO}}_B$, see Figure 2.6(c). It is possible for complex roots to be obtained when the UAV j is travelling faster than the cruising speed, which means that both constraints from Equations 2.7 and 2.9 can not be satisfied. In other words, there would be no avoidance velocity that i can take which will result in a minimum separation of S and achieve its cruising speed.

In the case that $|\mathbf{v}_j| = v_{\mathrm{CS}}$ then two of the four possible roots will be zero. We can discard these roots since they correspond to UAV i adopting the avoidance velocity $\mathbf{v}^j_i = \mathbf{v}_j$ and lead to the two UAVs maintaining their current separation. Finally, if $|\mathbf{v}_j| < v_{\mathrm{CS}}$ then Equation 2.10 will have a real negative and real positive root. We neglect the negative roots since these would lead to avoidance velocities where UAV i begins to reverse.

After all four roots have been found, complex roots and real roots equal to or less than zero are discarded. The remaining real positive roots are then turned in to potential avoidance velocities for i by substituting Equation 2.9 in to Equation 2.8. Given the possible avoidance velocities, UAV i will select the one with the smallest right hand (clockwise) turn with respect to its current velocity, as shown in Figure 2.7. The choice of the turn direction is arbitrary, and could be reversed, but reflects the current rules of the air [45]. The current rules specify that the aircraft without RoW should alter its heading to its right. Unlike [45] we do not develop a concept of RoW in an attempt to ensure a fairer manoeuvre. By enforcing this right-handedness on all UAVs we eliminate the need for negotiation between the UAVs and ensure that when two UAVs approach each other head on they do not choose mirrored avoidance velocities.

Given that an avoidance velocity $\mathbf{v}^j_i$ has been chosen, we prescribe the avoidance acceleration

$$\mathbf{a}^j_i = \frac{\mathbf{v}^j_i - \mathbf{v}_i}{t_\mathrm{c}}, \tag{2.11}$$

where t_c is the *time to conflict*, i.e., the time that will elapse before the inequality in Equation 2.5 is violated, based on the UAVs' current velocities. This approach is similar to the OVM used in

Algorithm 1: Overview of UAV Flight Model

```
for UAV i do
    Apply OVM:
      $\mathbf{a}_i^0 = \frac{1}{\tau}(v_{\mathrm{CS}}\hat{\mathbf{t}}_i - \mathbf{v}_i)$
    for Neighbour j where $j \neq i$ do
        Obtain $\mathbf{v}_j$ and $\mathbf{r}_j$
        Construct $\mathrm{VO}_{i|j}$
        if On a conflict course then
            Find the roots of:
              $|\mathbf{v}_{ij}^*|^2 + 2|\mathbf{v}_{ij}^*|\hat{\mathbf{r}}_A^{\mathrm{vo}} \cdot \mathbf{v}_j + (|\mathbf{v}_j|^2 - (v_{\mathrm{CS}})^2) = 0$
            and
              $|\mathbf{v}_{ij}^*|^2 + 2|\mathbf{v}_{ij}^*|\hat{\mathbf{r}}_B^{\mathrm{vo}} \cdot \mathbf{v}_j + (|\mathbf{v}_j|^2 - (v_{\mathrm{CS}})^2) = 0$
            Discard imaginary roots or real roots that are $\leq 0$
            Convert roots to avoidance velocities
            Select the avoidance velocity with the smallest right hand turn to be $\mathbf{v}_i^j$
            Generate avoidance acceleration component:
            if $|\mathbf{r}_{ij}| \leq S$ or no avoidance velocity is chosen then
                $\mathbf{a}_i^j = -\hat{\mathbf{r}}_{ij} a_{\max}$
            end
            else
                $\mathbf{a}_i^j = \frac{\mathbf{v}_i^j - \mathbf{v}_i}{t_{\mathrm{c}}}$
            end
        end
    end
    Combine acceleration components:
      $\mathbf{a}_i := \begin{cases} \sum_k \mathbf{a}_i^{(k)}, & \text{when } |\sum_k \mathbf{a}_i^{(k)}| \leq a_{\max}, \text{ or} \\ \frac{\sum_k \mathbf{a}_i^{(k)}}{|\sum_k \mathbf{a}_i^{(k)}|} a_{\max}, & \text{when } |\sum_k \mathbf{a}_i^{(k)}| > a_i^{\max}. \end{cases}$
end
```

Equation 2.2 but the rate parameter ($1/t_{\mathrm{c}}$) is no longer constant and depends on the state of both UAVs. Choosing the rate parameter in this way provides a simple method for weighting the avoidance acceleration components because as t_{c} goes to zero $|\mathbf{a}_i^j|$ goes to infinity. While this acceleration is never realised due to capping by $a_{\max}$, this mechanism ensures that the direction of the overall acceleration $\mathbf{a}_i$ is nearly parallel with $\mathbf{a}_i^j$ as the UAVs become close. Note, the time to conflict is given by the minimum positive root of

$$(t_{\mathrm{c}})^2|\mathbf{v}_{ij}|^2 - 2t_{\mathrm{c}}\mathbf{v}_{ij} \cdot \mathbf{r}_{ij} + (|\mathbf{r}_{ij}|^2 - (S)^2) = 0, \tag{2.12}$$

provided the distance between the two UAVs is larger than S.

The overall flight model for a UAV then is summarised, with key equations, in Algorithm 1, implemented in simulation. For every UAV in a system it first applies the OVM from Equation 2.2.

It then applies the conflict avoidance, as described in this section, for every neighbour with which it is on a conflict course with. The resulting acceleration components are then combined as described in Equation 2.4.

See Appendix A for more detail on how this avoidance method and the dynamical model from Section 2.2 are implemented in our simulation architecture.

2.4 Pairwise experimental setup

By combining the model for UAV dynamics from Section 2.2 with the conflict avoidance method developed in Section 2.3, it should be possible for a pair of UAVs on a conflict course to change their velocities so that they pass each other with a separation of at least S and then return to their desired heading. In order to test this we will propose an experimental setup to be implemented in simulation where two UAVs are initialised on a conflict course. A pairwise interaction is used here as this represents the simplest form of conflict avoidance. Note, in Chapter 3 we will explore more complex experimental setups with many UAVs sharing an airspace. In this section we will also present a method for quantifying the impact of conflict avoidance on a UAV's experience in terms of a time delay incurred.

We suppose that two UAVs $i = 1,2$ are initialised with positions $\mathbf{r}_i(0) = -R_i\hat{\mathbf{e}}_i$ and velocities $\mathrm{v}_i(0) = v_{\mathrm{CS}}\hat{\mathbf{e}}_i$, with $\hat{\mathbf{e}}_1 = (1,0)^{\mathrm{T}}$ and $\hat{\mathbf{e}}_2 = (\cos\theta, \sin\theta)^{\mathrm{T}}$, see Figure 2.8(a). Furthermore, we set $\hat{\mathbf{t}}_i = \hat{\mathbf{e}}_i$ so that, in the absence of collision avoidance manoeuvres, Equation 2.2 implies the UAVs will continue at constant velocity, and their paths will cross at angle θ at the origin. This models a scenario where two UAVs are converging at a point while travelling towards a waypoint that is infinitely far away. Note that the difference $\Delta R := R_2 - R_1$ constitutes a sort-of 'distance-phase' parameter. As such we can now describe an instance of this setup using the two parameters θ and ΔR in conjunction with a set of UAV parameters ($a_{\max}$, S etc.).

Here, we set R_1 and R_2 very large, to model UAVs approaching each other 'from infinity', so initially the time to conflict t_{c} is large, and the initial corrective motion is small. Subsequently, UAVs on a conflict course will avoid each other to the right, approaching at the minimum distance S, and then via Equation 2.2, equilibrate to a path that is parallel to their original course, see Figure 2.8(b) for a representative example flight path. The net effect of the interaction is to displace each UAV laterally and longitudinally with respect to its original path. In effect, the interaction costs each UAV a delay

$$T_i := d_i/v_{\mathrm{CS}}, \tag{2.13}$$

where d_i is the respective longitudinal deficit. The smaller the value of T_i, the more efficient we consider a conflict avoidance manoeuvre to be. We can also compare T_i for both UAVs in order to evaluate the fairness of the avoidance method.

Only those setups where the UAVs begin on a conflict course with each other are of interest since no conflict avoidance is needed otherwise. As such we will explore here analytically what

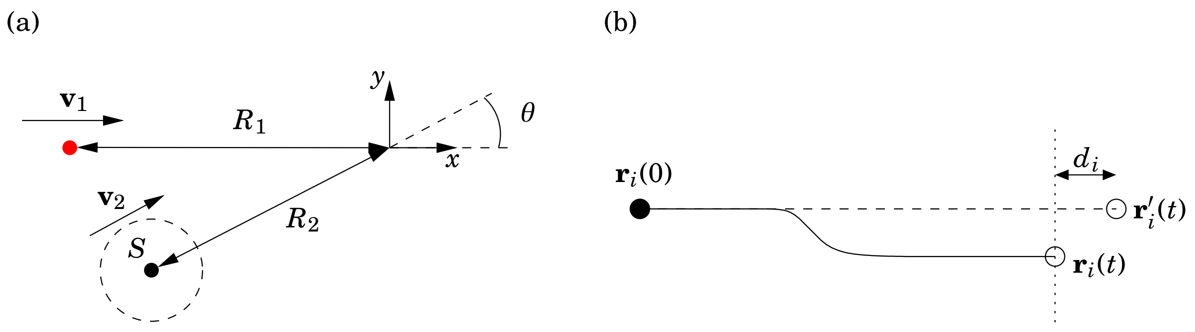


Figure 2.8: (a) Experimental setup where two UAVs begin on a conflict course with a relative displacement $\Delta R := R_2 - R_1$ to the origin and angle θ between them. (b) Comparison of linear flight path and representative flight path where an avoidance manoeuvre has been carried out. The longitudinal displacement d_i is used to derive the delay T_i incurred during the avoidance manoeuvre which we use as a metric to compare the effect of different setup parameters θ and ΔR.

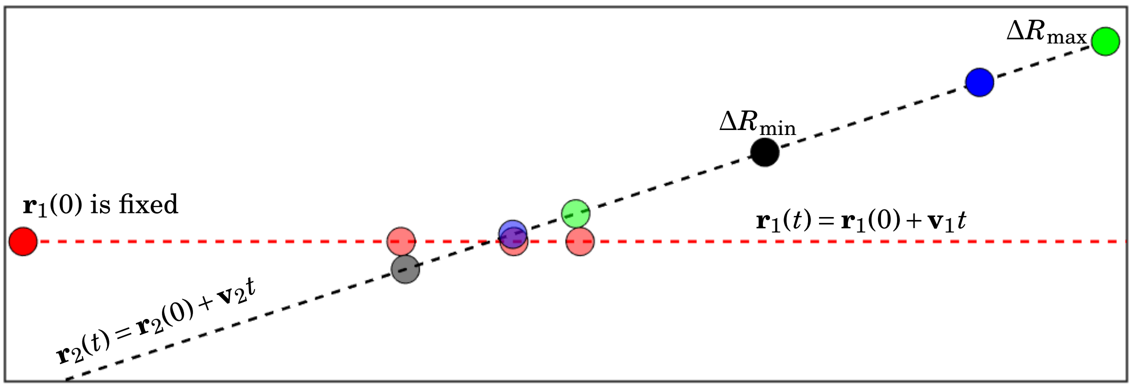

ΔR_{max}
ΔR_{min}
$\mathbf{r}_1(0)$ is fixed
$\mathbf{r}_1(t) = \mathbf{r}_1(0) + \mathbf{v}_1 t$
$\mathbf{r}_2(t) = \mathbf{r}_2(0) + \mathbf{v}_2 t$

Figure 2.9: We obtain the limits for ΔR which induce a conflict course between two UAVs, for any angle of approach θ, by considering the straight line paths the UAVs would take if no avoidance manoeuvre is enacted using Equation 2.14. In our experimental setups the position of UAV 1 (red opaque circle) is fixed and ΔR is achieved by moving UAV 2 closer to or further from the origin. Placing UAV 2 at starting positions that correspond to the maximum and minimum values for ΔR (green and black opaque circles) results in a minimum separation equal to S (translucent circles).

the limits of ΔR are that satisfy this requirement. Clearly, if $|\Delta R| < S$, the UAVs will come into conflict in the neighbourhood of the origin, whatever the approach angle θ. In contrast, in the perfect 'head-on' case $\theta = \pi$, the UAVs will come in to conflict irrespective of $|\Delta R|$. We now try to find the limits for ΔR which induce a conflict course for a general approach angle θ. First we consider that we can predict the position of UAV i at any point in the future using its initial position and velocity

$$\mathbf{r}_i(t) = \mathbf{r}_i(0) + \mathbf{v}_i t, \tag{2.14}$$

because the UAVs are setup with their velocity pointing in the desired direction. As such the UAVs would adopt a straight line path if no avoidance manoeuvre was undertaken, see Figure 2.9. Therefore the separation between UAVs at any time is given by

$$|\mathbf{r}_1(t) - \mathbf{r}_2(t)| = \sqrt{R_1^2 + 2\cos\theta(R_1 - v_{\mathrm{CS}}t)(v_{\mathrm{CS}}t - R_2) - 2v_{\mathrm{CS}}t(R_1 + R_2) + R_2^2 + 2v_{\mathrm{CS}}^2 t^2}, \tag{2.15}$$

using the parameters that describe the experimental setup. We also know that when

$$\frac{\mathrm{d}|\mathbf{r}_1(t) - \mathbf{r}_2(t)|}{\mathrm{d}t} = \frac{2v_{\mathrm{CS}}\sin^2\frac{\theta}{2}(R_1 + R_2 - 2tv_{\mathrm{CS}})}{|\mathbf{r}_1(t) - \mathbf{r}_2(t)|} = 0, \tag{2.16}$$

the separation between the UAVs is at a minimum. Rearranging Equation 2.16 and using our definition for ΔR we deduce that the UAVs will be at their closest when

$$t = \frac{\Delta R + 2R_1}{2v_{\mathrm{CS}}}, \tag{2.17}$$

regardless of the initial angle of approach. Using this time in Equation 2.15 provides us with the limits

$$-S\sec\frac{\theta}{2} \le \Delta R \le S\sec\frac{\theta}{2}, \tag{2.18}$$

for which the UAVs will begin on a conflict course. If this is the case, the conflict avoidance scheme described in Section 2.3 is activated, and the UAVs will thus deviate from their straight line paths for $t > 0$.

2.5 Initial results

The experimental setup presented in Section 2.4 is designed to allow us to validate the conflict avoidance method described in Section 2.3 for a pair of UAVs. We may also evaluate the performance of the conflict avoidance in terms of the delay incurred by the UAVs compared to their initial linear flight paths. This section then will initially discuss the choice of dimensional values for parameters as we implement the experimental setup in simulation. We will then present the results of these simulations and discuss how the behaviour of the UAVs influenced them. See Appendix A for more information about the simulation architecture we developed to implement these experiments and those performed later in the thesis.

Only those setups where the two UAVs begin on a conflict course and where the initial time to conflict is large are explored. We conduct a set of experiments that sweeps through values of θ in the range $(-\pi, +\pi)$, with $|\theta| \geq 2S/R_1$ so that the UAVs are not in conflict at $t = 0$. In fact, this ensures that the UAVs begin with a separation that is equal to or greater than $2S$ when $\Delta R = 0$. Correspondingly, ΔR is swept through the range $(-3S/2, +3S/2)$ which captures a variety of settings in which conflicts between the UAVs would occur. This means that for larger values of $|\theta|$ not all setups that induce a conflict will be explored. However, we will show that these setups are of little interest since they tend to result in small delays for both UAVs.

In these experiments the other problem parameters, see Table 2.1, are held at constant dimensional values. The internal UAV parameters are set to the same values for both UAVs, though this could be changed in future work. The values chosen are meant to capture reasonable estimates of typical performance expected of multi-rotor UAVs or limits imposed by the UTM system. In Table 2.1 we also set the distance R_1 to a constant value. Therefore, when the value for ΔR is chosen for a particular instance of the setup we set R_2 accordingly. By setting R_1 to a large value, we can explore small angles of approach, given the way we have defined the lower limit for θ. However, in addition to not starting in conflict with one another, we also want to ensure that the initial time to conflict $t_c(0)$ is large. As discussed in Section 2.3, in order for the UAVs to avoid each other they need access to each other's position and velocity. We suggest that however this is achieved, the UAVs' awareness will have a range much larger than S such that $t_c(0) \gg \tau$ normally. In reality this might not be the case, for example due to sensor and/or communication errors. Under such circumstances UAVs may have to enact more extreme avoidance manoeuvres in order to avoid a conflict. However the conflict avoidance tested here has been designed in an attempt to balance competing priorities alongside the need to ensure safe separation.

In order to ensure that $t_c(0)$ is large we obtain the following expression by finding the smallest root of Equation 2.19

$$t_c(0) = \frac{\mathbf{r}_{ij} \cdot \mathbf{v}_{ij} - \sqrt{(\mathbf{r}_{ij} \cdot \mathbf{v}_{ij})^2 - |\mathbf{v}_{ij}|^2(|\mathbf{r}_{ij}|^2 - S^2)}}{|\mathbf{v}_{ij}|^2}, \tag{2.19}$$

which can be formulated in terms of the setup parameters ΔR and θ. Using Equation 2.19 and values from Table 2.1, we can plot $t_c(0)$ for the range of θ and ΔR described above, see Figure 2.10.

Parameter	**Value**	**Units**
Cruising velocity (v_{CS})	20	ms^{-1}
Max possible acceleration ($a^{\max}$)	5	ms^{-2}
Desired avoidance separation (S)	30	m
Nominal distance to origin (R_1)	10^4	m

Table 2.1: Parameters and their dimensional values used for simulation results. Parameters for UAVs are inspired by previous work on UAV traffic management [31, 51]. We set the value of R_1 such that the initial time to conflict is large thus allowing us to explore small angles of approach, see Figure 2.10.

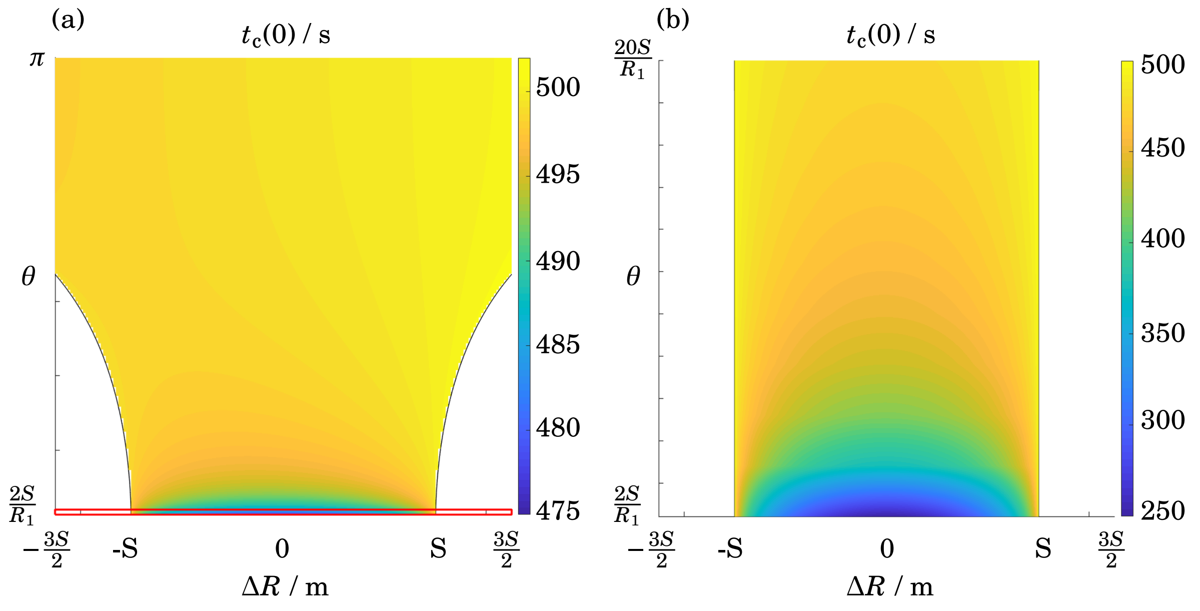


Figure 2.10: (a) The initial time to conflict for all values of ΔR and θ used. (b) A zoomed view of the area contained in the red box in (a), for small angles in the range $\frac{2S}{R_1} \leq \theta \leq \frac{20S}{R_1}$. In (a) and (b) the black lines show the limit of values for ΔR that induce a conflict course, i.e., the curves defined by Equation 2.18. Setups contained below these curves, the white areas, result in no avoidance manoeuvre for the UAVs. These plots are produced using the dimensional values for R_1, S and v_{CS} from Table 2.1. Note that, when $\Delta R \approx 0\,\text{m}$, then $t_c(0)$ rapidly tends to 0 s as θ tends to 0. This demonstrates that if we want $t_c(0)$ to be large and to explore small angle setups we require R_1 to be very large. The minimum $t_c(0)$ experienced by UAVs in any of these setups is 250 s, which is more than 30 times larger than the timescale τ defiend in Equation 2.3.

For $\theta \geq 20S/R_1$ the initial time to conflict is around 500 s, i.e., the time it takes to reach the origin when flying at speed v_{CS}. Note that there is a slight asymmetry introduced here due to the way in which we fix R_1 and lengthen or shorten R_2 which results in $t_c(0)$ being smaller for negative ΔR at the same value of θ. For $\theta \leq 20S/R_1$ the initial time to conflict reduces rapidly with θ to a minimum of 250 s when $\Delta R = 0\,\text{m}$ and $\theta = 2S/R_1$. This suggests that for our purposes $R_1 = 10\,\text{km}$ is sufficient to model two UAVs approaching each other with sufficient time to react since 250 s is more than 30 times greater than the timescale τ defined in Equation 2.3.

Although at this point both UAVs apply the same rules, for the sake of discussion we shall view $i = 1$ as the ego UAV and $i = 2$ as the alter UAV. Figure 2.11 shows the delay $T_1(\Delta R, \theta)$ experienced by UAV $i = 1$. The analysis of these simulations will focus on $T_1(\Delta R, \theta)$ since the UAVs were able to avoid each other successfully for all combinations of ΔR and θ. The geometric symmetry in the setup implies that $T_2(\Delta R, \theta) = T_1(-\Delta R, -\theta)$ and thus T_2 may be recovered from the T_1 plot. This means we can view UAV $i = 2$ as the ego and $i = 1$ as the alter by flipping the signs of ΔR and θ. The key observation from Figure 2.11 is that the problematic situations correspond to small values of $|\theta|$, where the vast majority of the delay is experienced by the UAV who is ego with positive θ. Thus a collaborative manoeuvre is giving rise to an unfair and thus

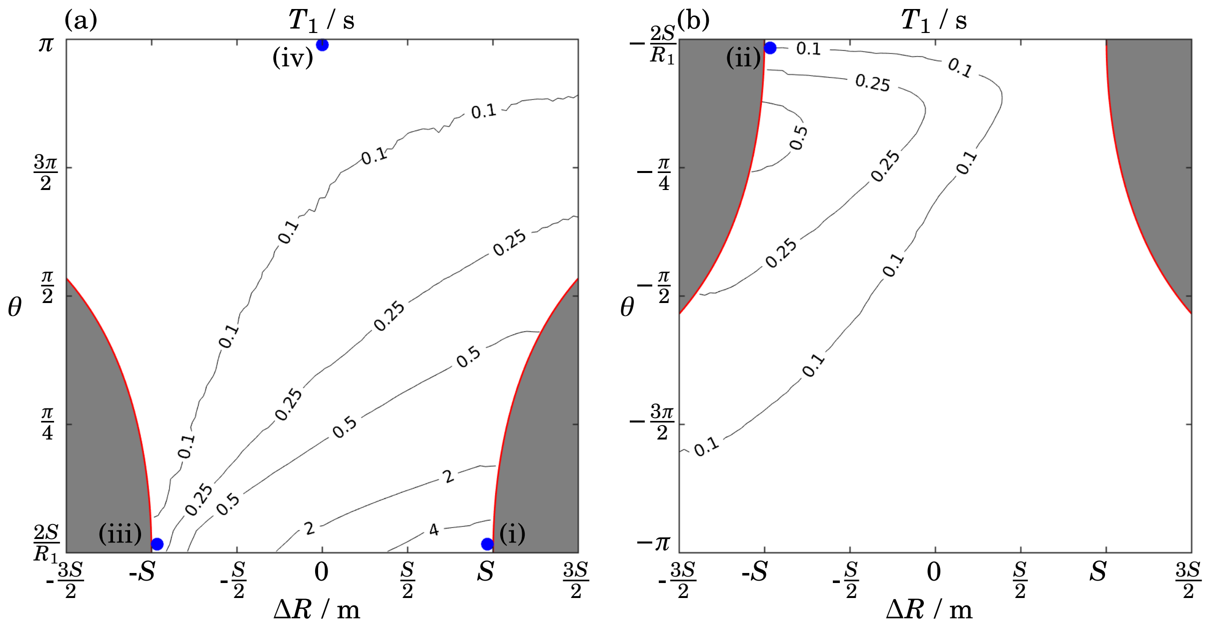


Figure 2.11: Delay experienced by UAV 1 when θ is (a) positive and (b) negative. These plots are produced from 10,000 simulations where θ and ΔR have a resolution of about 0.06 and 0.91 m respectively. The red lines correspond to the curves defined by Equation 2.18, such that the grey areas are setups where no avoidance manoeuvre is undertaken. The delay for UAV 2 is not shown since it can be obtained by reflecting (a) and (b) along both axes, i.e., $T_2(\Delta R, \theta) = T_1(-\Delta R, -\theta)$. Note that the worst delays for UAV 1 are experienced when θ tends to 0 and ΔR tends to is maximum value: this corresponds to the lower right section of (a) in the vicinity of point (i). The delay T_2 is also much smaller than T_1 in this region and therefore the manoeuvre is unfair. This suggests that other avoidance behaviours should be considered. We plot the trajectories of the UAVs for the setups at points (i) – (iv) in Figure 2.12.

undesirable outcome in the delays experienced.

This result is explained in the trajectory plot of Figure 2.12(a). Here, although UAV 1 has a ‘head start’ in the initial setup and therefore might be expected to pass in front of UAV 2, it turns to the right in the avoidance manoeuvre, and thus crosses behind UAV 2, incurring delay. In contrast, UAV 2 follows an almost perfectly linear path. This effect is most severe as the angle θ tends to zero and ΔR approaches its maximum positive value i.e., when UAV 1 is far enough ahead to almost be clear of a conflict course. In Figure 2.12(b) we show a similar situation where the roles of the UAVs have been reversed by flipping the signs of θ and ΔR. Again this corresponds to a scenario where the UAV which incurs the larger delay begins almost clear of the conflict course and in the ensuing manoeuvre the longitudinal order of the UAVs is reversed. In Figure 2.12(c) we show the trajectories that result from the same value of θ as in (a) but now $R_2 < R_1$. In this example UAV 1 still experiences a larger deviation from its original path compared with UAV 2 but the manoeuvre is more gradual resulting in a smaller delay and the longitudinal order is preserved. Finally, in Figure 2.12(d) both UAVs approach each other head

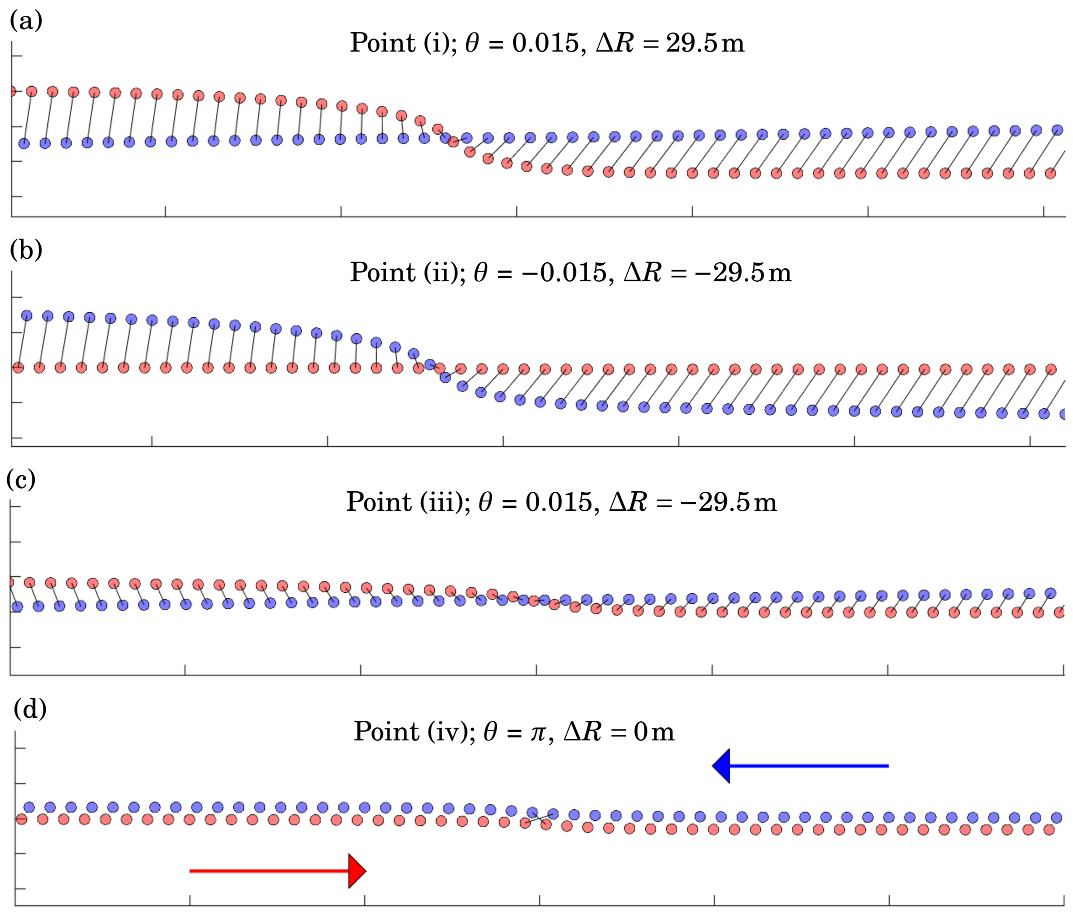


Figure 2.12: Simulated trajectories for UAV $i = 1$ (red) and UAV $i = 2$ (blue) which correspond to points (i) – (iv) in Figure 2.11. In (a) UAV 1 has a head start and would pass in front of UAV 2 but deviates to pass behind it and therefore incurs a significantly longer delay. This is due to an asymmetry in the avoidance velocities produced by the UAVs, see Figure 2.13. In (b) the same behaviour is observed with the roles reversed as the signs of θ and ΔR are flipped compared with (a). In (c) the value of θ is the same as in (a) but now UAV 2 has a head start and the longitudinal order of the UAVs is preserved. In (d) the UAVs approach head on and are both deviated in the same way, leading to small deviations and consequently small delays.

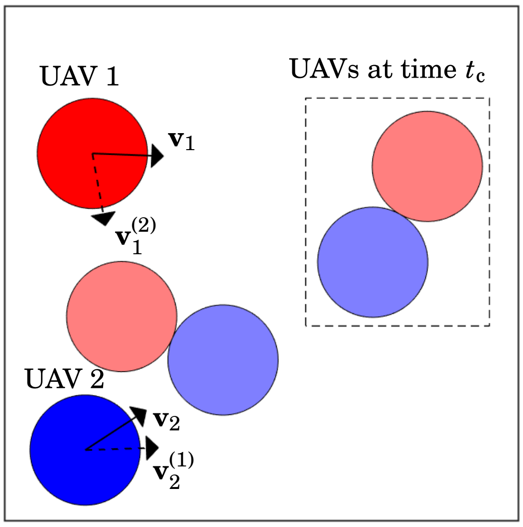


Figure 2.13: UAV 1 (red) and UAV 2 (blue) are on a conflict course when θ is small and ΔR is large. The future positions (translucent circles) inside the dashed box show the point where the two UAVs begin to come in to conflict. The dashed arrows show the avoidance velocity that the UAVs will accelerate towards, based on the conflict avoidance scheme from Section 2.3. This sort of setup leads to UAV 1 experiencing larger delays than UAV 2 because $|\mathbf{v}_1 - \mathbf{v}_1^{(2)}| \gg |\mathbf{v}_2 - \mathbf{v}_2^{(1)}|$ which results in a greater avoidance acceleration via Equation 2.11. Also observe that if UAV 1 adopted velocity $\mathbf{v}_1^{(2)}$ and UAV 2 maintained its velocity, then the longitudinal order of the UAVs is reversed, as we observe in Figure 2.12(a).

on. As such the UAVs undergo a symmetric avoidance manoeuvre where they are both deviated the same amount from their linear paths and as such incur the same delay.

We can understand why the undesirable behaviour in Figure 2.12(a) occurs by considering the avoidance velocities that are generated. In Figure 2.13 two UAVs are on a conflict course with a small angle of approach and UAV 1 would pass in front of UAV 2 when they begin to conflict at time t_c based on their current velocities. The UAVs must adopt avoidance velocities that involve a right hand turn, with a speed of v_{CS}, and assume that the other UAV will maintain its velocity, as laid out in Section 2.3. As such UAV 1 must alter its velocity more dramatically than UAV 2, i.e., $|\mathbf{v}_1 - \mathbf{v}_1^{(2)}| \gg |\mathbf{v}_2 - \mathbf{v}_2^{(1)}|$, which results in it experiencing a larger acceleration where it effectively takes on responsibility for the avoidance manoeuvre alone. In this scenario it might be more natural for UAV 1 to instead turn to its left which should preserve the longitudinal order of the UAVs and result in a smaller deviation. Indeed these results suggest that other avoidance rules, for example in which UAVs can choose an alternative direction of avoidance, might reduce delays for small $|\theta|$ situations. Clearly, large angles of approach do not present such problems as delays are small. As such we suggest that exploring larger values of $|\Delta R|$ for $\theta > \pi/2$ would be unnecessary.

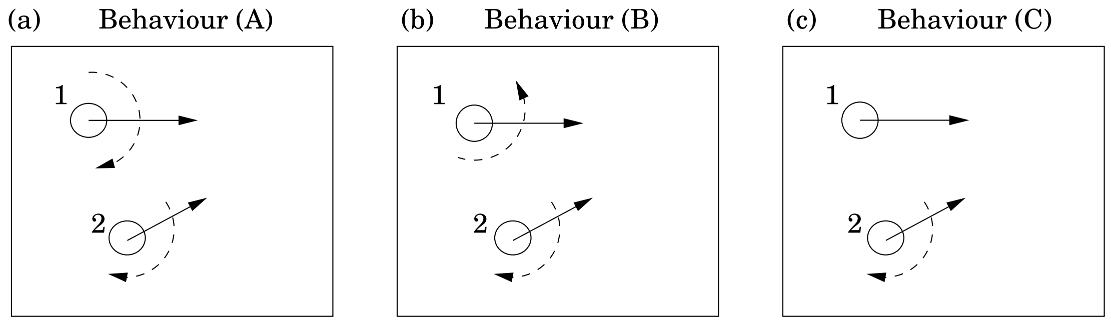


Figure 2.14: Three avoidance behaviours: (a) both UAVs turn to their right, (b) UAV 1 turns to its left, (c) UAV 1 does not engage in an avoidance manoeuvre. In (a), (b) and (c) UAV 2 follows behaviour (A) as it did in Section 2.5.

2.6 Modified avoidance behaviours

Section 2.5 has shown that the avoidance method implemented, while effective for larger angles of approach between the UAVs, can produce unfair outcomes when two UAVs are travelling along almost parallel paths. By requiring UAVs to turn to their right, which we now call behaviour (A), one UAV can experience a much longer delay than the other when they are set up near the limits of ΔR. In this section we consider adaptations to the conflict avoidance method where the behaviour of UAV 1 is changed. Figure 2.14(b) describes a behaviour (B) where UAV 1 turns to its left such that the UAVs turn away from each other. Figure 2.14(b) describes a behaviour (C) where UAV 1 does not employ any avoidance manoeuvre at all but instead carries on in a straight line. However, in any given scenario UAV 2 will continue to implement behaviour (A) as in Section 2.5.

We will only explore small positive angles of approach here, i.e., those setups where UAV 1 can experience long delays when behaviour (A) is used. Due to the symmetry of the setup discussed in Section 2.5, if θ is negative then UAV 2 becomes the ego UAV and would be susceptible to long delays. Therefore, under such circumstances, UAV 2 should adopt either behaviour (B) or (C). Also note that if UAV 1 adopts behaviour (B) while θ is negative and UAV 2 adopts behaviour (A) then the UAVs will instead turn towards each other, increasing the likelihood of a conflict. This scenario should be avoided and as such we suggest that the rules for deviating from behaviour (A) would have to be clearly laid out by the UTM system and obeyed by all participating UAVs.

We first consider the delay $T_1^{\mathrm{B}}(\Delta R,\theta)$ experienced by UAV 1 while it undergoes behaviour (B), see Figure 2.15(a). The delay experienced by UAV 2 is given by $T_2^{\mathrm{B}}(\Delta R,\theta) = T_1^{\mathrm{B}}(-\Delta R,\theta)$. Unlike before, the UAV which starts closer to the origin is generally better off in terms of delay. Compare with Figure 2.11(a) and observe that behaviour (B) dramatically reduces the delay for UAV 1 for combinations of $\Delta R \lessapprox S$ and $\theta \simeq 0$ for which the delay was worst for behaviour (A). This is at the expense, see Figure 2.11(b), of a modest increase in delay for UAV 2. Note, however, that behaviour (B) performs badly at large θ values compared to behaviour (A). This is explained by

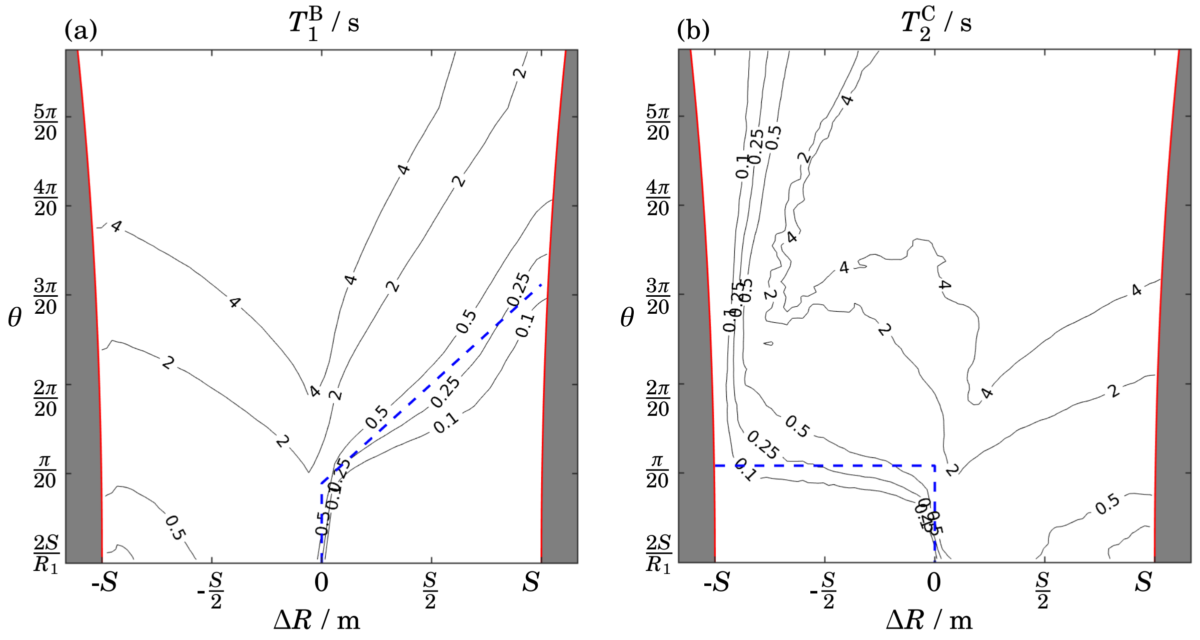


Figure 2.15: Delay experienced by (a) UAV 1 for behaviour (B) and (b) UAV 2 for behaviour (C). The delay T_2^B can be obtained by reflecting (a) in the ΔR axis. Both behaviour (B) and (C) clearly do not perform well at larger angles. The areas under the blue dashed lines are where we propose UAV 1 should adopt behaviour (B) and (C) respectively in accordance with Equations 2.20 and 2.21. This does however result in a small increase in the delay experienced by UAV 2.

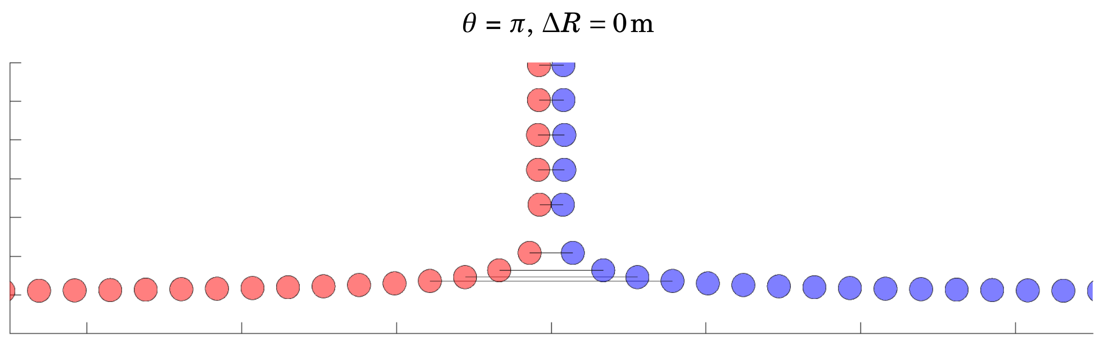


Figure 2.16: Simulated trajectories when UAV $i = 1$ (red) adopts behaviour (B) and UAV $i = 2$ (blue) adopts behaviour (A) when approaching each other head on. As both UAVs turn in opposite directions they now mirror each other's manoeuvre and adopt paths that are perpendicular to their original paths. This leads to extreme delays experienced by both UAVs.

the example flight paths in Figure 2.16 where two UAVs approach each other head on. Since UAV 1 now turns to its left while UAV 2 still turns to its right the two UAVs begin to fly parallel to each other along a path that is perpendicular to their desired direction of motion, leading to extreme delays.

For behaviour (C), UAV 2 experiences similar delays to those incurred when UAV 1 follows behaviour (B): compare Figure 2.15(b) with Figure 2.15(a) reflected in the $\Delta R = 0$ line. We might

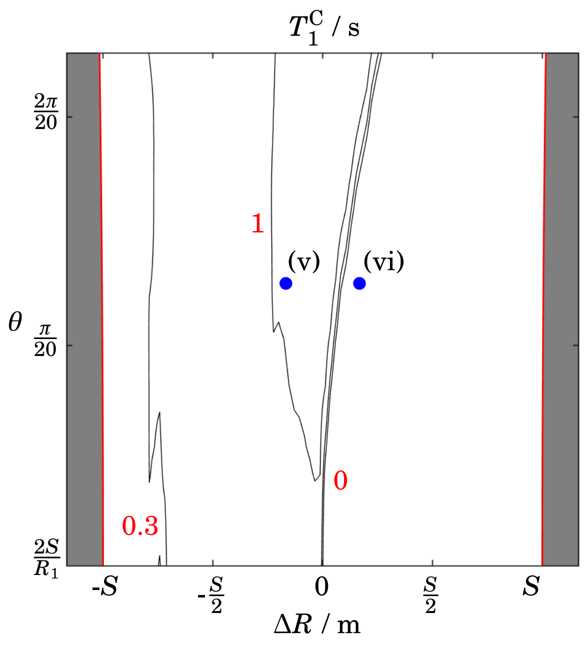


Figure 2.17: Delay experienced by UAV 1 when following behaviour (C). For small angles, when ΔR is negative it experiences a negative delay where it is sped up due to the emergency acceleration. The trajectories for both UAVs at points (v) and (vi) are shown in Figure 2.18(a) and (b) respectively.

expect UAV 1 to have a zero delay for behaviour (C) but this is not the case, see Figure 2.17, due to the effect of the emergency acceleration from Equation 2.6. In fact, when ΔR is positive, i.e., when UAV 1 has a head start, it can experience small negative delays. This is where the emergency acceleration effectively pushes UAV 1 forward, see Figure 2.18(a). However, when ΔR is negative the emergency acceleration acts to push UAV 1 further back. This leads to the worst experience when $\Delta R \approx 0\,\text{m}$ since, when this is the case, the emergency acceleration takes a long time to resolve the conflict course.

Based on the results both behaviour (B) and (C) can be used to reduce the delay incurred by UAV 1, but at the cost of additional delay to UAV 2. We therefore propose that UAV 1 should choose from behaviour (A), (B), or (C), broadly according to the following principles:

- Adopt behaviour (B) or (C) if

  $$T_1^{\text{B,C}} \le T_1^{\text{A}}, \tag{2.20}$$

  that is, UAV 1 improves its own experience by deviating from behaviour (A). Further

- we require

  $$\left[T_1^{\text{A}} - T_1^{\text{B,C}}\right] \ge \left[T_2^{\text{B,C}} - T_2^{\text{A}}\right], \tag{2.21}$$

  so that the reduction in UAV 1's delay is greater than any increase in UAV 2's delay, and thus there is overall net system benefit.

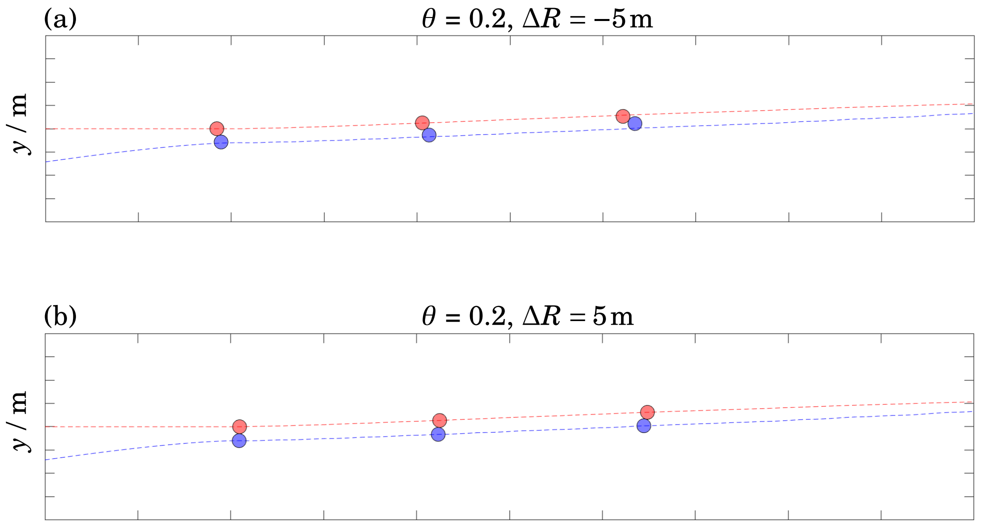


Figure 2.18: Simulated trajectories when UAV $i = 1$ (red) adopts behaviour (C) and UAV $i = 2$ (blue) adopts behaviour (A) when $\theta = 0.2$ and (a) $\Delta R = -5\,\mathrm{m}$ and (b) $\Delta R = 5\,\mathrm{m}$. Both UAVs follow a similar flight path where UAV 2 begins to avoid UAV 1 and turns to the right until both UAVs are travelling almost parallel to each other. However in (a) UAV 1 is pushed further behind UAV 2 due to the emergency acceleration while in (b) it is pushed further ahead, leading to a small negative delay.

If behaviour (B) and (C) satisfy these conditions at the same time, then we suggest that UAV 1 should adopt the behaviour which results in the smallest total delay for the pair of UAVs, see Figure 2.15. This provides a heuristic for a *hybrid avoidance method* which allows UAVs to choose their avoidance behavior, assuming behaviour (A) as the default, based on the parameters θ and ΔR. It is possible to apply this heuristic in more general scenarios by taking a pair of UAVs' current states and recovering the setup from Figure 2.8(a) in the ego UAV's frame of reference, e.g, by assuming the point at which their paths cross is based on their instantaneous velocities and by finding ΔR with respect to that point. From here on we will refer to the conflict avoidance method from Section 2.3, where both UAVs follow behaviour (A) regardless of ΔR or θ, as the *simple avoidance method*.

2.7 Conclusion

In Chapter 2 we have designed an S&A conflict management method. This method was based on the velocity obstacle method which first allows a UAV to determine if it and a neighbour are on a conflict course. If this is the case then the UAV produces an acceleration component that acts to change the UAV's velocity such that the UAV adopts a new trajectory where the minimum separation with its neighbour is equal to a safe distance S. Using simulations of pairs of UAVs, we developed a hybrid avoidance method where a UAV chooses its behaviour in order to minimise the delay incurred during the manoeuvre.

There are a number of possible extensions to this work. We have assumed throughout that all UAVs in a UTM system are of the same type and priority which is unlikely to be true of a real-world scenario. Likewise, we have not considered faulty or deliberately malicious UAVs. In the case of the latter it may be impossible to design an appropriate S&A method, particularly if the malicious UAV is capable of faster speeds, since then it could pursue benevolent agents and force collisions. Perhaps the most pressing extension however is applying the conflict management method and UAV model designed here to a scenario with many more UAVs. We will begin to explore this scenario in Chapter 3.

2.7.1 Contributions

C2.1 Designed a method for controlling autonomous UAVs by combining various required acceleration components. The acceleration components are produced according to rules designed to either drive a UAV towards a desired destination or to maintain a minimum separation with other UAVs.

C2.2 Based on current ICAO rules and the velocity obstacle method, we designed a pairwise avoidance rule where UAVs will turn to their right in order to avoid a conflict with a neighbour, provided they both know the position and velocity of the other.

C2.3 We simulated pairs of UAVs and showed that this right-handed rule ensured that conflicts are resolved for a wide range of setup parameters, but when the angle of approach between them was small it can lead to one agent experiencing a significantly longer delay.

C2.4 We designed a hybrid avoidance rule where the direction a UAV turns to avoid another is based on the angle between it and its neighbour's velocities and the relative distance to the point where their paths intersect. This hybrid rule then minimises the total delay experienced by both agents.

Chapter 3

Performance of Tactical Conflict Management for Representative UAV Traffic Scenarios

While the pairwise interactions explored in Chapter 2 were useful in the development of our conflict management method, they do not well model the full complexity of future airspace. As discussed previously, we expect the density of UAV traffic to increase in the future [30, 31]. Under such high demand we expect much more complex interactions to arise as UAVs need to avoid multiple neighbours at once or else encounter secondary avoidance interactions as a result of a previous avoidance manoeuvre. As such we need to design a test scenario which will not only involve large numbers of UAVs but induce conflicts between them. To frame this design we consider the use case of UAV goods delivery where two streams of UAVs cross at a point. This scenario, implemented in simulation, will then be used to answer **RQ2** as defined on page 14 by considering the time taken for a UAV to transit between two fixed points, compared with an ideal shortest time path.

In Section 3.1 we discuss various UAV use cases and their suitability for producing the high density, complex conflict avoidance traffic scenarios we want in order to test the designs developed in Chapter 2. We also provide a discussion for why UAV crossings will arise naturally, even in an unrestricted airspace, for the UAV delivery use case when deliveries are made between fixed points. We present a simple model for UAV ‘ports’ in Section 3.2 which can act as sources and sinks in simulation and therefore produce traffic streams. This port model is used in Section 3.3 to design a simple crossing setup, implemented in simulation, that uses four ports to produce two streams of traffic that cross at a point. We then use this setup to compare the performance of the simple and hybrid avoidance methods. We show that despite the two streams crossing perpendicular to each other, the hybrid avoidance method increases both the fairness and efficiency of the crossing. Further, we demonstrate that this is due to secondary avoidance interactions, where UAVs that are previously deviated from their straight line path reconverge with their stream near the destination port. Given these results, in Section 3.4 we conduct a more thorough exploration of the impact that demand has on the performance of a single crossing when hybrid avoidance is used. We sweep through a variety of demand levels on each stream independently and thus show that the performance of such a crossing deteriorates rapidly and non-linearly at high demands. Furthermore we find that the right-handedness of avoidance behaviour (A) (see Section 2.6) causes the performance of one stream in particular to be highly dependent on the demand experienced by the other. These results suggest that splitting a stream into two parallel streams that share the demand between them equally can improve system performance despite increasing the number of crossings in the system. As such we verify this in simulation in Section 3.5. In fact we show that this traffic splitting method can lead to greater than expected improvements: this is because an unsplit stream is spread out by interactions with the first crossing which reduces the effective demand at the second crossing. Finally, in Section 3.6, we provide a brief discussion of the results in this chapter along with an overview of our contributions.

3.1 Background

In this chapter we develop a UAV traffic scenario which will be likely to induce potential conflicts. Such a scenario will therefore provide us with a way to explore the effectiveness of the conflict avoidance methods from Chapter 2 for larger volumes of UAV traffic.

How UAVs will interact with the UTM system will depend on the use case they are used for. Some UAVs are going to operate within a well defined area, e.g, site survey [79–81] or search and rescue [82, 83]. We can imagine that the UTM system may designate such areas as ‘restricted flight zones’ (RFZ) upon which other UAVs should not encroach, a technique known as UAV Geofencing [84, 85]. If this were the case, we might expect that UAVs operating within such an RFZ would rarely need to use any conflict avoidance method and that, when it is needed, it would be in a similar pairwise interaction mode as those explored in Chapter 2. Other use cases, such as those where UAVs are used to supplement communications infrastructure [86–88], require that UAVs remain stationary or maintain a fixed flight pattern. As such, we might expect that UAVs employed for these use cases would be exempt from or otherwise given special priority by the conflict avoidance method employed by the UTM system.

We therefore suggest that use cases which routinely involve UAVs flying throughout the airspace would be better suited to induce more complex avoidance interactions. Two such use cases are surveillance [89, 90] and goods delivery [91]. An example surveillance mission might involve UAVs following set search paths throughout the airspace and then following a particular target once it is found. This sort of mission may induce conflicts between UAVs where search paths intersect or when a UAV must follow a target where the resulting flight path can not be

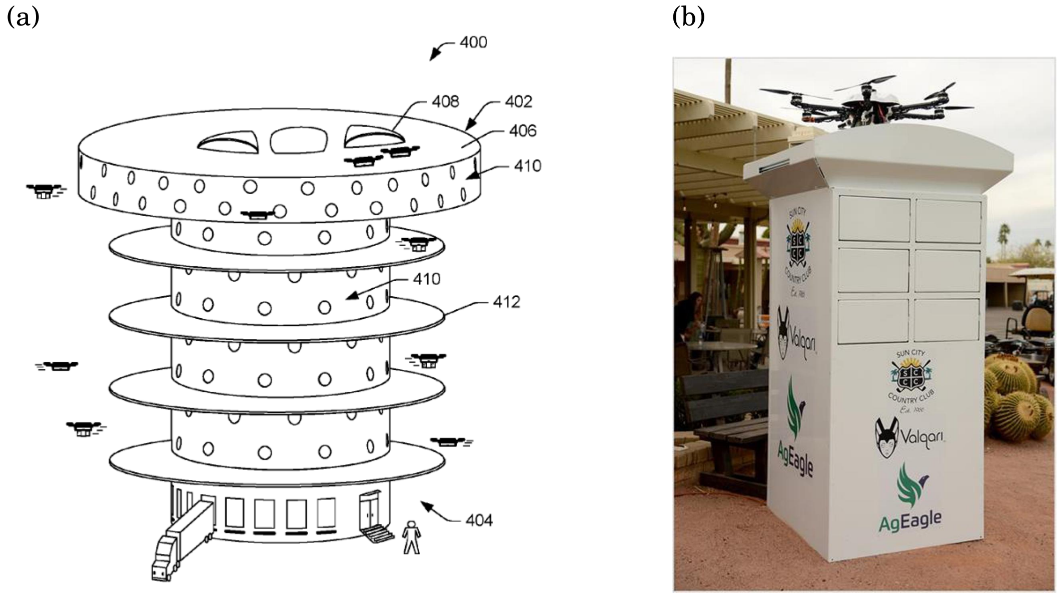


Figure 3.1: (a) Schematic of a "multi-level fulfillment center" proposed by Amazon. Trucks offload parcels at the ground floor which are then processed and loaded on to UAVs for delivery. UAVs then enter and exit the building through portals placed along the exterior and roof. Reproduced from Curlander *et al.*, 2017 [92]. (b) A communal mailbox, developed by Valqari, which enables a UAV to land on the roof and deposit a parcel which can then be picked up by a customer at a later point in time. Reproduced from Silver, 2021 [93].

known beforehand. However it is unlikely that even surveillance missions which employ multiple UAVs will lead to the sort of demand predicted for delivery applications. Combined with the added complexity of modelling aspects of a surveillance mission that are not directly related to a UTM controlled airspace (search routes, target behaviour, etc.), we will instead focus on goods delivery for the design of our UAV traffic scenario.

While there are several approaches to UAV goods delivery, such as the UAV-truck partnership discussed in Section 1.1, we will consider UAV goods delivery between fixed points which we call *point to point* (P2P). In P2P delivery missions, UAVs will first depart from fixed infrastructure, e.g, warehouses, then fly to the delivery location and finally return to its original location for refuelling and resupply, see Figure 3.1. For commercial delivery, the ability to deliver parcels directly to a customer's residence would be ideal but carries with it a number of challenges. Some of these are present for traditional delivery but exacerbated for an autonomous UAV, for example how to securely deliver a parcel to an address where no one is there to receive it. Others are unique to UAVs such as ensuring that there is sufficient space for the UAV to land without undue risk to pedestrians. The latter challenge is particularly pronounced in urban settings where many customers will not have access to a garden or other open space that can serve as a landing area. One solution is to place delivery infrastructure throughout a given area in the form of "delivery points" as proposed in [47], see Figure 3.2. These would enable UAVs to land and deposit their payload in a secure location for pick up at a convenient time by customers.

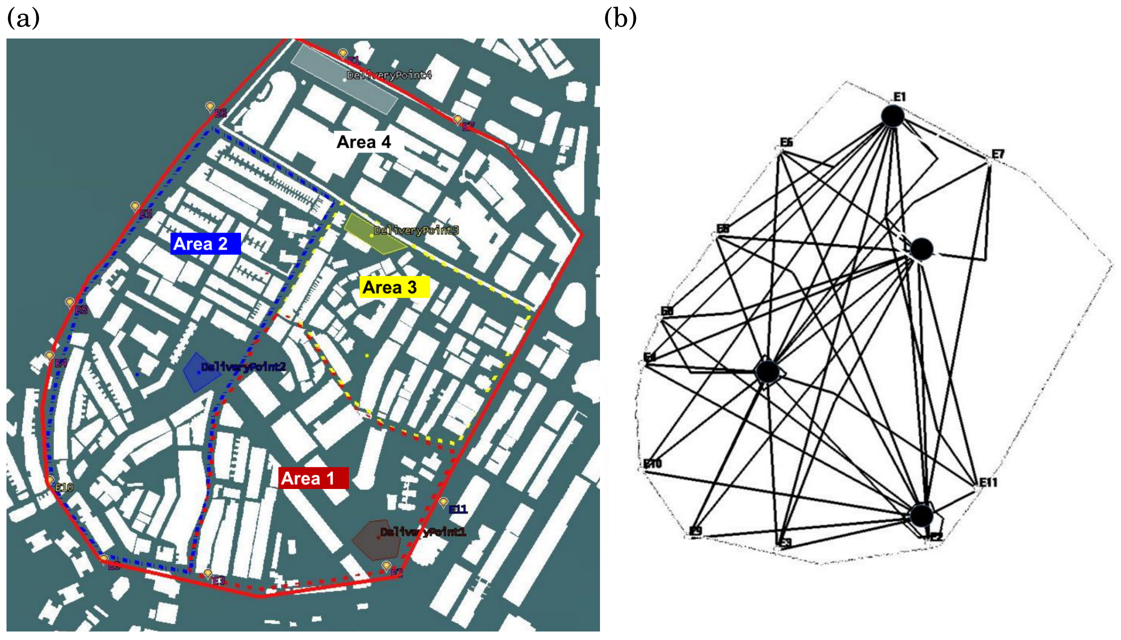


Figure 3.2: (a) A segment of airspace (defined by the solid red perimeter) over a residential area with several designated points of entry for UAVs placed along the border. The residential area is split in to four sectors each of which contains a UAV “delivery point” (shaded areas). (b) The proposed paths that UAVs will take between the various entry and delivery points. Note that even with only four delivery points there are many places where one path crosses another. UAV conflicts are likely to occur around these points and may involve many more UAVs than the pairwise interactions explored in Chapter 2. (a) and (b) are reproduced from Salleh *et al.*, 2017 [94].

This kind of P2P delivery between large warehouses and delivery points does not fully realize the convenience for customers that delivering directly to their door would. However, there are still advantages over traditional delivery methods. For example, provided there is an appropriately automated warehouse, UAV deliveries can be carried out on demand at any time of day. Even with an autonomous delivery truck, the system would have to wait until a certain number of orders have been placed before starting a delivery mission so that capacity is not wasted. We therefore believe that exploring this delivery method is useful both as a starting point for UAV traffic management more generally and in its own right as potential future use case.

The size and scale of these delivery points could vary depending on factors such as demand, available space and whether they are tied to specific delivery companies (e.g., Amazon lockers). It also seems reasonable to assume that locations such as universities, hospitals, large office blocks etc. could be fitted with their own delivery points. P2P delivery thus results in a UAV traffic network where the warehouses and delivery points form origin-destination (O-D) pairs. We can then model a UAV traffic scenario using an O-D network which generates UAVs whose motion we model as in Chapter 2.

While the concept of operations for P2P UAV goods delivery is still under development, it seems likely it will be more suited to urban areas. Urban areas provide a high density of customers which can be reached with relatively short flight distances. Baloch *et al.* [95], for example, considered how the number of UAV origins (warehouse facilities), their location, and the exact type of delivery services offered affect the economic desirability of a P2P UAV goods delivery system in New York City. Furthermore, they consider 10 km to be a reasonable maximum range which also ensures UAVs can make at least two deliveries per hour. However operating UAVs in the urban airspace comes with a variety of challenges. Some of these were discussed in Section 1.1, such as communication and localisation challenges. There are also several factors that may restrict the usable urban airspace which will lead to further increases in the traffic density. Perhaps the most obvious is the possibility of UAVs being expected to minimise risk to pedestrians by minimising the time spent flying overhead, for example by flying over existing roads or rail lines. Similarly, Sanjab *et al.* [96] argued that the security of UAVs should be considered when planning routes and suggested that UAVs should avoid flying above high risk areas. This cautious behaviour could lead to a further concentration of traffic in areas of airspace deemed safe. Finally there is the possibility that UAVs will be restricted to one or more flight layers, as discussed in Section 1.1.

For simplicity though we will consider an unrestricted layer of airspace, as in Chapter 2, such that the UAVs can manoeuvre freely in a 2D plane. We expect that potential conflicts will still be common in such a system due to the fixed location of the O-D pairs. One of the advantages of UAVs for delivery is the ability to reduce journey time by avoiding the congestion that affects ground vehicles and adopting more direct routes than are allowed by the road network. In an unrestricted airspace this advantage is maximised by UAVs taking the shortest straight line path to their destination. As such, we would expect UAVs to form streams of traffic along the edges of the O-D network. Given that an O-D network for an urban airspace is likely to consist of many nodes (at least one warehouse facility per delivery company and many delivery points located throughout the urban area) then we would expect some of these streams of traffic to overlap. This would create something analogous to a crossroads in the sky where UAVs are likely to be on a conflict course with one another.

The problem for a UTM system then becomes how to manage these crossings and ensure that traffic continues to flow while maintaining safe separation. As discussed in Section 1.1, while a centralised scheduling system could be used to facilitate such a crossing, it becomes increasingly difficult as traffic demand and the number of crossings increase. We will therefore explore in this chapter under what, if any, circumstances such a UAV crossing can be facilitated by the avoidance methods from Chapter 2. We will consider metrics for fairness and efficiency as before and study how the traffic demand and different network setups will affect these.

One potential use case for UAVs that is similar to goods delivery is *urban air mobility* (UAM). In UAM the UAVs effectively become flying taxis, able to bypass the busy and often congested

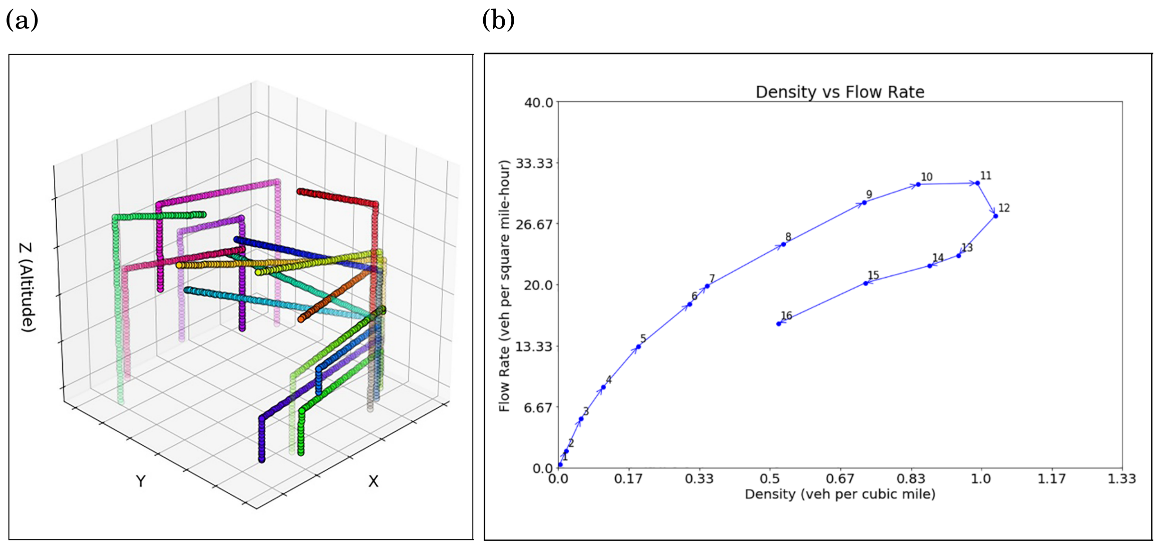


Figure 3.3: (a) Snapshot from simulations of a UAM traffic scenario which shows the paths taken by various UAVs between UAM pads. (b) Density and flow rate fundamental diagram produced by 600 UAVs whose departure times were normally distributed to emulate a peak demand. (a) and (b) are reproduced from Cummings *et al.*, 2021 [100].

urban road network and therefore hopefully provide shorter journey times. Similar to the P2P delivery concept we described above, UAM will require dedicated ground-based infrastructure to serve as landing and takeoff sites, as well as parking and other maintenance services. There is currently no agreed terminology for this infrastructure, but examples include “UAM port” or “pad” as used by NASA [97] which differentiates between large sites which can accommodate many UAVs and their maintenance, and smaller sites with smaller capacity.

One potential barrier to UAM, identified in a recent market study commissioned by NASA [98], is the assumption of one passenger per trip. This can result in low utilisation of the UAVs and therefore low profitability. A solution to this is to consolidate demand. Ale-Ahmad [99] presents two possible models of demand consolidation; “air sharing” and “air pooling”. The former is envisioned as being similar to ride-sharing where a passenger may experience multiple stops during a journey to allow other passengers to board and alight as needed. The latter involves all passengers boarding a UAV at the same origin and flying to the same destination. This model involves some of the passengers needing to undertake a “ground leg” of their journey prior to and / or after the “aerial leg”. From a UTM perspective, air pooling would likely reduce the number of simultaneous flights in a given airspace compared with a scenario where no demand consolidation is used. While this may seem like a positive (overall traffic density is reduced) it may also further concentrate traffic into a small number of busy routes (increasing local traffic density).

Regardless of the particular use case, it would be useful to know how different levels of demand will affect the flow of UAV traffic. Macroscopic fundamental diagrams (MFD) are well established in ground vehicle traffic modelling where they are used to describe the relationship between traffic flow and accumulation. Generally speaking, flow increases with accumulation up to some maximum level (the system capacity) before decreasing back towards zero (traffic jam). Recent work has shown that the MFD approach can be applied to UAV traffic. Cummings *et al.* [100] simulated a UAM traffic scenario where UAVs travel between UAM pads, see Figure 3.3(a). From this various relationships between flow rate, speed and density are obtained, see Figure 3.3(b). Similarly, Haddad *et al.* [101] also used simulation to produce an MFD for UAVs. With this they then designed an "adaptive boundary feedback flow control". Essentially, this is a method for segmenting the airspace and then controlling the flow between segments in order to improve system performance by ensuring traffic accumulation does not become too high in any segment of the airspace. While we do not develop these ideas in this thesis we will briefly discuss how we would develop an MFD for our dynamical model of UAVs in Chapter 5. As in previous work this can then be used in traffic management as a metric to decide when and how intervention is needed.

3.2 UAV origin-destination node model

In Section 3.1 we discussed the need to develop a UAV traffic scenario in order to test the conflict avoidance method designed in Chapter 2. To do this we haven chosen UAV delivery as the use case to inform the design of this scenario, the starting point of which is the assumption that UAVs will need to take off from and land at fixed points which we call *ports*. The size and scale of these ports could vary widely based on the particular details of the use case and constraints imposed by the operating environment. For example, a UAV port may be a purpose built building with many landing pads and other specialised systems designed for loading and maintaining the UAVs. On the other hand, it may be a single landing pad on top of a hospital. This section then will design a simple model for UAV ports which can be used to define a UAV traffic scenario.

Each port is defined in the 2D plane by some position, $\mathbf{p}_i$, and a radius R_{LZ} (where 'LZ' denotes landing zone) assumed to be common for all ports. As discussed in Section 3.1, the placement of these ports in reality is subject to many considerations and may have implications on the performance of the traffic scenario, but we will initially limit any investigation to simple setups, the first of which is described in Section 3.3. The main consideration for our purposes is to ensure that there is at least one point where the traffic streams cross in order to induce conflict avoidance manoeuvres.

The radius R_{LZ} does not necessarily represent the physical area taken up by the port but instead an area where the ports can reliably exert centralised control over UAVs, see Figure 3.4(a). We assume that ports have little to no control over a UAV once it has departed which is enabled by the decentralised nature of the conflict management. Therefore, according to this assumption, the UAVs operate in a distributed traffic mode which should reduce the requirement for a constant real time link between UAVs and ports, and thus relieves some pressure on what might already be a heavily loaded communication network given some of the issues discussed in Section 1.1. However, we assume that when a UAV begins to land, the port will take on responsibility for ensuring that the UAV lands safely. As such, when a UAV is within R_{LZ} of its destination port it is considered to be *landing* and is no longer considered by other UAVs for the purposes of conflict avoidance. UAVs which are not landing we call *active*. These assumptions are predicated on the ability of ports to ensure high quality communication and localisation systems are available in the area defined by R_{LZ}.

Ports may also be able to implement other techniques to help ensure that landing UAVs are separated from general traffic: for example, by reserving a layer of airspace below the general UAV traffic layer. This would allow landing UAVs to descend once they are within R_{LZ} and provide a ‘waiting’ area if there is no landing pad available, see point (ii) in Figure 3.4(a). For simplicity we assume that ports have a near infinite landing capacity, though this assumption could be relaxed in future work.

Each of the ports in a system form origin-destination (O-D) pairs with every other port. Streams of traffic between these can then be described using a demand parameter $\lambda_{i,j}$ which corresponds to an average rate at which UAVs depart from port i with port j as their destination. These demands can be used to construct a demand matrix and therefore the traffic scenario can be thought of as a graph where the nodes correspond to ports. Figure 3.4(b) presents a top down view of an example setup with five ports which is described in general by the demand matrix

$$\mathbf{M} = \begin{bmatrix} 0 & \lambda_{1,2} & \lambda_{1,3} & \lambda_{1,4} & \lambda_{1,5} \\ \lambda_{2,1} & 0 & \lambda_{2,3} & \lambda_{2,4} & \lambda_{2,5} \\ \lambda_{3,1} & \lambda_{3,2} & 0 & \lambda_{3,4} & \lambda_{3,5} \\ \lambda_{4,1} & \lambda_{4,2} & \lambda_{4,3} & 0 & \lambda_{4,5} \\ \lambda_{5,1} & \lambda_{5,2} & \lambda_{5,3} & \lambda_{5,4} & 0 \end{bmatrix}, \tag{3.1}$$

where the values of the leading diagonal ($\lambda_{i,i}$) are zero because we assume UAVs will never take off and land at the same port.

The total departure demand experienced by a port i is given by taking the row sum of the demand matrix

$$\lambda_i = \Sigma_j \lambda_{i,j}, \tag{3.2}$$

which can be used to generate UAVs at random. In the absence of real data on UAV delivery demand rates we have chosen to model this using a Poisson process. This is a parsimonious choice of distribution that is widely used in many areas of research. Also, since the only input is the average rate λ_i we do not need to make any other assumptions about the system.

When a UAV is generated in this way it takes off immediately with speed v_{CS} provided the airspace is clear: that is, when there are no other active UAVs within some distance $S_{\mathrm{TO}} \geq S$ of the port (where ‘TO’ denotes takeoff). This ensures that no UAV may take off while it would be in conflict with another UAV. Note also that, given the simplistic way we have modelled landing

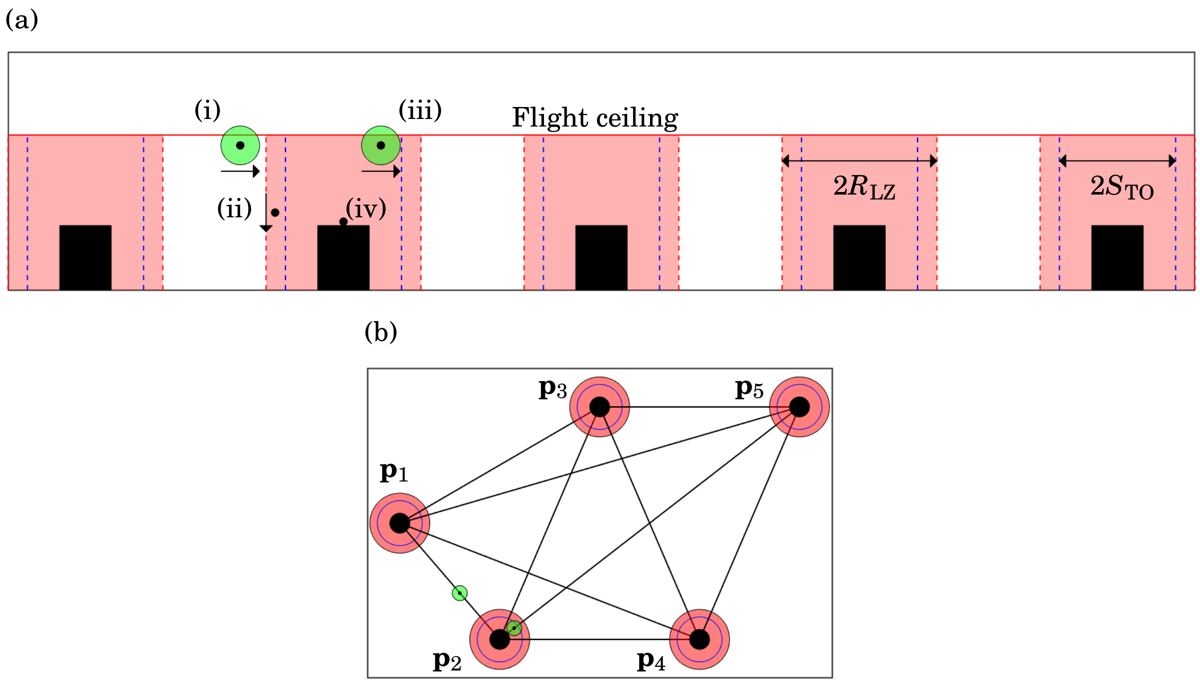


Figure 3.4: (a) A schematic diagram of five UAV ports (black area) viewed from the side. UAVs fly just below the flight ceiling imposed by regulation toward their destination port while trying to maintain a minimum separation of distance S (green area) from other UAVs (i). Once a UAV is within a distance R_{LZ} (red area) of their destination port they no longer take part in avoidance manoeuvres and begin to descend toward the port (ii). UAVs ascend from their port up to the flight ceiling before they begin to move toward their destination (iii). A UAV will not take off while another active UAV is within a distance S_{TO} (blue dashed lines) of the port (iv). (b) A top-down view of the example port layout. We assume that UAVs will fly along the edges of the network, where possible, until they undergo avoidance manoeuvres. As such there are points where two edges cross. We expect avoidance manoeuvres to be common at such points.

UAVs, so long as $S_{\mathrm{TO}} \leq R_{\mathrm{LZ}}$, then UAVs which are landing at the port will not stop a new UAV from departing.

In the case where a UAV can not depart it is added to a queue maintained by the port. At any point when the airspace is clear, a UAV from the queue may depart. In reality each port may only be able to accommodate queues of a certain size due to factors such as the number of landing pads, but for simplicity we consider ports to have an infinite queue capacity. However, so long as

$$\lambda_i \leq \frac{v_{\mathrm{CS}}}{S_{\mathrm{TO}}}, \tag{3.3}$$

then the queue should be stable (that is, the length of the queue will not grow indefinitely). Also, if λ_i tends toward this upper limit then the UAVs will tend to depart with uniform separation S_{TO}.

Upon departure a UAV is given its destination using a weighted sum technique based on the port's total demand and each component demand. So the probability that a UAV is assigned a

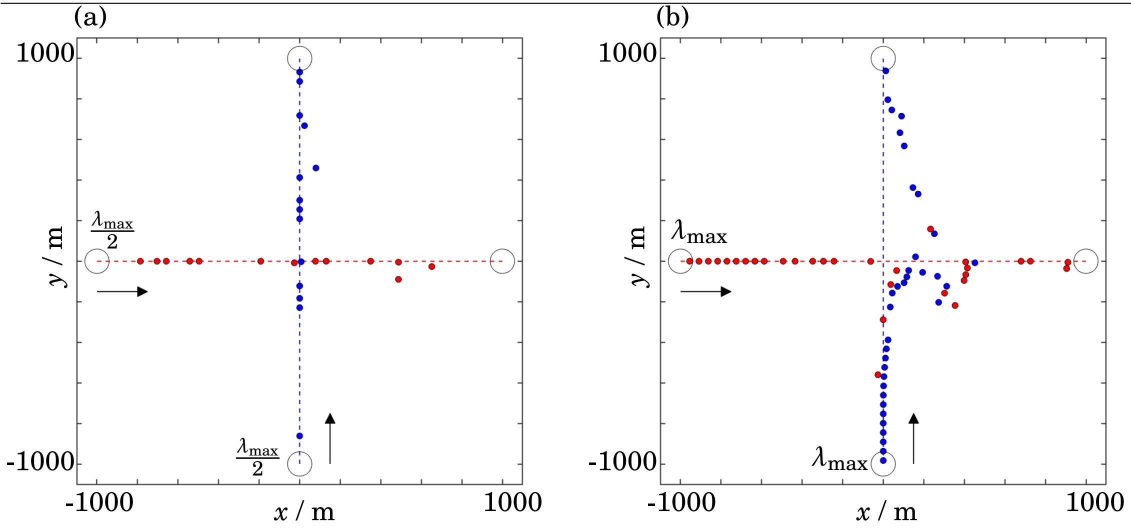


Figure 3.5: A snapshot of simulations using the experimental setup described in Section 3.3 when the demand at both origin ports is (a) $0.5\lambda_{\max}$ and (b) $\lambda_{\max}$, where $\lambda_{\max}$ is defined in Equation 3.9 on page 57.

particular destination port j is

$$P(j) = \frac{\lambda_{i,j}}{\lambda_i}, \tag{3.4}$$

the position of which also determines the UAV's initial velocity. The UAV is placed in the 2D plane at position $\mathbf{p}_i$ and the direction of the initial velocity is given by Equation 2.1 where $\mathbf{r}_i^{\mathrm{WP}} = \mathbf{p}_j$.

See Appendix A for more detail on how this model for UAV ports is implemented in our simulation architecture.

3.3 UAV crossing setup

Given the model of UAV ports described in Section 3.2 we can now construct a test setup with which to explore the conflict avoidance developed in Chapter 2 in scenarios with large numbers of UAVs. We will start by discussing the setup itself, including our choices of dimensional values for parameters and the different metrics we will use to analyse the output from simulation. Then we will present results which compare the performance for simulations where the UAVs either follow the original right-handed avoidance behaviour or the hybrid avoidance method.

For simplicity, we will initially consider test setups where only two streams of UAV traffic intersect at a point, see Figure 3.5. This sort of crossing can be considered as a building block for more realistic scenarios where many ports are connected in more complex layouts. In such a system a UAV might experience many intersections with other streams and so we aim to understand the behaviour at a single crossing. However, even a simple crossing like this can give rise to interesting behaviour for road vehicles, in particular for autonomous vehicles [102, 103].

Parameter	**Value**	**Units**
Distance between ports (L)	2000	m
Port Radius (R_{LZ})	60	m
Takeoff separation (S_{TO})	45	m

Table 3.1: Parameters and their dimensional values used for simulation results.

The setup shown in Figure 3.5 consists of four ports at positions $(\pm L/2, 0)^{\text{T}}$ and $(0, \pm L/2)^{\text{T}}$ which form two sets of origin-destination pairs. UAVs are generated at the origin ports at $(-L/2, 0)^{\text{T}}$ and $(0, -L/2)^{\text{T}}$, with initial velocities $v_{\text{CS}}(1,0)^{\text{T}}$ and $v_{\text{CS}}(0,1)^{\text{T}}$, according to independent Poisson processes as described in Section 3.2. There are thus two traffic streams, a *horizontal* stream (as viewed on the page) and a *vertical* stream (as viewed on the page) with Poisson rates λ_1 and λ_2 respectively.

Beside the demands, which we will vary throughout this chapter in order to explore their effect on the crossing, we will set the other setup parameters to fixed values, see Table 3.1. The port radius R_{LZ} and take off separation S_{TO} are assumed to be common to all the ports in the setup. Given the simplistic way in which we model UAVs landing, it is important that $R_{\text{LZ}} \ll L$ so that the UAVs operate in a distributed control mode for most of their journey, using the UAV model designed in Section 2.2. If this is not the case then the scenario becomes trivial in terms of conflict avoidance. However, a larger value for R_{LZ} is useful as it enables more UAVs to land simultaneously while also ensuring that those UAVs in the process of landing do not interact with any departing UAVs. We therefore set $R_{\text{LZ}} = 2S$. We also set $S_{\text{TO}} = 3S/2$ which ensures UAVs do not come into conflict in the early stages of their flight.

The UAVs themselves will use the same values for their internal parameters as those used in Chapter 2, see Table 2.1 on page 35. We also introduce the concept of a conflict horizon here. If two UAVs are on a conflict course but the time to conflict t_c is greater than the natural time scale τ, defined in Equation 2.3, then no conflict avoidance manoeuvre is undertaken. Once a UAV has taken off from its origin port, it will fly toward its destination on a straight path at constant speed v_{CS} until it interacts (by S&A manoeuvres) with another UAV. Interactions between the two traffic streams are very likely in the vicinity of the crossroads at $(0,0)^{\text{T}}$ and UAVs are thus often disturbed from their original paths. Once UAVs are clear of the crossroads, they begin to realign their paths towards their destination and in so doing may become involved with secondary S&A manoeuvres with UAVs from the same stream. Finally, as UAVs converge on their destination ports we assume that they are removed from the simulation when they enter the ‘landing zone’ disk of radius R_{LZ} centred upon their destination.

We measure the performance of a given setup by comparing the flight time between takeoff and entering the landing zone, to the time $(L - R_{\text{LZ}})/v_{\text{CS}}$ that each UAV would have taken in the absence of interactions with any other UAVs. This results in each UAV having a delay T_i that it experiences as a result of S&A interactions. These delays can then be used to quantify the

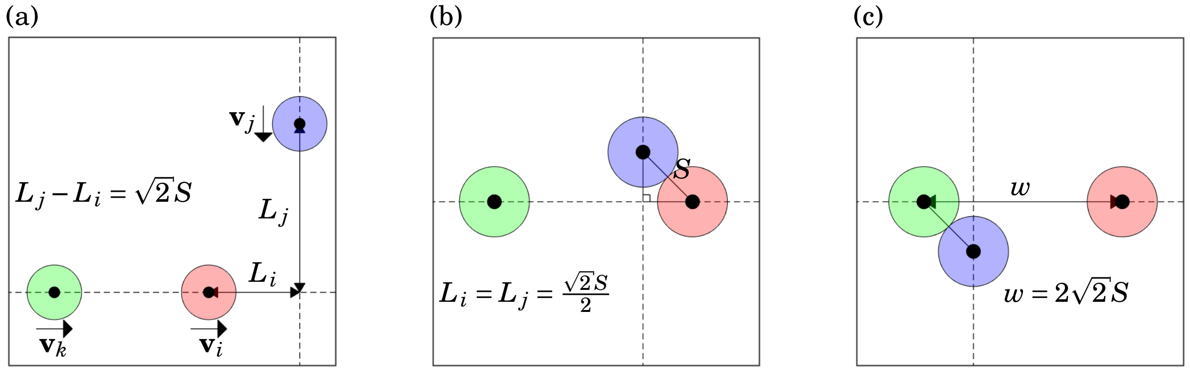


Figure 3.6: (a) Three UAVs approach a crossing at the cruising speed v_{CS}. UAV j is a distance $\sqrt{2}S$ further away from the crossing than UAV i. (b) Some time later both UAV i and j are the same distance $\sqrt{2}S/2$ away from the crossing at their closest point of approach such that the separation between them is S. (c) If the distance between UAV i and k is equal to $2\sqrt{2}S$ then UAV j can pass between them.

efficiency of a given setup by finding the mean delay $\bar{T}$, and the fairness, by finding the Gini coefficient G. The latter is a measure used in economics to describe wealth disparity in a given population. It is defined here as

$$G = \frac{\Sigma_i \Sigma_j |T_i - T_j|}{2n^2 \bar{T}}, \tag{3.5}$$

for n UAVs. If $G = 0$ then the UAVs all experience the same delay. As G increases so too does the level of disparity between delays experienced by UAVs.

In order to constrain the search space when exploring O-D demands, we consider what the maximum capacity of a crossing is in the absence of conflict avoidance. Consider the case where three UAVs approach a crossing with two UAVs in the horizontal stream and one UAV in the vertical stream, see Figure 3.6. Assuming that all three UAVs maintain a linear trajectory and that they each have the same avoidance separation S, what is the minimum width w between UAV j and k that allows i to pass between them?

Firstly, consider how far away from the crossing i needs to be in order to pass behind j so that their separation at the closest point of approach is equal to S. If i and j are both initially a distance L_i and L_j away from the crossing then

$$L_i = L_j + \sqrt{2}S, \tag{3.6}$$

satisfies our condition that i passes behind j. Note that i passes in front of j if the sign is flipped. By the same logic, UAV k must have the same relative displacement with respect to i in order to pass behind it. This implies that

$$w = 2\sqrt{2}S, \tag{3.7}$$

when two traffic streams are perpendicular to each other. Therefore, if two traffic streams had an infininte number of UAVs, uniformly separated, with width w between them, then

$$\lambda_{\text{cap}} = \frac{v_{\text{CS}}}{\sqrt{2}S}, \tag{3.8}$$

is the maximum rate at which UAVs can pass through the crossing. This capacity is split evenly between both streams and therefore

$$\lambda_{\max} = \frac{1}{2}\lambda_{\text{cap}}, \tag{3.9}$$

is a natural maximum demand for either origin port. Note that this satisfies the inequality in Equation 3.3 and therefore the departure queues will be stable.

In this section we will constrain the demand on both streams so that $\lambda_1 = \lambda_2 = m\lambda_{\max}/10$ and sweep through $m = 1,2,\dots,10$. These settings capture a range of setups from small demand values where UAV interactions are rare and delays are small, up to the notional maximum demand. Each simulation is run until 1,000 UAVs have taken off from each origin port, corresponding typically to 4,000–40,000 s of simulated time, of which the first 200 s of departures are discarded as simulation 'run-up' (where delays are anomalously short). We can then calculate the delay experienced by each of the remaining UAVs. In order to produce a robust estimate of the mean delay $\bar{T}$ experienced by these UAVs we use a simple bootstrapping method. Given a set of N delays we generate 1000 synthetic data sets by sampling from the original set, with replacement. The mean delay is calculated for each of these synthetic sets and the range of values provides us with a confidence interval for the mean of the original set.

We then conduct this experiment for two scenarios. In the first scenario all UAVs will adopt the simple avoidance behaviour, i.e., they will follow behaviour (A) when avoiding each other, as in Section 2.5. In the second scenario the UAVs will adopt the hybrid avoidance method presented in Section 2.6 where the avoidance behaviour will be chosen from behaviours (A), (B), or (C) based on the UAVs' positions and angle of approach. Given that the two streams are perpendicular to each other, we expected that there would be little difference between the two scenarios. This is because the hybrid avoidance method only deviates from behaviour (A) for small angles of approach. We will show here that this is not the case due to the impact of secondary avoidance interactions.

The mean delay experienced by UAVs increases with demand for both the simple and hybrid avoidance scenarios, see Figure 3.7(a). This is to be expected since the probability that two UAVs in different streams will need to avoid each other increases with demand. In fact, when $\lambda \ll \lambda_{\max}$ UAVs experience almost no delay on average as most UAVs transit the system without applying an avoidance manoeuvre. However, when $\lambda \geq 0.4\lambda_{\max}$ the hybrid avoidance scenario results in significantly shorter average delays compared with the simple avoidance method. A similar relationship is observed for the fairness of the system, see Figure 3.7(b). Again, both avoidance scenarios are about as fair as each other for small demand but the hybrid avoidance scenario

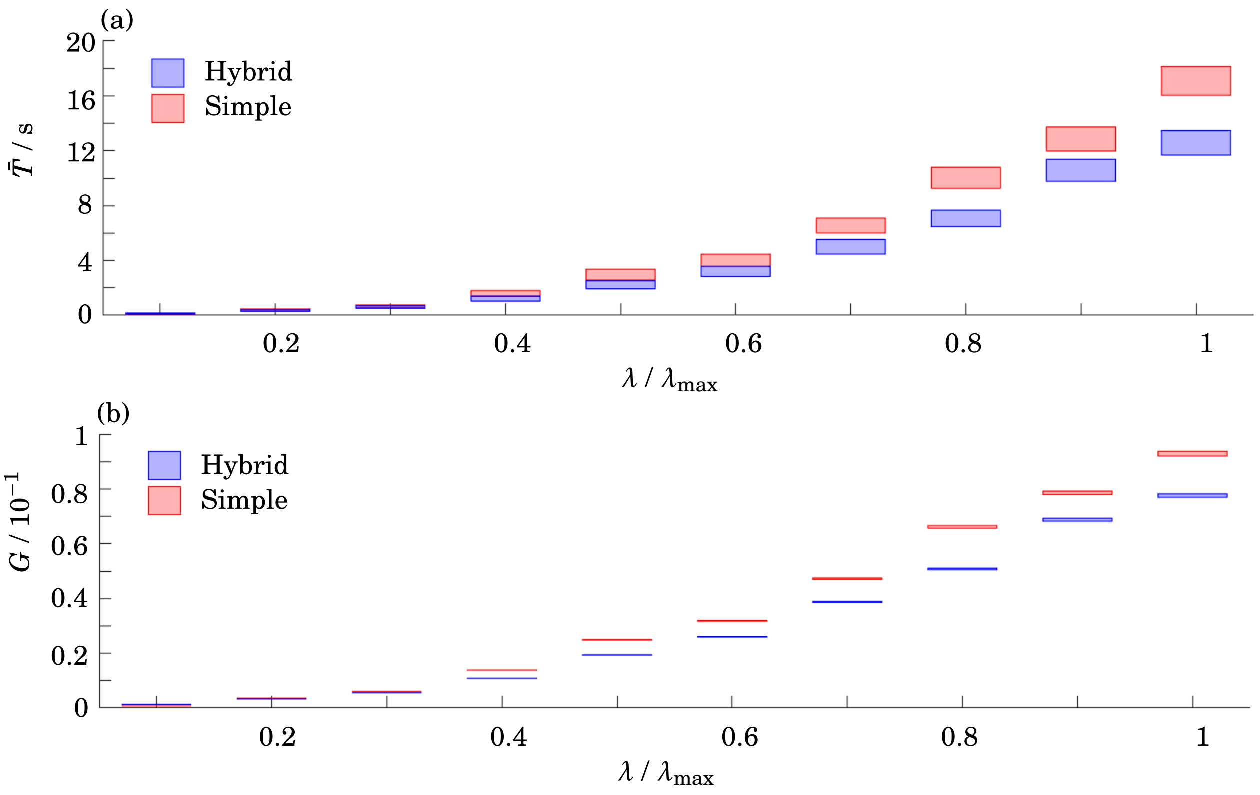


Figure 3.7: A comparison of the efficiency and fairness in terms of (a) mean delay $\bar{T}$ and (b) Gini coefficient G respectively, of a crossing where UAVs employ the simple (red) or hybrid (blue) avoidance methods from Chpater 2. For both $\bar{T}$ and G, the width of the bars is given by the bootstrapping process described above. For both metrics there is little difference between avoidance methods when $\lambda \leq 0.3\lambda_{\max}$. However both metrics are significantly improved when the hybrid avoidance method is used for heavy demand setups. We explain this in terms of the secondary interactions between UAVs of the same stream in the vicinity of their destination.

results in a significantly smaller value for G when $\lambda \geq 0.4\lambda_{\max}$. This implies that for setups with high demand the hybrid avoidance method is both more efficient and fairer than the simple method, similar to what we observed in Section 2.6.

This result can be explained by considering the secondary avoidance interactions a UAV experiences with UAVs from the same stream. These interactions occur when a UAV is displaced from its original flight path by an avoidance manoeuvre induced by a UAV from the other stream. After the initial avoidance manoeuvre is carried out, the UAV will realign itself with its destination through the OVM described by Equation 2.2. In doing so the UAV may find itself on a conflict course with another UAV from its stream. Since the UAVs are converging towards a point, the angle of approach between them is likely to be small and under behaviour (A) can lead to further extreme flight path deviations, see Figure 3.8(a). Given the same situation, if behaviour (B) or (C) is followed then the UAVs experience little to no deviation from their flight path, see Figure 3.8(b) and (c). In fact the UAVs will often land before the conflict course is resolved. The

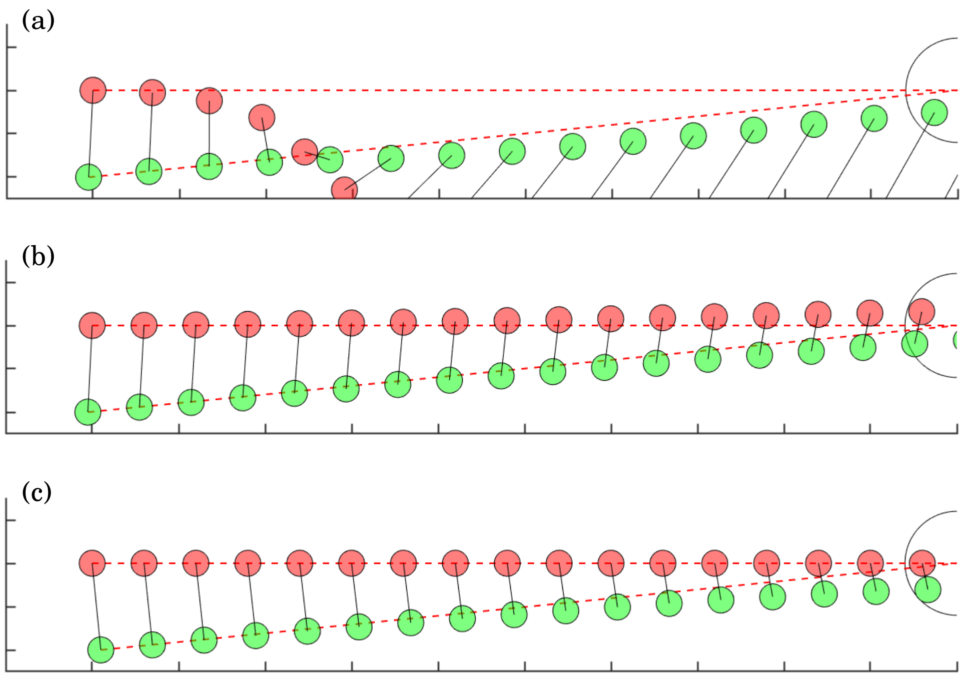


Figure 3.8: Trajectories of two UAVs converging on a destination port under (a) behaviour (A), (b) behaviour (B) and (c) behaviour (C). In (a) the red UAV experiences a large deviation as expected based on findings in Chapter 2. In (b) and (c) this behaviour is alleviated when the red UAV adopts behaviour (B) or (C).

frequency of these secondary interactions increases with demand as more UAVs are initially deviated from their original path. This observation would explain why the benefits of the hybrid avoidance method are amplified for higher demand setups.

However, some UAVs pass through the system without having to avoid another UAV from the same stream. We would expect the average delay experienced by these UAVs to be the same for both the simple and hybrid avoidance scenarios. We therefore produce bootstrapped estimates of delay for these UAVs in particular, see Figure 3.9(a). While the distributions do not fully overlap, particularly at extreme demands, there is much more cross over than is observed in Figure 3.7(a).

We can also consider the average delay per avoidance partner which can be thought of as the ‘interaction cost’ of each avoidance manoeuvre

$$\bar{I} = \frac{\Sigma T_i}{\Sigma N_i}, \tag{3.10}$$

where N_i is the number of other UAVs that UAV i had to avoid during its transit through the system. Since the secondary avoidance manoeuvres performed under the simple avoidance scenario can lead to much larger deviations, we would expect $\bar{I}$ to be reduced for the heuristic scenario. We therefore plot $\bar{I}$ for both the simple and hybrid avoidance scenarios in Figure 3.9(b). As expected the mean interaction cost is generally smaller for the hybrid avoidance scenario.

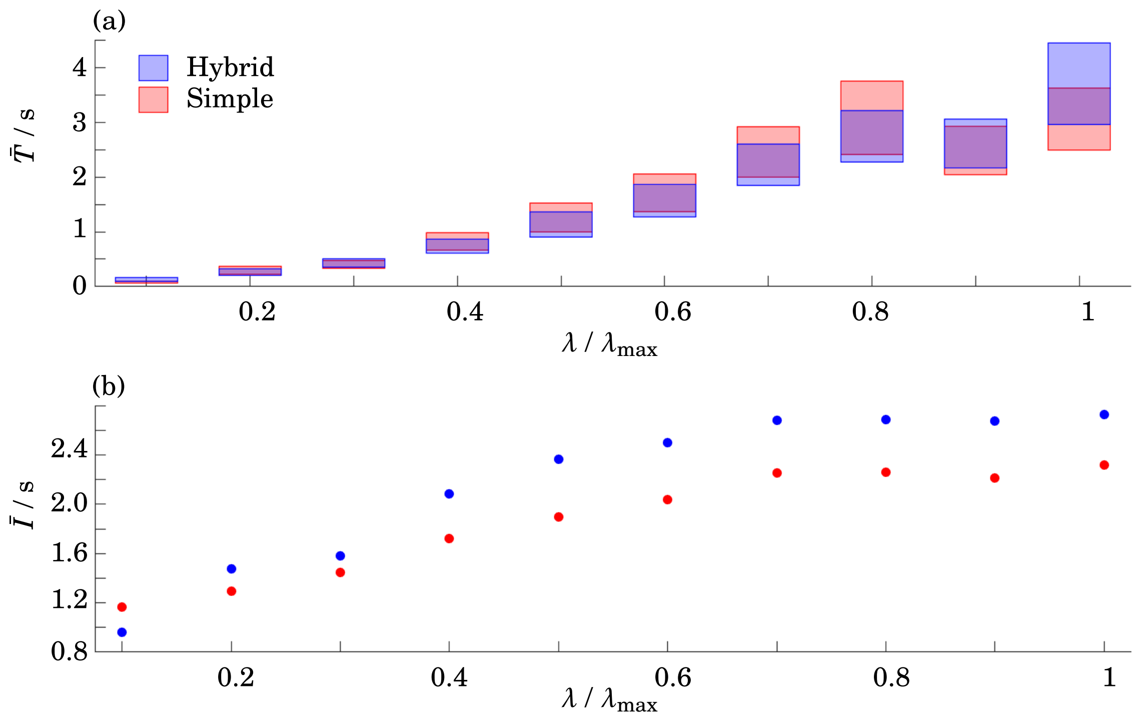


Figure 3.9: (a) The bootstrapped estimate of the average delay incurred by UAVs which only avoid UAVs from the other stream is roughly the same whether the hybrid or simple avoidance method is used. This is what we would expect if the reduction in delay observed in Figure 3.7(a) is due to the secondary interactions of UAVs from the same stream converging toward their destination. (b) The average interaction cost is also usually smaller for the hybrid avoidance, particularly at higher demands.

These results suggest that the hybrid avoidance tends to produce both more efficient and fairer outcomes for UAV traffic at a crossing compared with the simple avoidance. This is because under the hybrid avoidance scenario the interaction cost associated with having to avoid UAVs from the same stream is reduced. As such, the hybrid avoidance will be used throughout the rest of this thesis.

3.4 Impact of demand on UAV crossing performance

In Section 3.3 we introduced the crossroads experimental setup and showed that the hybrid avoidance method, see Section 2.6, produces fairer and more efficient performance compared with the simple avoidance behaviour, see Section 2.3. This effect is more pronounced for setups with higher total demand due to the higher frequency of secondary avoidance manoeuvres induced by UAVs in the same stream. However, in Section 3.3 the demand at each origin port was constrained to be the same. In this section we will explore setups where the demands λ_1 and λ_2 are varied independently over the same range as previously. We will also examine the fairness and efficiency

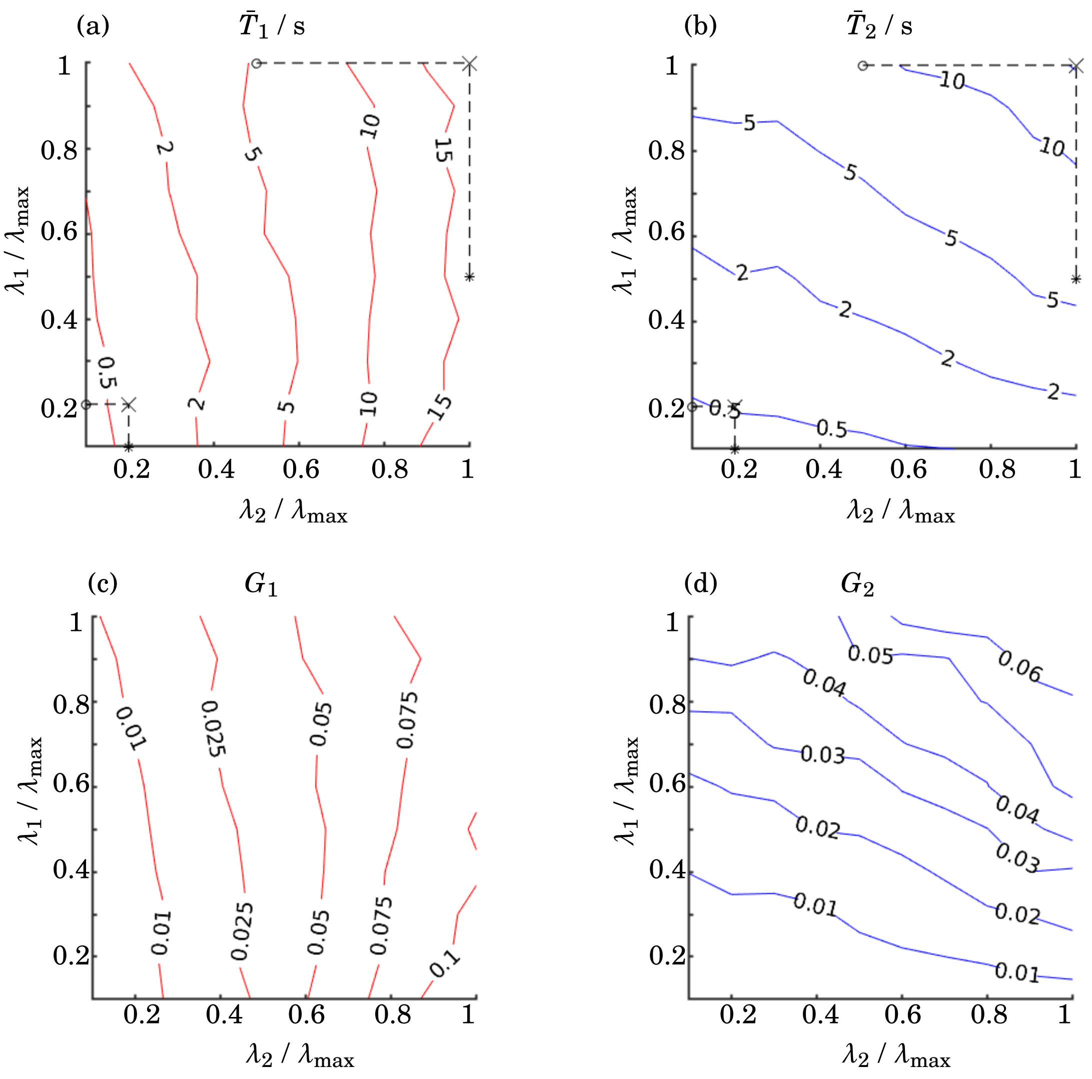


Figure 3.10: The mean delay experienced by UAVs in the (a) horizontal and (b) vertical streams. Note that for both streams there is a rapid, non-linear increase in delay as the total demand increases towards the maximum. Thus, halving the demand on one stream will more than halve the delay experienced by the other, shown by the dashed lines. We also calculate the Gini coefficient for the (c) horizontal and (d) vertical streams. This calculation implies that there is greater disparity in the delays experienced by UAVs in the horizontal stream. We believe this is due to the right-handedness of avoidance behaviour (A), see Figures 3.11 and 3.12.

experienced by the UAVs in each stream rather than just the whole system. The parameters and simulation methods are otherwise the same as in Section 3.3.

We produce contour plots showing how the mean delay experienced by UAVs in the horizontal, see Figure 3.10(a), and vertical, see Figure 3.10(b), streams, $\bar{T}_1$ and $\bar{T}_2$ respectively, depends on the demand experienced by both origin ports. In both cases there is a rapid, non-linear increase in delay as the total demand experienced by the system increases. In fact while the largest total demand is 10 times larger than the smallest, the mean delay increases by a factor of about 100.

The way in which the delay increases with demand is however quite different for both streams.

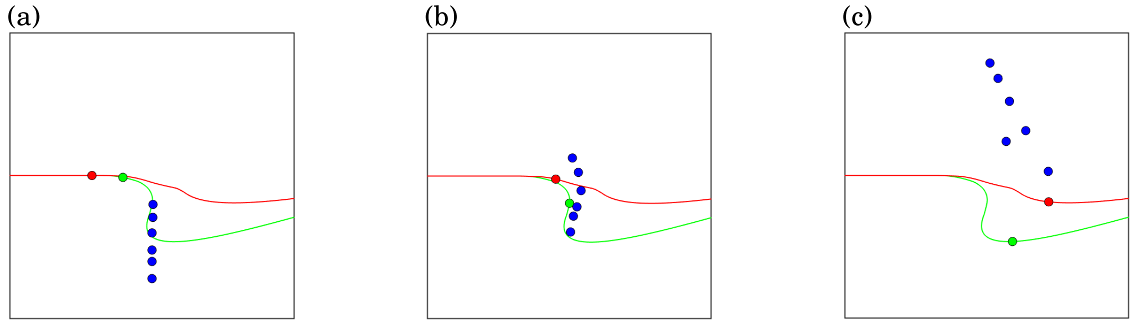


Figure 3.11: Snapshot of UAVs at a crossing which demonstrates why the delay $\bar{T}_1$ depends so heavily on demand λ_2 and why generally $G_1 > G_2$. (a) Several UAVs approach a crossing. (b) The green UAV, following behaviour (A), turns almost head on to the blue UAVs. (c) The green UAV consequently suffers a larger deviation than the red UAV and incurs a much longer delay. Conversely, the blue UAVs experience smaller and more similar deviations and delays on average. This behaviour would be reversed if the default avoidance behaviour was to turn to the left instead.

For the horizontal stream, $\bar{T}_1$ is almost independent of the demand λ_1. In other words, the delay experienced is dependent only on the demand experienced by the vertical stream. The reverse is not true. For the vertical stream, $\bar{T}_2$ depends on both λ_1 and λ_2. Note though that $\bar{T}_2$ tends to be larger when $\lambda_1 > \lambda_2$. That is, the delay is worse when the demand on the horizontal stream is higher than the vertical stream. Also note that the $\bar{T}_1$ is more likely to be higher than $\bar{T}_2$.

We also consider the fairness in terms of the coefficient G_1 and G_2 for the horizontal and vertical streams respectively, see Figure 3.10(c) and (d). By doing this we quantify the delay disparity within each stream. Both G_1 and G_2 experience similar relationships with respect to demand as their corresponding mean delay. That is, the fairness of a stream worsens along with delay. However, unlike delay, the fairness of the horizontal stream is consistently worse than the horizontal stream.

The greater level of delay disparity observed in the horizontal stream, and its strong dependence on the demand on the vertical stream, can be explained by the right-handedness of avoidance behaviour (A). Consider a toy example where 2 UAVs in the horizontal stream approach the crossing along with several UAVs in the vertical stream, see Figure 3.11(a). The first UAV in the horizontal stream (green) begins to turn to the right in order to avoid the first UAV in the vertical stream. In doing so it now has to avoid the next UAV in the vertical stream until it is eventually travelling almost head on to the vertical stream, see Figure 3.11(b). The green UAV therefore has to go around the entire group of blue UAVs, leading to a large lateral deviation compared with its ideal straight path, see Figure 3.11(c). The second UAV in the horizontal stream (red) however suffers a smaller lateral deviation due to the blue UAVs own small lateral deviations as a result of avoiding the green UAV.

This behaviour is worsened when the demand on the vertical stream is increased. To demon-

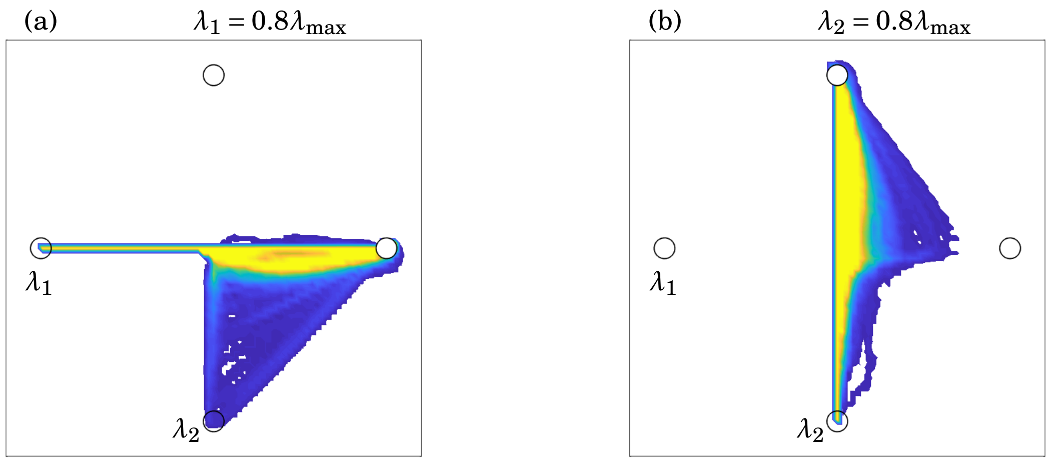


Figure 3.12: Density plots for UAVs in the (a) horizontal and (b) vertical streams, when $\lambda_1 = \lambda_2 = 0.8\lambda_{\max}$. The bright yellow corresponds to $> 10\%$ of UAVs. Most deviations experienced by UAVs in both streams are similar in magnitude. However, UAVs in the horizontal stream can experience much longer deviations in the worst case scenario where they are diverted around the bottom-most port. The UAVs that experience these extreme deviations are rare and thus lead to the greater level of disparity observed for the horizontal stream in Figure 3.10(c).

strate this we present density plots for both streams when $\lambda_1 = \lambda_2 = 0.8\lambda_{\max}$, see Figure 3.12(a) and (b). In both figures the brightest yellow corresponds to visits by more than 10% of UAVs. A small proportion of UAVs in the horizontal stream are sufficiently deviated as to pass around the origin port at $(0, -L/2)^{\mathrm{T}}$. Such drastic deviations are never observed for the vertical stream. However, the vertical stream does experience some lateral deviation before the crossing, unlike the horizontal stream. This is again due to the right handedness.

The right handedness of avoidance behaviour (A) then can lead to UAVs in the horizontal stream experiencing much longer deviations and consequently longer delays. As with delay and fairness we can explore how the lateral deviation experienced by UAVs depends on the demand. The maximum lateral deviation for a given UAV is defined as the maximum orthogonal distance from its ideal linear path. This can then be averaged over all UAVs in the horizontal and vertical stream to produce $\bar{D}_1$ and $\bar{D}_2$ respectively. We can then consider the relative size of the mean lateral deviations for both streams, see Figure 3.13(a). This roughly corresponds to the stream with the highest demand experiencing smaller lateral deviations on average. However, if we consider the worst 10% of lateral deviations, $\bar{D}_1^*$ and $\bar{D}_2^*$, then we see that the horizontal stream is more likely to experience worse or similar lateral deviations except when $\lambda_1 \gg \lambda_2$, see Figure 3.13(b).

In summary, these results show that for small traffic demands the hybrid conflict avoidance method can facilitate a fair and efficient UAV traffic crossing. However, we also show that as demand increases toward the theoretical capacity of the crossing the performance degrades rapidly, suggesting that other traffic management methods will be required. One such method is

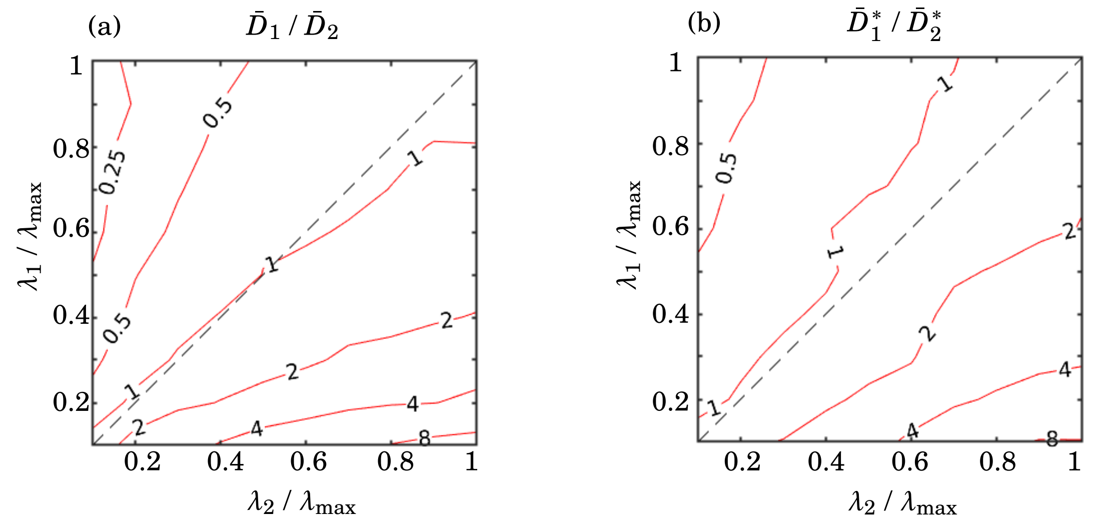


Figure 3.13: Comparison of the mean lateral deviation experienced by (a) all UAVs in both streams and (b) the UAVs with the worst 10% of lateral deviations. In general, the UAVs in a stream that experiences heavier demand than the other will experience smaller lateral deviations. However, the largest deviations experienced by the UAVs in the horizontal stream tend to be worse than the largest deviations experienced by UAVs in the horizontal stream. This is particularly pronounced as λ_2 tends toward $\lambda_{\max}$, i.e., when the vertical stream is under heavy demand and UAVs in the horizontal stream are forced to go around the vertical stream.

splitting a stream into two parallel streams and share the demand evenly between them.

3.5 Effect of splitting demand across multiple streams

In Section 3.4 we explored how the efficiency and fairness of a UAV crossing is affected by the demand on either stream. In doing so we identified how the right handedness of the avoidance behaviour (A) can result in worse performance for the horizontal traffic stream and showed how performance for both streams deteriorates rapidly as the total demand approaches the maximum capacity of the crossing. In this section then we will explore how splitting a traffic stream into two parallel streams might be used to improve performance. Crucially, while this strategy will increase the number of crossings in the system the contention at each is reduced.

We can see from Figure 3.10(a) and (b) that halving the demand on one of the streams can more than halve the delay experienced by the other. For example, $\bar{T}_1 \approx 15\,\text{s}$ when $\lambda_1 = \lambda_2 = \lambda_{\max}$ but when λ_2 is halved then $\bar{T}_1 \approx 5\,\text{s}$, a reduction of around two thirds. One possible interpretation, grounded in queuing theory, is for each traffic stream to view the crossroads as a server whose capacity is restricted by the flow of the other stream. Little's law [104] for Markovian M/M/1 queues gives a delay (response time) of $\lambda/\mu(\mu - \lambda)$ where λ is the arrival rate and μ is the service rate. It can be shown that doubling the number of servers can *more* than halve the wait time in such queues. A similar effect has also been shown in wireless communications networks [105] where increasing the number of routes by introducing intermediary relays was shown to increase the capacity of the network. Therefore we will explore whether this kind of effect can be exploited here by modifying the crossing setup presented in Section 3.3.

Here we will take either the vertical or horizontal traffic stream and split it into two parallel streams, see Figure 3.14(a) and (b) respectively. This is done by replacing the original O-D pair with two O-D pairs that are displaced by $(\pm\Delta L/2, 0)^{\mathrm{T}}$ or $(0, \pm\Delta L/2)^{\mathrm{T}}$ depending on whether the vertical or horizontal stream is being split. Thus the two resulting parallel streams are a distance ΔL appart. We suggest $\Delta L = 600\,\mathrm{m}$ as a reasonable dimensional value for simulation for two reasons. First, the parallel streams are only displaced by $300\,\mathrm{m}$ which is small compared with $L = 2\,\mathrm{km}$ and therefore should limit the impact on performance as a result of moving the crossing. Second, this value for ΔL is much larger than most of the lateral deviations experienced by UAVs in either stream. Therefore this should ensure that interactions between UAVs in one parallel stream with UAVs from the other parallel stream are rare.

For a particular instance of this kind of setup the unsplit stream experiences a demand λ and the two parallel streams experience demand $\lambda/2$. In Figure 3.14(a) and (b) $\lambda = 0.8\lambda_{\max}$. The result is a system with three streams of traffic and two crossings. A key modelling idea is to explain the total delays that result in terms of the delays incurred at each individual crossing. For example, when the vertical stream is split we assume that both crossings might each be modelled by the demand combination $(\lambda_1, \lambda_2/2)$ from the single crossing setup used in Section 3.4. The two vertical streams experience just one of these crossings each. We would therefore expect the UAVs in both streams to incur the same delay on average as the single crossing setup with the same delay profile. The unsplit horizontal stream however experiences both crossings and so we suggest that its average delay should be double the single crossing case.

In order to test the assumption that a system with multiple crossings can be thought of as a simple combination of single crossings, we adopt one-dimensional sweeps $\lambda_1 = \lambda_2 = m\lambda_{\max}/5$, $m = 1, 2, \ldots, 5$ and split the vertical and horizontal streams in turn. We follow the same experimental procedure as before to generate bootstrapped estimates for the mean delay for the horizontal and vertical streams. For this metric we consider all the UAVs in the two parallel streams together. When the vertical stream is split, see Figure 3.15(a), we can see then that the model does seem to provide a good predictor of the resulting delays $\bar{T}_1$ and $\bar{T}_2$. However, when the horizontal stream is split the model greatly over-estimates the mean delay at higher demand values, see Figure 3.15(b). Note as well that, despite the greater than expected reduction in delay observed when the horizontal stream is split, that the delay experienced by all streams tends to be smaller when the vertical stream is split.

In order to understand why there is a significant reduction in the delay experienced by the two parallel streams when the horizontal stream is split, compared with the model, we consider what the delay experienced by each of them is separately. For the $\lambda = 0.8\lambda_{\max}$ case we display the full bootstrapped distributions of the mean delay experienced by the first (red) and the second

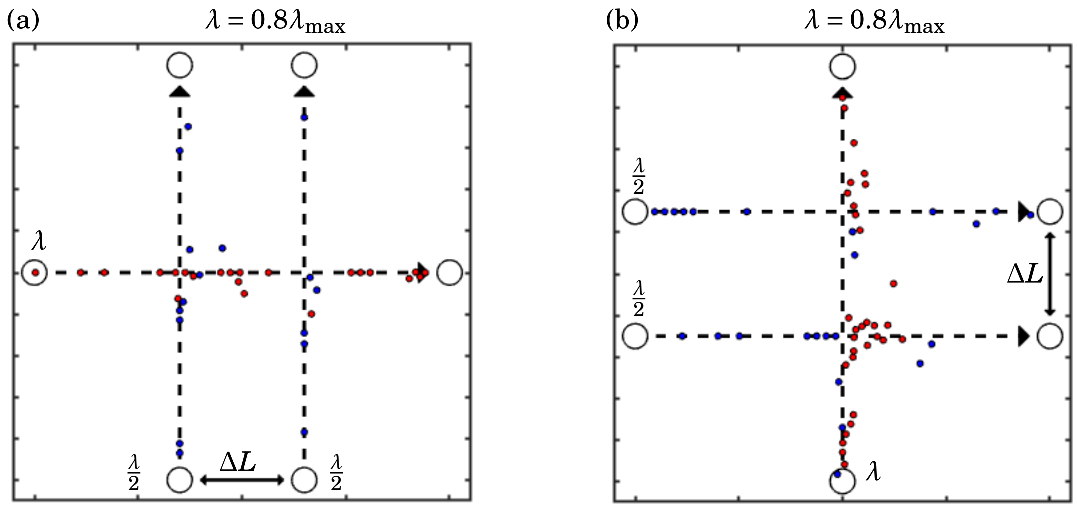


Figure 3.14: Test setup when the (a) vertical and (b) horizontal streams are split. When a stream is split we replace the original O-D pair with two O-D pairs which are displaced by a distance $\Delta L/2$ laterally from the original position. The demand on the two resulting parallel streams is then shared equally. We set ΔL large enough that interactions between UAVs in the parallel streams are rare.

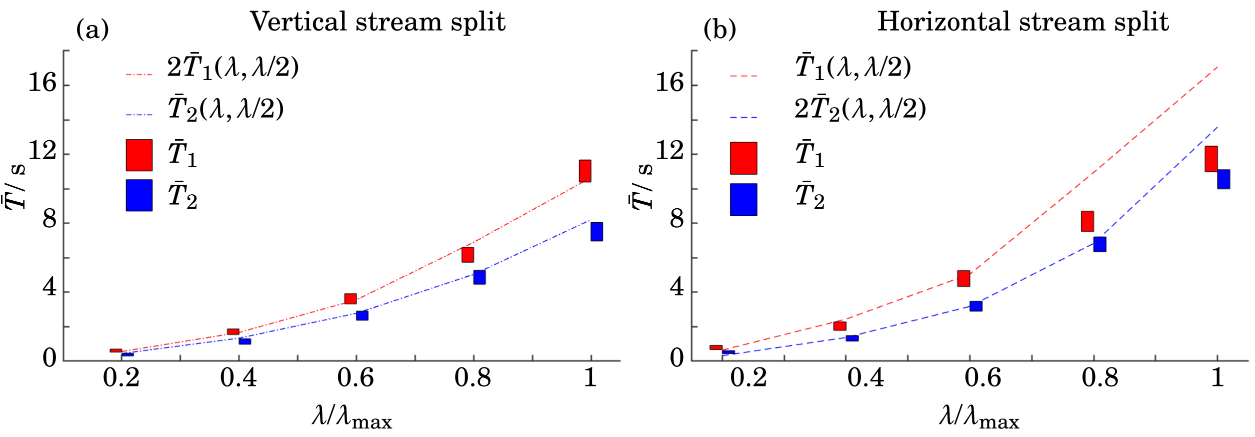


Figure 3.15: Comparison of the bootstrapped estimate of the mean delay from simulation with that predicted by the single crossing case when the (a) vertical and (b) horizontal streams are split. When the vertical stream is split the simple model of independent crossings provides a good estimate for delay. However, when the horizontal stream is split the model begins to significantly overestimate the delay when the demand is high. This can be understood by examining the delay of the two parallel streams separately, see Figure 3.16, and the traffic 'smear' effect observed in Figure 3.17.

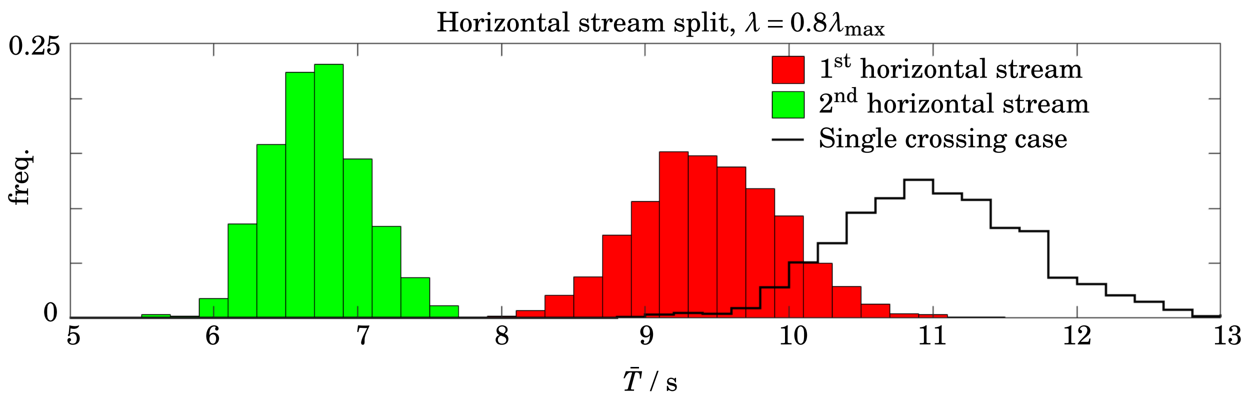


Figure 3.16: Bootstrapped estimate of the mean delay experienced by UAVs in the first (red) and second (green) horizontal streams, relative to the flow of the un-split vertical stream, when $\lambda = 0.8\lambda_{\max}$. If the two crossings are independent of each other, as we assumed, we would expect the two distributions to be the same. Here, the UAVs in the second horizontal stream experience significantly shorter delays on average compared with the UAVs in the first horizontal stream which suggests they experience fewer avoidance interactions, see Figure 3.17(a). Note as well that the UAVs in the first horizontal stream experience slightly shorter delays compared with the single crossing case. This is likely due to the reduced path length experienced by UAVs that undergo extreme lateral deviations as the first crossing is closer to the vertical stream's origin, see Figure 3.17(b).

(green) parallel streams, as encountered by the unsplit vertical stream, see Figure 3.16. The mean delay experienced by UAVs in the second parallel stream is significantly shorter than that observed for the first parallel stream whereas we expected them to be almost identical. Also, the delay experienced by the first parallel stream tends to be shorter than the single crossing case. Both of these behaviours can be understood by considering density plots for the different streams.

For the unsplit vertical stream, and the same demand value as in Figure 3.16, we see that traffic is deviated laterally before and in the vicinity of the first crossing, see Figure 3.17(a), similar to what is observed for the single crossing scenario. The result of this is that the UAVs are spread out along the horizontal axis: see the cluster of red UAVs near the first crossing in Figure 3.14(b). Due to the implementation of the waypoint following method, see Section 2.2, the deviated UAVs continue on almost linear paths as their destination is far away. Therefore this traffic 'smearing' effect is maintained up until the second crossing, effectively reducing the demand further and resulting in fewer avoidance interactions.

This effective reduction in the demand at the second crossing can be observed in the density plot for the two horizontal streams, see Figure 3.17(b). The lateral deviations experienced by the second horizontal stream are smaller and therefore incur smaller delays. Consider the scenario in Figure 3.11: if the blue UAVs were instead flying parallel to each other, instead of in series, then the large deviation experienced by the green UAV would have been averted. Also note that some of the UAVs in the first horizontal stream are deviated such that they go around the bottom port, compare with Figure 3.12(a). This explains the reduction in delay experienced by the first

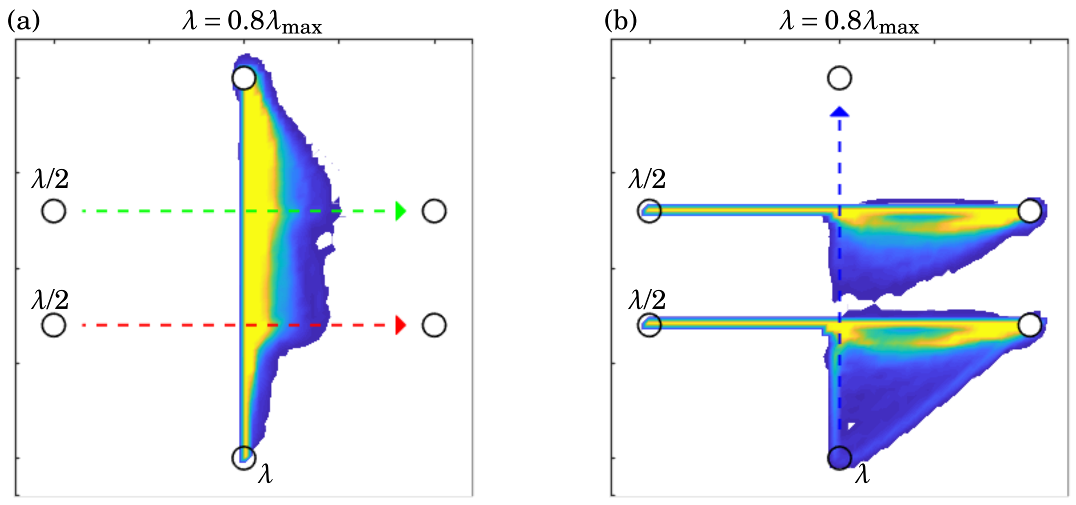


Figure 3.17: Density plots for UAVs in the (a) horizontal and (b) vertical streams, when $\lambda = 0.8\lambda_{\max}$ and the horizontal stream is split. The bright yellow corresponds to $> 10\%$ of UAVs. The vertical stream is spread out to the right around the first crossing as before. Due to the OVM used by the UAVs this lateral 'smear' effect is maintained as the UAVs approach the second crossing, thus the effective demand at the second crossing is reduced. This results in the UAVs in the second horizontal stream incurring shorter lateral deviations and thus shorter delays compared with the first horizontal stream. This effect could be exploited when designing an O-D network for UAVs.

horizontal stream, compared with the single crossing scenario, since the first crossing is now closer to the origin port and the worst lateral deviations are therefore shorter.

3.6 Conclusion

In this chapter we aimed to explore how the conflict avoidance methods developed in Chapter 2 performed for scenarios with many more UAVs than just a pair. Inspired by the P2P delivery use case, we developed a simple UAV port model to provide O-D pairs and thus designed a traffic scenario where two streams of UAV traffic cross at a point, forming a sort of crossroads around which the UAVs will need to avoid each other. We then evaluated the performance of this setup in terms of the efficiency and fairness, as per **RQ2** on page 14, by considering the time it takes a UAV to travel between its origin and destination compared with some ideal linear flight path.

For this crossing setup, in unrestricted airspace, we observe several features that emerge as a result of the OVM and conflict avoidance behaviours implemented by the UAVs. For example, when UAVs converge on their destination this can lead to secondary avoidance interactions with other UAVs from the same stream. Since the resulting conflict course is likely to have a small angle of approach, the hybrid avoidance method performs better than the simple avoidance method. Another emergent behaviour is the way in which the horizontal stream depends on the vertical stream. This is due to the right-handedness of avoidance behaviour (A) which introduces an asymmetry to the scenario. Despite all UAVs applying the same avoidance method, the horizontal stream effectively gives way to the vertical stream. This suggests that future work could focus on improving the performance of the horizontal stream in particular. Finally, we showed that for a system with multiple crossings, it is possible to effectively reduce the demand at a downstream crossing due to a smearing effect. This smearing effect is a result of the way UAVs are displaced by their avoidance behaviour and the way in which the OVM directs them to their destination. We could in fact eliminate this effect if we adopted a scheme in which UAVs attempt to return to their original flight path, which may be desirable in more restricted airspace. However, in an unrestricted airspace the smearing effect could be exploited when developing O-D networks to help improve network performance.

We also show how, due to the non-linear way in which the delay increases with respect to the demand for a single crossing, we can improve system performance by splitting a stream in to two parallel streams. Here we do this by effectively replacing the original O-D pair with two O-D pairs that results in two parallel streams of UAVs, each of which experiences half the demand of the original stream. This can lead to an average delay of about a third of the single crossing case for the stream that is split. In the highest demand setting used this corresponds to a reduction of about 10 s. While this may be small, especially compared to the minimum travel time of about 100 s, it is worth noting that in a real delivery operation a UAV may fly through many more of these crossings on missions with linear flight paths up to 5 times longer. This may lead to many small delays that become significant overall. Also, the method with which we split streams in this chapter can be thought of as an ideal case. Since the O-D nodes represent static infrastructure, a UTM system would not be able to replace nodes except on a much longer, strategic time scale.

For all the setups explored in this chapter we also show that the delays incurred by UAVs are negligible when the demand is small. This implies that under light loading, a UTM system may be able to operate using only a decentralised S&A scheme. However the performance of such a scheme will deteriorate as demand increases. The final chapter of this thesis then will focus on the development of traffic management techniques that can be applied on top of the tactical conflict management in order to improve the performance of UAV traffic crossings.

3.6.1 Contributions

C3.1 We presented a simple model for UAV ports and then used the ports to describe an experimental setup where two streams of UAV traffic form a crossing.

C3.2 We used this experimental setup to show that the hybrid avoidance method outperforms the simple avoidance method, despite the large angle of approach between streams, due to secondary avoidance interactions.

C3.3 We explored the effect of traffic demand on the efficiency and fairness when only the hybrid conflict avoidance method is used to manage the crossing.

C3.4 We showed that the right handedness of avoidance behaviour (A) can cause extreme deviations in the flight path of UAVs in the horizontal stream.

C3.5 We showed how system performance can be improved by splitting a traffic stream into two parallel streams with the demand shared equally between them.

C3.6 We showed that, due to a 'smearing effect', the effective demand at a second crossing can be lowered by avoidance interactions at a previous crossing. This may then be exploited when designing O-D networks for UAV delivery.

CHAPTER 4

PORT-LEVEL DESIGN SOLUTIONS FOR SUPPLEMENTAL TRAFFIC MANAGEMENT

In Chapter 3 we developed an experimental setup which was inspired by the P2P goods delivery use case for UAVs. This allowed us to explore how the tactical conflict management developed in Chapter 2 performs when there are many UAVs sharing an unrestricted airspace. Using this setup we showed that both in terms of efficiency and fairness, the tactical conflict management performed well when demand was low but this performance degraded rapidly as demand increased. However we also showed that we can improve performance at high demand by changing the layout of the setup in order to exploit the avoidance behaviour of individual UAVs. This chapter then will attempt to answer **RQ3** as defined on page 15 by developing several higher level traffic management methods that can be applied to our experimental setup in order to improve performance. The methods developed in this chapter are meant to show how a variety of approaches can be taken to traffic management. As such they provide a starting point for future work and should be considered more as 'proof of concept' methods which could be further developed and improved.

Based on the principles of simplicity and scalability, we suggest that UTM should be designed in such a way that it can easily be applied to all UAVs in the system, or large subsets of them. Therefore, the traffic management methods we will develop in this chapter will be applied at the port level, meaning that each port will be responsible for applying the appropriate traffic management method to the UAVs as they depart from it. This management could be as simple as altering the values of the internal parameters of the UAVs, e.g., reducing the cruising speed v_{CS}, or increasing the safe separation distance S. A port could change the way in which UAVs take off, e.g., increasing the size of the clear airspace radius S_{TO}, or forcing the UAV to wait to depart for some other criteria to be met. A port could even place virtual objects in to the airspace for the

UAVs to interact with, e.g., waypoints, or obstacles.

Note, it is critical that we assume whatever traffic management method is in use, it must be applied prior to a UAV's takeoff. This assumption ensures that, at the level of the UAVs, the traffic system still operates in a distributed mode as discussed at the start of Section 3.2. In other words, the UAVs themselves are not involved in the higher level management. In this way we will design methods that aim to change the traffic system in a structural way rather than directly control individual UAVs, and thus maintain scalability.

One of the key design principles for the traffic management methods designed in this chapter is to allow the UAVs to employ the same tactical conflict management as before. In this way, these higher level traffic management methods can be thought of as supplemental to, and not dependent on, any particular low-level S&A scheme employed by a UTM system, thus providing a neat modular design. Therefore the methods developed in this chapter may be used even if our dynamical model of multi-rotor UAVs and tactical conflict management, both developed in Chapter 2, are updated or generalised in future work. We will also aim to design traffic management methods that can be applied dynamically. In other words, they might be changed by each port in response to system-level metrics, such as demand. This is because we assume that the ports will have a backbone network that can be used to share information, such as the number of departed and landed UAVs, which can then be used to obtain said metrics. The information could be exchanged either directly with each other or indirectly depending on whether a 'centralised' or 'federated' architecture is used for the UTM system, as discussed on page 9. Indeed, if we assume the ports have a high-speed and high-quality communications infrastructure then ports could also potentially co-ordinate their control actions, over fine time-scales. However, this dynamic and co-operative port level traffic management is not explored in this chapter but may provide a rich area for future work.

In Section 4.1 we will put this chapter in to context by considering other traffic management methods, such as route assignment for ground and air vehicles, and floor fields used in pedestrian modelling, that will inform those designed later in the chapter. The first traffic management method is developed in Section 4.2. This method seeks to exploit the benefits we observed when splitting demand between multiple, streams but here we split streams by introducing waypoint-defined routes. Under this method a UAV is assigned a route which consists of intermediary waypoints that it should visit before reaching its destination. The second method is developed in Section 4.3. This method introduces the concept of a floor field zone which makes use of a velocity floor field to alter the motion of UAVs within the zone. Effectively, while a UAV is under the influence of a floor field, it will change the velocity it uses in the OVM in order to align itself with the floor field. The third and final method is developed in Section 4.4. In this method we force UAVs to wait until a certain number of UAVs are ready to depart, effectively producing small platoons of UAVs. Finally, we provide a discussion of these results and an overview of this chapter's contributions in Section 4.5.

4.1 Background

In Chapter 3 we have shown that tactical conflict management is capable of enabling a fair and efficient UAV crossing, in terms of the delays incurred by the UAVs, for low to medium demand setups. However, when demand increases both the efficiency and fairness of the crossing worsens as UAVs experience longer delays on average and one of the traffic streams is more adversely affected than the other. We therefore need to develop methods that can improve the experience of UAVs that pass through the crossing under heavy traffic conditions.

One approach to this could be in further developing the tactical conflict management. For example, we could explore variations of the way in which we combine acceleration components in Equation 2.4 or design other acceleration components based on factors like traffic density, e.g., where UAVs move away from large groups. In swarm robotics, the performance of a swarm has been shown to improve through the inclusion of "leader" agents [106]. These leader agents can be statically [107] or dynamically [108] chosen and then directly controlled in some way by a human operator in order to influence the emergent behaviour. A similar approach could be designed to work here but faces two main problems. First, there would need to be a good quality communication link between the human operators and the UAVs they directly control. This may be difficult due to the issues discussed in Section 1.1 (see page 5) about the communication capabilities of UAVs. Secondly, we do not know what percentage of UAVs would need to be directly controlled in order to effectively improve performance. Given the potential for large numbers of UAVs sharing the airspace, even if only a small percentage of UAVs need to become leaders this could represent a significant cost in terms of human personnel.

Instead we will design methods that can be applied by the UTM system in addition to the tactical conflict management rather than modify it. This approach is inspired by Mohamed-Salleh *et al.* [94] who suggests that as the density of traffic in an airspace increases, so too should the level of structure. In other words, they suggest that at low densities UAVs should be allowed to fly along direct paths where possible but when the density increases beyond some level, the UAVs transition to a more structured traffic layout, e.g., organised into layers according to heading. The idea then is to maintain the simplicity of the pairwise UAV interactions but change the traffic setup at a higher level. This approach to traffic management also highlights the fact that the UAV airspace is likely to be a dynamic one. Consider our primary use case of P2P goods delivery, as described in Section 3.1. The demand experienced by the network of ports may change throughout the day, experiencing peaks and troughs similar to those observed in other transportation networks [109]. Mohamed-Salleh *et al.* [94] further suggests that UTM should "enable transition from free flight to a structured one". For simplicity we will continue to use static demands for the simulations later in this chapter but, with this capability in mind, we will develop traffic management methods that can easily be altered in response to heuristics that describe the current state of the traffic network such as demand levels, or average flight time. Ideally the traffic management methods can be applied before the traffic enters an undesirable state, e.g., 'gridlock', and then prevent those states from arising. We also suggest that this dynamic airspace, combined with setup costs, would make traffic management that relies on static infrastructure infeasible.

We will now provide a brief summary of the literature which has inspired the three traffic management methods developed in this chapter. The first method we develop, in Section 4.2, is based on route assignment. As discussed in Section 3.1, the UAVs will default to taking the shortest path between their origin and destination ports on the basis that this produces the shortest time in flight for each individual UAV in an unrestricted airspace. We achieve this in practice by using the position of a UAV's destination in Equation 2.1 which sets the desired heading for the OVM, defined in Equation 2.2, used by our dynamical model of a UAV. The result of this behaviour for the crossing setup described in Section 3.3 is that all UAVs pass through the same point which makes avoidance manoeuvres both more likely and extreme. A similar effect is observed for road traffic where congestion is worsened when traffic density increases. One solution to this problem is to divide the road network into multiple regions and subregions. The density of these regions can then be observed and if it rises above a certain threshold then we take some control action. A popular control action in the academic literature is perimeter control which restricts the flow of traffic into certain regions [110]. In a particular implementation of perimeter control [111] the flows of traffic between regions and subregions are controlled in order to redistribute the traffic away from problem regions. This is done by assigning paths to individual vehicles according to a hierarchical route guidance scheme.

Mao *et al.* [112] consider a similar crossroads-like setup to ours, but for traditional aircraft. A control region is defined around the point where two streams of aircraft cross. When an aircraft enters this control region it adopts a change of heading that ensures it does not conflict with any other aircraft that has already entered the control region. This spreads the incoming stream of aircraft out so they no longer all pass through the same point. In fact, when the streams of aircraft are uniformly spaced they form 'chevron' patterns such that the stream splits into multiple streams that are perfectly phased to allow aircraft to pass each other, see Figure 4.1.

The problem with the route assignment in both [111] and [112] is that it applies updates to the routes mid-mission. Instead, we will develop a method by which a port can define a number of routes between it and another port which UAVs can then be assigned to upon takeoff. In this thesis we will consider a small number of routes per port that are defined on a longer timescale than a typical UAV mission. This limits complexity and avoids the scenario where a port designs a bespoke route for each UAV. We will then design routes that aim to alleviate the problems associated with a single crossing and in effect imitate the behaviour observed in Figure 4.1. However, in our system the traffic stream is split at the UAV's origin and then we allow the hybrid conflict avoidance to resolve any conflicts that arise mid-mission.

In the next method we develop, in Section 4.3, we instead attempt to control traffic by changing the environment in which the UAVs operate. We do this by designating areas where the UAVs will

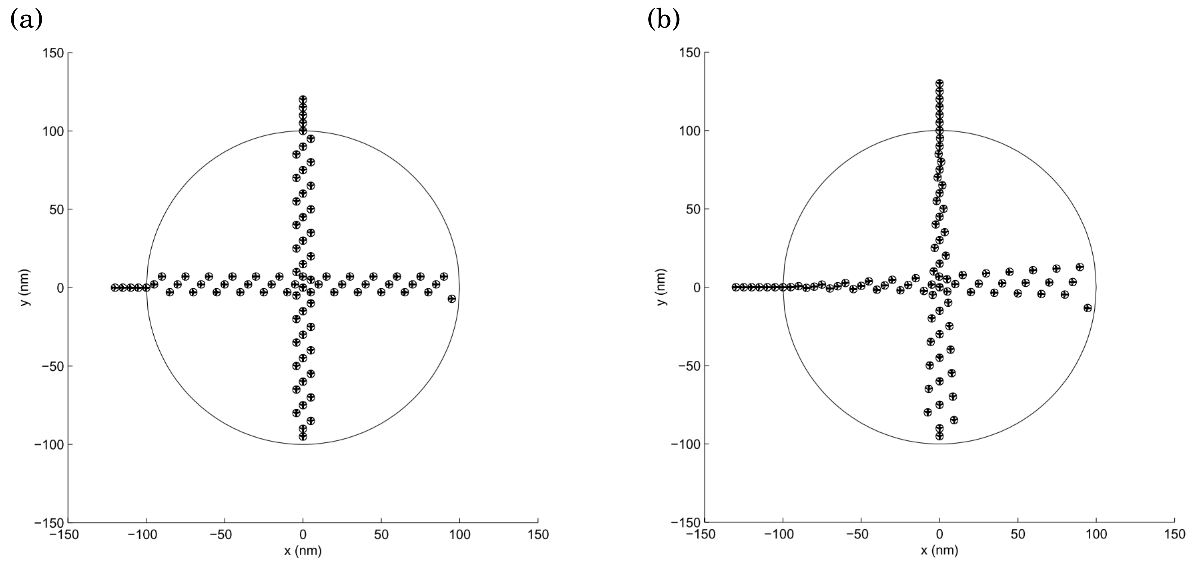


Figure 4.1: (a) When an aircraft enters the controlled area it must change its heading to ensure that it does not conflict with any other aircraft that entered the controlled area before it. This change of heading is modelled simply by displacing the aircraft laterally with respect to its original flight path. When both streams have a constant arrival rate, as they do here, the traffic is organised in to several lanes that form chevron patterns that allow the aircraft to pass each other. (b) The change of heading is modelled more realistically but the same lane formation and chevron patterns are obtained. Unlike (a) and (b) we will explore the possibility of forming UAVs into lanes at their origin, in Section 4.2, and allow the hybrid conflict avoidance developed in Section 2.6 to resolve mid-air conflicts. (a) and (b) are reproduced from Mao *et al.*, 2000 [112].

have some structure imposed on them which in turn will change their collective behaviour. To do this we again borrow from pedestrian modelling where the concept of a *floor field* is defined. The floor field is introduced by Burstedde *et al.* [113] as an extension of the cellular automaton model of pedestrians. In [113] pedestrians are modelled on a grid where each pedestrian has a preferred direction of motion which dictates the transition probability to each of its neighbouring cells. The floor field is used to introduce long range effects by modifying the transition probabilities. The authors also distinguish between a static and dynamic floor field. A static floor field can be used, for example, to designate areas of the grid as more attractive, e.g., evacuation points. A dynamic floor field can be modified by the actions of the pedestrians themselves which is an example of stigmergy, commonly associated with insects that lay pheromone trails [114]. In Section 4.3 we will use a static velocity floor field to impose some new collective behaviour by changing the desired heading used by the OVM in certain areas of the airspace. These floor fields will be digital and can therefore in principle be updated by the UTM system in a dynamic airspace as needed.

There are many other examples of where a change in the environment can induce a change in pedestrian or traffic behaviour. A counter-intuitive effect observed in pedestrian modelling by Helbing *et al.* [115] is how a temporary increase in the width of a corridor can result in worse

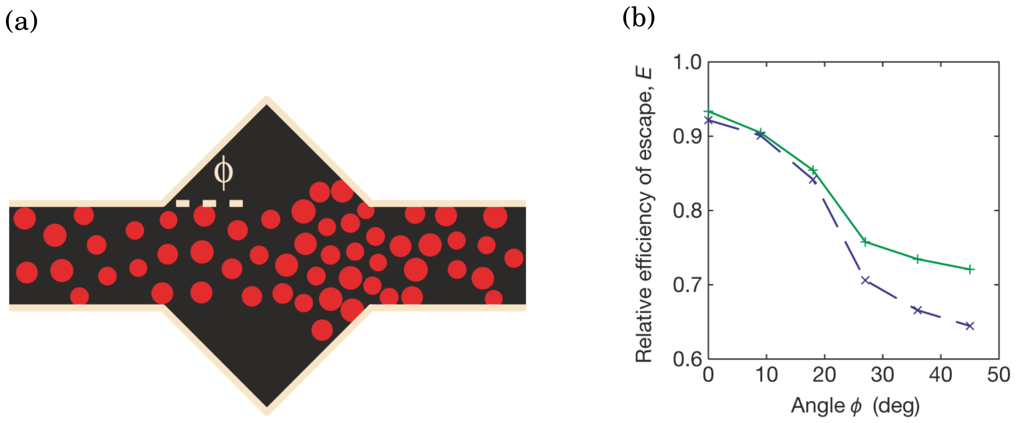


Figure 4.2: (a) Pedestrians (red circles) travel along a corridor which widens and then narrows. This change in width is characterised by an angle ϕ where $\phi = 0$ corresponds to a corridor of constant width. Note how the density of the pedestrians increases around the point where the corridor begins to narrow, effectively blocking the passage. (b) Results from simulation. As the angle ϕ is increased, the efficiency of escape decreases. This result is not necessarily intuitive since we might assume that increasing the width of the corridor would increase the capacity of the system, allowing for greater flows of pedestrians. (a) and (b) are reproduced from Helbing *et al.*, 2000 [115].

performance of an evacuation, see Figure 4.2. This behaviour can be considered loosely analogous to Braess' Paradox [116] which hypothesises that adding an edge to a traffic network can result in increased travel times. Therefore in both [115] and [116] an apparent increase in system capacity leads to worse system performance, though the underlying mechanisms are different. Fortunately we can invert this behaviour, as argued by Hughes [117]. For example we can improve evacuation of pedestrians by placing obstacles in front of their exit [118] or by walling off sections of a corridor and changing its shape, e.g., making it funnel shaped and constricting towards the exit [119]. In [118] and [119] the capacity of the system has been reduced but flow of pedestrians has improved. These insights can then be used to design effective spaces for pedestrians, though ideally we would be able to change the space in response to stimuli such as density or changes in pedestrian behaviour. This can be difficult to achieve when the interventions are physical, e.g., bollards. One related idea then, that we will not explore further in this thesis, is to use 'virtual' UAVs to form static obstacles in an airspace. This would have the advantage that UAVs could implement one of the conflict avoidance methods from Chapter 2 and so no new dynamics would need to be developed. We could then design different obstacles by placing many of these virtual UAVs along an outline.

The final method we develop, in Section 4.4, is inspired by vehicle platooning. It is well known that birds flying in formation expend less energy on average [120]. This idea has since been extended to road vehicles, in particular heavy goods vehicles (HGVs). A platoon of HGVs, that is a group of HGVs where a small inter-vehicle separation is maintained through collaborative control, has been shown to increase fuel efficiency by reducing the drag associated with air resistance [121]. This has led to further research questions such as how to plan vehicle routes so that platoons form en-route and therefore must balance issues such as time spent waiting at nodes for other vehicles to arrive versus the fuel efficiency gains [122]. While we do not model details such as air resistance, we will show that forming UAVs into platoons can reduce the delays incurred by UAVs passing through a crossing by exploiting large inter-platoon gaps.

4.2 Traffic splitting with waypoints

In Section 3.5 two parallel streams of traffic were produced by splitting a single O-D pair in two and sharing the demand between them. In this section we will explore the possibility of splitting traffic from a single origin through the implementation of intermediary waypoints. In doing so we aim to replicate the reduction in delay observed for a setup with multiple crossings in Section 3.5 when compared with a single crossing of the same total demand. To do this we use the waypoints to define multiple routes between the same O-D pair and assign UAVs to these routes at takeoff. Since these do not usually correspond to a straight line between the O-D pair the UAVs will incur a longer flight time than the ideal straight line paths provided by the parallel streams used in Section 3.5. However, we suggest that this method of routing UAVs with waypoints is worth exploring since moving and / or adding ports to the system is infeasible, at least over a short time period. In other words, while the results from Section 3.5 may be useful when designing a traffic network, it can not be easily implemented as a dynamic traffic management method.

4.2.1 Waypoint-defined routes

Up to this point simulations have been set up such that the nominal path traffic will take, in the absence of interactions with other UAVs, will be a straight line. This is one of the supposed advantages of UAVs over road-bound vehicles. We now introduce the concept of *waypoint-defined routes* in order to relax this assumption. Consider the example port layout from Figure 3.4(b) which results in a system where there are five potential crossings. We can use waypoints to define indirect routes between an O-D pair, see Figure 4.3(a). This can produce a system which contains many more crossings. However, those crossings that are formed by at least one stream that has been split between multiple routes will experience a smaller demand than in the un-split case. Based on the results from Section 3.5, these crossings will hopefully incur a smaller delay in those UAVs that transit through them. One important question then is whether any reduction in delay from this effect can outweigh the increased flight time associated with the increased length of the nominal flight path. Note also that these waypoint-defined routes can be used to bypass obstacles, a capability that may be vital in a busy urban airspace.

The routes themselves consist of a sequence of waypoints ending with the UAVs' final destination. Upon take off the positions of the waypoints for the UAV's selected route are provided by

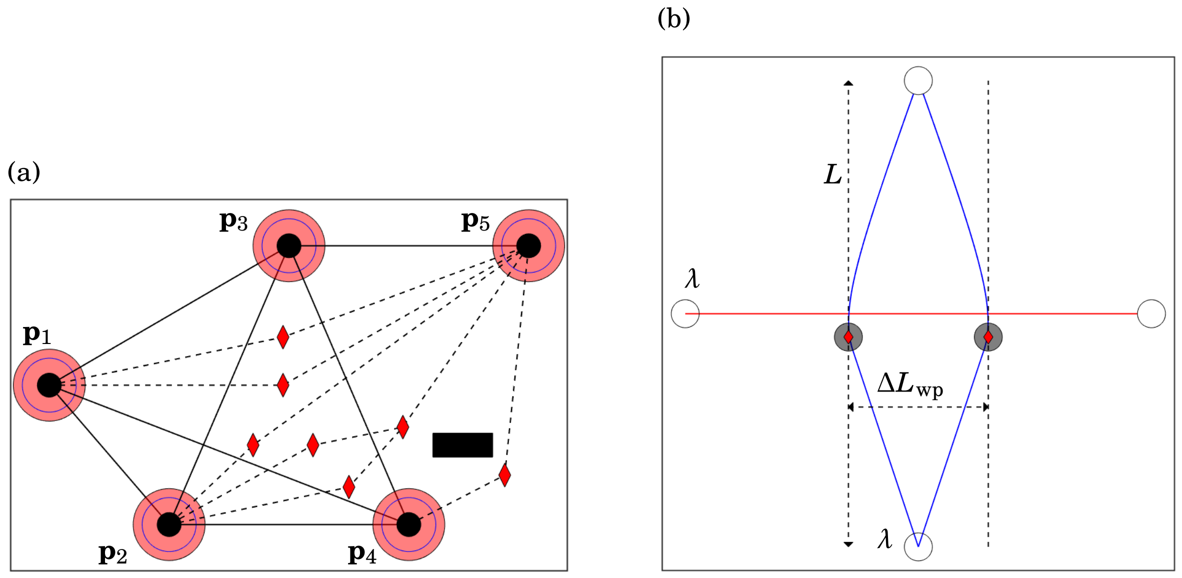


Figure 4.3: (a) The same port layout as in Figure 3.4(b). However, here we use waypoints (red diamonds) to define a variety of routes that UAVs can take between certain ports instead of straight lines. We can use these routes to go around obstacles (black rectangle) or to share demand between O-D pairs over multiple routes. The latter gives rise to a system with more crossings than in Figure 3.4(b). However, some of these crossings will have a lower contention and therefore should incur smaller delays in the UAVs that transit through them. (b) We emulate the setup in Figure 3.14(a) using two waypoint-defined routes. We use two waypoints (red diamonds) to produce two routes for the vertical stream and show the nominal path that a UAV will take (blue lines) if assigned to either route. This results in a system with two crossings (where the blue and red lines cross) which are a distance ΔL_{wp} apart. UAVs change their desired heading $\hat{\mathbf{t}}_i$ to align with their destination once they enter the grey circle around their waypoint. We have set the waypoints such that the angle of approach between streams at a crossing is still $\pi/2$ as is the case in Figure 3.14(a). As ΔL_{wp} increases, so too does the length of the nominal path of each route. The severity of this effect will depend on the size of L, i.e., how far apart the two ports are.

the port, the first of which is then used to set the desired heading $\hat{\mathbf{t}}_i$ described by Equation 2.1 which is used in the OVM described by Equation 2.2. While in flight a UAV will attempt to visit each of the waypoints in order and update its current waypoint when certain conditions are met, see Section 4.2.2. For simplicity we will also assume that the routes are static for a given experimental setup since we will also use constant values for the demand.

Using waypoint-defined routes we will aim to produce experimental setups that are similar to those used in Section 3.5, in particular the setup seen in Figure 3.14(a), but with only two O-D pairs each a distance L apart. We therefore split the vertical traffic stream by assigning UAVs who depart from the bottom-most port to one of two waypoint-defined routes, see Figure 4.3(b). Note that the routes have been designed so that the UAVs in the two vertical streams cross the horizontal stream at a right angle as is the case in Figure 3.14(a). For the work presented here we will assign routes to UAVs via a deterministic method. Taking the setup in Figure 4.3(b), the first UAV in the vertical traffic stream will be assigned the leftmost route, as viewed on the page, and then the routes are alternated for each subsequent UAV. Another simple method to assign routes would be to do so at random however this can lead to one route experiencing a higher demand over short time windows. While a route could consist of any number of waypoints, here we will only use two. The first waypoint will be off-centre to induce the traffic to split and the second will be the destination.

We accept that the way we have described waypoint-defined routes here is only one possible implementation. We can easily conceive of similar setups, for example where more routes are used or traffic is deliberately biased toward a particular route. However, fully exploring these options would likely constitute a significant body of work. We therefore reiterate that the aim of this chapter is to present a variety of traffic management methods which can then be further developed in later work.

4.2.2 Methods for updating a UAV's waypoint

As discussed above, the UAVs will attempt to visit each of their assigned waypoints in order. Once some criteria is met the UAV will update its current waypoint to the next one in the sequence and thus generate a new desired heading $\hat{\mathbf{t}}_i$. The first condition to cause a UAV to adopt its next waypoint is that it has visited its current waypoint. To determine whether a UAV has visited a waypoint we first define a 'visit radius' R_v, common to all waypoints. This is then compared with the UAV's current distance to its waypoint and, if that distance is less than R_v, the UAV updates its waypoint. We set R_v to be twice the safe separation distance S, the same as the landing radius discussed in Section 3.3. This ensures that R_v is small compared with the distance L between the O-D pair, which is desirable for intuitive route design. For example, if $R_\text{v} \approx L$ then the waypoints would need to be placed outside of the square formed by the four ports in order to achieve the same flight paths in Figure 4.3(b). Also, by setting $R_\text{v} > 0$ we make it less likely that UAVs who undergo conflict avoidance near their waypoint need to double back on themselves.

It is important to note here that the waypoints themselves do not represent anything crucial to the UAV's mission, their only purpose is to induce the traffic splitting effect. As such we will outline here a different condition which, if met, allows a UAV to update its waypoint without having visited the current waypoint. Consider a UAV i with position $\mathbf{r}_i$ which is currently travelling toward its n^th waypoint, see Figure 4.4(a). Let the positions of the current waypoint and the next waypoint be $\mathbf{r}_n$ and $\mathbf{r}_{n+1}$ respectively. We allow a UAV to adopt its next waypoint if

$$|\mathbf{r}_i - \mathbf{r}_{n+1}| \leq |\mathbf{r}_n - \mathbf{r}_{n+1}|, \tag{4.1}$$

that is if a UAV is ever closer to its next waypoint than the two waypoints are to each other. The benefit of this scheme is demonstrated in Figure 4.4(b) which shows the flight paths of two particular UAVs from a high demand simulation where UAVs may only update their waypoint after they have visited their current waypoint. Both UAVs are disturbed from their nominal path

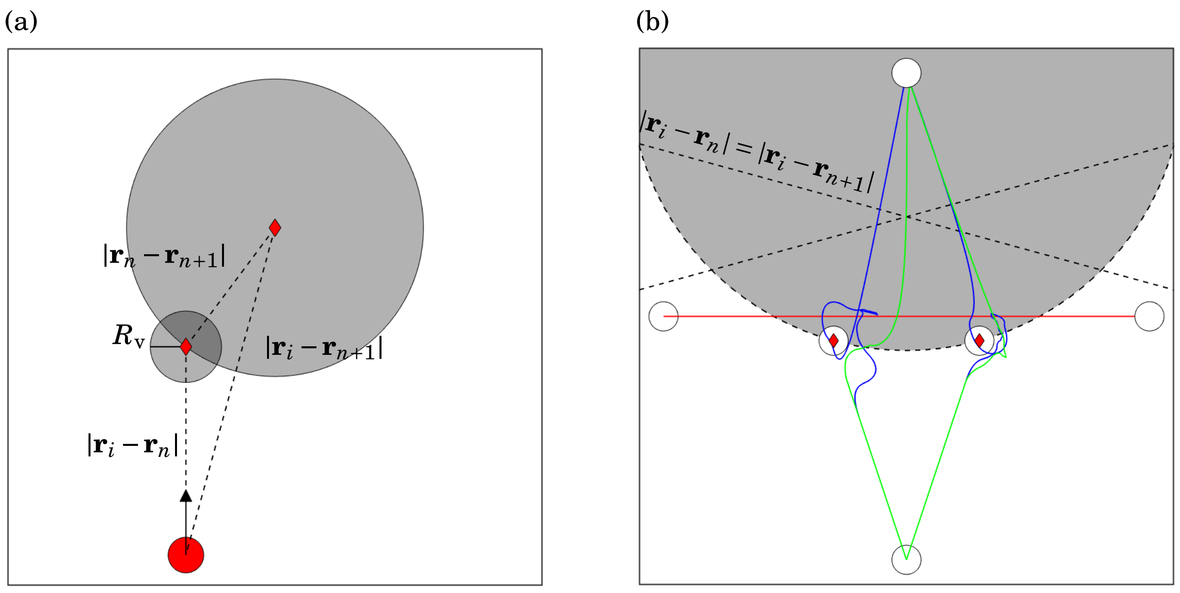

(a)
$|\mathbf{r}_n - \mathbf{r}_{n+1}|$
R_v
$|\mathbf{r}_i - \mathbf{r}_{n+1}|$
$|\mathbf{r}_i - \mathbf{r}_n|$
(b)
$|\mathbf{r}_i - \mathbf{r}_n| = |\mathbf{r}_i - \mathbf{r}_{n+1}|$

Figure 4.4: (a) A UAV i (red circle) is traveling toward its n^{th} waypoint. If the UAV enters either of the shaded circles, defined by radii of lengths R_v and $|\mathbf{r}_n - \mathbf{r}_{n+1}|$, then it will adopt its next waypoint. The benefit of the alternative update condition is demonstrated in (b). The flight paths of two UAVs are plotted for a high demand simulation where UAVs only update their waypoint after they have visited their current one (blue lines). The UAVs spend a long time in the vicinity of the horizontal stream as they undertake circuitous paths to reach their waypoint. This would not happen if they updated their waypoint when inside the grey circle defined by the radius $|\mathbf{r}_n - \mathbf{r}_{n+1}|$. This is demonstrated by the flight paths of two UAVs that do use the alternative update method (green lines). Also, the straight dashed lines show the regions where a UAV is closer to the first waypoint than the destination. This is clearly not a useful method for updating the UAV's waypoint as the UAVs are not deviated enough, though it could be effective if the waypoints were closer together.

due to avoidance manoeuvres and as such must double back on themselves in order to visit their current waypoint: in the case of the left flight path this means crossing the horizontal stream three times. In both cases, this undesirable behaviour would have been avoided if the alternative update rule had been in effect which is also demonstrated in Figure 4.4(b).

There are of course other possible waypoint update schemes, for example comparing the distance between a UAV and both its current and next waypoint. In Figure 4.4(b) the area above the dashed lines show where a UAV would update its waypoint under the condition that

$$|\mathbf{r}_i - \mathbf{r}_n| \geq |\mathbf{r}_i - \mathbf{r}_{n+1}|, \tag{4.2}$$

i.e., a UAV is closer to its next waypoint than its current waypoint. The effectiveness of each scheme will be dependent on the geometry of the traffic layout. The condition in Equation 4.2 would not have been satisfied before the UAVs in Figure 4.4(b) visited their current waypoint. The condition given in Equation 4.1 is effective because its border is near the two waypoints and thus it will likely catch those UAVs who are close to crossing the horizontal stream but are diverted due to conflict avoidance. Again, it is more important that the UAVs reach their final destination than visit the intermediary waypoint and allowing UAVs to congregate around the horizontal stream will likely lead to further degradation of performance.

4.2.3 Experiments with routes that induce a symmetric split

We will now take the waypoint-defined routing method and begin to explore its impact on system performance. We will begin by exploring similar simulation setups to that shown in Figure 4.3(b) where the vertical traffic stream is split in two, forming a left and right route as viewed on the page. The horizontal displacement between the two waypoints ΔL_{wp}, and thus the separation between the crossings, will be swept over along with traffic demand. We will use the same $\lambda_{\max}$ as that defined in Equation 3.9. The maximum value for ΔL_{wp} will be the same as the displacement between the two parallel streams used in Section 3.5, i.e., 600 m. As in previous sections each simulation will last until at least 1000 UAVs have departed from all origins and we discard the flight times of those UAVs in the simulation run up. Since we aim to demonstrate traffic management methods that will improve system performance in terms of delay we will define the *relative mean delay* as

$$\Delta\bar{T} = \bar{T} - \bar{T}_{\mathrm{sc}}, \tag{4.3}$$

which is the difference between the mean delay $\bar{T}$ experienced by UAVs in the simulations here and the mean delay $\bar{T}_{\mathrm{sc}}$ experienced by UAVs when there is only a single crossing with the same total demand, i.e. the delays shown in Figure 3.10(a) and (b). Therefore if $\Delta\bar{T} < 0$ then the delay incurred by UAVs is shorter on average while using the waypoint-defined routes. We will use this relative mean delay throughout this chapter. We also use the same bootstrapping method outlined in Section 3.3 to provide an estimate of the relative mean delay $\bar{T}$.

In Figure 4.5 we compare the relative mean delay for a low ($0.3\lambda_{\max}$), medium ($0.5\lambda_{\max}$), and high ($0.8\lambda_{\max}$) demand case and vary ΔL_{wp} between 6 m and 600 m. For horizontal traffic, see Figure 4.5(a) and (b), when the demand is low the mean delay experienced by UAVs is nearly the same as in the single crossing case. This is in keeping with the results from Section 3.5 and can be explained by the fact that at low demand UAV interactions are rare even when there is a single crossing. However, as the demand is increased the relative mean delay is decreased as the UAVs begin to experience shorter delays thanks to the effect of splitting the demand between multiple crossings. When the demand is high and ΔL_{wp} is small, see Figure 4.5(a), the relative mean delay is at its most extreme positive value. Under these conditions, splitting the vertical traffic stream between two narrowly separated routes can worsen system performance by increasing the average delay compared with a single crossing. This effect will be explored in more depth later in this subsection.

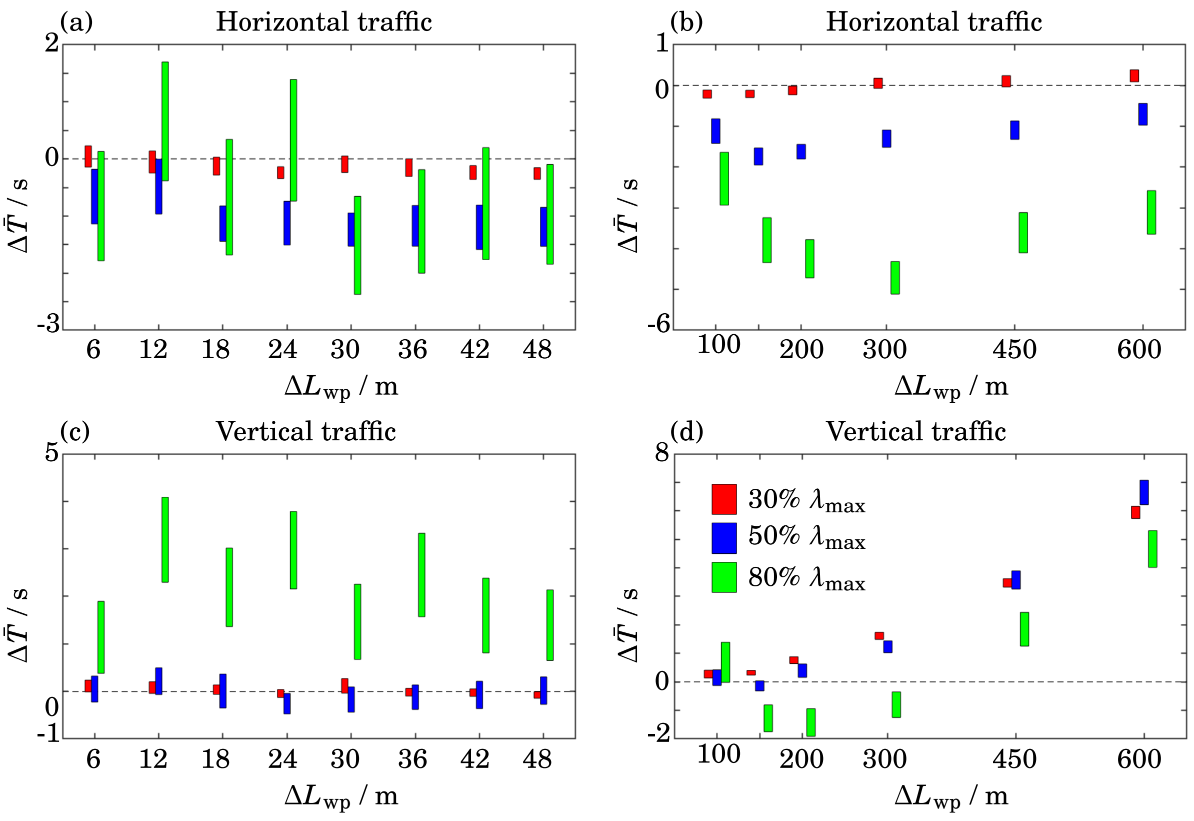


Figure 4.5: Estimated relative mean delay, Equation 4.3, experienced by UAVs in the horizontal stream for (a) small and (b) large values of ΔL_{wp} for three different demand levels; low (red), medium (blue) and high (green). When $150\,\mathrm{m} \leq \Delta L_{\mathrm{wp}} \leq 450\,\mathrm{m}$ and demand is high we observe the largest drop in relative mean delay. We also show the estimated relative mean delay for UAVs in the vertical streams (treated as one stream) for (c) small and (d) large values of ΔL_{wp}. As expected, when ΔL_{wp} becomes too large any benefit for the vertical streams is lost due to increased nominal flight times. However, we did not expect to find the relative mean delay increase for small values of ΔL_{wp}. This result is explained in Figure 4.6.

We will consider the relative mean delay for all the UAVs in the vertical stream, i.e., we combine the delays of UAVs in both the left and right route in to one set when carrying out the bootstrap process. The vertical traffic stream, see Figure 4.5(c) and (d), experiences similar relative mean delays for low and medium demand cases. That is, when ΔL_{wp} is small the relative mean delay is near zero but as ΔL_{wp} increases so too does the relative mean delay. This is as a result of the increased nominal flight time caused by the longer flight paths. These results imply that when the traffic demand is around 50% of $\lambda_{\max}$, system performance can be improved by introducing a split on the order of S to $2S$. This is wide enough to reduce the mean delay experienced by horizontal traffic without causing significant increases to the mean delay experienced by vertical traffic.

When the demand becomes large the experience of UAVs in the vertical streams change. The

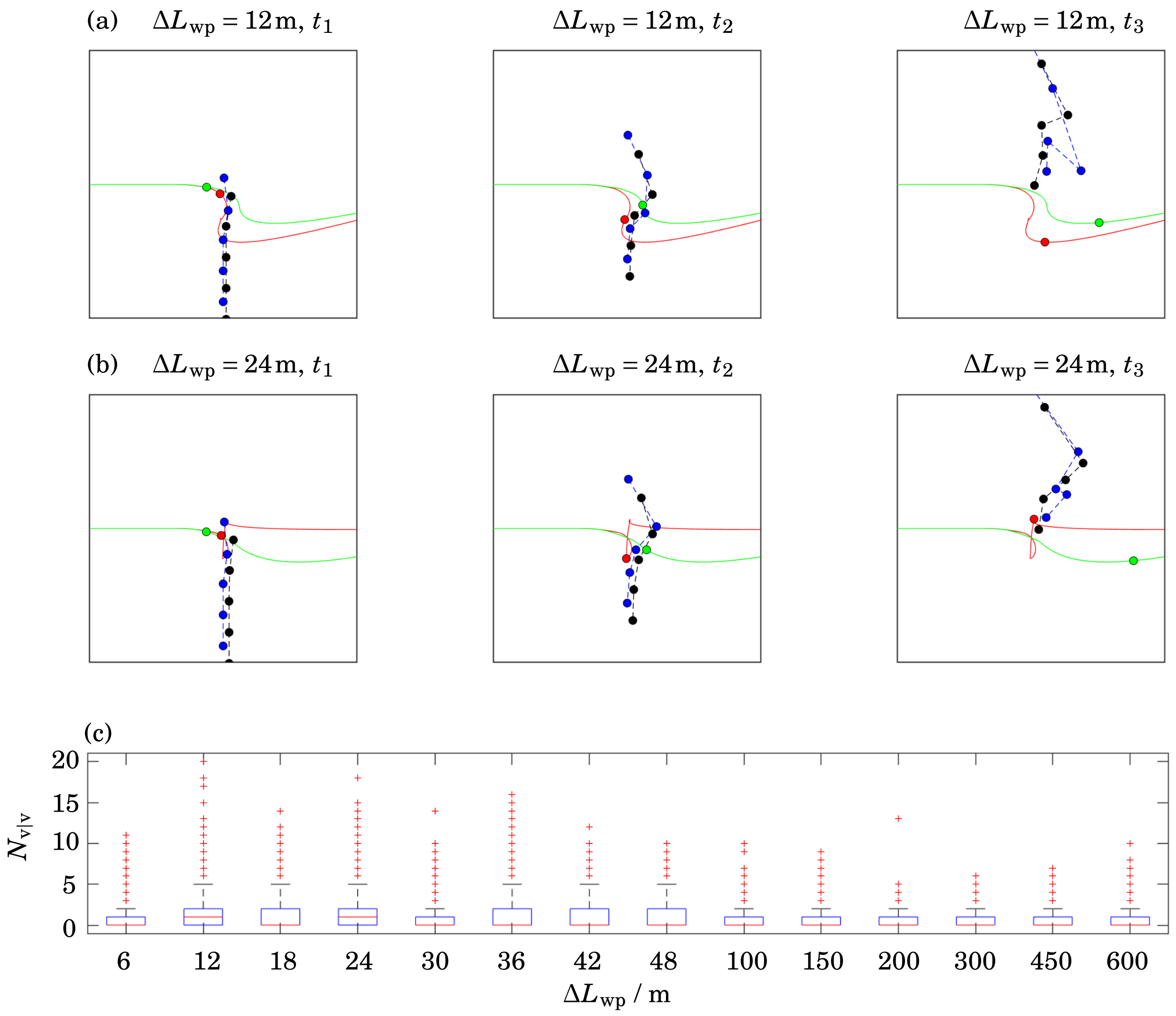
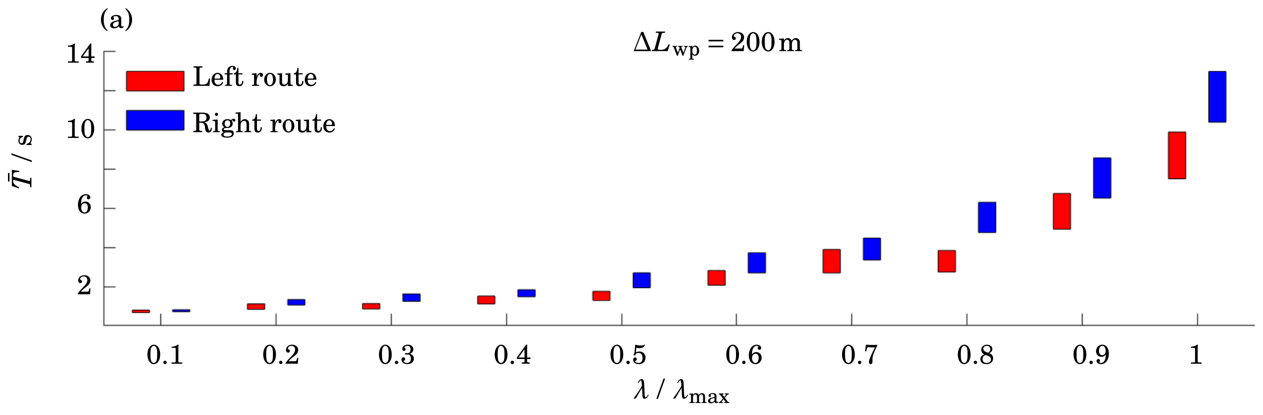


Figure 4.6: Snapshots from simulation at the same three points in time when (a) $\Delta L_{\rm wp} = 12\,{\rm m}$ and (b) $\Delta L_{\rm wp} = 24\,{\rm m}$. In both cases the UAVs in the left route (blue) are deviated to the right by their interaction with the horizontal stream UAVs (red and green) which causes the UAVs in the right route (black) to also engage in avoidance manoeuvres which then propagate down stream. These effects are exacerbated in (b) despite the two routes being further apart. (c) UAVs in the vertical streams are more likely to have to avoid other UAVs in the vertical streams when $\Delta L_{\rm wp}$ is small. The results shown here are for $\lambda = 0.8\lambda_{\rm max}$.



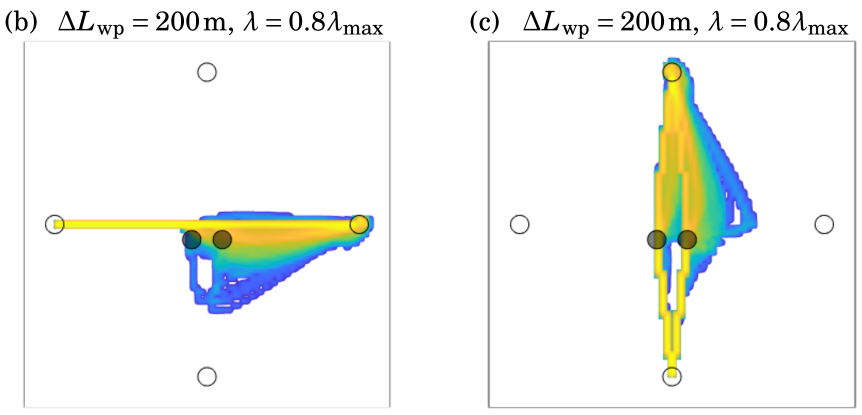


Figure 4.7: (a) Comparison of the mean delay experienced by UAVs in the vertical stream who take the left route (red) or the right route (blue). At small demand the choice of route has little impact on delay but as demand increases the UAVs which are assigned to the left route experience significantly shorter delays on average. This can be understood by examining the density plots for a particular simulation run when $\lambda = 0.8\lambda_{\max}$ for the horizontal (b) and vertical traffic (c). In (c) we can see that UAVs in both routes are deviated to their right as we would expect. As a result those on the left route shorten their flight path while those on the right path lengthen theirs.

relative mean delay decreases to a minimum between $100\,\mathrm{m} \leq \Delta L_{\mathrm{wp}} \leq 300\,\mathrm{m}$. When ΔL_{wp} is larger than this then the increased nominal flight time begins to outweigh any benefit gained from the reduced contention at the crossing points, as is the case for lower demands. However, similar to the horizontal traffic experience, when ΔL_{wp} is small ($\leq 100\,\mathrm{m}$) the relative mean delay is positive and sensitive to small changes in ΔL_{wp}, i.e., we have worsened system performance. We attempt to understand this behaviour by considering how a small group of UAVs interact with each other at a crossing.

Consider a setup where two UAVs from the horizontal stream approach a crossing while ten UAVs in the vertical stream do the same. The UAVs in the vertical stream are placed in to two groups with a horizontal separation ΔL_{wp}.In Figure 4.6(a) and (b) we show snapshots from simulation at the same three points in time (t_1 to t_3) when $\Delta L_{\mathrm{wp}} = 12\,\mathrm{m}$ and $24\,\mathrm{m}$ respectively. In Figure 4.6(a) at time t_1 we can see how the UAVs in the horizontal stream interact with the vertical stream and thus turn to the right. This is the same as the behaviour in Figure 3.11 which leads to secondary avoidance manoeuvres. Here the UAVs in the vertical stream's left route are also deviated to the right due to the avoidance manoeuvres such that they begin to cross in front of UAVs in the vertical stream's right route which then induce further avoidance manoeuvres. These interactions then propagate down the vertical stream until all of the UAVs have now been displaced from their straight line paths by t_3. Figure 4.6(b) shows similar behaviours with a few key differences. Now that ΔL_{wp} is larger, the UAVs in the horizontal stream initially interact more weakly with the UAVs in the vertical stream's right route, i.e., they produce smaller acceleration components than in (a), compare the UAVs' positions at t_1 for both setups. This leads to more extreme avoidance manoeuvres later on and thus the UAVs in the vertical stream experience longer delays as they interact with each other, despite the initial separation between them being larger.

We suspect that the behaviour observed in Figure 4.6(a) and (b) is at least partly responsible for the increase in the relative mean delay observed for small ΔL_{wp}. If this were the case then we would expect the number of UAVs in the vertical streams which interact with another UAV from the vertical streams, $N_{\text{v}|\text{v}}$, to be larger when ΔL_{wp} is small. This is what we see in Figure 4.6(c) which shows the distribution of $N_{\text{v}|\text{v}}$ when $\lambda = 0.8\lambda_{\max}$.

We also consider how the delay incurred by UAVs in the vertical streams differs depending on whether the left or right route is selected when $\Delta L_{\text{wp}} = 200\,\text{m}$, see Figure 4.7(a). We would naively expect the two routes to experience the same delay since the routes are symmetric, which is what we observe for small demands. Alternatively, we might expect that the right-hand route should incur a shorter mean delay based on the 'smear' effect identified in Section 3.5. In fact, when demand is high the opposite outcome is achieved with the left route producing a shorter mean delay. This can be understood by considering the density plots in Figure 4.7(b) and (c) produced for the $\lambda = 0.8\lambda_{\max}$ case. Note that traffic in both of the vertical streams are deviated to the right as we would expect based on the tactical conflict management. The effect of this behaviour is that UAVs following the left-hand route are deviated by avoidance manoeuvres in such a way as to shorten their flight path, in some cases effectively regaining their straight line path. UAVs following the right-hand route however are deviated further to the right, thus lengthening their flight path. This suggests that placing the waypoints in a non-symmetric way could be used to exploit the tactical conflict management to further enhance the benefits of using waypoint-defined routes to split traffic.

4.2.4 Experiments with routes that induce a non-symmetric split

Based on the results in Figure 4.7(a) we will now explore a variation of the setup from Section 4.2.3 where the waypoints will not be placed symmetrically. Instead we will fix $\Delta L_{\text{wp}} = 200\,\text{m}$, since this results in the best performance for the vertical stream, and vary the placement of the waypoints across eight possible setups where the routes lean either to the left or the right. The first and last

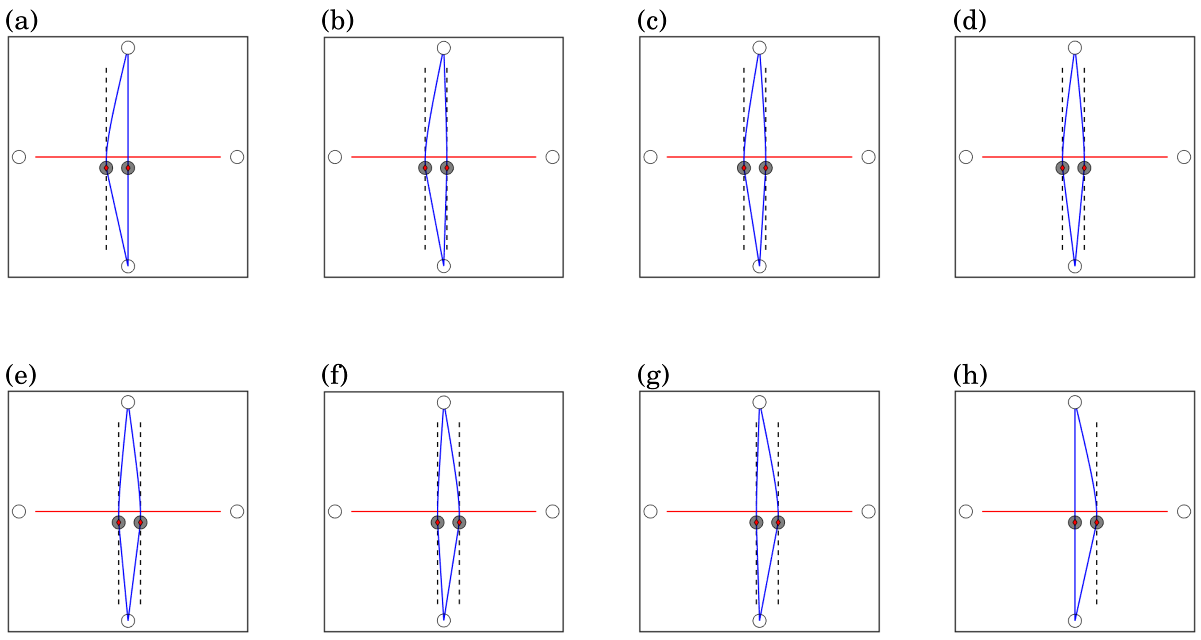


Figure 4.8: For all waypoint layouts $\Delta L_{\mathrm{wp}} = 200\,\mathrm{m}$. Layouts (a)-(d) are left leaning while (e)–(h) are right leaning. The (a) right route and (h) left route recovers a straight line path. Based on Figure 4.7(a) we expect that those setups where the routes lean more to the left will provide shorter relative mean delays.

setup used are the two extremes where one route recovers the original straight line path, see Figure 4.8(a) and (h), i.e., the most left leaning and right leaning setups respectively.

We follow the same simulation procedure and produce estimates of the relative delay for a high and low demand case, see Figure 4.9(a) and (b) respectively. As expected, when the demand is high, the relative mean delay experienced by UAVs in the vertical stream is lower when the waypoints produce routes that lean to the left. In particular the best and worst performance comes when the routes are most left or right leaning respectively. While the absolute difference between the various setups is small it should be noted that all of the left leaning setups produce smaller delays when compared to the equivalent right leaning setup. However, the relationship between relative delay and the left-right leaning setup changes for the intermediate demand case. Now, the best performance is achieved when there is a small lean to the left and performance degrades if the lean becomes too extreme. This is likely because at lower demand levels interactions between UAVs are less common and thus a higher proportion of UAVs will follow their nominal path which in the case of the extreme leaning setups leads to longer inherent delays. This suggests that there is a balance between the effect identified in Figure 4.7(a) and the inherent delays associated with long nominal paths. Furthermore we suspect that the optimal setup for the waypoints will thus be dependent on the current demand experienced by the different streams and is something that should be dynamically altered for the best system performance.

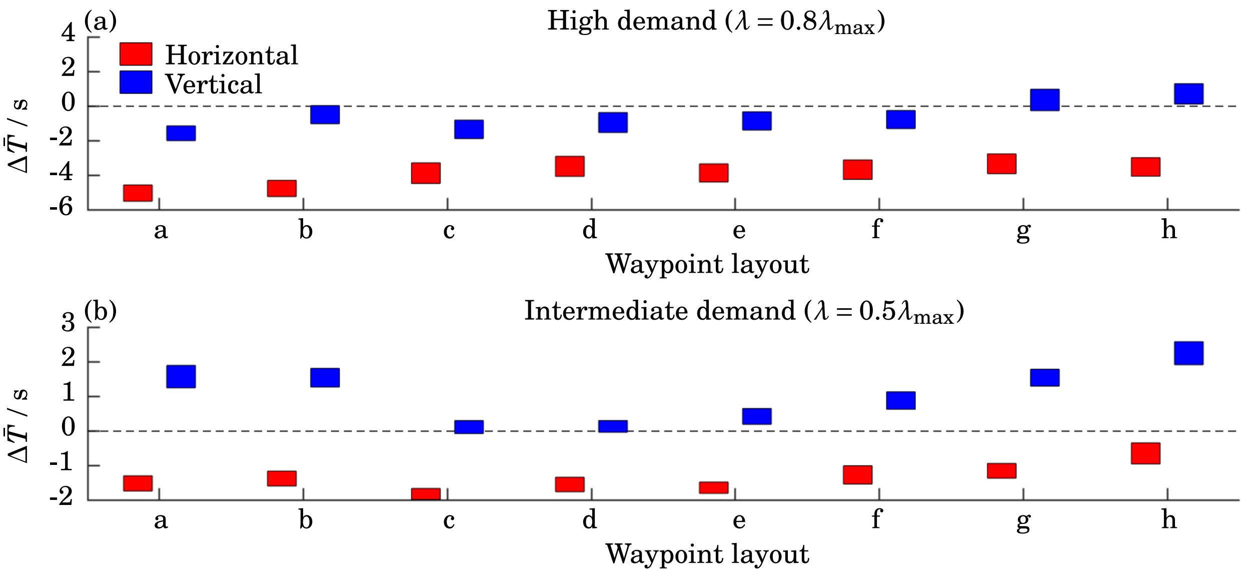


Figure 4.9: A comparison of the relative mean delays experienced by UAVs in the horizontal (red) and vertical (blue) streams when demand is (a) high and (b) intermediate. In the high demand case we can see that, for UAVs in the vertical stream, the most extreme left and right leaning setups correspond to the best and worst performance, as we expected. However there is little difference between the intermediate leaning setups. For the intermediate demand case the vertical stream performs best when the routes lean slightly to the left. If the routes lean too much to the left however the performance degrades as the long nominal path lengths outweigh the effect identified in Figure 4.7. This suggests that the optimum layout for the waypoints will depend on the current demand experienced by the streams.

4.3 Changing UAV behaviour with floor field zones

In Section 4.2 we altered the layout of the traffic network through the introduction of intermediary waypoints, while individual UAV behaviour was unchanged. We used these waypoints to split a stream of traffic across two routes and therefore introduced an extra crossing to the system. We then showed how this could be used to improve system performance compared with a single crossing, since the demand is now shared between multiple crossings. In this section we will instead change the behaviour of some UAVs by introducing a new airspace object. So far we have assumed that the OVM described in Equation 2.2 uses the UAV's current waypoint wherever they are in the airspace. However, we will now explore a design solution in which this assumption is relaxed such that areas of the airspace can be designated as *floor field zones* inside of which certain UAVs will alter their desired heading in a predetermined way. This idea is inspired by pedestrian modelling where floor fields are used to describe complex routing features of pedestrian motion, see Section 4.1.

The use of floor field zones is predicated on the assumption that many features of an airspace will be known about at a strategic level. This may be because they are registered with an authority, for example, a no fly zone due to some other operation or a physical obstacle such as a tall building.

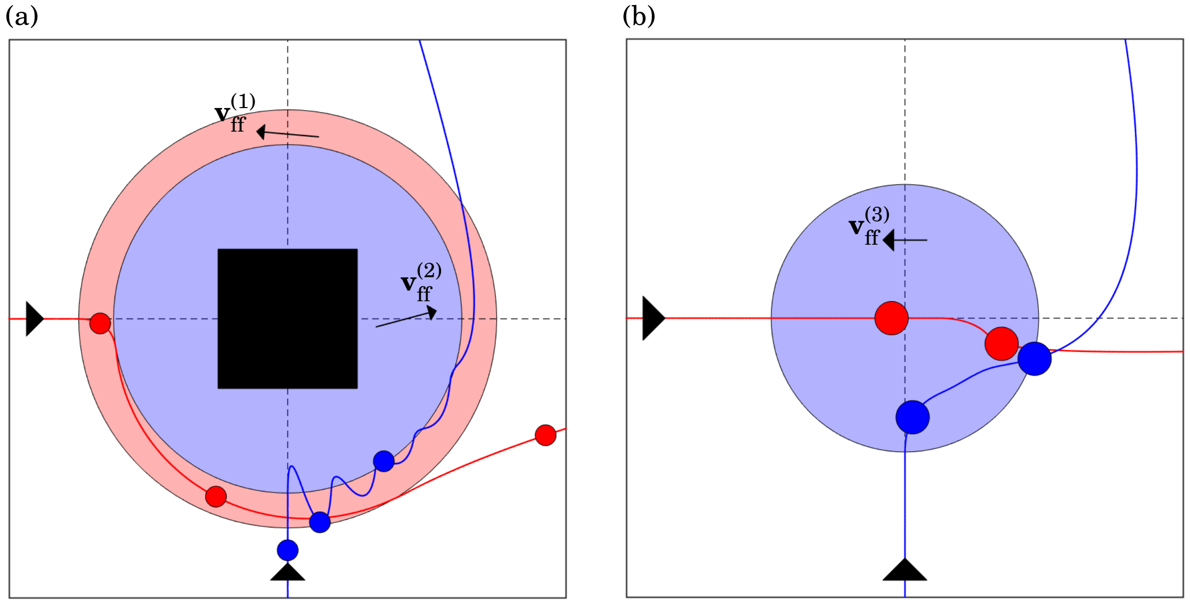

(a)
(b)
$\mathbf{v}_{\mathrm{ff}}^{(1)}$
$\mathbf{v}_{\mathrm{ff}}^{(2)}$
$\mathbf{v}_{\mathrm{ff}}^{(3)}$

Figure 4.10: (a) An obstacle (black square) is placed on the point where two streams of UAVs cross. We setup two floor field zones, the first will affect UAVs in the horizontal stream (red) and has a velocity floor field where the velocity $\mathbf{v}_{\mathrm{ff}}^{(1)}$ is almost perpendicular to the radius that defines the zone. This results in a smooth trajectory where the UAV travels around the obstacle. The second floor field will affect UAVs in the vertical stream (blue) and has a velocity floor field where the velocity $\mathbf{v}_{\mathrm{ff}}^{(2)}$ is almost perpendicular to the radius. This results in oscillatory motion but the UAV does not penetrate the floor field as deeply. We could imagine a more complex floor field where the angle of $\mathbf{v}_{\mathrm{ff}}$ changes the closer a UAV is to the obstacle become more repulsive.(b) We setup a single floor field zone where the velocity $\mathbf{v}_{\mathrm{ff}}^{(3)}$ is perpendicular to the radius. The avoidance interaction between the red and blue UAV is altered compared to the simple crossing. Compared to Figure 3.11, here the red UAV experiences the smaller deviation from its nominal path.

Alternatively a feature may naturally arise due to the layout of origins and demands, such as busy intersections of traffic flows similar to those used for simulation in this thesis. Whatever the case may be, given that these features are known about, a floor field zone, with appropriately designed floor field, can be setup around them. In Figure 4.10 we show two examples of such floor field zones and their effect on the flight paths of two UAVs. In Figure 4.10(a) both UAVs fly around an obstacle. The floor fields, applied within the large red and blue circles, force UAVs away from the obstacle in such a way that the UAVs move around the circumference of the zone. The field parameters could then be tuned to allow for a smoother flight path. In Figure 4.10(b) the avoidance interaction between two UAVs has been altered by subjecting the blue UAV to a floor field zone that causes it to change direction. This means that the angle between the two UAVs' headings is smaller when they begin to avoid each other and this leads to a small deviation from the nominal path for the red UAV and a large deviation for the blue UAV, the reverse of what is generally experienced for the simple crossing as described in Section 3.4.

4.3.1 Floor field zone design

Similar to the routes in Section 4.2 we assume that the floor field zones and their associated floor fields will be designed and implemented at an operational or strategic level. As such we will assume that they are static objects in the system though this assumption could be relaxed in future work where zones could be turned on and off as needed or the floor fields they contain could change over time. We expect that the number of features that warrant a floor field zone will be small, at least compared with the number of UAVs, and as such this approach will maintain scalability as each feature will have one or two zones associated with it. The zones themselves will be setup prior to simulation and each zone will have an ID associated with it which will be used to determine which UAVs are influenced by its floor field. UAVs will be prescribed a zone on take off so no subsequent strategic control decision is taken at the UAV level.

A floor field zone could take on any desired geometry with complex floor fields contained therein. In order to constrain the problem we will only explore zones with circular geometry and floor fields where the velocity is perpendicular to the zone's radius, inspired by roundabouts seen in traditional road transport networks. The floor field associated with a zone is then implemented in simulation by placing a square grid placed over the world. Any cell who's center lies within the zone then has a floor field velocity $\mathbf{v}_{\mathrm{ff}}$ ascribed to it, see Figure 4.11(a). We assume that $|\mathbf{v}_{\mathrm{ff}}| = v_{\mathrm{CS}}$. If a UAV is within one of these cells and is subject to this particular floor field zone then it will change the waypoint following rule to

$$\mathbf{a}_i^0 = \frac{1}{\tau}(\mathbf{v}_{\mathrm{ff}} - \mathbf{v}_i), \tag{4.4}$$

which replaces the waypoint following velocity with $\mathbf{v}_{\mathrm{ff}}$. Therefore the UAV will tend to align its velocity with the floor field.

Unlike cars and roundabouts, where a driver will choose the appropriate exit, a floor field that induces circular motion, such as that in Figure 4.11(a), could lead to an undesirable outcome where a UAV cannot escape the floor field zone. The way we solve this problem here, in an effort to maintain simplicity and remove decision making from the UAV level, is to remove a segment of the floor field zone, see Figure 4.11(b). This ensures that the UAV will eventually leave the zone just by following the nominal path at which point the original waypoint following behaviour will take over. Therefore, based on the assumptions and constraints presented in this section, the floor field zones will be described by two parameters, the radius R_{ffz} and an angle θ_{ffz}.

4.3.2 Experiments with a UAV crossing setup and a single floor field zone

As with previous sections we will focus on exploring whether these floor field zones, as described in Section 4.3.1, can be used to improve performance at a crossing point between two streams of traffic. Similar to the waypoint-defined routes used in Section 4.2.3 we will begin by only applying a floor field zone to traffic in the vertical stream. An example of the experimental layout can be seen in Figure 4.11(b) along with the nominal paths taken by horizontal and vertical traffic. The

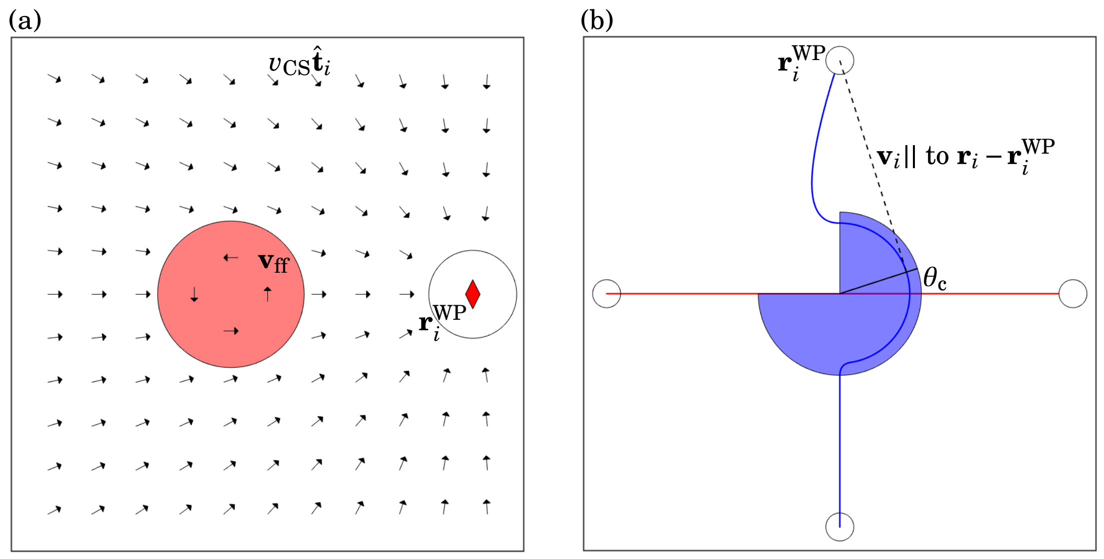

(a)
$v_{\mathrm{CS}}\hat{\mathbf{t}}_i$
$\mathbf{v}_{\mathrm{ff}}$
$\mathbf{r}_i^{\mathrm{WP}}$
(b)
$\mathbf{r}_i^{\mathrm{WP}}$
$\mathbf{v}_i \| \text{ to } \mathbf{r}_i - \mathbf{r}_i^{\mathrm{WP}}$
θ_{c}

Figure 4.11: (a) A generic floor field zone (red circle) with its associated velocity floor field which has been set up to induce circular motion. The arrows outside the zone show the desired velocity $v_{\mathrm{CS}}\hat{\mathbf{t}}_i$ associated with the waypoint (white circle) at $\mathbf{r}_i^{\mathrm{WP}}$ a UAV would align with outside of the zone as per Equations 2.1 and 2.2. (b) Proposed design for a floor field zone that aims to improve the performance of a UAV crossing as described in Section 3.3. Here we only apply the floor field zone to UAVs in the vertical stream. When a UAV enters the zone circular motion is induced (blue path). If we remove the segment of the floor field zone above the solid black line, such that the zone makes an angle θ_{c} with the horizontal axis, then the shortest possible flight path for a floor field zone with radius R_{ffz} is achieved. This is because when the UAV exits the zone its velocity is aligned with its destination.

vertical traffic begins to turn to the right when it enters the floor field zone and then adopts a circular path until it exits the zone and begins to change direction to once again intersect with its destination. The floor field inside the zone is designed to produce this anti-clockwise motion so that when two UAVs from different streams begin to avoid each other the angle between the UAVs' velocities should be smaller than in the simple crossing case. The idea is to produce a smoother integration of the two streams and reduce the occurrence of large deviations experienced by UAVs in the horizontal stream.

Another design choice shown in Figure 4.11(b) is the way we remove a segment of the floor field zone. Since traffic in the vertical stream starts from the bottom origin, as viewed on the page, we apply the floor field zone to all parts of the circle below the horizontal traffic stream which results in a semi-circle common to all such zones. Note that, similar to waypoint-defined routes, UAVs subject to a floor field zone will necessarily incur a delay compared with the nominal straight line path, however there are ways to minimise this. In the example we set θ_{z} to $3\pi/2$, just removing the top left segment, so that the UAVs leave the zone at the closest point to their destination. The problem with such a setup is that when the UAV leaves the zone it needs to accelerate in order to realign its velocity with its destination, the curved part of the path

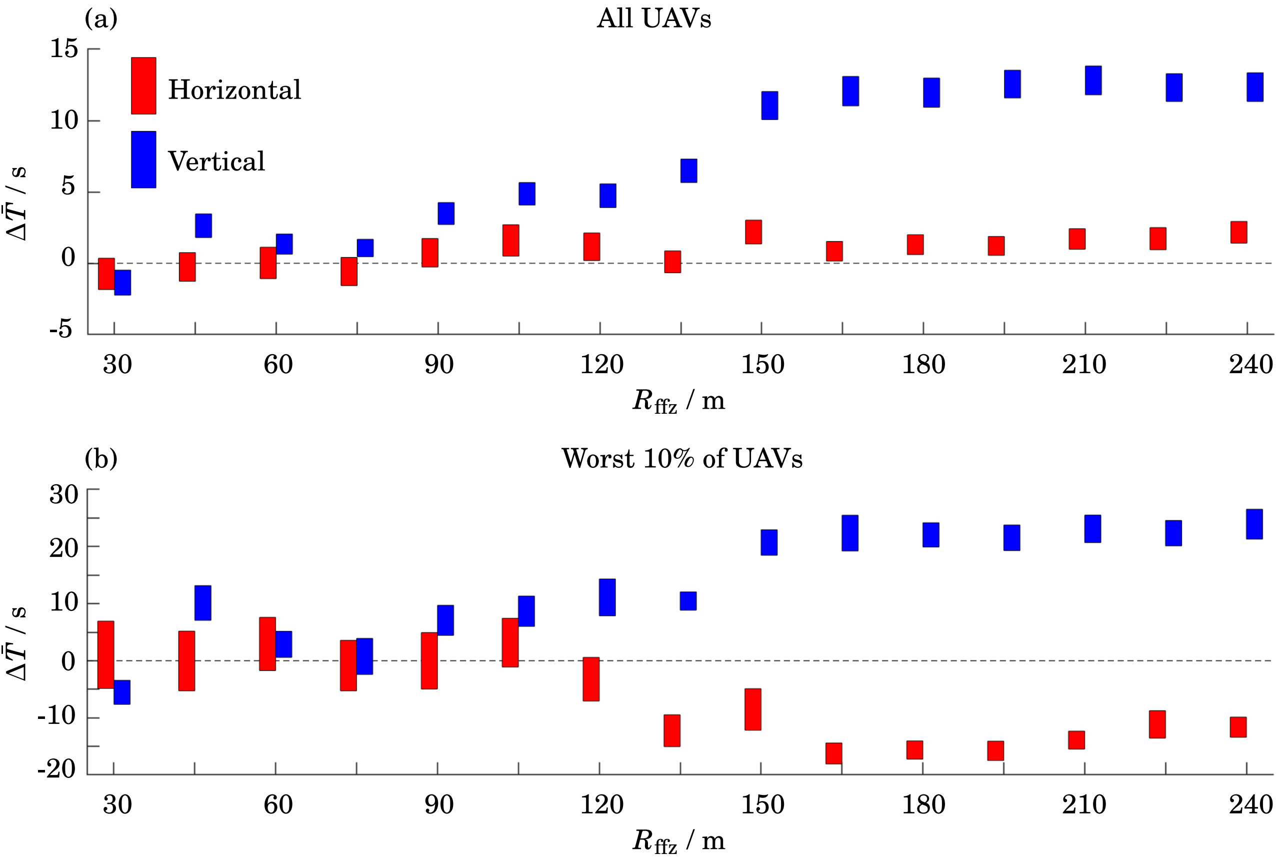


Figure 4.12: (a) Compare the relative delay for the horizontal (red) and vertical (blue) traffic using a variety of sized floor fields when $\lambda = 0.8\lambda_{\mathrm{max}}$. In (b) we show the same analysis but only for UAVs with the worst 10% of delays. From (a), the floor field has not significantly altered the mean delay for UAVs in the horizontal stream. However, from (b), the worst delays experienced by UAVs in the horizontal stream are reduced by about 10 s or more, when $R_{\mathrm{ffz}} \geq 135\,\mathrm{m}$. This comes at the cost of an increase to the delay experienced by UAVs in the vertical stream in both (a) and (b). These results can be explained by the behaviour observed in Figure 4.13 and 4.14.

where the UAV bends back on itself. Instead, if we removed the segment above the black line shown in the figure, then when the UAV left the zone its velocity would already point toward its destination and no change of heading would be needed. This would lead to a straight line path after the UAV has exited the zone leading to a shorter flight time compared with the example flight path. We therefore call the value of θ_z that leads to this behaviour the critical angle and it will depend on R_{ffz}. We will therefore sweep over values for R_{ffz} from 30 m to 240 m and set θ_{ffz} to the corresponding critical value. This ensures that the nominal path for UAVs in the vertical stream incurs the smallest possible delay.

We consider how the relative mean delay, as defined in Section 4.2.3, depends on R_{ffz} when $\lambda = 0.8\lambda_{\mathrm{max}}$ for both streams. In other words we will only consider a high demand case since this is when additional traffic management is most needed. From Figure 4.12(a), the floor field zone seems to have little to no impact on the delay incurred by UAVs in the horizontal stream.

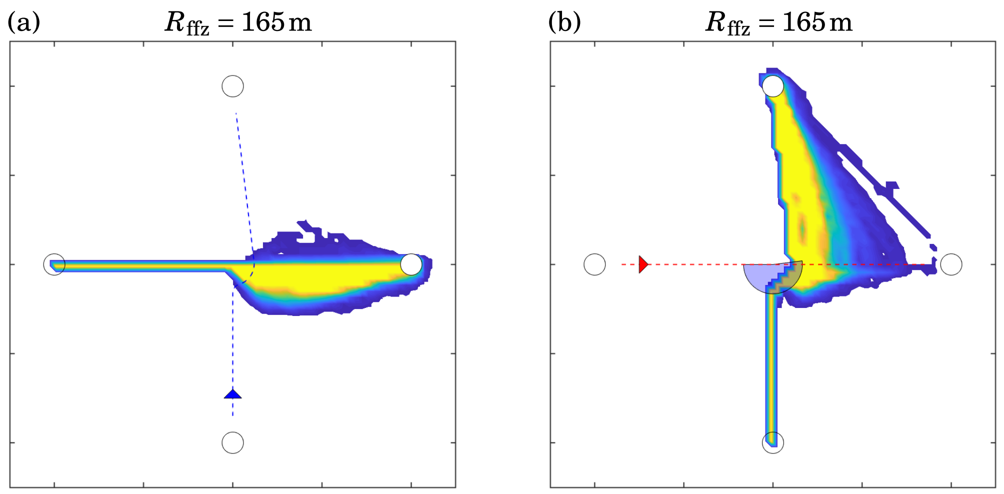


Figure 4.13: Density plots for UAVs in the (a) horizontal and (b) vertical streams when $R_{\mathrm{ffz}} = 165\,\mathrm{m}$. The floor field zone experienced by UAVs in the vertical stream is represented by the blue circular segment in (b). The dashed lines in (a) and (b) show the nominal flight paths for the vertical and horizontal streams respectively. We can see that the 'give way' behaviour experienced by UAVs in the horizontal stream has been eliminated.

However, this changes if we consider the mean delay experienced by UAVs with the top 10% of delays, see Figure 4.12(b). When considering these UAVs with the worst delays we can see that, when $R_{\mathrm{ffz}} \geq 135\,\mathrm{m}$, the mean delay for UAVs in the horizontal stream is reduced by about 10 s or more. This suggests that when the floor field zone is large enough, UAVs in the horizontal stream tend to experience slightly longer delays compared to the single crossing setup from Chapter 3 but do not experience the long delays associated with the right-handedness discussed in Section 3.4.

UAVs in the vertical stream however experience a steady increase in mean delay as R_{ffz} increases up to 150 m for both the overall mean and the mean of the worst 10% of UAVs, see Figure 4.12(a) and (b) respectively. This seems to correspond to the increase in nominal delay associated with the increased path length that results from larger values of R_{ffz}. However, when R_{ffz} increases beyond 150 m, up to the maximum value used, the delay is unaffected. For the overall mean delay this corresponds to an increase of about 10 s for large values of R_{ffz}.

In order to understand the relationships between R_{ffz} and the delays incurred by UAVs in both streams we consider density plots for the $R_{\mathrm{ffz}} = 165\,\mathrm{m}$ case, see Figure 4.13. The dashed lines show the nominal flight paths that UAVs would take without interaction with another UAV and the transparent circular segment in Figure 4.13(b) shows the outline of the floor field zone. Compare these with Figure 3.12(a) and (b). The large deviations undertaken by some UAVs in the horizontal stream in Figure 3.12(a) have been eliminated here. Now, no UAV in the horizontal stream undergoes a lateral deviation of more than a couple hundred meters. However, more UAVs now experience small deviations, compare the width of the two yellow areas in

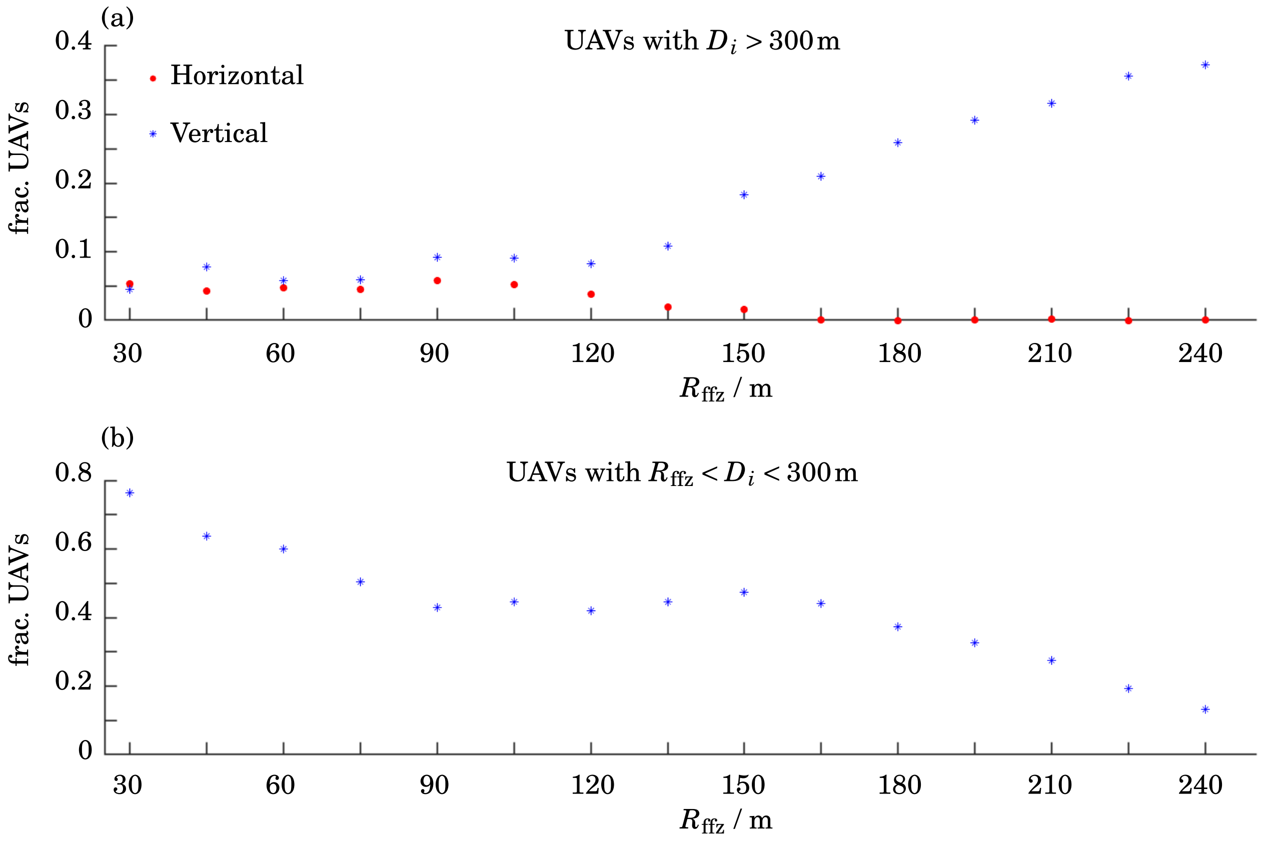


Figure 4.14: From (a) we can see that R_{ffz} needs to be larger than 165 m in order to ensure that the worst deviations experienced by UAVs in the horizontal stream are prevented. However, the fraction of UAVs in the vertical stream that experience large deviations steadily increases up to 37% when $R_{\text{ffz}} = 240\,\text{m}$. (b) The fraction of UAVs in the vertical stream that experience intermediate deviations decreases steadily as R_{ffz} increases. This explains why the mean delay experienced by the vertical stream remains the same when $R_{\text{ffz}} \geq 150\,\text{m}$. The deviations experienced by UAVs in the vertical stream become more polarised and thus balance each other out.

Figures 4.13(a) and 3.12(a). This suggests that, since the overall mean delay does not change much, the elimination of the extremely large deviations is balanced by an increase in the mean deviations.

As for the vertical stream, we can clearly see where the UAVs deviate from their straight line path as they enter the floor field zone and begin to be influenced by the velocity floor field. The angle of approach between UAVs in the two streams now changes throughout their interaction and this leads to a sort of role reversal between the two streams, compared with Figure 3.12. In Figure 4.13(b) we see that UAVs in the vertical stream now undergo larger deviations than those experienced by UAVs in the horizontal stream, though not quite as extreme as those observed in Figure 3.12(a). This effect becomes more pronounced as R_{ffz} gets larger, see Figure 4.14(a). While $R_{\text{ffz}} \leq 120\,\text{m}$ the rate at which UAVs experience lateral deviations of more than 300 m is about the same for both streams. Once R_{ffz} increase beyond this though large deviations quickly become rare for the horizontal stream while becoming increasingly likely for the vertical stream. When R_{ffz} is at the maximum value used, around 37% of UAVs in the vertical stream experience large lateral deviations.

Given that the likelihood of a UAV in the vertical stream experiencing large deviations increases with R_{ffz} this would suggest that the delay incurred should also increase. However, as previously noted, the delay remains largely unchanged for $R_{\text{ffz}} \geq 150\,\text{m}$. This seems to be explained by the fact that while large deviations are more common at large values of R_{ffz}, so too are small deviations. In other words, the distribution of lateral deviations experienced by the UAVs in the vertical stream becomes more polarised as R_{ffz} increases so that UAVs tend to either experience small or large deviations with relatively few in between, see Figure 4.14(b). This leads to a region where the mean delay remains static.

These results show that floor field zones may indeed be used to change the nature of a UAV crossing. The way in which we have setup the floor field zone here has led to the burden of conflict management shifting from the horizontal stream to the vertical stream. This may be a satisfactory outcome in many scenarios, for example where one traffic stream should be given priority over another. Alternatively, if the goal of ensuring some upper limit on delay is maintained is deemed more important than the goal of minimising the mean delay then this setup would seem to be a better choice than the simple single crossing. However, since we have primarily focused on the former goal throughout this thesis, in the next section we will explore a variation of the floor field zone presented here which aims to make use of the traffic splitting idea in a new context.

4.3.3 ID controlled floor field zones

In Section 3.5 we showed how splitting demand between two parallel streams could improve the performance of a UAV crossing. We called this ideal splitting since it involved introducing extra ports and changing the layout of the system. Then in Section 4.2 we explored how we could use waypoints to form routes so that UAVs that depart from a single port can be split in to multiple streams, thus achieving some of the benefits observed for the ideal splitting without having to redesign the layout of the experimental setup. The aim here then is to use the same principle of providing UAVs multiple routes to their destination in order to lower contention at each of the new crossings that arise from these routes. In this section we will combine this principle with the floor field zone to produce a system where UAVs in the vertical stream have two possible routes, see Figure 4.15(a).

The way in which we produce the two routes in Figure 4.15(a) is by allowing some UAVs in the stream to be unaffected by the floor field zone. As stated, we assume that all the floor field zones in an airspace are generated at the port level and generally on a longer operational timescale than individual UAV mission times. Therefore when a UAV departs from a port it can be assigned to various floor field zones that it will be influenced by, if entered in to. However, not

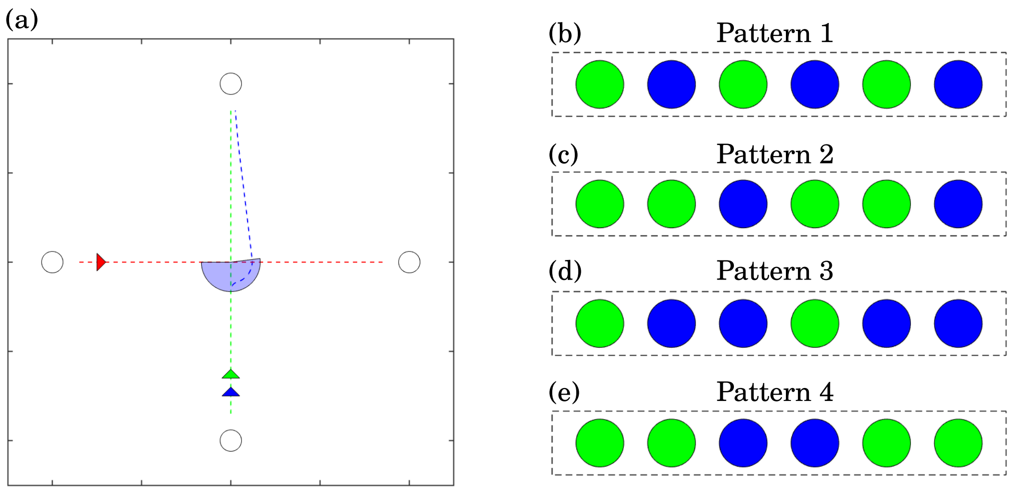


Figure 4.15: (a) Three nominal paths are shown by the dashed lines, one for the horizontal stream (red) and two for the vertical stream when UAVs are affected by the floor field zone (blue) and when they are not (green). The idea here is to exploit the same sort of splitting effect as that observed in Section 3.5. UAVs in the vertical stream are assigned to the floor field zone in a deterministic way according to four possible patterns; simple even split (b), biased to the straight line path (c), biased to the floor field zone (d) or clustered even split (e).

all UAVs with a common destination need to be subject to the same floor field zones. We achieve this simply in simulation by assigning each floor field zone a unique ID and assign each UAV on departure a list of IDs which correspond to the floor field zones it is affected by. In this work this effectively becomes a binary flag since only one floor field zone is used but this ensures that we can precisely control which UAVs are affected by which floor field zone in larger, more complex scenarios.

In this section we will fix R_{ffz} at 165 m since this is the smallest floor field zone that effectively eliminates the large deviations experienced by UAVs in the horizontal stream. Besides this we will maintain the same setup parameters as in Section 4.3.2, i.e., using the critical angle for the floor field zone and the same demand value. When a UAV in the vertical stream departs from its origin then it is assigned to one of the routes in a deterministic way similar to Section 4.2.1. However, here we will explore four different patterns for route assignment. The first and most simple is to split traffic between the routes evenly and to alternate between them as each new UAV departs, see Figure 4.15(b). We will also consider setups where the UAVs are biased toward the straight line route or curved route, see Figure 4.15(c) and (d) respectively. The last pattern used is one in which the UAVs are split equally between the routes again but alternate after two UAVs depart, see Figure 4.15(e). This has the effect of clustering UAVs in to small groups. This is obviously not an exhaustive set of options and other methods could be explored, such as assigning UAVs to routes via a stochastic method, but is meant to explore if this has any effect on the performance.

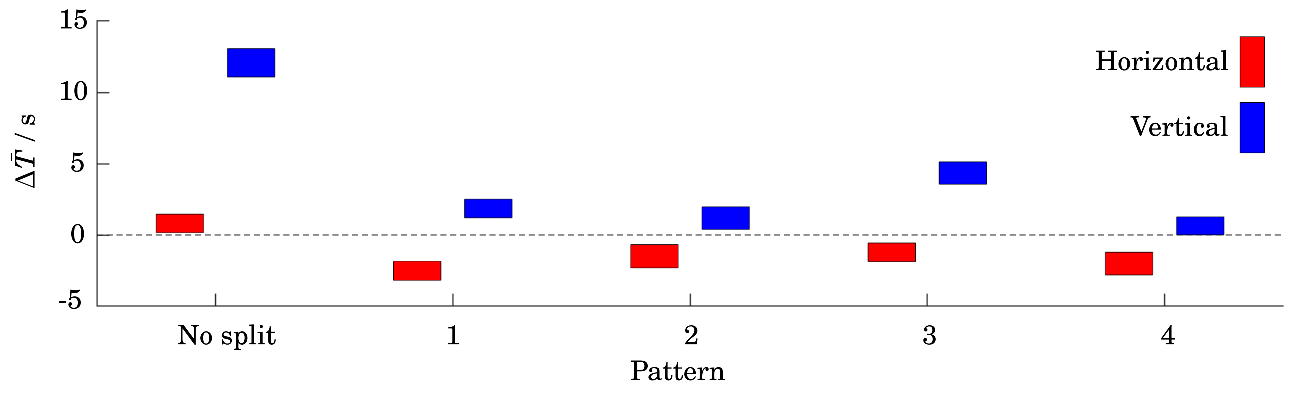


Figure 4.16: Average delay incurred by UAVs in the horizontal (red) and vertical (blue) streams compared with the single crossing case from Chapter 3. For all the patterns used the delay experienced by the horizontal stream is reduced at the cost of an increase to the delay experienced by UAVs in the vertical stream. This increase is largest when the vertical stream is biased toward being affected by the floor field zone. The other three patterns result in a net benefit or no change to the overall delay.

Following the same simulation procedure as before we produce bootstrapped estimates for the mean delay experienced by the UAVs in the vertical and horizontal streams for each of the four route assignment patterns used, see Figure 4.16. If we compare this to Figure 4.12 we can see that UAVs incur significantly shorter delays when the vertical stream is split across the two routes. The UAVs in the vertical stream still increase their overall delay compared with the single crossing case but this is now around 2 s or less for most of the route assignment patterns used. Importantly, the overall mean delay of UAVs in the horizontal stream is now reduced compared with the single crossing case.

Under the same logic that motivated the development of the hybrid avoidance rule, near the end of Section 2.6, we suggest that an increase in the overall mean delay of one stream is permissible if the other stream receives a similar or greater decrease, thus providing an overall system benefit. Particularly since here the increase is incurred by the vertical stream which tends to experience smaller delays than the horizontal stream for the single crossing case. This requirement then is satisfied for all but one of the route assignment patterns and thus suggests that this method for traffic management may be useful but further work is needed to determine the best possible implementation.

4.4 Platoon formation at takeoff

In Section 3.4 we show how the right handedness of the default avoidance rule can cause some UAVs to undergo large deviations from their nominal flight path due to secondary avoidance interactions. In the crossroad setup presented in Chapter 3 traffic in the horizontal stream are

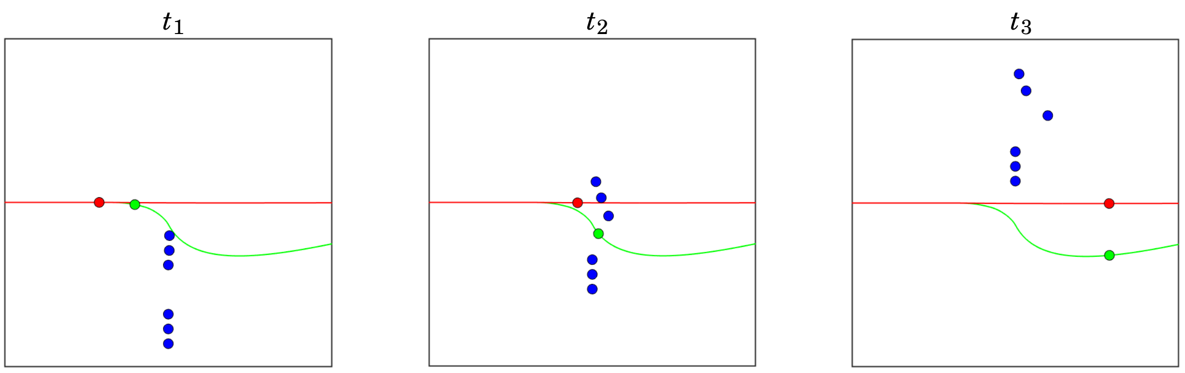


Figure 4.17: Three snapshots from a simulation demonstrate how forming the UAVs in to platoons might help alleviate the impact of the right handedness on horizontal traffic. At t_1 we see the UAVs still following their nominal path. At t_2 the green UAV has turned head on to the vertical stream after avoiding the UAVs in the first platoon but has started to change course due to the waypoint following and is now passing through the inter-platoon gap. At t_3 all the UAVs are clear of the crossroad. Note that the red UAV and the UAVs in the second platoon are undisturbed from their nominal paths. Compare this to the behaviour seen in Figure 3.11.

most adversely affected by this right handedness. In particular, as demand on the vertical stream increases the UAVs in the horizontal stream are more likely to undergo longer deviations as they turn into the vertical stream and therefore must avoid many more UAVs. We will attempt to alleviate this effect through the implementation of UAV *platoon formation*.

Up to this point UAVs have been generated using a random process and take off as soon as they are able, see Section 3.2, a scheme which we now call *on demand*. We propose that the large deviations experienced by UAVs in the horizontal stream, caused by the right handedness, can be reduced by forcing UAVs to wait to depart until a group of N UAVs is formed. In doing so the UAVs will form small, tightly packed, 'platoons' with gaps between the platoons which are larger than the average gap between UAVs under the on demand scheme. We illustrate this using a similar setup to that shown in Figure 3.11. In Figure 4.17 however the blue UAVs have been formed in to two platoons, i.e., $N = 3$. The green UAV still undergoes avoidance manoeuvres in which it turns to its right, nearly head on with the vertical stream, but now it only interacts with the blue UAVs in the first platoon before passing through the inter-platoon gap. The red UAV now interacts with none of the blue UAVs and none of the blue UAVs in the second platoon interact with any other UAVs. In this section then we will explore whether this behaviour can improve delay in larger scale crossroad setups through the implementation of platoon formation at UAV origins.

4.4.1 Queue based UAV platoon formation

We implement platoon formation by assuming that each origin is assigned a number N which dictates the number of UAVs in a platoon. As with the waypoint-defined routes discussed in Section 4.2.1 we assume that the platoon size is controlled at a higher operational or strategic level and therefore we will consider static values for N as a proof of concept. However, in future work we could consider a case where a given origin will adjust N based on different factors such as demand or queue capacity.

During operation UAVs must wait until a platoon has formed before departing from their origin. A platoon is formed when N UAVs have been generated with the same destination. Once a platoon is formed the UAVs in the platoon will begin to take off from the origin in the order they were generated. Importantly, the UAVs will still respect the minimum safe take off separation S_{TO} defined in Section 3.3. This, combined with other parameters such as demand λ and velocity v_{CS} will provide us with various ways to describe different platoon setups which we will discuss more in Section 4.4.2.

In the experimental setup described in Section 3.3 each origin has a queue associated with it, which we now call Q_{a}, which is used to store UAVs that have been generated but can not depart. We will implement platoon formation by adding another queue Q_{b} to each origin where N is larger than 1. Once there are N or more UAVs in Q_{a} then a platoon is formed and the first N UAVs are transferred over to Q_{b}, see Figure 4.18(a). UAVs in the platoon will then depart as described above until Q_{b} is empty again, see Figure 4.18(b) and (c). Meanwhile, UAVs are added to Q_{a} by a random process as before.

One of the disadvantages of UAVs forming these platoons is that a UAV may spend time waiting in Q_{a} when it would otherwise not. We will therefore consider two flight times for all UAVs which undergo platoon formation: simple flight time and modified flight time. The simple flight time is the same as the flight time described in Section 3.3. The modified flight time will take in to account the time UAVs in a platoon spend waiting for the platoon to form. This wait time t_{w} is the time between Q_{b} being empty and then being re-filled. Note that we only start to measure t_{w} when Q_{a} is not empty i.e., at least one UAV is actually waiting. The modified flight time for UAV i in a platoon of size N is then

$$t_{i,\mathrm{p}} = t_{\mathrm{f}} + t_{\mathrm{w}}\frac{N-i}{N-1}, \tag{4.5}$$

which adds a portion of the wait time based on the UAV's position in the platoon to UAV's recorded flight time. We do not consider time UAVs spend in Q_{b} since this is no different to UAVs in the on demand scheme waiting for the safe take off separation to be met before they depart.

4.4.2 Platoon formation with the crossroad setup

In this section we will apply the platoon formation scheme described in Section 4.4.1 to the crossroad setup. Since the primary goal here is to reduce the impact of long diversions on UAVs in

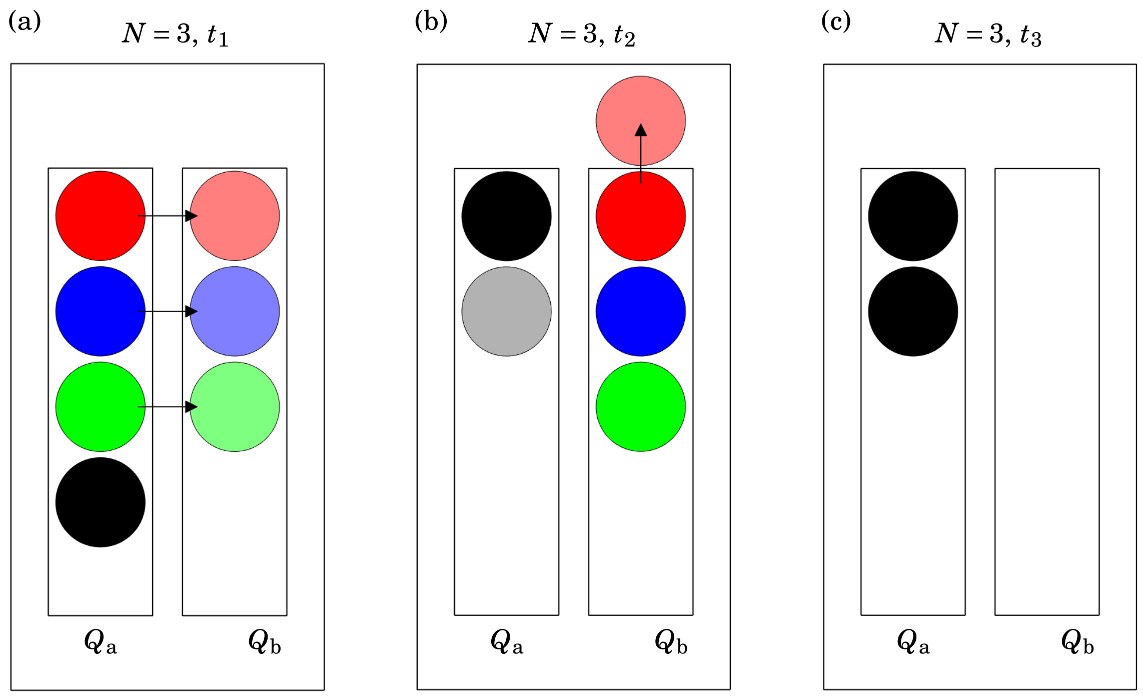


Figure 4.18: (a) There are 4 UAVs in Q_a and Q_b is empty. Therefore the first 3 UAVs in Q_a are moved to Q_b. (b) UAVs will depart from Q_b, respecting S_{TO}, until it is empty again. UAVs are added to Q_a according to a random Poisson process as in previous setups. (c) UAVs will wait in Q_a until there are at least N UAVs. We only consider time spent waiting in the queue while Q_b is empty.

the horizontal traffic stream, we will only apply platoon formation to the vertical stream. Given the platoon formation scheme described in Section 4.4.1 we can begin by performing a simple analysis on how different platoon features depend on the various parameters of the cross road setup and from this establish a range of platoon sizes to be explored.

Consider two platoons that have departed from the same origin, see Figure 4.19. The length of a platoon measured from the centre of the first UAV to the center of the last is

$$L_p = (N-1)S_{TO}, \tag{4.6}$$

which is fixed here since N and S_{TO} do not change during a single simulation. The length L_p also gives us the time it takes for a platoon to complete its departure

$$t_d = \frac{L_p}{v_{CS}}, \tag{4.7}$$

since all UAVs depart from their origin at the cruising speed. The other important time period associated with a given platoon size is the average time it takes for N UAVs to be generated in Q_a

$$\tau_p = \frac{N}{\lambda}, \tag{4.8}$$

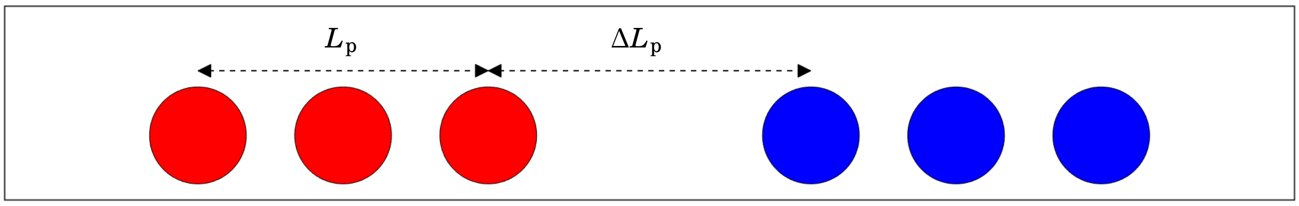


Figure 4.19: Diagram of two platoons showing the definitions of the platoon length L_{p} which should remain constant and the inter platoon gap ΔL_{p} which is stochastic. The average width of ΔL_{p} will depend on the cruising speed, the time it takes a platoon to depart and the average time it takes for a platoon to form.

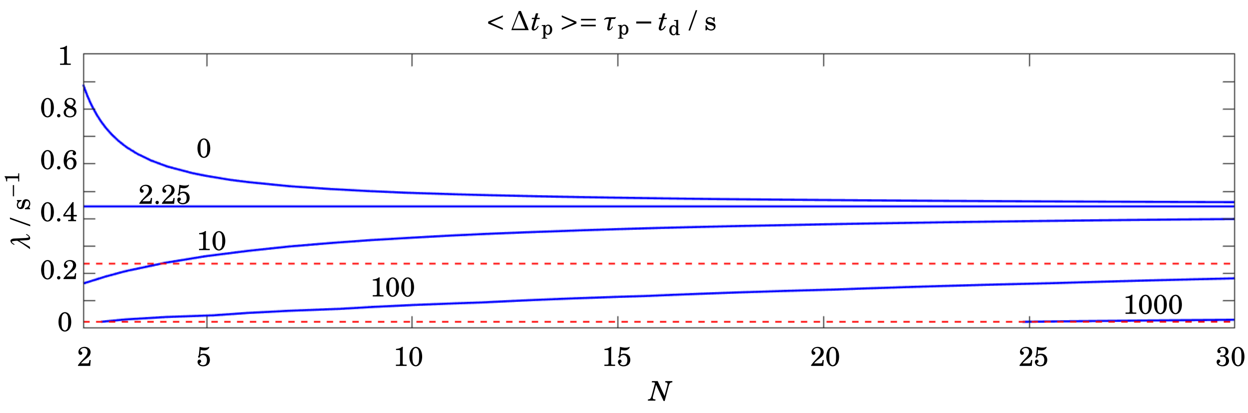


Figure 4.20: Contour plot that shows how the mean inter-platoon time depends on the demand λ and the size of the platoon N.

which depends on the demand at the origin.

The average inter-platoon width is then given by

$$< \Delta L_{\mathrm{p}} >= \max\left(\frac{< \Delta t_{\mathrm{p}} >}{v_{\mathrm{CS}}}, \frac{S_{\mathrm{TO}}}{v_{\mathrm{CS}}}\right), \tag{4.9}$$

where $< \Delta t_{\mathrm{p}} >= \tau_{\mathrm{p}} - t_{\mathrm{d}}$, because if the time taken for a platoon to depart is greater than the time it takes for N UAVs to be generated, then the inter-platoon width will tend toward a lower bound given by the safe takeoff separation and the cruising speed. In Figure 4.20 we present a contour plot showing how the difference between τ_{p} and t_{d} changes given a particular platoon size and demand for the values of v_{CS} and S_{TO} established in Sections 2.5 and 3.3 respectively. The minimum and maximum demand values of $0.1\lambda_{\max}$ and $\lambda_{\max}$ are shown by the two red dashed lines. The region above $\lambda \approx 2.25\,\mathrm{s}$ is where the inter-platoon width tends toward S_{TO}. Therefore, for the demand range we will be exploring, the average inter-platoon width will always be larger than S_{TO}.

If this kind of platoon formation scheme were to be used in practice then one of the relationships that would have to be tuned for is the inter-platoon width and time spent waiting for a platoon to form. Larger inter-platoon widths should allow for more UAVs in the horizontal stream to travel without undergoing long diversions and are therefore desirable. However, time spent waiting for a platoon to form is generally undesirable since it results in longer mission times. The best balance between these will likely be very context or mission specific. For example, some high priority deliveries (such as medical supplies) will want to minimise their individual mission time. Other types of traffic (such as consumer goods delivery) may prioritise overall traffic performance and therefore be willing to endure longer platoon formation times. It is also worth noting here that time spent waiting at an origin is different to extra time spent in flight. A UAV can only remain in flight for a given period of time before it is forced to land and refuel. A UAV that has yet to depart can wait at the origin indefinitely.

Using the same simulation procedure as before we will sweep over the demand range $0.1\lambda_{\max}$ to $\lambda_{\max}$ and set the platoon size such that $N = 2,...,5$. We suggest that $N = 5$ is a natural upper limit since, when demand is at the minimum value used, the average time between platoons is a little over 200 s. This means that the first UAV in the platoon will at least triple its overall mission time without accounting for any delays incurred through the tactical conflict management. As discussed, while time spent waiting on a landing pad can be considered materially different to delays incurred in the air, it is unlikely that waiting times longer than this would be tolerable for any individual member of a platoon. However it is worth exploring these larger sized platoons since at high demand $< \Delta t_{\mathrm{p}} >$ is about 12 s which is equivalent to the delay it would experience in the single crossing setup without platooning. We also confirm that our model for the average inter-platoon time is correct in Figure 4.21 by comparing it to the observed values for $< \Delta t_{\mathrm{p}} >$ for the various simulation parameters.

We will consider what effect platooning has on the relative delay using the simple flight times experienced by UAVs in the horizontal and vertical streams, see Figure 4.22(a) and (b) respectively. At low demands, both streams show that platooning has little impact on the delay compared with the on demand single crossing. In fact there is a tendency for larger sized platoons to cause small increases to the mean delay, up to about $\lambda = 0.4\lambda_{\max}$. This is likely due to the fact that, while interactions between UAVs are rare at low demands, those UAVs in the horizontal stream that do have to undergo avoidance manoeuvres now interact with many more UAVs than they otherwise would. The relationship between the relative mean delay and platoon size becomes more complex though when $\lambda \geq 0.5\lambda_{\max}$. At these higher demands platooning does provide some significant reductions to the mean delay for both streams and some sizes of platoon. However, some demand and platoon size combinations can also lead to significantly longer mean delays, for example when $\lambda = \lambda_{\max}$ and $N = 4$. Also, what benefits are gained for the vertical stream are lost when we consider the modified flight time, see Figure 4.22(c). In other words, when we incorporate the time spent waiting for a platoon to form then the vertical stream always experiences an increase in the mean delay. This is unsurprisingly most prominent when demand is low and the inter-platoon time is at its largest.

These results suggest that there is some benefit to employing platooning when demand is

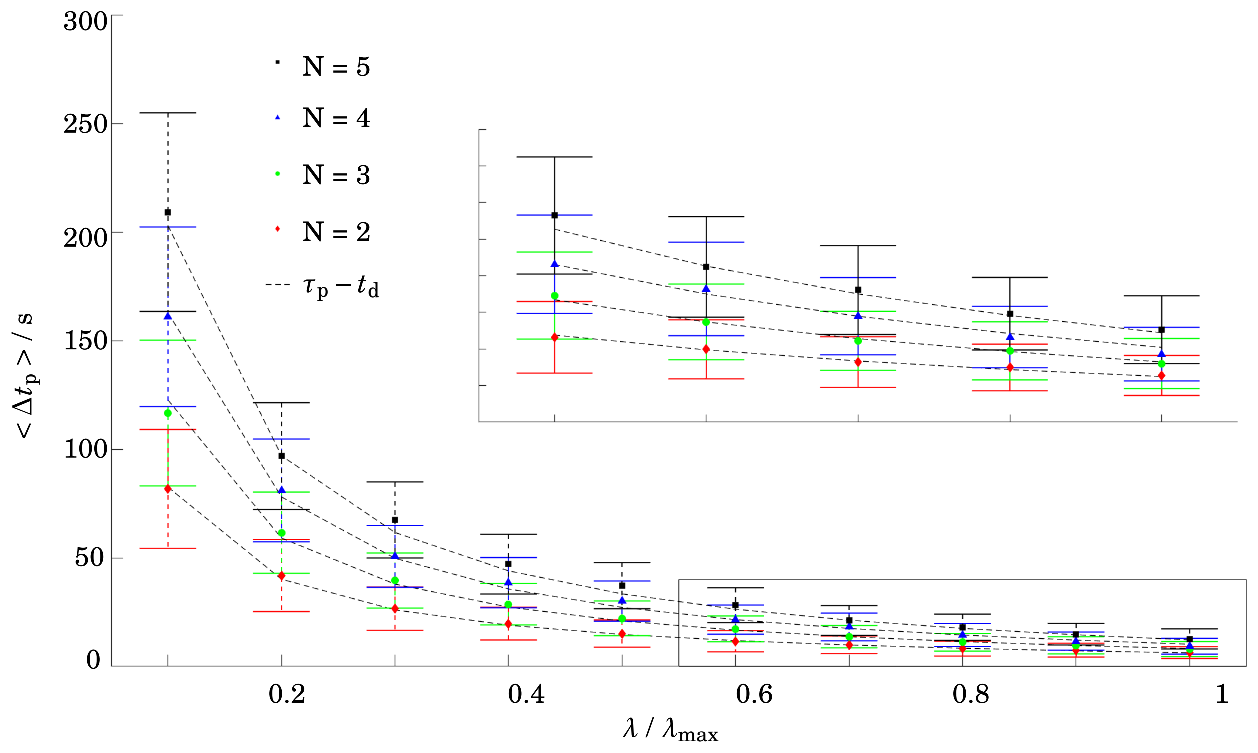


Figure 4.21: We record the inter-platoon time throughout each simulation and use this to generate an estimate of $< \Delta t_p >$. We then compare this to the expected value for $< \Delta t_p >$ based on Equations 4.7 and 4.8 (dashed lines). There appears to be good agreement between the model and observed mean inter-platoon times for all demand levels and platoon size.

high but further work is needed to determine when and how it can be used to greatest effect. One possible refinement would be to employ a dual system where a platoon of UAVs can depart either when the platoon reaches a certain size or when some maximum wait time has been reached. This could then be used to limit the worst waiting times.

4.5 Conclusion

In this chapter we have shown how we can improve the efficiency of a UAV crossing, i.e., reduce the relative mean delay experienced by UAVs, by supplementing the hybrid conflict avoidance method from Chapter 2 with other traffic management methods which are applied at the port level rather than by in-flight centralised control of individual UAVs. Inspired by previous work in traffic and pedestrian modelling we have designed and developed three traffic management methods based on three core concepts, *waypoint-defined routes*, *floor field zones*, and UAV *platoons*.

We used waypoint-defined routes to relax the assumption that UAVs will travel along the shortest path from their origin to their destination. A port can define several alternative routes between itself and another port by using intermediary waypoints. A UAV must then visit the intermediary waypoints before reaching its destination. Since the waypoints are not an integral

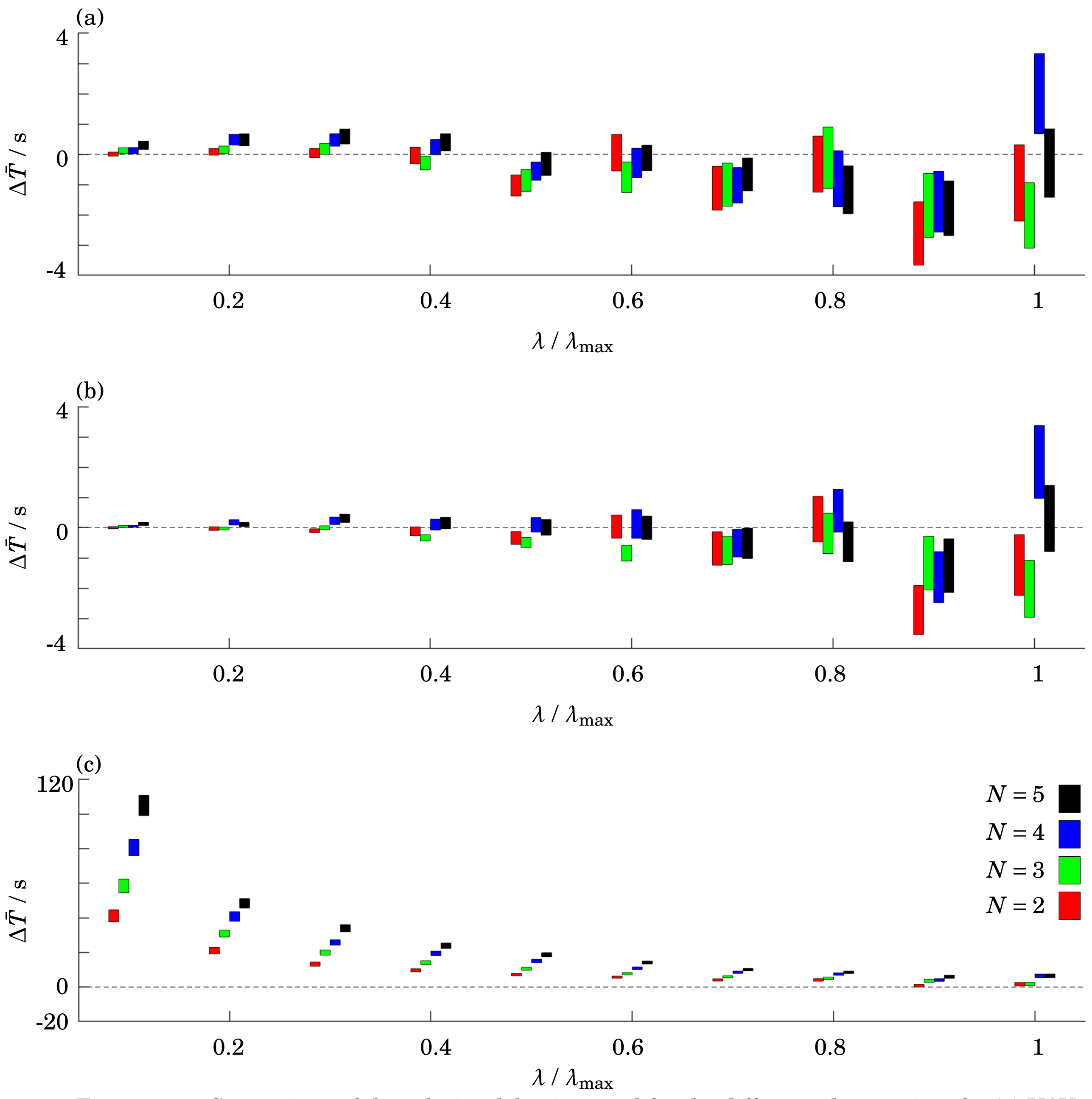


Figure 4.22: Comparison of the relative delay incurred for the different platoon sizes for (a) UAVs in the horizontal stream, (b) UAVs in the vertical stream using their simple flight times and (c) UAVs in the vertical stream using their modified flight times. From (a) and (b) we can see that platooning offers little to no benefit at low demand and can actually lead to a small increase of the mean delay. For larger demands the relationship between platoon length and relative delay becomes more complex. Some platoon sizes lead to a significant reduction in delay while others can lead to an increase in mean delay. When we consider the modified flight time for UAVs in the vertical stream in (c) we can see that platooning at small demands leads to extreme delays as UAVs spend a long time waiting for the platoon to form.

part of the UAV's mission though, as we discussed in Section 4.2.1, we used conditions other than the separation between the UAV and its current waypoint to determine whether it had 'visited' that waypoint. This ensured that UAVs did not waste time and take circuitous paths.

The waypoint-defined routes were then used to split a stream of UAV traffic into two. The idea here was to exploit the effect demonstrated in Section 3.5 where, for the same total demand, two crossings produced shorter delays than a single crossing. We first explored how the separation between the two routes ΔL_{wp} affects the relative mean delay. When the demand was high we found that when $100\,\mathrm{m} \leq \Delta L_{\mathrm{wp}} \leq 300\,\mathrm{m}$ then this resulted in a significant reductions to the average delay experienced by UAVs in both streams. However, when ΔL_{wp} is outside this range then the relative mean delay experienced by the UAVs in the vertical stream increases. This was expected of large values of ΔL_{wp} since these result in routes that are significantly longer than the ideal straight line path. For small values of ΔL_{wp} this seems to be the result of secondary avoidance manoeuvres that cascade down the vertical stream as UAVs that follow the left route are deviated in to the path of the UAVs that follow the right route. Finally, we also showed how, for higher demands, biasing the routes to the left can further improve the relative mean delay by exploiting the right-handedness of the tactical conflict management.

We used the floor field zones to collectively change the behaviour of UAVs. To do this a port defines an area of the airspace as a floor field zone which contains a velocity floor field. Any UAVs which are subject to the floor field zone then use this velocity floor field to produce the desired velocity used in their OVM, Equation 2.2, when inside the zone. The idea here is to alleviate the behaviour where UAVs in the horizontal stream can undergo large deviations due to the right-handedness of the tactical conflict management, see Figure 3.11, by changing the way the two streams interact at the crossing.

To do this we defined a floor field zone as a circular segment, with radius R_{ffz}, centred on the crossing, that would induce an anti-clockwise circular motion in the UAVs in the vertical stream. Since the floor field zone is a segment of a circle the UAVs will eventually leave the zone and can then steer themselves towards their destination. In order to reduce the negative impact on the vertical stream we setup the floor field zone such that a UAV that follows its nominal path will exit the zone with its velocity aligned with its destination which results in the shortest flight time for a given length of R_{ffz}. We swept over values for R_{ffz} and showed that the relative mean delay for either stream is not improved. However, when $R_{\mathrm{ffz}} \geq 165\,\mathrm{m}$ the longest deviations experienced by the UAVs in the horizontal stream are eliminated. Furthermore, we show that if some UAVs in the vertical stream are allowed to adopt straight line linear paths, i.e., are not subject to the floor field zone, then the relative mean delay experienced by the horizontal stream can be reduced at the cost of a small increase to that experienced by the vertical stream.

We used platooning to ensure that UAVs form small groups with small inter-UAV gaps but large inter-platoon gaps. As with the floor field zones, the idea here is to alleviate the impact of UAVs in the horizontal stream being diverted around the vertical stream. The idea here is that so long as the inter-platoon gaps are large enough a UAV in the horizontal stream will at most have to go around the UAVs in a single platoon. In order to produce a platoon with N UAVs, we do not allow UAVs to take off from their origin until there are N UAVs ready to depart. We showed that when demand is high platoons up to $N = 5$ can reduce the relative mean delay. The exact relationship between the size of N and relative mean delay however is not well understood. Also, as N increases the time spent by UAVs waiting at ports to take off also increases. This leads to longer mission times for some UAVs but this time increase may be preferred over increased flight times where UAVs are at risk of running out of fuel.

All three of these traffic control methods are applied at the port level and therefore do not require frequent communication between ports and UAVs. Nor do they rely on any physical infrastructure and therefore can be applied as needed by the ports, possibly with coordination by the overall UTM system. We expect that the ports within a system will have the ability to communicate with each other on a regular basis through the UTM system and therefore will be able to keep track of metrics such as overall demand or the flight times of UAVs between origin and destination. These metrics could then be used to dynamically apply the traffic control methods proposed in this chapter, individually or in combination.

4.5.1 Contributions

C4.1 We developed the concept of waypoint-defined routes where UAVs are assigned a sequence of intermediary waypoints in order to produce alternative routes to their destination and thus split traffic from a single origin. We showed that this method for splitting traffic can still reduce the delay compared to the single crossing case even with the inherent delay associated with non-straight line nominal paths.

C4.2 We developed the concept of floor field zones which replace the desired heading associated with the UAV's waypoint with a desired heading given by a floor field within a certain area. We then showed how this can be used to eliminate the large deviations experienced by UAVs in the horizontal stream by effectively reversing the outcome of the single crossing interactions.

C4.3 We developed the concept of UAV platoons where a UAV must wait to depart until a certain number of other UAVs are also ready. We showed that at high demands this can be used to reduce the simple flight time and thus delay experienced by UAVs in both streams but at the cost of some UAVs incurring on-pad wait times.

Chapter 5

Extensions for Future Work

Throughout Chapters 2 to 4 we have explored how different pairwise conflict avoidance schemes and traffic management methods affect the performance of UAVs. We have done this both for experimental setups of just two UAVs and for larger setups with many UAVs sharing an airspace. While we believe that this work forms a strong foundation for understanding the challenges posed by future UAV use cases, we have had to make a number of assumptions to constrain the scope of this work. For example, in terms of our dynamical model, we have only considered multi-rotor UAVs flying in a 2D airspace. Also, for our experimental setups, we have focused on the 'crossing-like' setup where two streams of UAV traffic cross perpendicular to each other.

In this chapter, we aim to show how some of these assumptions might be relaxed, and how the types of experimental setups might be expanded, in order to extend this work and thus bring it closer to real world application. We do this by showing several examples of how future work could be progressed and will show some preliminary results where possible. We split these extensions into two sections; dynamical models (Section 5.1), and experimental setups (Section 5.2).

5.1 Extensions for dynamical models

In this section we will explore possible extensions to the dynamical models we have used to describe how the UAVs navigate through the airspace. We will first lay out a simple model for fixed wing UAV dynamics. In this new model we will constrain UAVs such that they must maintain a constant speed. Next we will describe how the simple avoidance method from Section 2.3 can be extended to a 3D airspace. Finally, we will discuss how a 3D airspace might affect our choice of model for steering a UAV toward a desired direction when it is not avoiding a neighbour.

5.1.1 Fixed wing dynamical model

In Section 2.2 we presented a simple dynamical model for multi-rotor UAVs which allows a UAV to accelerate in any direction with a magnitude equal to, or less than, some maximum value. This means that a UAV's speed can change throughout the simulation, independently of its current velocity, and thus the UAVs can become stationary. While hovering in place is undesirable for multi-rotor UAVs, due to the energy inefficiency, it is possible. This is not the case for a fixed wing UAV as we will discuss below. Therefore, we will present a simple dynamical model for fixed wing UAVs that could be used in future work to explore how traffic scenarios with these types of UAVs differ from the multi-rotor scenarios explored in this thesis.

For a simple model of fixed wing UAVs, suppose that the speed of each UAV is fixed as v_{fw} and that when a UAV changes direction it does so by an acceleration $\mathbf{a}_i$ at a right angle to its velocity, thus the speed is preserved. For fixed wing UAVs, it is critical that they maintain a high enough speed to produce the required lift to keep the UAV airborne, known as the stall speed. For a given value of v_{fw} the UAV is capable of a maximum magnitude of acceleration a_{fw}. For fixed wing craft, the faster the craft is moving, the steeper tilt it can make when turning and thus is capable of greater magnitudes of acceleration. If this maximum acceleration is applied constantly throughout a manoeuvre then the craft will undergo circular motion and trace out a circle with diameter $D_{\mathrm{turn}} = v_{\mathrm{fw}}^2/a_{\mathrm{fw}}$. Therefore in an ideal scenario where a_{fw} is infinitely large the fixed wing UAV would turn on the spot to instantly face its desired heading $\hat{\mathbf{t}}_i$ and then follow a straight line path until the desired heading changes. Therefore, in this simple model of fixed wing UAV dynamics we apply a_{fw} as a constant so that in any manoeuvre the UAV has the tightest turning circle possible. Also note that, given the definition of D_{turn}, a UAV could theoretically tighten its turning circle by reducing its speed. Given the dependence of a_{fw} on v_{fw} this might not always be the case and other factors, such as the desire for short mission times, may incentivise larger values for v_{fw}. In a real world implementation the UTM system will need to balance the desire for short mission times with the ability to react and manoeuvre in a crowded airspace.

Given this model for fixed wing UAVs we suggest a simple control scheme where a UAV will try to align itself with its destination by choosing a left or right turn that traces out the smallest segment of a circle possible. We illustrate what the resulting behaviour would look like in Figure 5.1 which shows the trajectory undertaken by a fixed wing UAV in simulation where the UAV steers itself to visit four waypoints placed in the world. We can see how the UAV traces out a segment of a circle while turning until it points towards its next waypoint. Once the velocity is aligned with the waypoint the UAV stops applying the steering acceleration and adopts a linear trajectory until it reaches its waypoint. Once the waypoint is reached the UAV immediately begins to turn to face the next waypoint. Under this simple scheme it is important that the waypoints are further apart than D_{turn}. If a waypoint is within D_{turn} of the previous waypoint then it will be unreachable, as the UAV adopts a circular trajectory which it can not

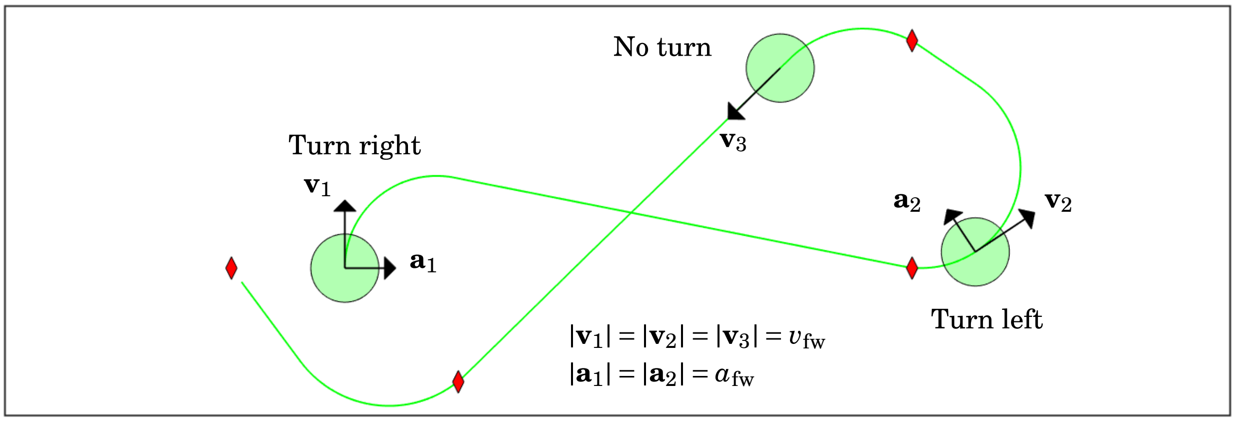

No turn
Turn right
$\mathbf{v}_1$
$\mathbf{a}_1$
$\mathbf{v}_3$
$\mathbf{a}_2$
$\mathbf{v}_2$
Turn left
$|\mathbf{v}_1| = |\mathbf{v}_2| = |\mathbf{v}_3| = v_{\text{fw}}$
$|\mathbf{a}_1| = |\mathbf{a}_2| = a_{\text{fw}}$

Figure 5.1: The trajectory of a fixed wing UAV, shown by the green curve, undertaken in simulation using our proposed model and steering scheme. The UAV has to visit each of the waypoints (red diamonds) in turn and may only steer by turning to the left or right through the application of an acceleration that is always perpendicular to the velocity. We show the UAV (green circle) at three points along its trajectory, first when it makes a right turn, second when it makes a left turn and lastly when its velocity is aligned with a waypoint and therefore does not apply any turn. All of the waypoints are further apart than D_{turn} to ensure that they are reachable given the simple steering scheme.

leave. This can be resolved in practice by applying an extra layer to the UAV's behaviour. For example, if a UAV is closer than D_{turn} to its next waypoint, it could adopt a linear trajectory for a set period of time until it is further than D_{turn}, at which point it can begin to turn towards its waypoint as before.

5.1.2 Avoidance in 3D space

Throughout this thesis we have assumed that UAVs will be restricted to operating below some maximum altitude, as is the case under current regulations [20, 34, 35]. We further assume that the UAVs will try to fly at this maximum altitude whenever possible to maintain maximum separation with the ground. Given these two assumptions and a multi-rotor UAV's relative inefficiency while ascending, we have modelled the UAVs in a 2D plane. Therefore, one obvious extension for future work is to develop the conflict avoidance in 3D. In this subsection then, we will show how the velocity obstacle method can be extended to a 3D airspace and show two UAVs avoiding each other in simulation using a simple implementation of this new 3D avoidance.

Consider two UAVs i and j in 3D space, Figure 5.2(a). UAV i, which we consider 'ego', has a position $\mathbf{r}_i$ and velocity $\mathbf{v}_i$. We maintain the separation condition that the relative displacement between the UAVs, $|\mathbf{r}_{i,j}|$, should be greater than or equal to some minimum value S, see Equation 2.5. In 3D this corresponds to a sphere with radius S centred on $\mathbf{r}_j$ that UAV i will try to avoid.

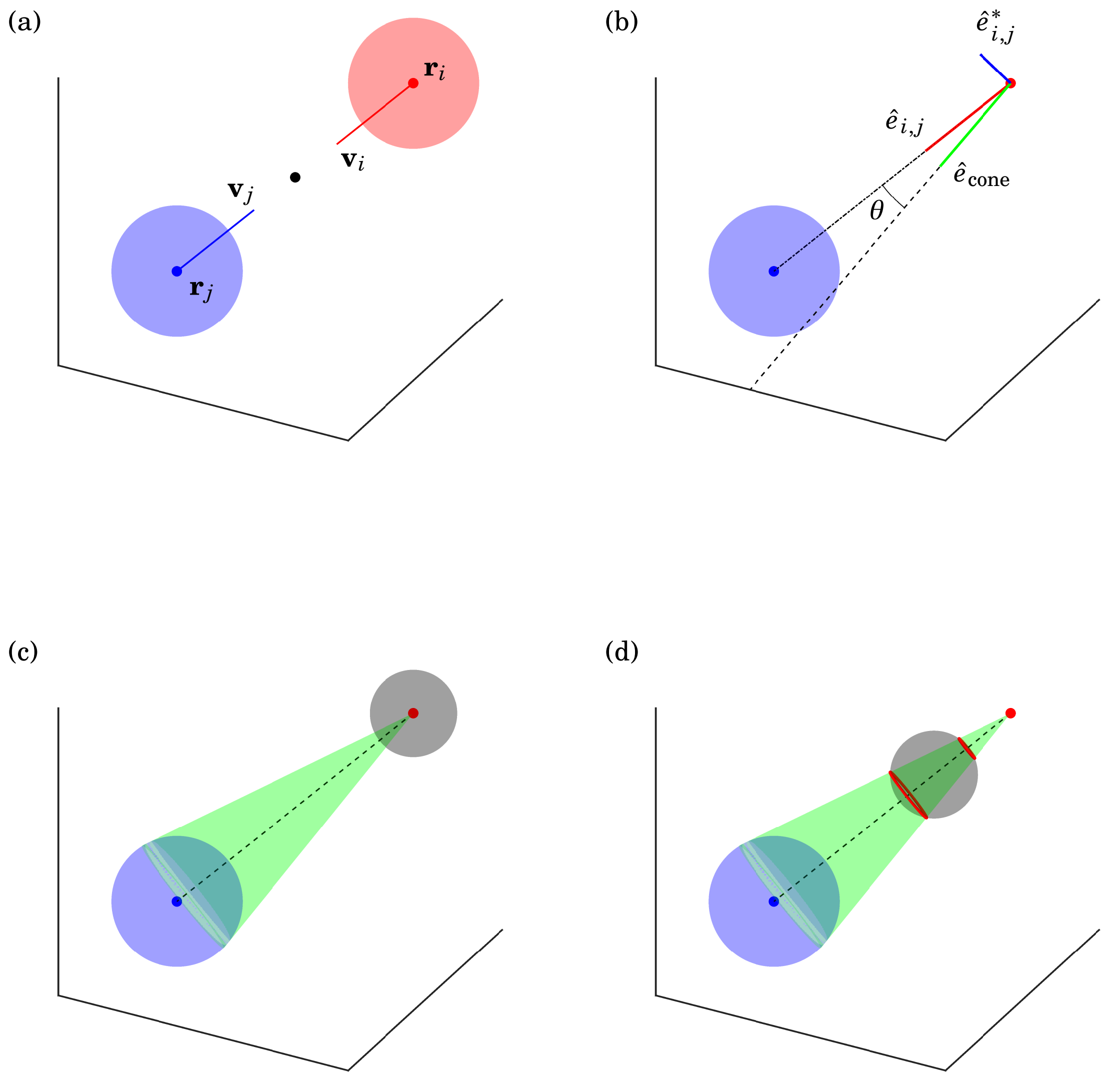


Figure 5.2: (a) Two UAVs are flying toward each other in 3D space. If their velocities (solid lines) do not change, then they will collide at the halfway point between them (black dot). (b) We construct the velocity obstacle cone by finding a unit vector $\hat{e}_{\text{cone}}$ that lies on its surface. We do this by rotating the unit vector $\hat{e}_{i,j}$ around the unit vector $\hat{e}^*_{i,j}$ by some angle θ. Note that $\hat{e}_{i,j}$ and $\hat{e}^*_{i,j}$ are perpendicular to each other. We can find θ by using simple trigonometry since the velocity obstacle cone is tangential to the blue sphere. (c) The velocity obstacle cone is shown by the green surface. Also, the constraint that $|\mathbf{v}_i^j| = v_{\text{CS}}$ defines the grey sphere centered on the red UAV. (d) Given the constraint on $\mathbf{v}_i^j$, we can then find the relative velocities which result in a separation equal to S at the closest point of approach. These velocities are where the grey sphere and green cone intersect (shown in red).

To construct the velocity obstacle cone in 3D we can again use simple trigonometry to find the angle θ between the unit vector $\hat{e}_{i,j}$, which points from the center of UAV i to the center of UAV j, and the surface of the cone. Since the velocity obstacle cone is tangential to the sphere centred on UAV j we can use the expression $\sin(\theta) = S/|\mathbf{r}_{i,j}|$, Figure 5.2(b). We can then obtain a unit vector that lies on the surface of the cone by rotating the vector $\hat{e}_{i,j}$ by the angle θ around the vector

$$\hat{e}^*_{i,j} = \frac{[0, \hat{e}_{i,j}(z), -\hat{e}_{i,j}(y)]^T}{|[0, \hat{e}_{i,j}(z), -\hat{e}_{i,j}(y)]^T|}, \tag{5.1}$$

which are perpendicular to each other. Note that this requires the z and y components of $\hat{e}_{i,j}$ to be non-zero but other formulations can be used if this is the case. The result is the unit vector $\hat{e}_{\text{cone}}$, which can in turn be rotated around $\hat{e}_{i,j}$ to produce the velocity obstacle cone, the green surface in Figure 5.2(c). As in the 2D case, if the relative velocity $\mathbf{v}_{i,j}$ lies inside this cone, then the UAVs will come into conflict with each other at some point in the future. Also in keeping with the 2D case, we maintain the assumption that $\mathbf{v}_j$ will remain constant and we constrain the avoidance velocity such that $|\mathbf{v}^j_i| = v_{\text{CS}}$. The latter constraint defines the grey sphere in Figure 5.2(c).

In Figure 5.2(d) the centre of the grey sphere is translated by $-\mathbf{v}_j$ and shows all possible relative velocities which obey the constraint we placed on $|\mathbf{v}^j_i|$. Therefore, the relative velocities where the grey sphere and green cone intersect are those relative velocities which ensure that the UAVs will pass each other with the minimum separation S at their closest point of approach, shown in Figure 5.2(d) in red. This is where the 2D and 3D velocity obstacle avoidance deviate from one another. In the 2D case, there were at most four possible avoidance velocities. These corresponded to a forward left or right turn and a reversing left or right turn. In the 3D case we have two curves which correspond to a set of forward velocities and a set of reversing velocities. Both of these sets can be continuous, as is the case in Figure 5.2(d). We are therefore faced with a much larger (infinite) set of possible avoidance velocities and must design an appropriate way to choose $\mathbf{v}^j_i$ from them.

For simplicity, we will demonstrate how we can extend the simple avoidance method, where UAVs always turn to their right, to the 3D case. In order to do this we first define a right and left hand turn as one where the UAV's velocity is rotated about the z axis by $-\pi/2$ or $\pi/2$ respectively. This results in two new vectors $\mathbf{v}^{\text{R}}_i$ and $\mathbf{v}^{\text{L}}_i$, see the blue and red dashed lines in Figure 5.3(a). We can then work out the distance between any point on either curve of possible avoidance velocities and these two new vectors. These distances can then be used to group those possible avoidance velocities in to those that result in the UAV turning to its right (blue portion of the curves in Figure 5.3(a)) or to its left (red portion). Again for simplicity, we choose from amongst the possible avoidance velocities which result in the UAV turning to its right by choosing the one with the smallest distance to $\mathbf{v}^{\text{R}}_i$. In Figure 5.3(b) we show the simulated trajectories of two UAVs that implemented this 3D simple avoidance behaviour.

This method represents only one possible way to choose an avoidance velocity and we suspect there are many others that would result in better performance. For example, if the UAVs must

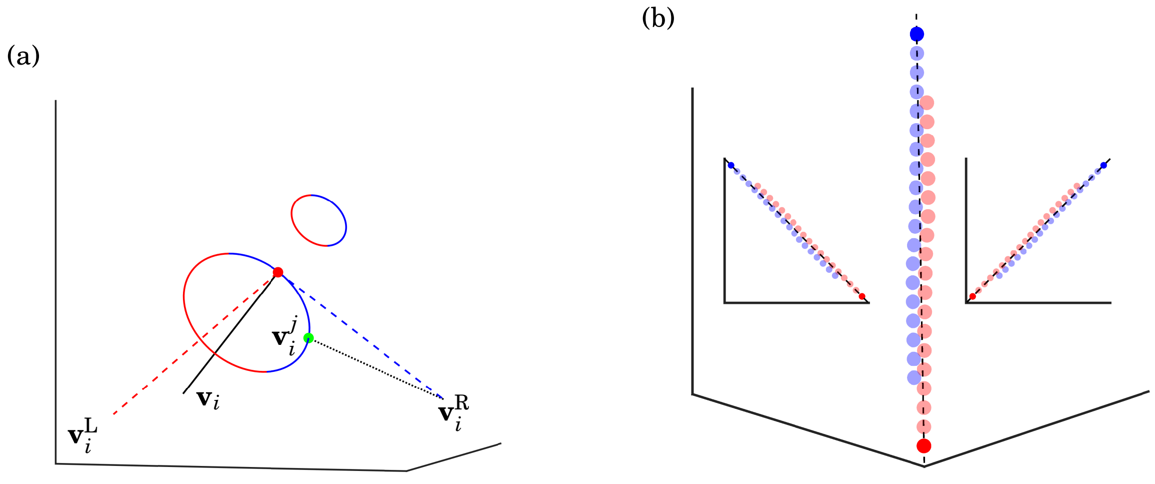


Figure 5.3: (a) UAV i must choose its avoidance velocity $\mathbf{v}_i^j$ from two sets of possible avoidance velocities. We adapt the simple avoidance method, presented in Section 2.3, to 3D by defining the vectors $\mathbf{v}_i^L$ and $\mathbf{v}_i^R$. These correspond to rotating the UAV's velocity $\mathbf{v}_i$ around the z-axis by $\pi/2$ or $-\pi/2$ respectively. Here we choose $\mathbf{v}_i^j$ (green dot) by selecting the point on either circle that is closest to $\mathbf{v}_i^R$. (b) The simulated trajectories of two UAVs using this 3D simple avoidance. Three different viewing angles are shown. The start positions are shown by the solid red and blue spheres.

operate in a tight layer of airspace we might prefer to choose an avoidance velocity that tries to maintain the UAV's current altitude. We therefore suggest that exploring this design space would be an interesting area for future work.

5.1.3 Waypoint following in 3D space

Now that we have described an extension of the conflict avoidance method to 3D, it is natural for future work to consider UAV traffic scenarios with a 3D airspace. However, for the various reasons discussed throughout this thesis, it is unlikely to be a completely unrestricted airspace. A simple model of a future airspace then is one which consists of three important altitudes; a ceiling above which UAVs should not ascend, a floor below which UAVs should not descend, and a cruising altitude which UAVs should try to maintain in the absence of avoidance. It is also reasonable to assume that in such an airspace the UAVs will ascend to the cruising altitude above their port before flying toward their waypoint. Furthermore, we suggest that waypoints should be placed in the plane of the cruising altitude. This would ensure that UAVs which have been displaced from the cruising altitude return to it if they implement a 3D version of the optimal velocity model (OVM) from Equation 2.2, Figure 5.4(a).

A potential problem with the OVM in 3D though arises when the horizontal distance to a waypoint is much larger than the vertical displacement of the UAV. Under such circumstances the UAV will take a long time to return to the cruising altitude. One solution to this would be to

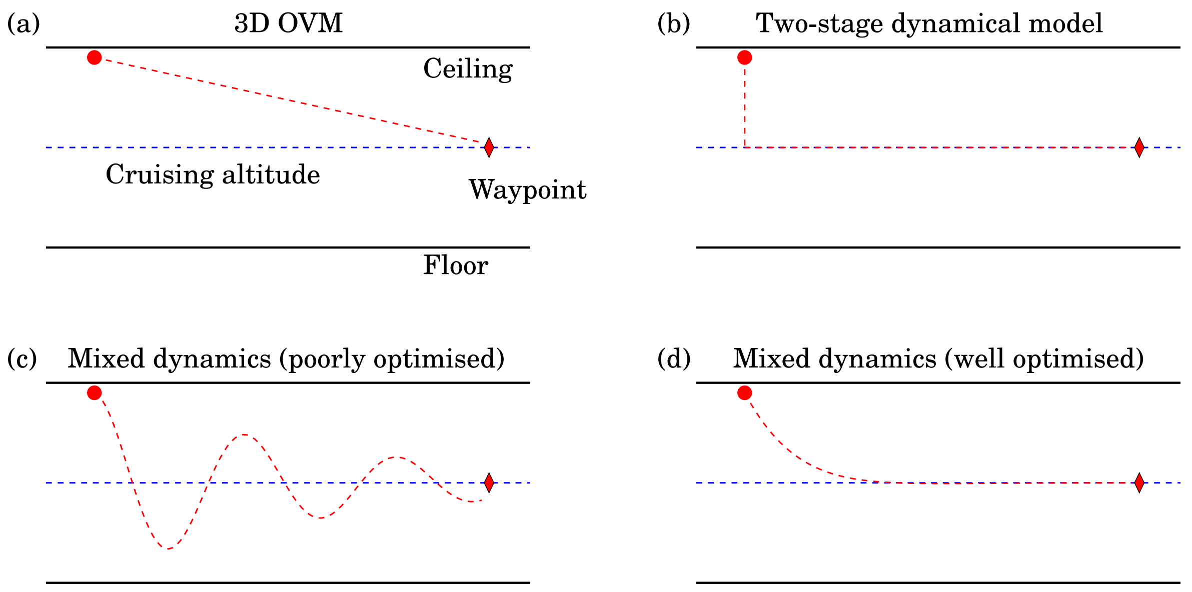

(a)
3D OVM
Ceiling
Cruising altitude
Waypoint
Floor
(b)
Two-stage dynamical model
(c)
Mixed dynamics (poorly optimised)
(d)
Mixed dynamics (well optimised)

Figure 5.4: Trajectory of a UAV that has been displaced from its cruising altitude when it implements (a) a 3D extension of the OVM described in Equation 2.2, (b) a 'two-stage' dynamical model where it first descends back to the cruising altitude and then implements the OVM, (c) a poorly optimised mixed dynamical model which introduces unwanted oscillatory motion, (d) a well optimised mixed dynamical model which provides a balance between returning to the cruising altitude while also moving toward its waypoint.

adopt a two-stage dynamical model. That is, when a UAV has been displaced from the cruising altitude it first tries to ascend or descend directly back to the cruising altitude and only then applies the OVM, Figure 5.4(b). While this two-stage dynamical model is simple, we suspect it may be unsuitable in some situations, for example when traffic demand is high. This is because under such conditions a UAV may be unable to return to the cruising altitude as other UAVs pass below or above it. This could lead to a UAV effectively becoming stuck.

We suggest that a mixed dynamical model would be more appropriate. That is a dynamical model which results in behaviour that is a mix between that demonstrated in Figure 5.4(a) and (b). This can be achieved in a variety of ways but it is important to ensure that we do not introduce unwanted oscillatory behaviour, Figure 5.4(c), which could negatively impact the energy consumption of a UAV. Ideally we want UAVs to exhibit behaviour like that in Figure 5.4(d), where they quickly return to the cruising altitude and then smoothly transition back to the OVM when the UAV is close to the cruising altitude. The development of this mixed dynamical model represents another potential area for future work.

5.2 Extensions for experimental setups

In this section we will explore possible extensions to the experimental setups used in Chapters 3 and 4. We will first explore how we can simulate more complex crossing setups where multiple streams of UAV traffic may cross at a single point with different angles of approach. We will also explore how the traffic management methods from Chapter 4 can be applied to these complex crossings as well as how they might be combined to produce further reductions in delay. We will then discuss how we might draw on existing work which applies ideas from ground traffic modelling to future UAV traffic scenarios.

5.2.1 Complex crossings

Up to this point, this thesis has made use of a base case where two streams of UAVs approach one another at right angles, what we now call a ‘simple crossing’. In Section 3.5 and Chapter 4 we began to explore variations of this simple crossing and developed methods that could improve the performance of UAVs travelling through such a system. One potentially rich area for future work then is to take these traffic management methods and apply them to a more complex crossing setup. In this subsection we will present a complex crossing setup where multiple streams approach one another at a variety of angles. We will then present some preliminary results of simulations where some of the traffic management methods developed in Chapter 4 are applied to the complex crossing.

Here we will consider a complex crossing where, in the absence of any traffic management, all traffic is routed through a single point. It should be noted that such a crossing is unlikely to occur naturally. Consider a scenario where origin-destination pairs are placed in an empty airspace at random. The chances that any two routes cross at exactly the same point are small. However, it is possible that multiple different routes from multiple crossings near each other. Therefore we can consider the complex crossing presented here as a sort of simplification. Future work then could consider these more natural multi-stream crossings and compare them with the complex crossing presented here, for example, in order to determine if there is any benefit in routing traffic that form multiple crossings such that they are consolidated in to one complex crossing.

We start by placing two ports at positions $\mathbf{p}_1$ and $\mathbf{p}_2$ with some distance L between them, similar to the simple crossing. We can then define a number of angles θ_n which we use to add further pairs of ports. The complex crossing we will use in this subsection consists of three pairs of ports, Figure 5.5(a). Also like the simple crossing, the distance between each pair of ports is kept constant and UAVs will only travel between ports in the same pair. This is done so that the nominal flight time of all UAVs is the same and we can easily compare the delay each UAV experiences. Unlike the simple crossing we will allow each port to act as both an origin and destination. Therefore the demand matrix $\mathbf{M}$ for the setup in Figure 5.5(a) is,

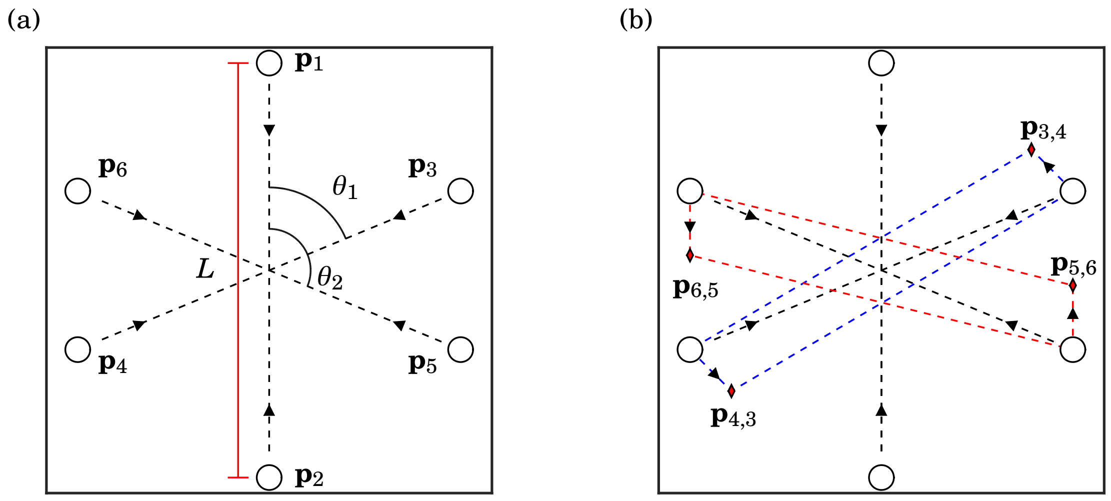


Figure 5.5: The complex crossing setup for (a) the base case where no traffic management method is applied, and (b) when waypoint (diamonds) defined routes are used to split the traffic in to multiple streams. Unlike in previous setups, each port in a pair serves as both an origin and a destination. The values and dimensions of the parameters we use to simulate this complex crossing are given in Table 5.1.

$$\mathbf{M} = \begin{bmatrix} 0 & \lambda & 0 & 0 & 0 & 0 \\ \lambda & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \lambda & 0 & 0 \\ 0 & 0 & \lambda & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \lambda \\ 0 & 0 & 0 & 0 & \lambda & 0 \end{bmatrix}, \tag{5.2}$$

which means there are six traffic flows passing through the central crossing.

In this way the complex crossing is closer to the envisioned delivery use case since it is reasonable to assume that UAVs will have to return to their origin after completing a delivery. However, this kind of setup is still much simpler than the sort of traffic scenarios envisioned for UAV delivery. As such, the setup in Figure 5.5(a) is primarily intended to explore how factors such as the number of, or angle between, streams affects the previously developed traffic management methods. Future work may further develop this by using existing infrastructure to describe a more realistic traffic scenario which consists of many complex crossings. In Section 5.2.2 we will explore one such setup inspired by other work in the UAV literature. Another possible example would be to imagine a scenario where UAVs are used to deliver documents between university buildings which can then be used to describe the location of origins and destinations in a simulation.

We will simulate four versions of the complex crossing setup in Figure 5.5(a). First, a 'base case' version where no traffic management methods are applied. This will serve as a baseline against which we will measure the other three. See Table 5.1 for the value of the parameters we

Parameter	**Value**	**Units**
Distance between port pairs (L)	2000	m
Angle between first and second pair (θ_1)	$3\pi/8$	
Angle between first and third pair (θ_2)	$5\pi/8$	
Mean demand (λ)	$0.19 \approx 0.8\lambda_{\max}$	s^{-1}

Table 5.1: Parameters and their dimensional values used for simulation results.

used. We set the distance between pairs of ports to $L = 2\,\text{km}$ since this is the same as the simple crossing, thus allowing us to compare delays directly. We will then simulate the complex crossing when platooning or waypoint defined routes are used. We only apply platooning to the UAVs travelling between $\mathbf{p}_1$ and $\mathbf{p}_2$, in both directions. We set the platoon length to three UAVs. For the waypoint defined routes we specify waypoints at the positions $\mathbf{p}_{3,4}$, $\mathbf{p}_{4,3}$, $\mathbf{p}_{5,6}$, and $\mathbf{p}_{6,5}$, which correspond to the red diamonds in Figure 5.5(b). We then assign each UAV that takes off from one of these four ports either the route defined by the nearest waypoint (red or blue dashed lines) or allow them to follow the original straight line path. This results in nine separate crossings. Finally, we will simulate a complex crossing where both of these traffic management methods are used at the same time.

As with previous results, we simulate each complex crossing setup until at least 1,000 UAVs have departed from each origin. We then measure the flight time of each UAV and produce a delay based on this and the ideal flight time that would be achieved if the straight line path was taken without deviation. From these individual delays we produce an estimate for the mean delay $\bar{T}$ by performing the bootstrapping process described in Section 3.3. Here, we perform this bootstrapping process on all UAVs in the system, Figure 5.6. In future work we could separate the UAVs by stream as we have done in Chapter 3 to provide more insight.

From Figure 5.6 we see that the mean delay experienced by UAVs in the base case complex crossing is much worse than the simple crossing when the demand is the same at each origin, about 7 s and 31 s respectively. There is also little to no change in the mean delay when only the platooning or waypoint defined routing traffic management methods are applied on their own. However, when both methods are applied there is a small but significant drop in the mean delay. Another direction for future work then may be in exploring how these traffic management methods can be combined to maximise the benefit they provide. This might be in the way that they exploit emergent behaviour of UAVs, such as the traffic 'smearing' effect we demonstrated in Section 3.5. However, it might also be the case that we can limit the negative effects of the traffic management methods when they are combined. For example, allowing diverted UAVs to take shorter alternate routes by enforcing platooning on other streams, and thus achieving the same, or better, reductions in delay.

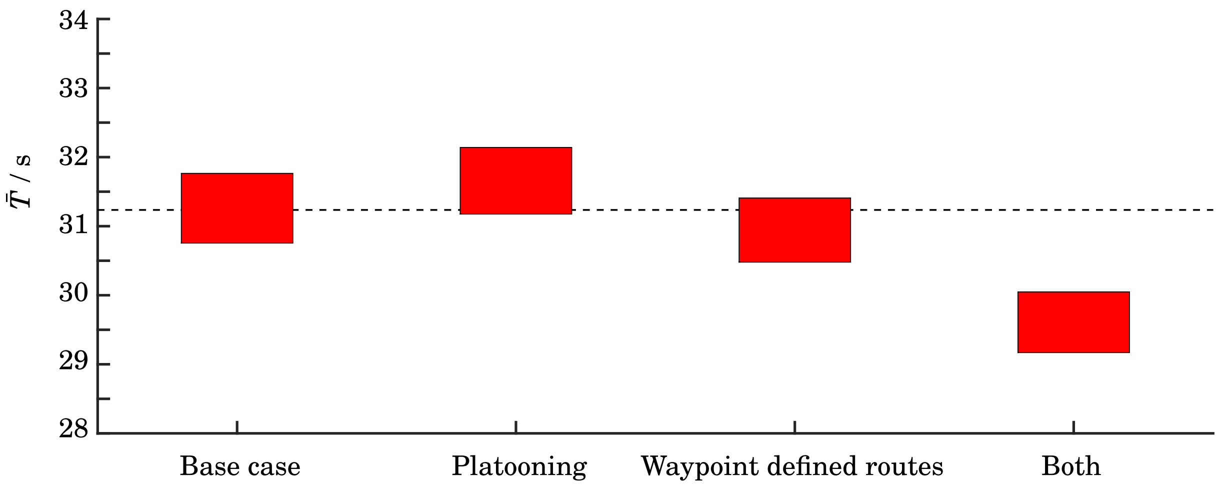


Figure 5.6: Bootstrapped estimates of the mean delay experienced by UAVs for the complex crossing setup shown in Figure 5.5(a) when different combinations of traffic management methods are applied. When platooning or waypoint defined routes are applied individually there is little to no change in the mean delay. However, when both traffic management methods are applieds at the same time there is a small but significant drop in delay.

5.2.2 Realistic traffic scenarios

The complex crossing setup described in Section 5.2.1 is useful in helping us to understand how the various traffic management methods are affected by different aspects of the setup, such as the angle between streams. However, it is still much smaller than the sorts of traffic scenarios envisioned for future UAVs, both in terms of the number of ports, and the variety of routes. It will therefore be necessary to explore even larger and more complex setups.

One of the potential questions we might want to answer about these larger setups is how the capacity changes in response to demand and the number of ports. As discussed in Section 3.1, there is a growing literature where ideas from the macroscopic fundamental diagram (MFD) for road traffic are being applied to UAVs. Therefore, in this subsection we will recreate a similar setup to that used by Cummings *et al.* [100], where several origins and destinations are placed on the border of some experimental arena. In Figure 5.7(a) sixteen origins are placed on the circumference of a circular arena defined by a diameter L. Each origin is then assigned the same total demand λ which is shared equally between the destinations. We suggest that in such a setup we will not connect nearest neighbours since the streams that would result from this do not form a crossing with any other stream. This experimental setup can then be implemented using the simulation architecture used throughout this thesis. See Appendix A for an overview of this simulation architecture.

In order to analyse the results in terms of a delay to the travel time, as we have done throughout this thesis, we now need to define a relative delay for each UAV

$$T_i^* = t_i^{\text{actual}}/t_i^{\text{ideal}}, \tag{5.3}$$

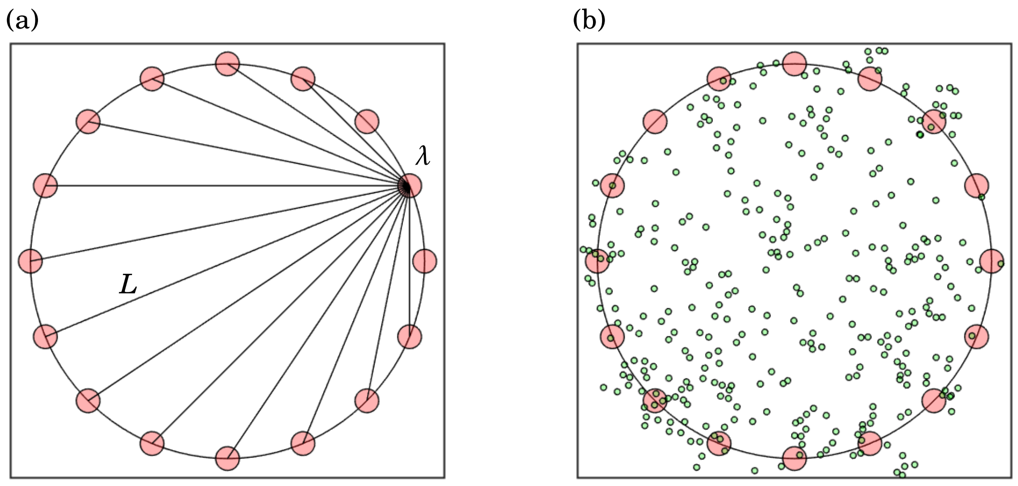


Figure 5.7: (a) Experimental setup with sixteen origins placed on the circumference of a circle with a radius of 1 km such that the longest straight line path between two ports has a distance equal to L as defined in Table 3.1. Each origin has a total demand λ which is set at the start of an experiment and is shared equally between its neighbours except for its two nearest neighbours. (b) Snapshot of a simulation run.

where t_i^{ideal} is the time it would take for a UAV to travel between its O-D pair with a constant speed and no avoidance manoeuvres, while t_i^{actual} is the actual flight time that it experiences in simulation. As a first step we simulated a number of demand cases using this setup for which the preliminary results are presented in Figure 5.8. A similar degradation of performance is observed for the single crossing in Figure 3.7(a) though it is more pronounced for this more complex setup. For example, for the largest value of λ used with this new setup, a UAV on average experiences a flight time that is more than double its ideal flight time. This result further demonstrates the need for traffic management at higher demands in order to supplement low-level conflict avoidance.

Another challenge presented by setups with many ports, such as that in Figure 5.7(a), is knowing how and where to apply the traffic management methods developed in Chapter 4. One possible approach is to use techniques such as machine learning or evolutionary algorithms to, for example, generate waypoints and alternative routes at random and then iteratively perform simulations to find configurations that improve performance. However, this would require a more efficient simulation architecture that is capable of running simulations much faster than real time.

5.3 Conclusion

In this chapter we have presented a number of examples for how the work in Chapters 3 and 4 could be extended. In Section 5.1, we considered extensions to the dynamical models. We first presented a new dynamical model for fixed wing UAVs based on a ‘constant speed’ constraint.

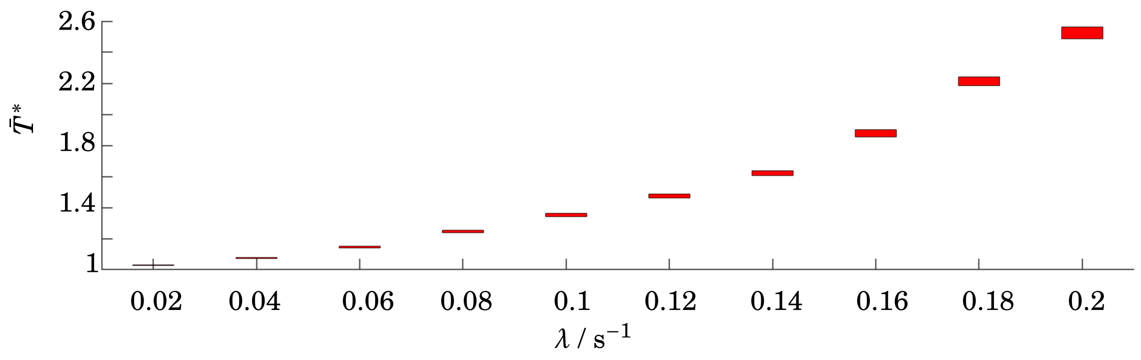


Figure 5.8: Relationship between the average fractional delay and the demand. This is similar to the behaviour we observed for a single crossing, see Figure 3.7. However, the increased complexity of the setup has led to a more pronounced degradation of performance. At the largest demand value used here the average UAV experiences a flight time that is more than double what it would experience without avoidance manoeuvres.

Then we considered how the velocity obstacle conflict avoidance method could be extended to 3D and what effect this would have on the optimal velocity model we use to steer UAVs toward their waypoint. In Section 5.2 we presented more complex traffic scenarios with some preliminary results from simulations. Importantly we demonstrated how combining two traffic management methods (waypoint routing and platooning) from Chapter 4 could improve system performance when neither method proved effective on its own. This is by no means an exhaustive list of possible extensions for this work but highlights some of the key areas where further development is needed in order to apply these avoidance and traffic management methods to real UAV fleets.

Chapter 6

Conclusion

The current vision for unmanned traffic management (UTM) for UAVs is based on the UTM service provider (UTM-SP) model. Under this model the UTM-SPs will need to provide a wide range of services, as discussed in Section 1.1, that will require many different functionalities. This thesis has been primarily concerned with designing and developing the functionality that will enable UAVs to maintain a safe separation in flight and the functionality that will enable the UTM system to improve traffic performance by changing the traffic setup. We have designed these functionalities with simplicity and scalability as primary design principles. Of course, we expect that a UTM system in reality will also need other functionality such as enabling communication between stakeholders, ensuring cyber security, maintaining the registration of both the UAVs and their operators etc.

We first designed a simple conflict avoidance method and then developed it in to a hybrid conflict avoidance method that allows a pair of UAVs to maintain a safe separation by choosing the most appropriate avoidance behaviour, see Chapter 2. We then explored how these conflict avoidance methods performed in a setup with many more simultaneous UAVs, see Chapter 3. Finally we designed three different high-level traffic management methods which we showed could be used to improve the system performance without directly controlling individual UAVs or altering the underlying conflict avoidance methods, see Chapter 4.

We will now re-state the research questions we proposed in Section 1.2.1 and discuss how they were addressed.

RQ1 How can two UAVs be made to avoid each other safely while also ensuring efficiency and fairness?

We designed a simple conflict avoidance method where a UAV uses a velocity obstacle method to generate several possible avoidance headings based on the current position and velocity of itself and its neighbour. We call this first avoidance method ‘simple’ because it dictates that UAVs will always select the avoidance heading which involves the smallest right-hand turn. We then simulated pairs of UAVs on a variety of initial paths and showed that this simple avoidance method does allow the UAVs to pass each other safely such that a minimum safe separation between the UAVs is maintained throughout the manoeuvre. We also showed that the resulting manoeuvre is fair (delay is split evenly) when the two UAVs approach each other head on. However, as the angle of approach between the UAVs tends to zero, the efficiency and fairness of the resulting manoeuvre worsens, as one of the UAVs experiences a much longer delay compared with the other. We therefore designed a hybrid avoidance method which allows a UAV to choose from three avoidance behaviours based on the current angle of approach and a relative distance. This ensures a more equitable share of the delay between UAVs for small angles of approach and the total delay is smaller than that produced by the simple avoidance method, thus improving the efficiency of the manoeuvre.

RQ2 How does the performance of a UAV traffic scenario, where UAVs implement some tactical conflict avoidance, depend on traffic demand and other factors?

We set up experiments with two origin-destination (O-D) pairs in simulation which generate two streams of UAVs. These streams cross at a point and therefore produce a sort of crossroads in the sky. UAVs will come on to conflict course with each other in the vicinity of this crossing and therefore will undergo avoidance manoeuvres. We first demonstrated that the hybrid avoidance method was more efficient (UAVs incurred a shorter delay on average) and fairer (produced a smaller Gini coefficient) than the simple avoidance method due to the effects of secondary avoidance interactions. Furthermore, we showed how both the efficiency and fairness of a crossing degrades non-linearly as the demand on the system increases to a theoretical maximum. Based on this behaviour, we developed a new experimental setup where one O-D pair is split in two and the demand is shared between them. This results in a system with two parallel streams of UAV traffic and two crossings. For high demands, this resulted in a much smaller average delay compared with a single crossing with the same total demand.

RQ3 How can we supplement a tactical conflict avoidance scheme with other traffic management methods in order to improve the performance of a UAV traffic crossing?

We designed and investigated three potential high-level traffic management methods that can be used to change the experimental setup and thus improve performance. For all three methods, the UAVs continue to use the hybrid conflict avoidance method to resolve conflicts that arise in mid-air. The first method introduced intermediary waypoints to define alternative routes between O-D pairs that UAVs could be assigned to. In doing so, we exploited the behaviour observed in the two-crossing setup, where sharing the demand between multiple crossings shortens the average delay experienced by UAVs. However, this sharing is now at the cost of longer nominal flight paths, with inherent delay due to the UAVs no longer taking the shortest straight line path between each O-D pair. The second method introduced the idea of floor field zones. When a UAV is inside one of these zones, it tries to align its velocity with a system-defined floor field. We thus changed the collective behaviour of selected UAVs and demonstrated how this intervention could be used to shorten the average delay. The third and final method forced UAVs to wait for takeoff until a certain number of other UAVs were ready to take off at the same location. Consequently the affected UAVs formed platoons. We then showed how the resulting inter-platoon gaps could reduce the average delay at high demands, but at low demands some UAVs experienced long waiting times at their origin, prior to departure.

While we have provided answers to all three of the research questions we have posed, there are many opportunities to further develop the work in this thesis. At the more technical level some extensions could include, for example, developing the conflict avoidance methods for use by fixed wing UAVs, or conducting experiments using heterogeneous fleets of UAVs (as one might expect to be realised in a future with a wide variety of UAV operator entities). Note, if any of the conflict avoidance or traffic management methods developed in this thesis were to be used in a real UTM system, then some experiments with physical UAVs would also be necessary, to achieve a fuller verification and validation of our methods that simulation alone cannot achieve.

In this thesis we have only considered small traffic setups, with one or two crossings, which we have considered to be a sort of 'building block' element for a future UAV airspace. In contrast, we expect that in real-world setups, there will be many O-D pairs and therefore very many crossings with different layouts, e.g., differing angles between streams, varying numbers of streams per crossing etc. Future work then should also focus on exploring these larger, more complex setups in order to determine to what degree the learnings about the small setups presented here are still valid.

In conclusion, this thesis has begun to develop some of the traffic management capabilities that a UTM system will need in order to ensure that an airspace is safe, efficient and fair. We have shown that, for small O-D networks, when the demand is low, UAVs should be able to travel between O-D pairs using only a distributed conflict avoidance method. However, as demand increases and performance degrades, a UTM system should be able to mitigate this through the use of high-level traffic management methods like those we have designed as an answer to **RQ3**. We have demonstrated three proof of concept high-level traffic management methods that change the traffic setup. We hope that this work can be developed further for use in real UTM systems in order to enable large scale UAV operations in the coming decades.

APPENDIX A

Simulation architecture and implementation

Throughout this thesis we have presented results from experiments which were carried out in simulations. These simulations in turn have implemented the various models we described in Chapters 2 and 3. While some of these experiments have been quite different from each other, for example a pair of UAVs used in the experiments in Chapter 2 compared with many simultaneous UAVs in Chapters 3 and 4, they make use of a common simulation architecture that I developed as part of my thesis programme. In this appendix then we aim to provide more details on this simulation architecture, our design philosophy and some of the decisions made in the implementation of the architecture.

Before we discuss the generic simulation architecture it is important to understand what we need the simulations to do. The aim of this thesis is to "develop a set of traffic management methods for large scale, low flight ceiling UAV operations...", as laid out in Section 1.2 along with the core assumptions we make about the system we have modelled. As such, we will need to simulate many UAVs at the same time as they move through the space, which in turn will require the implementation of different dynamical models. Since we have developed and iterated different traffic management methods, from pairwise avoidance to larger scale interventions, it is important that we are able to change between these methods easily and modify them without interference with other parts of the simulator.

We also made the assumption that "all UAVs will be engaged in delivering small packages between fixed locations", see page 12. We therefore will also have to simulate what we call 'ports' from which the UAVs will depart and land at, see Section 3.2 for details on how we model these. The UAVs and ports then are the two types of agents that we will need to define and which will act (i.e., own execution) in the simulation. We provide an overview of the different attributes that define the UAVs and ports respectively in Table A.1 and A.2, along with the corresponding data type.

Property	Id	Velocity	Position	Time at takeoff	Waypoint	Parameters
Definition	N	$[v_x, v_y]$	$[x, y]$	t_0	$[x^*, y^*]$	$[P_1, P_2 ... P_n]$

Table A.1: Properties that define the UAV agent. The ID, parameter, and takeoff time properties are static and do not change throughout the simulation. The velocity and position properties are updated dynamically. The waypoint property is only dynamic if intermediary waypoints are used to give a UAV a non-straight line path to its destination.

Property	Id	Position	Demand	Queue	Parameters
Definition	N	$[x, y]$	$[\lambda_1, \lambda_2 ... \lambda_N]$	I_N	$[P_1, P_2 ... P_n]$

Table A.2: Properties that define the port agent. In this thesis, all properties are static, with the exception of the queue which changes both as UAVs are generated at the port and when the UAVs take off.

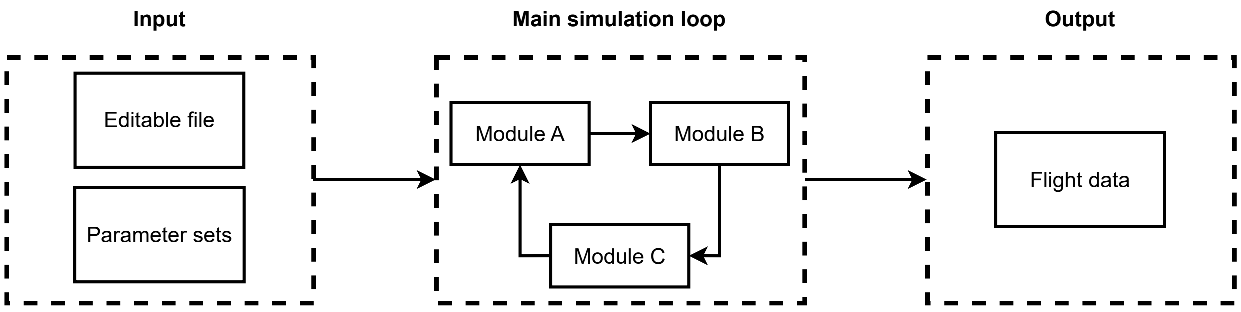


Figure A.1: Generic simulation architecture which consists of three stages; an input, main simulation loop, and output stage. The input stage corresponds to a script file which results in a single (editable file) or multiple (parameter sets) simulation runs. The main simulation loop is where the various models are implemented and actions are undertaken by the various agents. The loop consists of different modules which may be selected during the input stage. The output stage is where the relevant data produced by the main simulation loop is packaged ready to be analysed. The main data set used throughout this thesis is the flight data produced by UAV agents.

With these needs in mind we suggest the following generic simulation architecture which is broken down into three main stages; *input*, *main simulation loop*, and *output*, see Figure A.1. The input stage is where values for parameters are set. This will include parameters that define the agents to be used in the simulation, such as the cruising speed for the UAVs and the control radius for the ports. This is also where the experimental setup is defined which will include information such as how long the simulation will last, which traffic management methods are to be used, and initial conditions for agents. The input then takes the form of a single script file that, when called, will result in one or more instances of a simulation loop being run. When we developed some new agent behaviour or experimental setup this script file was designed to be readily editable by a user. Alternatively, we could define a range of parameter values in the script file which are then swept over, each resulting in a different simulation run. When performing this sort of parameter sweep we might also use functions to automate the setup of certain parameters. An example of this is the pairwise avoidance experiments presented in Chapter 2. Due to the symmetry of the experimental setup, we defined a particular setup in terms of three parameters. These can then be used to produce the initial conditions for the two UAVs in the appropriate format that conforms to that shown in Table A.1, e.g., the initial position in Cartesian coordinates.

The second stage of the simulation architecture is the main simulation loop. This is where the agents are instantiated and the various dynamical models are implemented. We will explain the contents of the main simulation loop in more detail below, see Figure A.2. For a generic simulation architecture however, it is sufficient to show that the main simulation loop consists of various *modules*. Each of these modules will be responsible for carrying out an aspect of the simulation and may contain *sub-modules* which typically correspond to the implementation of a particular model. We have adopted this modular approach to the simulation architecture in order to make the design and testing of new traffic management methods as easy as possible. By organising the various capabilities and functions of the main simulation loop into these modules we can quickly change or remove modules between simulation runs. This then allows us to use a single architecture for the different experimental setups we needed to explore. Another key benefit of this modular approach is that it ensures a level of robustness as development can be carried out on individual modules without impacting others. This allows for easier debugging as we can focus on individual sections of code.

The final stage of the architecture is the output. A simulation is only useful if it produces data which can then be analysed. We therefore need to know what data to record from the main simulation loop and then store it in an appropriate format. On page 14 we described how we defined efficiency and fairness, two of the main metrics we use throughout the thesis, in terms of the delay a UAV incurs during its flight. There are many different ways to define this delay, depending on the particular experimental setup, but in all cases this will require recording some flight data from each UAV. This flight data is recorded either when a UAV lands at a port agent or when the simulation ends. The data will include the time and position from both the start and end of the UAV agent's flight.

In order to carry out simulations and produce results we needed to choose a programming language in which to implement the generic simulation architecture outlined above. While this could be achieved in different ways in different programming languages, we suggest that an interpreted language would be most appropriate. This is because the work in this thesis is exploratory in nature and required us to develop and iterate modules that enabled, for example, different avoidance behaviours or experimental setups. This process can be quicker in an interpreted language where many similar modules do not need to be individually compiled. The development process also then benefits from easier debugging as the execution of a module can be tracked line by line through the language's interpreter.

We chose MATLAB as the language in which to implement the simulation architecture. This

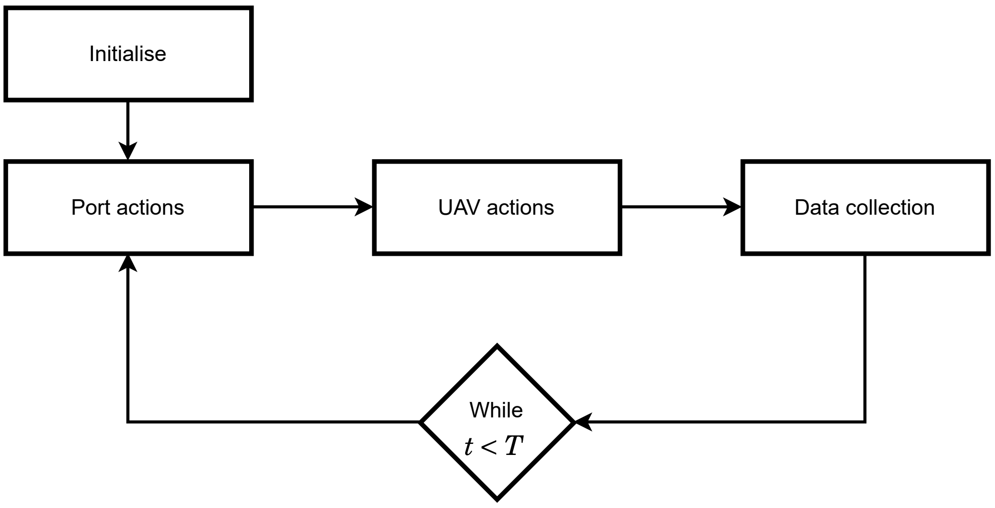


Figure A.2: The main simulation loop used throughout the thesis. After an initialisation step, which sets up any agents required by the input stage, the main simulation loop moves through three steps. In the first and second steps actions are undertaken by the port and UAV agents respectively, these steps are explained in greater detail below, see Figures A.3 to A.5. In the third step data collection is handled. This loop will repeat until the simulation time (t) is equal to some total time (T) set during the input stage.

choice was primarily based on our existing familiarity with MATLAB from previous projects which allowed for quick early development and prototyping. We also used MATLAB to analyse the data produced by our simulations which was made easier by the simulation itself being developed in MATLAB.

It should however be possible to implement this architecture in any language, for example Python which has become a popular choice in academia and the emerging data science industry. Indeed a compiled programming language could be used to implement a version of this architecture for simulations that are much larger scale than anything considered in this thesis. This is because a compiled language will tend to produce simulations that are more efficient when executed compared to the same simulation produced by an interpreted language. However, this increased efficiency at run time would likely not be worth the likely longer development times.

With the choice of programming language in mind, we will discuss the structure and flow of the main simulation loop as used throughout this thesis, as well as some of the choices made in its implementation. The main simulation loop then can be broken down into four stages; *initialisation*, *port actions*, *UAV actions*, and *data collection*, Figure A.2. The second and third stages reflect the fact that there are two types of agents and we collect the various modules that are responsible for their actions together. Both the port and UAV actions stages will be explored in more detail in their own sections below, see Figures A.3 to A.5.

The first stage, initialisation, is carried out once at the start of the simulation loop. This is where appropriate data structures and initial agents are set up. We do not consider the dynamic creation and destruction of ports at any point in this thesis and therefore all ports must be defined before the simulation loop begins and are set up in this stage. Some UAVs can also be set up at this stage, which is how we carry out the pairwise avoidance experiments presented in Chapter 2 as they do not dynamically generate UAVs. This is also where simulation time t is set to 0 s.

After this initialisation stage the other three stages take place in turn during each time step of the simulation. First all port agents carry out their actions, i.e., this is where new UAVs are introduced into the simulation. Next all the UAVs carry out their actions. This is where we implement the different dynamical models and UAVs leave the simulation by landing at their destination. We order the agent actions in this way because the experiments that make use of ports begin with no UAV agents. Finally we collect data, such as UAV states upon landing, and format it appropriately.

Once these three stages have been carried out we increment the simulation time by a time step Δt. For all simulation work in this thesis we set $\Delta t = 0.05\,\mathrm{s}$ which is more than one hundred times smaller than the time scale $\tau = 8\,\mathrm{s}$ defined in Equation 2.3, using the dimensional parameter values found in Table 2.1.

UAV actions

In Figure A.1 above we introduced our generic simulation architecture and then discussed the modular design philosophy we have adopted for this work. We also briefly outlined the different stages of the main simulation loop. Given that the simulation consists of two types of agents whose actions drive the simulator, we have grouped their actions into two stages in the main simulation loop, i.e, the stages correspond to a particular type of agent. In this section then, we will provide more detail on how we implement the various actions that the UAV agents can undertake and how these actions affect their state.

During a time step of the simulation each UAV agent performs four actions; steering toward a desired heading, avoiding its neighbours, producing a resultant acceleration, and updating its state, Figure A.3. These actions are implementations of the various dynamical models described in Chapter 2. The way this works in practice is that the main simulation loop will call a module with two inputs; a list of the current UAV agents with their states, and a list of sub-module handles. These sub-module handles are used to determine which models the UAV agents will follow, e.g., whether the desired heading for a UAV is given by a constant vector or some waypoint. Once all UAV agents have performed all four actions the module will return a list of UAV agents with updated states to be used in the next time step.

The first action that a UAV performs is to steer toward a desired heading. This is achieved by implementing the optimal velocity model described by Equation 2.2 to produce an acceleration that is proportional to the difference between the UAV's current velocity and a desired velocity. If

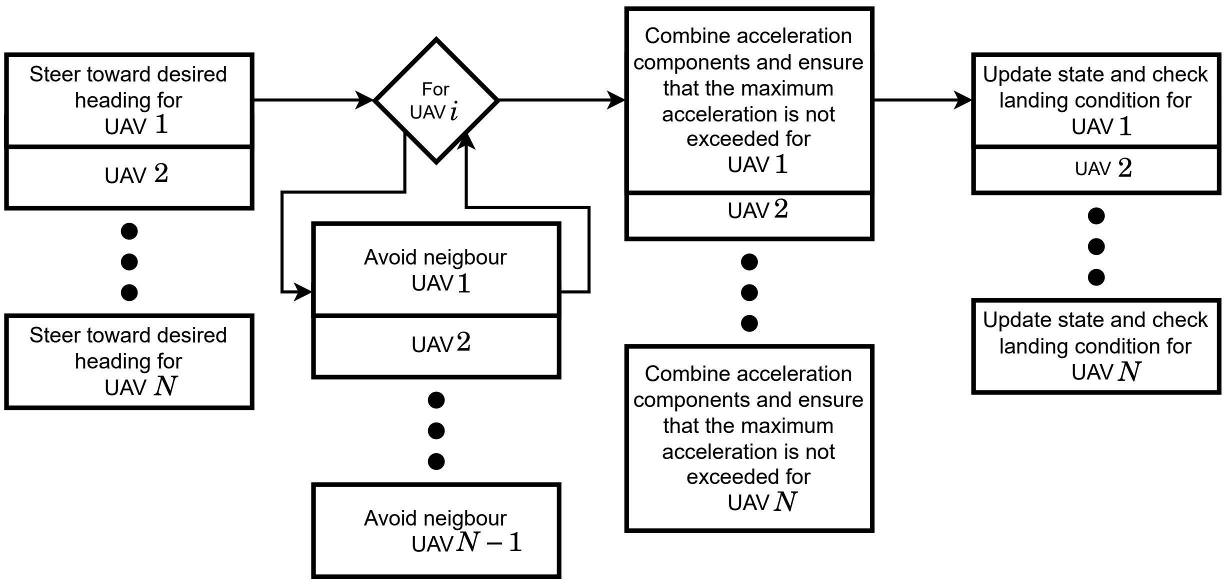

Steer toward desired heading for UAV 1
UAV 2
Steer toward desired heading for UAV N
For UAV i
Avoid neigbour UAV 1
UAV 2
Avoid neigbour UAV $N-1$
Combine acceleration components and ensure that the maximum acceleration is not exceeded for UAV 1
UAV 2
Combine acceleration components and ensure that the maximum acceleration is not exceeded for UAV N
Update state and check landing condition for UAV 1
UAV 2
Update state and check landing condition for UAV N

Figure A.3: The various actions that each UAV undertakes during a single time step of the main simulation loop. The first two actions both produce acceleration components that steer the UAV toward a desired heading or to avoid their neighbours respectively. These components are then combined and scaled in the third action to produce a resultant acceleration. The last thing the UAVs do is to update their state based on their velocity at the start of the time step and the resultant acceleration. The actions of each UAV are shown here as stacked on top of each other since we implement them in a vectorised way.

the UAV's current velocity is the same as its desired velocity then an acceleration is still produced but with the x and y components set to zero.

The next action each UAV performs is to avoid its neighbours. This is the most complex action, typically composed of several sub-modules, and will therefore be explored in more detail in its own subsection below. Note, however, that this action will also result in an acceleration that corresponds to each UAV, and that this acceleration's x and y components can both be zero, i.e., a UAV is not on a conflict course with any other UAVs.

The first two actions then result in each UAV being assigned two accelerations. The third action combines these two accelerations and ensures that the UAV's maximum acceleration is respected, as described in Equation 2.4. We do this by adding the two accelerations together to produce a resultant acceleration. Any resultant accelerations with a magnitude greater than some maximum value are rescaled so that the direction is preserved but the magnitude is now equal to the maximum allowed value.

The final action is where the UAV updates its state. Firstly, the position of the UAV is updated based on its velocity at the start of the time step, $\mathbf{r}_i(t+\Delta t) = \mathbf{r}_i(t) + \mathbf{v}_i(t)\Delta t$. Then the velocity is updated based on the resultant acceleration, $\mathbf{v}_i(t+\Delta t) = \mathbf{v}_i(t) + \mathbf{a}_i(t)\Delta t$. Finally the UAV checks its landing condition. In this thesis we use a simple landing condition that checks what the distance

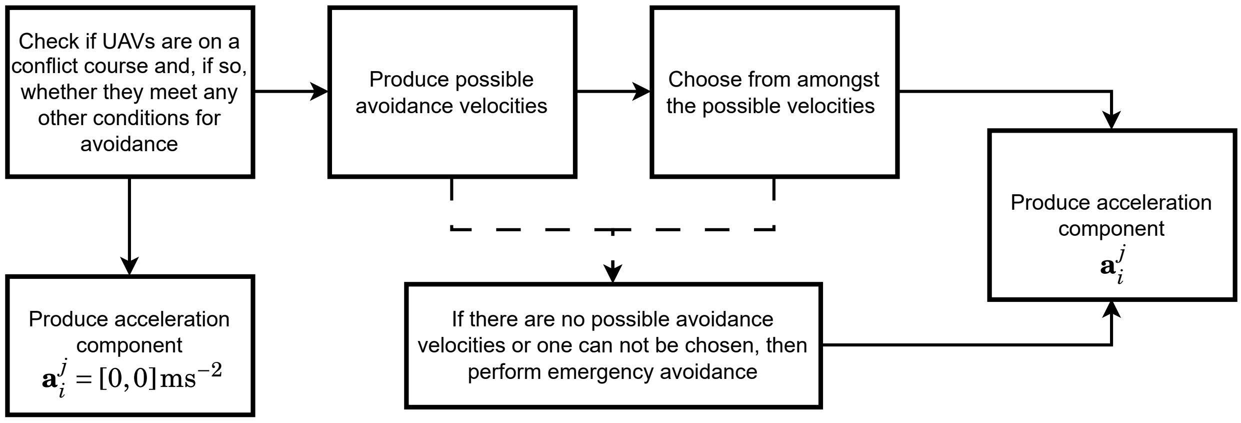

Check if UAVs are on a conflict course and, if so, whether they meet any other conditions for avoidance
Produce possible avoidance velocities
Choose from amongst the possible velocities
Produce acceleration component $\mathbf{a}_i^j$
Produce acceleration component $\mathbf{a}_i^j = [0,0]\text{ms}^{-2}$
If there are no possible avoidance velocities or one can not be chosen, then perform emergency avoidance

Figure A.4: The avoidance action for a single UAV and its neighbour is broken down in to several components. Each of these components can be implemented in its own sub-module so that different avoidance behaviours can be explored between simulation runs.

between the UAV and its destination is. If this distance is less than or equal to some parameter value set during the input stage of the simulation, then the UAV is considered to be landing. This means that it is flagged for data collection and removed from the list of active UAVs.

In our chosen programming language, MATLAB, it is usually more efficient to perform operations in a vectorised way, rather than using 'for loops'. As such we have tried to vectorise these actions as much as possible, represented in Figure A.3 by the way the actions for different UAVs are stacked on top of each other. It is only during the avoidance action that we were not able to fully vectorise the action. Instead a particular UAV i will avoid each of their $N-1$ neighbours in a vectorised way, but we must then loop over each UAV to perform the avoidance action.

Despite the use of a for loop we still maintain a parallel update scheme, that is the UAVs can be thought of as updating their state at the same time. This is because the first, second, and third actions all use the UAVs' states from the start of the time step to produce the resultant acceleration. Therefore a UAV which has updated its state does not affect another UAV until the next time step.

For this sort of discrete-time, agent-based simulation the choice of update scheme can result in different system behaviour. Other potential update schemes include sequential and stochastic updating. In the former, agents are updated according to some deterministic order while in the latter agents update at random. We have implemented a parallel update scheme as we assume that during each time step the UAV agents will apply the dynamical models in the same way. In future work it may be necessary to explore other update schemes, for example a stochastic update scheme that captures the random nature of wireless communication between UAVs.

Avoidance action

As discussed above, the avoidance action the UAV agents perform is more complex than the others and can be further broken down into components. In Figure A.4 we provide a generic architecture for the implementation of the avoidance action in terms of these components. The components themselves are implemented in sub-modules which allows the simulator to change between avoidance behaviours easily between runs.

From Figure A.4, the first thing a particular UAV i does during the avoidance action is to check whether it and its neighbour j are on a conflict course. We do this through the implementation of a velocity obstacle inspired method which is described in Section 2.3. If the two UAVs are not on a conflict course then the UAV i produces an acceleration component $a_i^j = [0,0]\,\mathrm{ms}^{-2}$.

In addition to checking whether two UAVs are on a conflict course, we might have other conditions that are checked before continuing with the avoidance action. We could conceive of many possible conditions, for example having UAVs only avoid a certain number of nearest neighbours. In this thesis however, for experiments with many UAVs, we used a condition on the 'time to conflict' to determine whether a UAV needed to avoid its neighbour. We did this in part to reduce the number of neighbours a UAV needs to avoid which in turn reduces the number of computations in the latter components of the avoidance action. These kinds of conditions could also be used to model communication range and other factors in future work.

If all of the conditions for avoidance are met, then the UAV will produce a list of possible avoidance velocities. The mathematics of how this is done is also described in Section 2.3. This will result in between one and four possible avoidance velocities which the UAV must then choose between. We separate out the generation of possible avoidance velocities and the act of choosing between them into their own sub-modules. This enables us to try different combinations of different models. In this thesis we use the same method to generate possible avoidance velocities throughout and only change the method with which UAVs choose between the velocities, e.g., the simple or hybrid avoidance. Again, this modular approach would allow future work to explore other methods for generating the avoidance velocities without having to replicate any of the methods for choosing.

If there are no possible velocities, or an avoidance velocity can not be chosen, then the UAV will implement some emergency avoidance behaviour. This could simply be to do nothing, i.e, produce an acceleration component $a_i^j = [0,0]\,\mathrm{ms}^{-2}$ as though the UAVs were not on a conflict course. However, in this thesis we implement a simple electrostatic-like repulsion, Equation 2.6. This emergency behaviour is only rarely enacted in situations such as two UAVs already being too close to each other.

Assuming that an avoidance velocity is chosen, then the UAV will produce an avoidance acceleration component a_i^j. This is done for each of the UAV i's neighbours in a vectorised way. This produces a list of acceleration components that are added together to produce a single avoidance acceleration for each UAV. This means that the output of the avoidance action is a list of accelerations with the same number of elements as the first action in Figure A.3 (steering toward a desired heading). This makes the third action, where the acceleration components are combined and scaled, easier since the two lists can be added together.

Port actions

In the above section we discussed how the various actions of the UAV agents are structured in our simulator. In keeping with the modular design philosophy outlined in our overview of the generic simulation architecture, the various actions were implemented in one or more sub-modules. In this section then, we will do the same for the port agents.

Unlike the UAV agents, in this thesis the ports only do one thing, i.e., generate new UAVs during a simulation. Therefore we only require one sub-module to handle this action and we will explain how this works in more detail below. However, it would be possible to add other actions, using a similar structure to the UAV actions in Figure A.3, for example to generate new ports during a simulation.

We break down the action of generating new UAVs in to various components, Figure A.5, as we did for the UAV avoidance action. Since the number of ports is usually small, especially when compared to the number of UAVs, we therefore loop over each of the ports in turn. This is a static, sequential update scheme, i.e, the order in which ports go through this does not change between time steps. However this should not impact the simulation results since the UAVs are still updated in parallel and any new UAVs are added before the UAV actions take place.

The port begins by generating a random number between 0 and 1. If this is bigger than a probability determined by the total demand that the port was assigned during the input stage of the simulation, then no new UAVs are generated. If the random number is smaller than this probability then a new UAV is generated and added to the port's queue. This is simply an integer that records how many UAVs are at the port waiting to take off. See Section 3.2 for more details on the mathematical model being used.

If the port's queue is not empty it will then check if the airspace around it is clear. It does this by checking the distance between itself and all UAVs currently active in the simulation. If no UAVs are within a certain distance of the centre of the port then the airspace is considered clear.

If the airspace is clear then the port will generate a new UAV by adding it to the list of current UAVs. This new UAV will have a position that is the same as that of the port and its velocity will point towards its destination. We assign the destination by normalising a cumulative sum of the demand list associated with the port and then generating another random number between 0 and 1. This is what allows us to model the port's queue as an integer which keeps the data structure for the ports simple. In principle, for our purposes, it is only important to know where a UAV is going once it is in the air.

If the airspace around the port is not clear, or its queue is empty, then the next port will begin

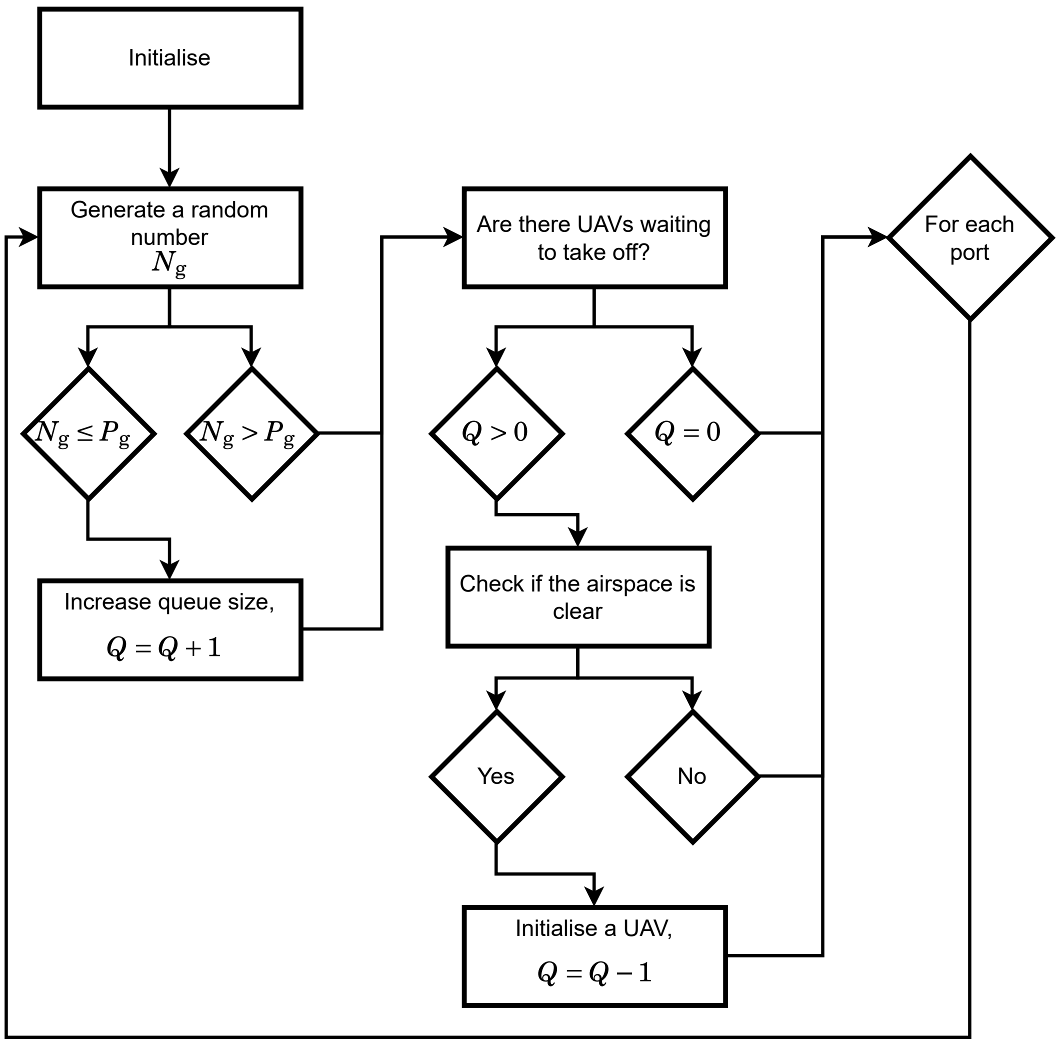

Initialise
Generate a random number N_g
$N_g \leq P_g$
$N_g > P_g$
Increase queue size, $Q = Q + 1$
Are there UAVs waiting to take off?
$Q > 0$
$Q = 0$
Check if the airspace is clear
Yes
No
Initialise a UAV, $Q = Q - 1$
For each port

Figure A.5: A breakdown of how each port tries to generate a new UAV during a single time step of the main simulation loop. A port will first try to add a UAV to its queue which will depend on the demand it was assigned at the input stage of the simulation. If the port's queue is not empty, then it will check if the surrounding airspace is clear. If the airspace is clear, then a new UAV is generated, at which point it is assigned its destination and initial state. This process is then repeated for all ports. Note that a port can only generate one UAV per time step.

this action. Once the last port has finished trying to generate a new UAV the main simulation loop continues to the UAV actions.

Summary

In this appendix we have discussed the simulation architecture, the implementation of this architecture, and the two types of agents that act within the simulation. The generic simulation architecture we have developed is comprised of three main stages with the main simulation loop (i.e., where UAVs move around and avoid one another) being the second stage. We have shown how the design principle of modularity has been central to the development of the main simulation loop which is composed of different modules responsible for implementing the various models we use to describe the system. The two main modules then correspond to the two main agents; ports and UAVs. The details of how these agents carry out their various actions have been detailed in their respective sections.

Appendix B

List of publications

- W. Bonnell, and R. E. Wilson, “Velocity Obstacles and Emergent Rules-of-the-Air for Autonomous Drone Traffic Management”, *techrxiv* (preprint), 2021
- W. Bonnell, and R. E. Wilson, “Improving Unmanned Aerial Vehicle Traffic Flow at a Crossroads by Splitting Demand into Parallel Streams”, in *TRISTAN 11*, 2022

References

[1] D. Donald, “U.S. Air Force Ends Predator Operations.” https://www.ainonline.com/aviation-news/defense/2018-03-13/us-air-force-ends-predator-operations, 2018.
Accessed: 2022/09/03.

[2] “Call of Duty: Modern Warfare.” Activision: Santa Monica, 2007.

[3] A. Webster, “AR.Drone coming to Android, gets new multiplayer games.” ars technica, https://arstechnica.com/gaming/2011/06/ardrone-coming-to-android-gets-new-multiplayer-games/, 2011.
Accessed: 2022/09/03.

[4] L. Eadicicco, “How This New Drone Can Track Your Every Move.” TIME, https://time.com/4243394/dji-phantom-4-activetrack/, 2016.
Accessed: 2022/09/03.

[5] M. Waibel, “Video: Throwing and catching an inverted pendulum - with quadrocopters.” https://robohub.org/video-throwing-and-catching-an-inverted-pendulum-with-quadrocopters/, 2013.
Accessed: 2022/09/03.

[6] S. Hamaza, I. Georgilas, and T. Richardson, “Energy-Tank Based Force Control for 3D Contour Following,” in *Towards Autonomous Robotic Systems*, 2019.

[7] Markets and Markets, “Drone Service Market - Global Forecast to 2025.” https://www.marketsandmarkets.com/Market-Reports/drone-services-market-80726041.html, 2019.
Accessed: 2022/09/03.

[8] PWC, “Skies without limits: Drones - taking the UK’s economy to new heights.” https://www.pwc.co.uk/intelligent-digital/drones/Drones-impact-on-the-UK-economy-FINAL.pdf, 2018.
Accessed: 2022/09/03.

[9] PWC, “Skies Without Limits V2.0.” https://www.pwc.co.uk/intelligent-digital/drones/skies-without-limits-2022.pdf, 2022.
Accessed: 2023/05/26.

[10] Department for Business, Energy and Industrial Strategy, “Regulation for the Fourth Industrial Revolution: White Paper.” https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/807792/regulation-fourth-industrial-strategy-white-paper-web.pdf, 2019.
Accessed: 2022/09/03.

[11] J. Karp and A. Pasztor, “Drones in Domestic Skies?.” The Wall Street Journal, 2006.

[12] A. Sato, “The RMAX Helicopter UAV.” https://apps.dtic.mil/dtic/tr/fulltext/u2/a427393.pdf, 2003.
Accessed: 2022/09/03.

[13] H. Shakhatreh *et al.*, “Unmanned Aerial Vehicles (UAVs): A Survey on Civil Applications and Key Research Challenges,” *IEEE Access*, vol. 7, pp. 48572–48634, 2019.

[14] Nesta, “Flying High: Shaping the future of drones in UK cities.” https://media.nesta.org.uk/documents/Flying-High-full-report-and-appendices.pdf, 2018.
Accessed: 2022/09/03.

[15] A. Oakey and T. Cherrett, “A drone service to support the isle of wight nhs in the uk,” in *Proceedings of the 14th ITS European Congress*, 2022.

[16] A. Oakey *et al.*, “Integrating drones into nhs patient diagnostic logistics systems: flight or fantasy?,” *PLoS ONE*, vol. 17, 2022.

[17] Business Insider Intelligence, “Here’s how drones are transforming news media.” https://www.businessinsider.com/heres-how-drones-are-transforming-news-media-2017-1?r=US{&}IR=T, 2017.
Accessed: 2022/09/03.

[18] R. Brandom, “Ferguson’s no-fly zone was about keeping the media out, according to new documents.” The vergre, https://www.theverge.com/2014/11/3/7149445/fergusons-drone-blackout-was-about-keeping-the-media-out-faa, 2014.
Accessed: 2022/09/03.

[19] BBC, “Guidance: Use of drones.” https://www.bbc.com/editorialguidelines/guidance/drones, 2022.
Accessed: 2022/09/03.

[20] Civil Aviation Authority (UK), "The Drone and Model Aircraft Code." https://register-drones.caa.co.uk/drone-code, 2019.
Accessed: 2022/09/03.

[21] J. Stewart, "A Drone-Slinging UPS Van Delivers the Future." Wired, `https://www.wired.com/2017/02/drone-slinging-ups-van-delivers-future/`, 2017.
Accessed: 2022/09/03.

[22] C. C. Murray and A. G. Chu, "The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery," *Transportation Research Part C: Emerging Technologies*, vol. 54, pp. 86–109, 2015.

[23] P. Bouman, N. Agatz, and M. Schmidt, "Dynamic programming approaches for the traveling salesman problem with drone," *Networks*, vol. 72, pp. 528–542, 2018.

[24] E. Es Yurek and H. C. Ozmutlu, "Traveling salesman problem with drone under recharging policy," *Computer Communications*, vol. 179, pp. 35–49, 2021.

[25] E. Ackerman and M. Koziol, "In the Air With Zipline's Medical Delivery Drones," *IEEE Spectrum*, 2019.

[26] M. Strohmeier *et al.*, "Realities and challenges of nextgen air traffic management: The case of ADS-B," *IEEE Communications Magazine*, vol. 52, pp. 111–118, 2014.

[27] R. M. Guterres *et al.*, "ADS-B surveillance system performance with small UAS at low altitudes," in *AIAA Information Systems-AIAA Infotech at Aerospace*, 2017.

[28] F. Minucci, E. Vinogradov, and S. Pollin, "Avoiding Collisions at Any (Low) Cost: ADS-B like Position Broadcast for UAVs," *IEEE Access*, vol. 8, pp. 121843–121857, 2020.

[29] A. Bazzi *et al.*, "On the performance of IEEE 802.11p and LTE-V2V for the cooperative awareness of connected vehicles," *IEEE Transactions on Vehicular Technology*, vol. 66, pp. 10419–10432, 2017.

[30] M. Doole, J. Ellerbroek, and J. Hoekstra, "Drone Delivery: Urban airspace traffic density estimation," in *8th SESAR Innovation Days*, 2018.

[31] A. Oosedo *et al.*, "Unmanned Aircraft System Traffic Management (UTM) Simulation of Drone Delivery Models in 2030 Japan," *Journal of Robotics and Mechatronics*, vol. 33, pp. 348–362, 2021.

[32] Federal Aviation Administration (US), "FAA National Forecast 2019-2039." `https://www.faa.gov/data_research/aviation/aerospace_forecasts/media/FAA_Aerospace_Forecasts_FY_2019-2039.pdf`, 2019.
Accessed: 2022/09/03.

[33] Federal Aviation Administration (US), “FAA Aerospace Forecast Fiscal Years 2022-2042.” https://www.faa.gov/sites/faa.gov/files/2022-06/FY2022_42_FAA_Aerospace_Forecast.pdf, 2022.
Accessed: 2022/09/03.

[34] European Union Aviation Safety Agency, “Easy Access Rules for Unmanned Aircraft Systems.” https://www.easa.europa.eu/sites/default/files/dfu/Easy Access Rules for Unmanned Aircraft Systems (Revision from September 2021).pdf, 2021.
Accessed: 2022/09/03.

[35] Federal Aviation Authority (US), “Summary of small unmanned aircraft rule (Part 107).” https://www.ecfr.gov/current/title-14/chapter-I/subchapter-F/part-107#107.73, 2016.
Accessed: 2022/09/03.

[36] T. Jiang *et al.*, “Unmanned Aircraft System traffic management: Concept of operation and system architecture,” *International Journal of Transportation Science and Technology*, vol. 5, pp. 123–135, 2016.

[37] N. Zhu *et al.*, “GNSS Position Integrity in Urban Environments: A Review of Literature,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, pp. 2762–2778, 2018.

[38] P. D. Groves *et al.*, “Intelligent Urban Positioning using Multi- Constellation GNSS with 3D Mapping and NLOS Signal Detection,” in *25th International Tecnical Meeting of the Satellite Division of The Institute of Navigation*, 2012.

[39] World Economic Forum, “Advanced Drone Operations Toolkit : Accelerating the Drone Revolution.” https://www3.weforum.org/docs/WEF_Advanced_Drone_Operations_Toolkit.pdf, 2018.
Accessed: 2022/09/03.

[40] Joint Authorities for Rulemaking of Unmanned Systems (JARUS), “JARUS guidelines on Specific Operations Risk Assessment (SORA), EDITION 2.0.” http://jarus-rpas.org/sites/jarus-rpas.org/files/jar_doc_06_jarus_sora_v2.0.pdf, 2019.
Accessed: 2022/09/03.

[41] Federal Aviation Administration (US), “Unmanned Aircraft System (UAS) Traffic Management (UTM): Concept of Operations v2.0.” https://www.faa.gov/uas/research_development/traffic_management/media/UTM_ConOps_v2.pdf, 2020.
Accessed: 2022/09/03.

[42] Connected Places Catapult, “Enabling UTM in the UK,” 2020.

[43] European Union Aviation Safety Agency, “Opinion No 01 / 2020: High-level regulatory framework for the U-space.” `https://www.easa.europa.eu/sites/default/files/dfu/OpinionNo01-2020.pdf`, 2020.
Accessed: 2022/09/03.

[44] ICAO, “Doc 9854, Global Air Traffic Management Operational Concept.” `https://www.icao.int/Meetings/anconf12/DocumentArchive/9854_cons_en[1].pdf`, 2005.
Accessed: 2022/09/03.

[45] ICAO, “Annex 2 - Rules of the Air - Tenth Edition.” `https://www.icao.int/Meetings/anconf12/DocumentArchive/an02_cons[1].pdf`, 2005.
Accessed: 2022/09/03.

[46] H. Alturbeh and J. F. Whidborne, “Visual flight rules-based collision avoidance systems for UAV flying in civil aerospace,” *Robotics*, vol. 9, pp. 1–35, 2020.

[47] M. F. Bin Mohammed Salleh *et al.*, “Preliminary concept of adaptive urban airspace management for unmanned aircraft operations,” in *AIAA Information Systems-AIAA Infotech at Aerospace*, 2018.

[48] E. Sunil *et al.*, “Analysis of airspace structure and capacity for decentralized separation using fast-time simulations,” *Journal of Guidance, Control, and Dynamics*, vol. 40, pp. 38–51, 2017.

[49] E. Sunil *et al.*, “Metropolis: Relating airspace structure and capacity for extreme traffic densities,” in *Proceedings of the 11th USA/Europe Air Traffic Management Research and Development Seminar*, 2015.

[50] A. Battista and D. Ni, “A Comparison of Traffic Organization Methods for Small Unmanned Aircraft Systems,” *Transportation Research Record*, vol. 2672, pp. 20–30, 2018.

[51] L. Sedov and V. Polishchuk, “Centralized and Distributed UTM in Layered Airspace,” in *8th International Conference on Research in Air Transportation*, 2018.

[52] Z. Liu, R. Sengupta, and A. Kurzhanskiy, “A power consumption model for multi-rotor small unmanned aircraft systems,” in *International Conference on Unmanned Aircraft Systems*, 2017.

[53] C. W. Reynolds, “Flocks, herds, and schools: A distributed behavioral model,” in *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, 1987.

[54] G. Vásárhelyi *et al.*, “Optimized flocking of autonomous drones in confined environments,” *Science Robotics*, vol. 3, pp. 368–378, 2018.

[55] C. W. Reynolds, “Steering Behaviors For Autonomous Characters Steering Behaviors For Autonomous Characters,” in *Game developers conference*, 1999.

[56] D. Helbing and P. Molnar, “Social Force Model for Pedestrian Systems,” *Physical Review E*, vol. 51, pp. 4282–4286, 1995.

[57] B. M. Albaker and N. A. Rahim, “A survey of collision avoidance approaches for unmanned aerial vehicles,” in *International Conference for Technical Postgraduates 2009*, 2009.

[58] O. Khatib, “Real-time obstacle avoidance for manipulators and mobile robots,” in *IEEE International Conference on Robotics and Automation*, 1985.

[59] K. Sigurd and J. How, “UAV trajectory design using total field collision avoidance,” in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2003.

[60] J. Barraquand, B. Langlois, and J.-C. Latombe, “Numerical potential field techniques for robot path planning,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 22, pp. 224– 241, 1992.

[61] T. Schouwenaars *et al.*, “Mixed integer programming for multi-vehicle path planning,” in *European Control Conference*, 2001.

[62] A. Richards *et al.*, “Spacecraft trajectory planning with avoidance constraints using mixed-integer linear programming,” *Journal of Guidance, Control, and Dynamics*, vol. 25, pp. 755–764, 2002.

[63] J. How, E. King, and Y. Kuwata, “Flight Demonstrations of Cooperative Control for UAV Teams,” in *AIAA 3rd "Unmanned Unlimited" Technical Conference, Workshop and Exhibit*, 2004.

[64] D. Mellinger, A. Kushleyev, and V. Kumar, “Mixed-integer quadratic program trajectory generation for heterogeneous quadrotor teams,” in *Proceedings - IEEE International Conference on Robotics and Automation*, 2012.

[65] J. Schulman *et al.*, “Finding Locally Optimal, Collision-Free Trajectories with Sequential Convex Optimization,” *Robotics: science and systems*, vol. 9, pp. 1–10, 2013.

[66] D. Morgan, S. J. Chung, and F. Y. Hadaegh, “Model predictive control of swarms of spacecraft using sequential convex programming,” *Journal of Guidance, Control, and Dynamics*, vol. 37, pp. 1725–1740, 2014.

[67] C. Tomlin, G. J. Pappas, and S. Sastry, “Conflict resolution for Air Traffic Management: A study in multiagent hybrid systems,” *IEEE Transactions on Automatic Control*, vol. 43, pp. 509–521, 1998.

[68] Y. Kumar, A. Manoharan, and P. Sujit, "Right of Way Rules based Collision Avoidance Approach Using Model Predictive Control," in *Sixth Indian Control Conference*, 2020.

[69] Y. F. Chen *et al.*, "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning," in *IEEE International Conference on Robotics and Automation*, 2017.

[70] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using the relative velocity paradigm," in *IEEE International Conference on Robotics and Automation*, 1993.

[71] P. Fiorini and Z. Shiller, "Motion Planning in Dynamic Environments using Velocity Obstacles," *The International Journal of Robotics Research*, vol. 17, pp. 760–772, 1998.

[72] J. D. Van Berg, M. Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," in *IEEE International Conference on Robotics and Automation*, 2008.

[73] J. Snape *et al.*, "The hybrid reciprocal velocity obstacle," *IEEE Transactions on Robotics*, vol. 27, pp. 696–706, 2011.

[74] J. D. Van Berg *et al.*, "Reciprocal n-body collision avoidance," *Springer Tracts in Advanced Robotics*, vol. 70, pp. 3–19, 2011.

[75] J. A. Douthwaite, S. Zhao, and L. S. Mihaylova, "Velocity Obstacle Approaches for Multi-Agent Collision Avoidance," *Unmanned Systems*, vol. 7, pp. 55–64, 2019.

[76] S. Gupte, P. I. T. Mohandas, and J. M. Conrad, "A survey of quadrotor unmanned aerial vehicles," in *IEEE SOUTHEASTCON*, 2012.

[77] B. B. Deepak and P. Singh, "A survey on design and development of an unmanned aerial vehicle (quadcopter)," *International Journal of Intelligent Unmanned Systems*, vol. 4, pp. 70–106, 2016.

[78] M. Bando *et al.*, "Dynamical model of traffic congestion and numerical simulation," *Physical Review E*, vol. 51, pp. 1035–1042, 1995.

[79] C. Deng *et al.*, "Unmanned aerial vehicles for power line inspection: A cooperative way in platforms and communications," *Journal of Communications*, vol. 9, pp. 687–692, 2014.

[80] P. Liu *et al.*, "A review of rotorcraft unmanned aerial vehicle (UAV) developments and applications in civil engineering," *Smart Structures and Systems*, vol. 13, pp. 1065–1094, 2014.

[81] Y. Ham *et al.*, “Visual monitoring of civil infrastructure systems via camera-equipped Unmanned Aerial Vehicles (UAVs): a review of related works,” *Visualization in Engineering*, vol. 4, pp. 1–8, 2016.

[82] J. Scherer *et al.*, “An autonomous multi-UAV system for search and rescue,” in *Proceedings of the 2015 Workshop on Micro Aerial Vehicle Networks, Systems, and Applications for Civilian Use*, 2015.

[83] M. Silvagni *et al.*, “Multipurpose UAV for search and rescue operations in mountain avalanche events,” *Geomatics, Natural Hazards and Risk*, vol. 8, pp. 18–33, 2017.

[84] F. Reclus and K. Drouard, “Geofencing for fleet & freight management,” in *9th International Conference on Intelligent Transport Systems Telecommunications*, 2009.

[85] M. Hosseinzadeh, “Chapter 22 - UAV geofencing: navigation of UAVs in constrained environments,” in *Unmanned Aerial Systems* (A. Koubaa and A. T. Azar, eds.), Advances in Nonlinear Dynamics and Chaos (ANDC), pp. 567–594, Academic Press, 2021.

[86] R. Bor-Yaliniz, A. El-Keyi, and H. Yanikomeroglu, “Efficient 3-D placement of an aerial base station in next generation cellular networks,” in *IEEE International Conference on Communications*, 2016.

[87] Y. Zeng, R. Zhang, and T. J. Lim, “Throughput Maximization for UAV-Enabled,” *IEEE Transactions on Communications*, vol. 64, pp. 4983–4996, 2016.

[88] Y. Chen, W. Feng, and G. Zheng, “Optimum Placement of UAV as Relays,” *IEEE Communications Letters*, vol. 22, pp. 248–251, 2018.

[89] D. Kingston, R. W. Beard, and R. S. Holt, “Decentralized perimeter surveillance using a team of UAVs,” *IEEE Transactions on Robotics*, vol. 24, pp. 1394–1404, 2008.

[90] N. H. Motlagh, M. Bagaa, and T. Taleb, “UAV-Based IoT Platform: A Crowd Surveillance Use Case,” *IEEE Communications Magazine*, vol. 55, pp. 128–134, 2017.

[91] R. D’Andrea, “Guest editorial: can drones deliver?,” *IEEE Transactions on Automation Science and Engineering*, vol. 11, pp. 647–648, 2014.

[92] J. C. Curlander *et al.*, “Multi-level fulfillment center for unmanned aerial vehicles.” Patent, 2017.

[93] D. Silver, “Smart Mailboxes Will Unlock Drone Delivery, According To Valqari.” Forbes, 2021.

[94] M. F. B. M. Salleh, D. Y. Tan, C. H. Koh, and K. H. Low, “Preliminary concept of operations (ConOps) for traffic management of unmanned aircraft systems (TM-UAS) in urban environment,” in *AIAA Information Systems-AIAA Infotech at Aerospace*, 2017.

[95] G. Baloch and F. Gzara, “Strategic network design for parcel delivery with drones under competition,” *Transportation Science*, vol. 54, pp. 204–228, 2020.

[96] A. Sanjab, W. Saad, and T. Basar, “Prospect theory for enhanced cyber-physical security of drone delivery systems: A network interdiction game,” in *IEEE International Conference on Communications*, 2017.

[97] G. Price *et al.*, “Urban air mobility operational concept (opscon) passenger-carrying operations.” NASA, `https://ntrs.nasa.gov/api/citations/20205001587/downloads/UAMPassenger-carryingOpsCon-v14GPaccept.pdf`, 2020.
Accessed: 2022/09/03.

[98] S. Hasan, “Crown consulting uam market study.” Crown Consulting Inc., `https://www.nasa.gov/sites/default/files/atoms/files/uam-market-study-executive-summary-v2.pdf`, 2019.
Accessed: 2022/09/03.

[99] H. Ale-Ahmad and H. S. Mahmassani, “Capacitated location-allocation-routing problem with time windows for on-demand urban air taxi operation,” *Transportation Research Record*, vol. 2675, pp. 1092–1114, 2021.

[100] C. Cummings and H. S. Mahmassani, “Emergence of 4-d system fundamental diagram in urban air mobility traffic flow,” *Transportation Research Record*, vol. 2675, pp. 841–850, 2021.

[101] J. Haddad, B. Mirkin, and K. Assor, “Traffic flow modeling and feedback control for future Low-Altitude Air city Transport: An MFD-based approach,” *Transportation Research Part C: Emerging Technologies*, vol. 133, p. 103380, 2021.

[102] S. Le Vine, A. Zolfaghari, and J. Polak, “Autonomous cars: The tension between occupant experience and intersection capacity,” *Transportation Research Part C: Emerging Technologies*, vol. 52, pp. 1–14, 2015.

[103] I. H. Zohdy and H. A. Rakha, “Intersection management via vehicle connectivity: The intersection cooperative adaptive cruise control system concept,” *Journal of Intelligent Transportation Systems*, vol. 20, pp. 17–32, 2016.

[104] J. D. Little, “A proof for the queuing formula: $L = \lambda W$,” *Operations research*, vol. 9, pp. 383–387, 1961.

[105] W. Jones and R. E. Wilson, “Effects of routing on the capacity of multi-hop wireless networks,” in *Modeling, Analysis and Simulation of Wireless and Mobile Systems*, pp. 155–162, 2018.

[106] I. D. Couzin *et al.*, “Effective leadership and decision-making in animal groups on the move,” *Nature*, vol. 433, pp. 513–516, 2005.

[107] P. Walker *et al.*, “Human control of leader-based swarms,” in *IEEE International Conference on Systems, Man, and Cybernetics*, 2013.

[108] P. Walker *et al.*, “Control of swarms with multiple leader agents,” in *IEEE International Conference on Systems, Man and Cybernetics*, 2014.

[109] L. Alexander *et al.*, “Origin-destination trips by purpose and time of day inferred from mobile phone data,” *Transportation Research Part C: Emerging Technologies*, 2015.

[110] N. Geroliminis, J. Haddad, and M. Ramezani, “Optimal perimeter control for two urban regions with macroscopic fundamental diagrams: A model predictive approach,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, pp. 348–359, 2013.

[111] M. Yildirimoglu, I. I. Sirmatel, and N. Geroliminis, “Hierarchical control of heterogeneous large-scale urban road networks via path assignment and regional route guidance,” *Transportation Research Part B: Methodological*, vol. 118, pp. 106–123, 2018.

[112] Z.-H. Mao, E. Feron, and K. Bilimoria, “Stability of intersecting aircraft flows under decentralized conflict avoidance rules,” in *18th Applied Aerodynamics Conference*, 2000.

[113] C. Burstedde *et al.*, “Simulation of pedestrian dynamics using a two-dimensional cellular automaton,” *Physica A: Statistical Mechanics and its Applications*, vol. 295, pp. 507–525, 2001.

[114] M. Dorigo and G. Di Caro, “Ant colony optimization: A new meta-heuristic,” in *Congress on Evolutionary Computation*, 1999.

[115] H. Helbing, I. Farkas, and T. Vicsek, “Simulating dynamical features of escape panic,” *Nature*, vol. 407, pp. 487–490, 2000.

[116] D. Braess, A. Nagurney, and T. Wakolbinger, “On a paradox of traffic planning,” *Transportation Science*, vol. 39, pp. 446–450, 2005.

[117] R. L. Hughes, “The flow of human crowds,” *Annual Review of Fluid Mechanics*, vol. 35, pp. 169–182, 2003.

[118] G. A. Frank and C. O. Dorso, “Room evacuation in the presence of an obstacle,” *Physica A: Statistical Mechanics and its Applications*, vol. 390, pp. 2135–2145, 2011.

[119] A. Johansson and D. Helbing, “Pedestrian flow optimization with a genetic algorithm based on Boolean grids,” in *Pedestrian and Evacuation Dynamics*, 2007.

[120] H. Weimerskirch *et al.*, “Energy saving in flight formation,” *Nature*, vol. 413, pp. 697–698, 2001.

[121] A. Alam *et al.*, “Heavy-duty vehicle platooning for sustainable freight transportation: A cooperative method to enhance safety and efficiency,” *IEEE Control Systems*, vol. 35, pp. 34–56, 2015.

[122] A. K. Bhoopalam, N. Agatz, and R. Zuidwijk, “Planning of truck platoons: A literature review and directions for future research,” *Transportation Research Part B: Methodological*, vol. 107, pp. 212–228, 2018.