3D Path Planning for UAVs in Dynamic Environments in the Presence of Uncertainties

Zammit, C.

DOI
10.4233/uuid:9a3302f6-6b8f-4aad-b877-b270bee7aa78

Publication date
2021

Document Version
Final published version

Citation (APA)
Zammit, C. (2021). *3D Path Planning for UAVs in Dynamic Environments in the Presence of Uncertainties*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:9a3302f6-6b8f-4aad-b877-b270bee7aa78

3D PATH PLANNING FOR UAVS IN DYNAMIC ENVIRONMENTS IN THE PRESENCE OF UNCERTAINTIES

Proefschrift

ter verkrijging van de graad van doctor aan de Technische
Universiteit Delft,
op gezag van de Rector Magnificus prof. dr. ir. T. H. J. J. van der Hagen,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op
dinsdag 5 oktober 2021 om 15:00 uur

door

Christian ZAMMIT

Master of Science in Engineering,
University of Malta, Malta
geboren te Victoria, Gozo, Malta.

Dit proefschrift is goedgekeurd door de

promotor: prof. dr. ir. M. Mulder
copromotor: dr. ir. E. van Kampen

Samenstelling promotiecommissie:

Rector Magnificus,	voorzitter
Prof. dr. ir. M. Mulder,	Technische Universiteit Delft, promotor
Dr. ir. E. van Kampen,	Technische Universiteit Delft, copromotor
Onafhankelijke leden:	
Prof. dr. G.C.H.E. de Croon,	Technische Universiteit Delft
Prof. dr. R.R. Negenborn,	Technische Universiteit Delft
Prof. dr. ir. P.N.A.M. Visser,	Technische Universiteit Delft
Prof. dr. K. Alexis,	Noors Universiteit van Wetenschap & Technologie, Noorwegen
Dr. A. Franchi,	Universiteit Twente

Keywords: UAV, Path Planning, 3-Dimension, Real-time, Uncertainty, Obstacle avoidance, Dynamic Environments, Artificial Intelligence

Printed by: Ipskamp Printing, The Netherlands

Front & Back: The cover and bookmark were designed by Ms. Bregje Jaspers using resources from Shuttershock

ISBN 978-94-6421-492-5

An electronic version of this dissertation is available at
http://repository.tudelft.nl/.

To my beloved parents

Lill-ġeneturi tiegħi

CONTENTS

Summary **xi**
References . . . xvi

Samenvatting **xvii**
References . . . xxiii

1 Introduction **1**
1.1 Background . . . 2
1.1.1 Definitions . . . 2
1.1.2 Need for Path Planning . . . 2
1.1.3 History of Path Planning . . . 6
1.2 Challenges . . . 6
1.2.1 Challenge 1 . . . 7
1.2.2 Challenge 2 . . . 7
1.2.3 Challenge 3 . . . 8
1.3 Research Goal and Research Questions . . . 9
1.3.1 Research Goal . . . 9
1.3.2 Research Question 1 . . . 9
1.3.3 Research Question 2 . . . 10
1.3.4 Research Question 3 . . . 10
1.3.5 Research Question 4 . . . 11
1.3.6 Research Question 5 . . . 11
1.4 Research Scope . . . 11
1.4.1 Research Question 1: 3D UAV Path Planning . . . 12
1.4.2 Research Question 2: Real-time UAV Path Planning . . . 13
1.4.3 Research Question 3: Real-time Path Planning of UAVs in Dynamic Environments . . . 13
1.4.4 Research Question 4: Uncertainty Identification and Modelling . . . 14
1.4.5 Research Question 5: Uncertainty Integration . . . 14
1.5 Main Contributions . . . 14
1.5.1 Contributions to Path Planning Algorithms . . . 14
1.5.2 Contributions to Real-time Path Planning . . . 15
1.5.3 Contributions to Dynamic Obstacle Modelling . . . 15
1.5.4 Contributions to Uncertainty Identification and Modelling . . . 15
1.6 Thesis Outline . . . 15
References . . . 18

2 Comparison between A* and RRT Algorithms for 3D Path Planning 23
2.1 Introduction . . . 24
2.2 The A* and RRT Algorithms . . . 26
2.2.1 The A* Algorithm . . . 26
2.2.2 Rapidly-Exploring Random Tree (RRT) Algorithm . . . 26
2.3 Literature Review . . . 28
2.3.1 Graph-based Approaches . . . 29
2.3.2 Sampling-based Approaches . . . 31
2.3.3 Comparing Approaches . . . 34
2.4 Path Planning Algorithm Enhancements and Test Environment . . . 34
2.4.1 Introduction . . . 34
2.4.2 The A* Ripple Reduction Algorithm (A^*_R) . . . 34
2.4.3 The RRT Algorithm without Step Size Constraint . . . 35
2.4.4 Multiple Rapidly-Exploring Random Tree (MRRT) . . . 35
2.4.5 Smoothing Algorithm . . . 35
2.4.6 Vehicle Model Definition . . . 36
2.4.7 Experimental Scenarios . . . 37
2.5 Results . . . 37
2.5.1 Introduction . . . 37
2.5.2 A* Algorithm . . . 39
2.5.3 A* Ripple Reduction Algorithm (A^*_R) . . . 40
2.5.4 Rapidly-Exploring Random Trees (RRT) . . . 41
2.5.5 Rapidly-Exploring Random Trees (RRT) without Step Size . . . 42
2.5.6 Multiple Rapidly-Exploring Random Trees (MRRT) . . . 43
2.5.7 New Smoothing Algorithm . . . 44
2.5.8 Conclusion . . . 45
2.6 Conclusion . . . 45
References . . . 46

3 Comparison of A* and RRT in Real-time 3D Path Planning of UAVs 57
3.1 Introduction . . . 58
3.2 Real-time Path Planning Literature Review . . . 59
3.2.1 Introduction . . . 59
3.2.2 Optimisation Algorithms . . . 60
3.2.3 Graph-based Methods . . . 61
3.2.4 Sampling-based Methods . . . 62
3.2.5 Conclusion . . . 62
3.3 The A*, RRT and Smoothing Algorithms . . . 62
3.3.1 The A* Algorithm . . . 63
3.3.2 The RRT Algorithm . . . 63
3.3.3 The Smoothing Algorithm . . . 63
3.4 The Real-time Algorithm . . . 63
3.4.1 Problem statement . . . 64
3.4.2 Parameter Definition and Initiation . . . 65
3.4.3 The Move Function . . . 67
3.4.4 Main Real-time Algorithm . . . 71

3.4.5 Conclusion . . . 72
3.5 Parameter Definition and Experimental Scenario Definition . . . 72
3.5.1 Real-time Algorithm Parameter Assignment . . . 72
3.5.2 Experimental Scenarios . . . 76
3.6 Results . . . 76
3.6.1 Speed (v_{UAV}) . . . 78
3.6.2 Look-ahead Distance (d_{int_goal}) . . . 81
3.6.3 Maximum Intermediate Time ($t_{iterate\_max}$) . . . 83
3.6.4 Distance to Travel per Iterate (d_{s_step}) . . . 85
3.6.5 Maximum Time to Generate a Path ($t_{path\_gen\_max}$) . . . 87
3.6.6 Distance Reduction Factor (d_{factor}) . . . 89
3.6.7 Conclusion . . . 90
3.7 Conclusion and Future Work . . . 91
References . . . 92

4 3D Real-time Path Planning of UAVs in Dynamic Environments 99
4.1 Introduction . . . 100
4.2 Path Planning in Dynamic Environments Review . . . 101
4.2.1 Introduction . . . 101
4.2.2 Dynamic Environment Definition . . . 102
4.2.3 The Need for Dynamic Path Planning . . . 102
4.2.4 Environmental Assumptions . . . 103
4.2.5 Constraints, Obstacle and Threats (COT) Sensing and Modelling Systems . . . 104
4.2.6 Solutions . . . 106
4.2.7 Conclusion . . . 107
4.3 A*, RRT, Smoothing and Real–time Algorithms . . . 108
4.3.1 Introduction . . . 108
4.3.2 The A* Algorithm . . . 108
4.3.3 The Rapidly–Exploring Random Tree (RRT) Algorithm . . . 108
4.3.4 The Smoothing Algorithm . . . 109
4.3.5 The Real–time Algorithm . . . 109
4.3.6 Conclusion . . . 109
4.4 The Obstacle Generation Algorithm . . . 110
4.4.1 Introduction . . . 110
4.4.2 Theoretical Rationale . . . 110
4.4.3 Implementation . . . 110
4.4.4 Conclusion . . . 111
4.5 Enhancements to the Real–time Path Planning Algorithm . . . 111
4.6 Results . . . 114
4.6.1 Introduction . . . 114
4.6.2 A* Results . . . 114
4.6.3 RRT Results . . . 117
4.6.4 A* vs. RRT . . . 120
4.6.5 Conclusion . . . 122

4.7 Conclusion and Future Work . . . 123
References . . . 124

5 3D Real-time Path Planning of UAVs in Dynamic Environments in the Presence of Uncertainty 133
5.1 Introduction . . . 134
5.2 Path planning in the Presence of Uncertainty Review . . . 136
5.2.1 Introduction . . . 136
5.2.2 The Need for Path Planning in the Presence of Uncertainty . . . 136
5.2.3 Uncertainty Sources . . . 137
5.2.4 Uncertainty Modelling . . . 138
5.2.5 Uncertainty Quantification and Reduction . . . 139
5.2.6 Path planning Solutions under Uncertainty . . . 140
5.2.7 Conclusion . . . 143
5.3 A*, RRT, Smoothing and Real-time Algorithms . . . 143
5.3.1 Introduction . . . 143
5.3.2 The A* Algorithm . . . 143
5.3.3 The Rapidly-Exploring Random Tree (RRT) Algorithm . . . 144
5.3.4 The Smoothing Algorithm . . . 144
5.3.5 The Real-time Algorithm . . . 145
5.3.6 Conclusion . . . 145
5.4 Environmental Scenarios, UAV Model, Path Planner Parameter Definition and Uncertainty Modelling and Quantification . . . 145
5.4.1 Introduction . . . 145
5.4.2 Environmental Scenarios . . . 146
5.4.3 UAV Model and Path Planner Parameter Definitions and Constraints . . . 146
5.4.4 Bounded Uncertainty Definitions . . . 148
5.4.5 Uncertainty Quantification . . . 150
5.4.6 Conclusion . . . 152
5.5 Results . . . 152
5.5.1 Introduction . . . 152
5.5.2 A* and RRT with No Uncertainty . . . 153
5.5.3 A* and RRT with Uncertainty in UAV Position . . . 154
5.5.4 A* and RRT with Uncertainty in Obstacle Position and Orientation . . . 157
5.5.5 A* and RRT with Uncertainty in Obstacle Position and Orientation and UAV Position . . . 161
5.5.6 Conclusion . . . 164
5.6 Conclusion and Future Work . . . 165
References . . . 166

6 Conclusions and Recommendations 173
6.1 Conclusions . . . 174
6.1.1 Main Findings . . . 174
6.1.2 Implications to Real UAV Path Planning . . . 178
6.1.3 Summarised Conclusions . . . 179

6.2 Limitations, Future Work and Main Recommendations 180
6.2.1 Limitations and Future Work 180
6.2.2 Main Recommendations 182
6.2.3 Closing Statements 182
References 184

Acknowledgements **185**

Curriculum Vitæ **187**

List of Publications **189**
.1 Journal Papers 189
.2 Peer-Reviewed Conference Papers 189

SUMMARY

Unmanned Aerial Vehicles (UAVs) are being integrated into all spheres of life, for a wide range of application in civil, commercial and military applications in both indoor and outdoor environments. UAV onboard intelligence is a paramount requirement in the realisation of UAV Traffic Management System (UTM) and Air Traffic Management (ATM) integration. The UAV onboard intelligence requirement is more envisaged in indoor applications where the use of Global Positioning Systems (GPS) is severely restricted and more complex localisation technology is required and traffic management systems are less supportive.

For UAVs to be considered for specific tasks, their use must positively outweigh the use of other established, conventional systems. A key feature for UAVs would be a capability to perform autonomous, onboard real–time path planning. Path planning is defined as the process of automatically generating feasible and optimal paths to a predefined goal point in view of static and dynamic environmental and model constraints and uncertainties. This functionality allows UAVs to require minimal human intervention once its working environment and goals are defined. Therefore, autonomous and robust path planning is fundamental for UAVs to be considered for indoor applications in industrial, commercial, military and home applications.

The need for autonomous path planning initiated with the introduction of robotics in industrial repetitive applications several decades ago. Since then, path planning extended outside factory floors evolving from 2D to 3D, operating in both static and dynamic environments with a wide spectrum of constraints and uncertainties. Path planning algorithms for autonomous vehicles can be broadly categorised into three main categories: Graph–based or Grid–based algorithms; Sampling–based algorithms and Interpolation algorithms.

Although the use of UAVs has increased, the UAVs' potential is far from reached. This can be mainly attributed to a number of challenges that have not been fully tackled and are hindering the use of small UAVs in indoor environments. This research will focus on path planning challenges in indoor, obstacle–rich environments with no UTM availability except for goal point definitions. In such scenario's, the UAV is expected to operate using only onboard facilities. In this regard, three challenges are identified, which can be summarised as follows:

> **Construct in *real-time*, non-colliding paths from the current UAV position to a goal position using only onboard UAV resources in the presence of both *static* and *dynamic* obstacles and in the presence of *uncertainties*.**

The following research goal is formulated to address these three challenges for the realisation of path planning algorithm of UAVs in indoor environments.

Assess the performance of state-of-the-art path planning rationales in the context of UAVs operating in *3D real–time, dynamic* indoor environments in the presence of *uncertainty* and identify a customised configuration based on the application.

To tackle this research goal, five research questions are formulated:

Research Question 1: *What is the state-of-the-art in the field of path planning for UAVs in 3D and how do these algorithms compare?*

To investigate the potential of different path planning algorithms, the current state-of-the-art in all fields of engineering are considered. The literature review shows that graph–based and sampling–based methods are potential candidates for 3D UAV path planning. The most often utilised algorithms from each category, that is the A* and Rapidly– Exploring Random Tree (RRT), and their variants, namely RRT without step size constraints and the Multiple RRT (MRRT) are tested in 3D scenarios of different complexity. A path smoothing interpolation algorithm is also developed to attenuate non–optimal paths, especially for the sampling-based methods.

The same path smoothing algorithm is implemented on each path planning variant with the same parameters to offer a fair comparison. These algorithms are tested on the same set of different complexity 3D scenarios using the same computer. For comparison, the path length and the computational time are the considered performance measures.

The A* with a spectrum of resolutions, the standard RRT with different step–size constraints, RRT without step size constraints and the Multiple RRT (MRRT) with various seeds are implemented and their performance measures compared. For A*, tests show an inherent ripple in path length with change in resolution for all scenarios. This results due to the grid-based nature of the A* algorithm that creates situations in which a small increase in resolution, which theoretically shall slightly decrease the path length, effectively generates longer or shorter paths. This ripple is mitigated by randomly shifting the environment in all three dimensions by a distance varying between zero and half the distance between adjacent graph points.

Results confirm that all algorithms are able to generate a path in all scenarios for all resolutions, step sizes and seeds considered. In comparison, the A* algorithm generates shorter paths in less time with respect to RRT algorithms, although the A* algorithm only explores areas necessary for path construction while RRT algorithms explore the environment evenly. Results show that A* outperformed the RRT, both in terms of path length and path generation time in offline situations with static obstacles, with 100% success rate for both in all scenarios considered.

A* allows the environment to be discretised differently according to different exigencies of different parts of the scenario, making optimal use of resources. Oppositely, RRT and its variants are suited to generate paths efficiently in evenly distributed and focused 3D area exploration applications. Based on the results obtained, and their implication to UAV path planning, the second research question is tackled.

Research Question 2: *Can the selected path planning algorithms be applied in real-time static environments using the computational resources onboard small UAVs?*

This research question assumes that all path planning computation, sensing and environmental modelling and actuator controls must be computed onboard and in real–time. Another implication is that the path planner can only visualise the environment within the sensing distance determined by the on-board sensing systems and therefore can only construct, if possible, a path to an intermediate goal point.

For the scope of this research question, a sphere equal to the sensing range of the UAV is considered, assuming that the sensing system has a 360^o field-of-view (FOV) in all three dimensions. It is further assumed that static obstacles within the sensing range are known with certainty, while other obstacles are unknown and become visible only if the UAV moves in their direction. To simulate real-time path planning, the computational time must be less than or equal to the time needed by the UAV to move from the current position to a new position. The same test environment used to tackle Research Question 1 is used, using the same performance measures.

Results show that the A* algorithm again outperforms the RRT algorithm in both path length and computational time for all scenarios considered, with the difference increasing with scenario complexity. A* is successful 90% or more of all tests for all scenarios considered provided the look-ahead distance is at least double the distance moved per iterate. In general, the RRT algorithm results in a lower success rate than A* owing to the longer computational time required to construct intermediate paths with respect to A*.

The UAV speed, sensor range and computational power are defined based on different studies that analyse these parameters onboard a range of UAVs [1–3]. The path planning results, based on these UAV parameters, show that 3D real-time path planning can be realised using only UAV onboard systems. The results outline the best empirical values for the different parameters. The setting of these parameters will configure the 3D real-time path planning platform, optimising its performance to each particular indoor application.

Research Question 2 considered only static obstacles but in real UAV application obstacles can move and rotate, hence a dynamic environment needs to be considered to assess the usability of the developed 3D real-time UAV path planning algorithm. This requirement is investigated in the following research question:

Research Question 3: *What is the effect on path planning performance if static obstacles are replaced with dynamic obstacles?*

The inclusion of dynamic environments is external to the path planning algorithm but it can affect the path that the UAV will traverse. Dynamic obstacles within an indoor environment can be represented by symmetrical shapes. For the scope of this work, four different scenarios with different complexity are constructed. These incorporate rotating and non-rotating cubes, rotating V-shaped obstacles and static 2D planes with windows.

Both obstacle movement and orientation are considered in the dynamic environment modelling. The random obstacle movement speed is assumed to be smaller than or equal to the speed of the UAV, as otherwise obstacle avoidance is not possible.

A real-time environment with a limited range creates situations where an intermediate goal point is not available. In this regard, two different rationales are developed to mitigate this situation. In the *waiting* rationale, the UAV waits in its current position until the defined intermediate goal position becomes available. In the *moving* rationale, the intermediate goal position is moved closed to the current UAV position, consequently increasing the chances of the UAV moving closer to the final goal position. Both rationales are integrated within the A* and RRT path planning algorithms and tested in all scenarios with dynamic obstacles.

Results show that the *moving* option yields better overall results in terms of path length, computational time and success rate for A* and RRT with respect to the *waiting* option. Both A* and RRT produce similar results in relatively simple scenarios with RRT recording better results in path length, computational time and success rate. For complex scenarios, the RRT is better if time is not limited while the A* algorithm is less susceptible to time constraints. Also, as speed increases in complex scenarios the success rate drops due to lack of path planning time in both A* and RRT.

The results show that the developed 3D real-time path planning platform with both A* and RRT algorithms has potential to be used in low obstacle density dynamic obstacle scenarios. The *waiting* variant is suited in situations where safety is paramount. In home environments, this is usually the case as the UAV cannot collide with obstacles, especially if these are humans. The *moving* variant would be ideal in situations where goal achievement is more important than safety. Such situations include search and rescue.

Until now it is assumed that no uncertainties are present within the UAV systems. In real scenarios a range of uncertainties are present. In the next research question, uncertainty in a UAV operating in an indoor environment is investigated.

Research Question 4: *Do uncertainties affect 3D path planning of UAVs? If yes, how can these uncertainties be modelled?*

This research question queries whether uncertainties affect path planning of UAVs in indoor environments. This requires a thorough literature survey and consequently the identification and modelling of uncertainty sources that might affect path planning performance. For the scope of this work, only uncertainties within the UAV model and the environment (perceived through UAV onboard sensing systems), are considered. Other uncertainties, such as communication with user/s, are not considered as they are out of scope for this analysis.

Literature identifies the need for uncertainty consideration in real-time 3D UAV path planning, due to the possible negative implications on path planning performance if uncertainties are neglected. The fidelity with which uncertainties can be predicted is essential in determining the usability of the proposed path planning algorithms. Furthermore, literature portrays the bounding shapes and probabilistic distributions methods as key candidates for uncertainty modelling in UAV applications. After considering the characteristics of both methods, uncertainty is modelled using bounded shapes around the current UAV position and obstacle volume.

Once uncertainty sources are identified, estimated and modelled, the developed 3D real-time path planning algorithms are assessed in the presence of dynamic obstacles and uncertainties.

Research Question 5: *Can uncertainties be mitigated to ensure collision-free 3D path planning of UAVs in real-time in the presence of dynamic obstacles?*

The same test environment constructed to assess Research Question 3 is considered using the same real-time 3D UAV path planning platform. Tests are performed using both A* and RRT path planning algorithms with the *moving* method. Uncertainty bounds are quantified based on literature and varied between 2% and 20% for both UAV position and obstacles. Uncertainty is included by adding an offset to the actual respective parameter. The effect of each uncertainty source will be analysed independently and collectively with dynamic obstacles to identify how real-time path planning algorithms can safely operate.

Results show that both sources of uncertainty (UAV position and obstacles), deteriorate path planning performance of both A* and RRT algorithms, for all scenarios considered, with RRT exhibiting the larger effect. The concurrent inclusion of both uncertainty sources further deteriorates path planning performance. RRT results in the fastest and shortest paths, with approximately the same success rate as A*, for relatively simpler scenarios, while A* performs better in the relatively complex case. Furthermore, RRT has a higher risk of collision than A* as RRT nears obstacles more often than A*.

From the results it is confirmed that uncertainty must be considered as it has an effect on path planning performance. The accuracy with which uncertainty is modelled affects path planning performance for both path planning rationales considered.

In this thesis, each research question built on the previous one, in such a way to reach the final research goal and tackling the research challenges. In the process of assessing the path planning performance (first part of research goal) of each algorithm, the response of each method with respect to each additional complication, can be independently analysed. This knowledge can be used to guide future UAV designers in selecting the best configuration, based on their application, hence reaching the second part of the research goal.

The implementation of the developed 3D real-time path planning algorithms to configure a real UAV for autonomous 3D UAV navigation in an indoor, obstacle-rich environment is the ultimate future goal that can lead into the commercialisation of this system for use in domestic applications. Moreover, this real-time 3D UAV path planning system can be proposed for integration in outdoor UAVs. Finally, this dissertation's ultimate aim is to contribute in reducing the gap that still exists to integrate UAVs into domestic environments with the aim of improving current and future services that rely on UAVs with the ultimate aim of enhancing people in their daily lives.

REFERENCES

[1] Hrabar, S., "3D Path Planning and Stereo-based Obstacle Avoidance for Rotorcraft UAVs", *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nice, France, 22-26 Sep. 2008, pp. 807-814.

[2] Yao, P., Wang, H. and Su, Z., "Real-time path planning of unmanned aerial vehicle for target tracking and obstacle avoidance in complex dynamic environment," *Aerospace Science and Technology*, Vol. 47, pp. 269-279, 2015.

[3] Ferguson, D. and Stentz, A. "Using interpolation to improve path planning: The field D* algorithm", *Journal of Field Robotics*, Vol. 23, No. 2, pp. 79-101, 2006.

SAMENVATTING

Onbemande luchtvaartuigen (UAV's) worden steeds verder geïntegreerd in de samenleving. UAVs worden gebruikt voor een breed scala aan civiele, commerciële en militaire toepassingen, zowel binnenshuis als buiten. Voor integratie van UAVs in een UAV-verkeersleidingssyteem (UTM) en integratie in bestaande luchtverkeersmanagement systemen (ATM) is een mate van intelligentie aan boord van de UAV vereist. De noodzaak voor deze intelligentie is sterker voor toepassingen binnen in gebouwen, waar ondersteuning van luchtverkeersleidingssystemen minder is en het gebruik van positiebepalingssystemen zoals GPS beperkt is, waardoor meer complexe lokaliseringstechnieken nodig zijn.

Voordat inzet van een UAV voor een bepaalde taak overwogen wordt, moeten eerst de voordelen daarvan ten opzichte van conventionele systemen afgewogen worden. Een belangrijke eigenschap van een UAV is de mogelijkheid om autonoom en real-time het pad te plannen dat gevlogen dient te worden. Pad-planning is gedefinieerd als het automatisch genereren van optimale paden tot een van te voren gedefinieerd doel, onder invloed van statische en dynamische onzekerheden en beperkingen in de UAV en in de omgeving. Deze functionaliteit staat toe dat minimale interventie van menselijke bestuurders nodig is zodra de omgeving en het doel gedefinieerd zijn. Om deze reden is het autonoom en robuust plannen van het te vliegen pad een fundamentele eigenschap van UAV's die bepaalt of de UAV ingezet kan worden voor toepassingen binnenshuis, zowel industrieel, commercieel, militair, als in de thuissituatie.

De vraag naar autonome pad-planningsalgoritmes kwam op gang door de introductie van robots in industriële toepassingen enkele decennia geleden. Sindsdien hebben pad-planningsalgoritmes zich verspreid buiten de fabriekstoepassingen, van 2-dimensionaal naar 3-dimensionaal, zowel in statische als dynamische omgevingen met een breed scala aan beperkingen en onzekerheden. Pad-planningsalgoritmes voor autonome voertuigen kunnen in drie hoofdcategorieën verdeeld worden: Graaf-methodes, rooster-methodes en interpolatiemethodes.

Ondanks de groei in het gebruik van UAV's, is het volledige potentieel van UAV's nog lang niet benut. Dit kan voornamelijk verklaard worden door een aantal uitdagingen die nog niet volledig aangepakt zijn en die het gebruik van kleine UAV's binnenshuis beperken. Dit onderzoek legt de focus op de uitdaging van binnenshuis pad-plannen in omgevingen met veel obstakels en een gebrek aan verkeersleidingsystemen, afgezien van een definitie van de doel-positie van de UAV. In zulke scenario's wordt de UAV geacht zelfstandig te vliegen door alleen gebruikt te maken van systemen aan boord van de UAV. In dit licht kunnen drie uitdagingen geïdentificeerd worden, welke als volgt samengevat kunnen worden:

Creëer, in *real-time*, een pad zonder botsingen van de huidige UAV positie naar een doel-positie, waarbij alleen systemen aan boord van de UAV gebruikt mogen worden, in een omgeving met zowel *statische* als *dynamicsche* obstakels en *onzekerheden*.

Het volgende onderzoeksdoel is geformuleerd om deze drie uitdagingen aan te pakken en daarmee pad-plannen van UAV's binnen gebouwen mogelijk te maken.

Beoordeel de prestaties van state-of-the-art pad-planningsalgoritmes in de context van UAV's in een *3-D real-time, dynamische* omgeving met *onzekerheden* en identificeer een configuratie op basis van de toepassing.

Om dit onderzoeksdoel te bereiken zijn vijf onderzoeksvragen geformuleerd:

Onderzoeksvraag 1: *Wat is de state-of-the-art op het gebied van 3-D pad-planning voor UAV's en hoe vergelijken deze algoritmes zich tot elkaar?*

Om het potentieel van de verschillende pad-planningsalgoritmes te onderzoeken, wordt de state-of-the-art in alle technische vakgebieden overwogen. Het literatuuronderzoek laat zien dat graaf-methodes en bemonsterings-methodes potentiële kandidaten zijn voor 3-D UAV pad-planning. Het meest gebruikte algoritme uit ieder van deze categorieen, A* en Rapidly– Exploring Random Tree (RRT) en hun varianten, de RRT zonder stapgrootte-beperking en Multiple RRT (MRRT), worden getest in 3-D scenario's van verschillende complexiteit. En pad-afvlak interpolatiealgoritme is ontwikkeld om niet-optimale paden te verminderen, in het bijzonder voor de bemonsterings-methodes.

Hetzelfde pad-afvlak interpolatiealgoritme is toegepast op elke pad-planning variant met dezelfde parameters, om zo een eerlijke vergelijking te kunnen maken. Deze algoritmes zijn getest op dezelfde set van 3-D scenario's op dezelfde computer. In de vergelijking worden de pad lengte en de rekentijd beschouwd als maat voor de prestatie van het algoritme.

Het A* algoritme, met een set van resoluties, de standaard RRT met verschillende stapgrootte, de RRT zonder stapgroottebeperking en de MRRT zijn geïmplementeerd en de prestaties zijn vergeleken. Testen met A* laten voor alle scenario's een rimpelachtige pad lengte zien met veranderingen in resolutie. Dit wordt veroorzaakt door het rooster waar het algoritme op gedefinieerd is, waardoor een kleine toename in resolutie, die theoretisch de pad lengte zou moeten verkleinen, effectief een langer pad geeft. Het effect van deze rimpel in pad lengte is verminderd door het rooster een willekeurige afstand (ergens tussen 0 en de halve afstand tussen punten in het rooster) te verplaatsten in alle drie dimensies.

De resultaten bevestigen dat alle algoritmes een pad kunnen genereren voor alle scenario's, voor iedere resolutie, stapgrootte, en MRRT startpunt. A* genereert kortere paden in minder tijd vergeleken met RRT algoritmes, ook al ontdekt A* alleen gebieden die noodzakelijk zijn voor de constructie van het pad, terwijl RRT de omgeving gelijkmatig ontdekt. Resultaten laten zien dat A* beter presteert dan RRT, zowel in pad lengte als

padconstructie-tijd, in offline situaties met statische obstakels, met een 100% succeskans voor beide algoritmes in alle scenario's.

A* staat toe dat de omgeving gediscretiseerd wordt naar gelang de behoefte van verschillende delen van het scenario en maakt daarmee optimaal gebruikt van de middelen. In tegenstelling zijn RRT en varianten geschikt om efficiënt paden te construeren in gelijk verdeelde 3-D toepassingen. Op basis van de resultaten en de impact hiervan op UAV pad-planning, wordt de tweede onderzoeksvraag opgesteld.

> Onderzoeksvraag 2: *Kunnen de gekozen pad-planningsalgoritmes toegepast worden in real-time statische omgevingen door alleen gebruik te maken van systemen aan boord van kleine UAV's?*

Deze onderzoeksvraag gaat ervan uit dat alle berekeningen, metingen en modellering van omgeving aan boord in real-time gedaan moeten worden. Bovendien kan de pad-planning alleen een beeld vormen van de omgeving die binnen het bereik van de sensoren valt en daarmee alleen een pad plannen tot een tussentijdse doel-locatie.

Voor deze onderzoeksvraag word een bol met straal gelijk aan het bereik van de sensoren van de UAV gekozen, met als aanname dat het bereik van de sensoren in alle richtingen gelijk is. Bovendien wordt aangenomen dat statisch obstakels binnen het bereik van de sensoren met zekerheid bekend zijn, terwijl overige obstakels onbekend zijn en pas zichtbaar worden als de UAV dichtbij genoeg is. Om real-time pad-planning te garanderen moet de rekentijd kleiner zijn dan de tijd die de UAV nodig heeft om het pad te vliegen. Dezelfde testomgeving als bij onderzoeksvraag 1 wordt gebruikt, met dezelfde prestatiemetrieken.

Resultaten laten zien dat A* opnieuw voor alle scenario's beter presteert dan RRT, in zowel pad lengte als rekentijd, waarbij het verschil groter wordt bij toenemende complexiteit van het scenario. A* is voor 90% of meer van alle testen succesvol mits het sensorbereik ten minste tweemaal de afstand is die gevlogen wordt per iteratie. In het algemeen heeft RRT een lagere succeskans dan A* doordat de rekentijd voor het construeren van het pad langer is dan bij A*.

De snelheid, het sensorbereik en de beschikbare rekenkracht van UAV's zijn gedefinieerd volgens verschillende studies die deze eigenschappen bestudeerd hebben voor een scala aan UAV's[1–3]. De pad-planning resultaten, op basis van deze UAV parameters, laten zien dat 3-D real-time pad-planning gerealiseerd kan worden met enkel gebruik van systemen aan boord van de UAV. De resultaten geven richtlijnen voor de empirische waarden van deze parameters. De waardes van deze parameters configureren het 3-D real-time pad-planningsplatform, waarbij de prestaties geoptimaliseerd worden per toepassing van binnenshuis vliegen.

Onderzoeksvraag 2 beschouwt alleen statische obstakels, maar in werkelijkheid kunnen obstakels ook bewegen en roteren, dus een dynamische omgeving is nodig om de bruikbaarheid van het ontwikkelde 3-D pad-planningsalgoritme te evalueren. Deze voorwaarde is in de volgende onderzoeksvraag verder onderzocht:

> Onderzoeksvraag 3: *Wat is het effect op de prestatie van het pad-planningsalgoritme als statische obstakels worden vervangen door dynamische?*

Het toevoegen van dynamische omgevingen is extern aan het pad-planningsalgoritme, maar het kan het pad beïnvloeden dat de UAV af gaat leggen. Dynamische obstakels binnenin een gebouw kunnen gerepresenteerd worden door symmetrische vormen. Voor de scope van dit werk zijn vier verschillende scenario's met verschillende complexiteit geconstrueerd. Deze bestaan uit roterende en niet-roterende kubussen, roterende V-vormen en statische 2-D vlakken met raamvormige openingen.

Zowel de oriëntatie als de beweging van obstakels worden meegenomen in de modellering van de dynamische omgeving. De snelheid van obstakels is willekeurig gekozen, maar kleiner dan of gelijk aan de snelheid van de UAV, omdat ontwijken van de obstakels anders niet mogelijk is.

Een real-time omgeving met een beperkt sensorbereik kan situaties opleveren waar een tussentijds doel-locatie niet beschikbaar is. In deze situatie zijn er twee mogelijkheden tot een oplossing. In de *wacht-optie* zal de UAV wachten in de huidige positie totdat de tussentijdse doel-locatie beschikbaar wordt. In de *beweeg-optie* wordt de tussentijdse doel-locatie dichterbij de UAV gebracht, waarmee de kans groter wordt dat de UAV dichter bij het einddoel komt. Beide opties zijn geïntegreerd in zowel A* als RRT en getest op alle scenario's met dynamische obstakels.

Resultaten laten zien dat de *beweeg-optie* in het algemeen betere resultaten geeft ten opzichte van de *wacht-optie* in termen van pad lengte, rekentijd en succesratio voor A* en RRT. Zowel A* als RRT produceren soortgelijke resultaten in simpele scenario's, waar RRT betere resultaten heeft in termen van pad lengte, rekentijd en succesratio. For complexe scenario's is RRT beter als er geen tijdslimiet is, terwijl A* minder gevoelig is voor tijdsbeperkingen. Wanneer de snelheid van de UAV toeneemt in complexe scenario's, daalt de succesratio voor zowel A* als RRT door gebrek aan planningstijd.

De resultaten laten zien dat de ontwikkelde pad-planningsalgoritmes op basis van A* en RRT de potentie hebben om gebruikt te worden in scenario's met lage obstakeldichtheid. De *wacht-optie* is geschikt in situaties waar veiligheid het meest belangrijk is. Dit is vaak het geval in toepassingen binnenin gebouwen, aangezien de UAV hier niet met obstakels mag botsen, vooral als dit mensen zijn. De *beweeg-optie* is ideaal wanneer het bereiken van de doel-locatie belangrijker is dan veiligheid. Zulke situaties bestaan bijvoorbeeld uit zoek- en reddingsvluchten.

Tot nu toe is aangenomen dat er geen onzekerheden zijn in de UAV systemen. In echte toepassingen zijn echter meerdere soorten onzekerheden. In de volgende onderzoeksvraag zal onzekerheid in UAV systemen worden onderzocht.

> Onderzoeksvraag 4: *Hebben onzekerheden invloed op de pad-planning van UAV's? Zo ja, hoe kunnen deze onzekerheden gemodelleerd worden?*

Deze onderzoeksvraag bekijkt of onzekerheden effect hebben op pad-planning van UAV's voor toepassingen binnen gebouwen. Dit vereist een grondige literatuurstudie waaruit de bronnen van onzekerheden geïdentificeerd en gemodelleerd kunnen worden. Voor de scope van dit onderzoek worden alleen onzekerheden onderzocht in het UAV model en in de omgeving, zoals waargenomen door de sensoren aan boord van de UAV. Overige onzekerheden, zoals onzekerheden in communicatie met gebruikers, worden niet beschouwd.

Uit de literatuur blijkt dat er een noodzaak is tot het meenemen van onzekerhe-

den, aangezien deze negatieve gevolgen voor de prestatie van het 3-D real-time pad-planningsalgoritme hebben. De betrouwbaarheid van de voorspelling van onzekerheden is essentieel voor het bepalen van de bruikbaarheid van een pad-planningalgoritme.

De literatuur laat twee categorieën van onzekerheidsmodellering zien, enerzijds door het modelleren van de limieten van de onzekerheid, anderzijds door het modelleren van de distributie van onzekerheid. Na de karakteristieken van deze twee categorieën vergeleken zijn, is gekozen voor het modelleren van de limieten van de onzekerheden met bepaalde geometrische vormen rondom de huidige UAV positie en rondom de obstakels.

Wanneer de onzekerheden geïdentificeerd, geschat en gemodelleerd zijn, kunnen de 3-D real-time pad-planningsalgoritmes geëvalueerd worden in omgevingen met dynamische obstakels en onzekerheden.

> Onderzoeksvraag 5: *Kan het effect van onzekerheden verminderd worden zodat er gegarandeerd geen botsingen optreden in het 3-D real-time pad-planningsalgoritme wanneer er dynamische obstakels zijn?*

Voor onderzoeksvraag 5 wordt dezelfde testomgeving gebruikt als bij onderzoeksvraag 3, met hetzelfde 3-D real-time UAV pad-planningsplatform. Testen zijn uitgevoerd met zowel A* als RRT met de *beweeg-optie*. Onzekerheidslimieten zijn vastgesteld op basis van literatuur en variëren van 2% tot 20% voor zowel UAV-positie als obstakel-positie. De onzekerheid is meegenomen door een offset toe te voegen aan de echte parameter. Het effect van elke onzekerheid wordt zowel onafhankelijk als collectief met dynamische obstakels onderzocht, om zo te identificeren hoe real-time pad-planningsalgoritmes veilig kunnen werken.

De resultaten van de testen laten zien dat beide onzekerheden (UAV- en obstakel positie) zorgen voor een slechtere prestatie van zowel A* als RRT, voor alle scenario's, waarbij er op RRT en groter effect is. Het tegelijkertijd meenemen van beide onzekerheden verslechtert de prestatie nog meer. RRT geeft de snelste en kortste paden voor de simpele scenario's, met ongeveer dezelfde succeskans als A*, terwijl A* beter presteert in complexere scenario's. Bovendien heeft RRT een grotere kans op botsingen met obstakels en nadert RRT de obstakels dichter en vaker dan A*.

De resultaten bevestigen dat onzekerheden meegenomen moeten worden, aangezien deze een effect hebben op de prestaties van het pad-planningsalgoritme. De nauwkeurigheid waarmee onzekerheden gemodelleerd worden heeft effect op de prestaties van beide pad-planningsalgoritmes.

In dit proefschrift bouwt iedere onderzoeksvraag voort op de voorgaande, om zo tot het hoofddoel te komen. Door de prestaties van ieder algoritme te evalueren voor iedere toevoeging van complexiteit/realisme, kan het effect hiervan onafhankelijk geanalyseerd worden. Deze kennis kan gebruikt worden door toekomstige UAV-ontwerpers om de configuratie te selecteren die het beste past bij de toepassing.

De implementatie van de ontwikkelde 3-D real-time pad-planningsalgoritmes om een echte UAV te configureren voor navigatie in een obstakelrijke omgeving binnen een gebouw, is het ultieme doel wat kan zorgen voor commercialisering van dit systeem voor huiselijke toepassingen. Bovendien kan dit 3-D real-time UAV pad-planningssysteem ook gebruikt worden voor toepassingen buitenshuis. Tot slot is het ultieme doel van dit proefschrift om de integratie van UAV's in thuissituaties dichterbij te brengen, met als

doel om toekomstige diensten en taken van UAV's mogelijk te maken die het dagelijkse leven van mensen kunnen verbeteren.

REFERENCES

[1] Hrabar, S., "3D Path Planning and Stereo-based Obstacle Avoidance for Rotorcraft UAVs", *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nice, France, 22-26 Sep. 2008, pp. 807-814.

[2] Yao, P., Wang, H. and Su, Z., "Real-time path planning of unmanned aerial vehicle for target tracking and obstacle avoidance in complex dynamic environment," *Aerospace Science and Technology*, Vol. 47, pp. 269-279, 2015.

[3] Ferguson, D. and Stentz, A. "Using interpolation to improve path planning: The field D* algorithm", *Journal of Field Robotics*, Vol. 23, No. 2, pp. 79-101, 2006.

NOMENCLATURE

Acronyms

$2D$	2-Dimensional
$3D$	3-Dimensional
A^*_R	A* Ripple Reduction Algorithm
$AD*$	Anytime Dynamic A*
$AD–RRT$	Anytime Dynamic Rapidly-exploring Random Tree
$ARA*$	Anytime Repairing A*
ATM	Air Traffic Management
$ATRA*$	Anytime Tree Restoring A*
$AT–RRT$	Anytime Transition–based Rapidly-exploring Random Tree
AUV	Autonomous Underwater Vehicle
$BVLOS$	Beyond Visual Line Of Sight
$CL–RRT$	Closed Loop Rapidly-exploring Random Tree
$CORUS$	Concept of Operation for European Unmanned Aerial Vehicle Traffic Management Systems
$COTs$	Constraints, Obstacles and Threats
$CPRM$	Cell–based Probabilistic Roadmap Method
$D*$	Dynamic A*
$DARPA$	Defense Advanced Research Projects Agency
$DDRRT$	Dynamic Domain Rapidly–Exploring Random Tree
DoF	Degrees of Freedom
DRM	Dynamic Roadmap
$DRRT$	Dynamic Rapidly-exploring Random Tree

ERRT	Execution Extended Rapidly–Exploring Random Tree
FAA	Federal Aviation Administration
FADPRM	Flexible Anytime Dynamic Probabilistic Roadmap Method
FOV	Field of View
GA	Genetic Algorithm
GNC	Guidance, Navigation and Control
GPS	Global Positioning System
GRIP	Greedy Incremental Path–Directed
ICAO	International Civil Aviation Organisation
IFDS	Interfered Fluid Dynamical System
IkRRT	Iterative k–Nearest Rapidly-exploring Random Tree
JPS	Jump Point Search
KIAST	Korea Institute of Aviation Safety Technology
LGVF	Lyapunov Guidance Vector Field
LPA∗	Lifelong Planning A*
LQG	Linear Quadratic Gaussian
LQGMP	Linear Quadratic Gaussian Motion Planning
LRF	Lazy Reconfigurable Forest
MAPF	Modified Artificial Potential Field
MDP	Markov Decision Processes
MILP	Mixed Integer Linear Programming
MIRRT	Multiple Incremental Rapidly-exploring Random Tree
MPC	Model Predictive Control
MP–RRT	Multipartite Rapidly-exploring Random Tree
MRRT	Multiple Rapidly–Exploring Random Tree
NASA	National Aeronautics and Space Administration
OBBs	Oriented Bounding Boxes
PC	Personal Computer

$POMDP$	Partially Observable Markov Decision Process
PRM	Probabilistic Roadmap Method
PSO	Particle Swarm Optimisation
QVG	Quantized Visibility Graph
RET	Rapidly Exploring Evolutionary Trees
RHC	Receding Horizon Control
$RHIST$	Finite Receding-Horizon Incremental-Sampling Tree
RRF	Reconfigurable Random Forests
RRM	Rapidly-exploring Random Tree Roadmap
RRT	Rapidly–Exploring Random Tree
$SESAR$	Single European Sky Air Traffic Management Research
$SLAM$	Simultaneous Localisation and Mapping
SOP	Standard Operating Procedures
SSD	Shell Space Decomposition
TCL	Technical Capability Level
$TD*$	Truncated Incremental Dynamic A*
$TLPA*$	Truncated Incremental Lifelong Planning A*
$T–RRT$	Transition–based Rapidly-exploring Random Tree
UAS	Unmanned Aircraft Systems
UAV	Unmanned Aerial Vehicle
$UOMS$	Civil Unmanned Aircraft System Operation Management System
USV	Unmanned Surface Vehicles
UTM	Unmanned Aerial Vehicle Traffic Management System
$VTOL$	Vertical, Take-Off and Landing

Mathematical Parameters

$\%_u$	Percentage uncertainty
Γ	Successor operator in A*

$\hat{f}(n)$	Estimated cost of $f(n)$ in A*
$\hat{g}(n)$	Estimated cost of $g(n)$ in A*
$\hat{h}(n)$	Estimated cost of $h(n)$ in A*
τ	Tree nodes and their associated edges from the starting node in RRT
$bound_{max}$	Maximum bound increase
c_i	Cost in A*
D	Preset tree branch length in RRT
d_{env_space}	Distance of one axis in the environmental space
d_{factor}	Distance reduction factor
$d_{int\_(i)-to-(i+1)}$	Distance between the smoothed path points i^{th} to $(i+1)^{th}$
d_{int_goal}	Distance between the current UAV position and the prospective intermediate goal point
d_{int_s-to-g}	Distance between the current UAV position and the final goal point
d_{mov}	Distance the UAV will move from the current position. The new point will be the current UAV position at the next iteration
d_{obst_centre}	Distance from the centre of the obstacle to vertex
d_{obst_vertex}	Distance from the centre to the new vertex
d_{plane_vertex}	Distance from an original plane vertex to the new vertex
$d_{s\_step\_part}$	Distance to be moved by the UAV ($d_{s\_step}-d_{total}$) from $path_{smooth\_pre}(i)$
d_{s_step}	Distance covered by the UAV in every iterate
d_{step_RRT}	Step size in RRT
d_{total}	Distance from s_{cur} to a previous smoothed path point in $path_{smooth\_pre}(i)$
env_{para}	Environmental parameters namely the size of the environmental space and the size and position of obstacles in the environmental space
$f(n)$	Evaluation function in A*
$flag_{path}$	Path possibility flag
$flag_{small}$	A Boolean parameter that denotes whether the distance between d_{s_step} is smaller than $path_{smooth\_pre}(1)$ and $path_{smooth\_pre}(2)$
$g(n)$	Actual cost from starting point to node n in A*

$goal_{int}$	Intermediate goal point in the next iterate
$h(n)$	Actual cost from node n to the goal point of n in A*
K	Maximum preset iterates to construct a tree from start to goal in RRT
l_{near}	Nearest point on edge to x_{rand} in RRT
lim	Limits of 3D boundaries
n	Generic node in A*
n_{path_points}	Number of points in the previously generated smoothed path
$path_{smooth\_pre}$	Constructed path in the previous iteration
$path_{smooth}$	All points of the smoothed path generated in the current iterate
r_{u_sphere}	Radius of the uncertainty sphere
res	Resolution in A*
s_{cur_new}	Next iterate UAV's current UAV position
s_{cur_part}	Previous smoothed path point prior which the total distance from s_{cur} was smaller than d_{s_step}
s_{cur}	Current UAV position
$t_{iterate\_max}$	Maximum time to generate a path between iterates
$t_{path\_gen\_max}$	Maximum time allocated to reach the goal point from the start position
$time_{iterate}$	Iterate time
v	Vertex
v_{UAV}	UAV speed
x_{edge}	Nearest node on edge in RRT
X_{free}	Obstacle-free environment space in RRT
x_{goal}	Goal point in A* and RRT
x_{init}	Starting point in A* and RRT
x_{near}	Nearest node to x_{rand} in RRT
x_{new}	New node of tree a D distance from x_{near} or l_{near} equal to the preset tree branch length in RRT
x_{rand}	Random node selected to construct new branches in RRT

1

INTRODUCTION

One, remember to look up at the stars and not down at your feet. Two, never give up work. Work gives you meaning and purpose and life is empty without it. Three, if you are lucky enough to find love, remember it is there and don't throw it away.

Stephen Hawking

1

1.1. BACKGROUND

1.1.1. DEFINITIONS

Unmanned or Uninhabited Aerial Vehicles (UAVs), Unmanned Aircraft or more commonly known as drones are defined by the International Civil Aviation Organisation (ICAO) as *"pilotless aircraft, in the sense of Article 8 of the Convention on International Civil Aviation, which is flown without a pilot-in-command on-board and is either remotely and fully controlled from another place (ground, another aircraft, space) or programmed and fully autonomous"* [1], page B-6. Moreover, Unmanned Aircraft Systems (UAS) extend beyond the aircraft and incorporate all associated elements necessary to operate efficiently and safely an aircraft without a pilot on board [2].

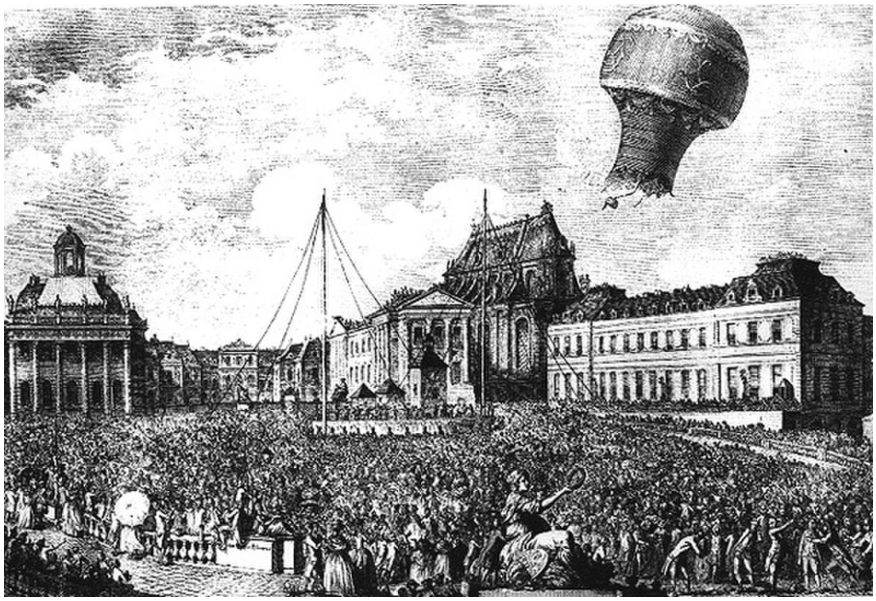
1.1.2. NEED FOR PATH PLANNING

The definition of what constitutes a UAV incorporates a wide spectrum of vehicles that have been, are currently existing and are being proposed to be used in a wide range of civil, commercial and military applications. The first unmanned flight of about 10 minutes with a range of 2km and altitude of 5,200 to 6,600ft (not within ICAO definition) was demonstrated to a crowd of dignitaries in Annonay, France on 4th June, 1783, by Joseph-Michel and Jacques-Étienne Montgolfier [3]. Later, on the 19th September, 1783 the flight was repeated in Versailles in the presence of the French king, Marie Antoinette and a crowd of 130,000, see Figure 1.1b. This time the flight lasted 8 minutes, travelled for 3.2km and carried a sheep, a duck and a rooster [3].

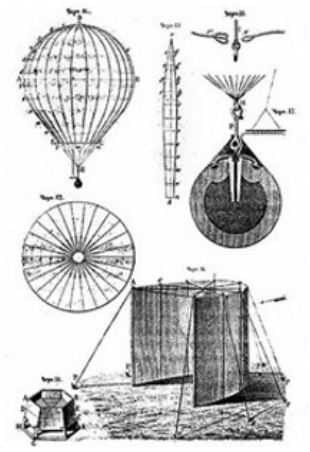
In military applications, the use of unmanned aerial vehicles (not within ICAO definition) have been recorded since 1849 when the Austrian army used balloons carrying explosives to attack Venice [4, 5]. Figure 1.1b illustrates an artistic impression of their design. These vehicles were not controlled and the first controlled pilotless aircraft was developed during World War I [4] and illustrated in Figure 1.1c. Since then the use, performance, capabilities and applications in military have increased. On the civil and commercial side, the price drop of UAVs (to less than $100 in 2016 [4]), owing to the increasing demand and reduced production costs, have resulted in a surge in UAV numbers over the last two decades. The Federal Aviation Administration (FAA) noted that by 10th March 2020 in the US, over 1.1M and 440k UAVs are registered for recreational and commercial use, respectively [6]. Furthermore, over 170,000 persons are certified remote pilots [6].

These ever-increasing numbers are a consequence of the advantages that these vehicles have in comparison to more traditional ways of doing the same task. But, the integration of UAVs into the airspace requires robust guidance, navigation and control (GNC) systems, because of the increasing numbers but also due to the wide variety of sizes, shapes, performance, capabilities, accuracy, precision of the vehicle, and last but not least, the abilities of the UAV pilot. The resultant UAV Traffic Management System (UTM) must ensure safe navigation in view of uncertainties for all UAVs in the presence of piloted aircraft, ground constraints (buildings, ground vehicles, trees) and people present within the volume the UTM system is expected to manage.

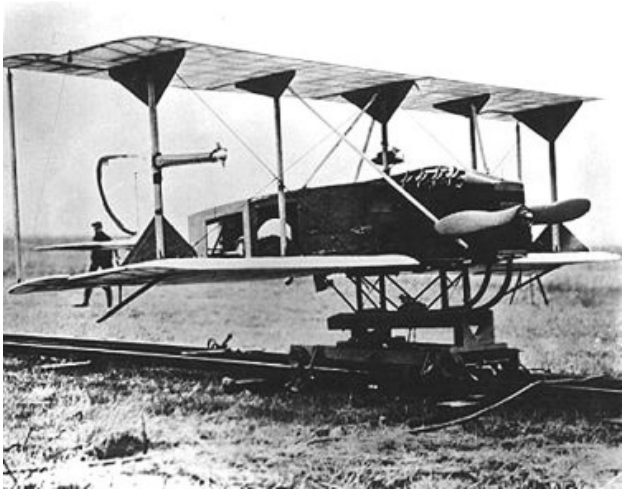
In the USA, the National Aeronautics and Space Administration (NASA) embarked into the UAS Traffic management project in partnership with the FAA, other federal agencies, academics and research partners with the aim of understanding the requirements



(a) First Hot Air Balloon flight



(b) Unmanned Austrian Balloons



(c) Hewitt-Sperry Automatic Airplane

Figure 1.1: Early UAVs: An artistic impression of the first balloon flight with non human passengers (a sheep, a duck and a rooster) that took off on September 19, 1783 © National Air and Space Museum, Smithsonian Institution, (b) An artistic impression of the design of the unmanned Austrian balloons that attached Venice © Prof. Jurij Drushnin, Monash University and (c) The Hewitt-Sperry Automatic Airplane developed in 1916 © warhistoryonline.com

to safely integrate UAVs in complex low-altitude airspace [7]. The envisioned system is expected to provide situational awareness, enhance communication with UAV operators, allow UAV pilots to submit flight plans and consequently determine the safety of the proposed flight plan [8]. Figure 1.2 illustrates the proof of concept of this proposed system. The project is currently in the final stages of completion having started Technical Capability Level (TCL) 4 testing in May 2019 [7]. Owing to the importance and success, this project was awarded, in June 2020, the 2020 NASA Government Invention of the Year. Similar projects are currently pursued in other countries such as the Concept of Operation for European Unmanned Aerial Vehicle Traffic Management Systems (CORUS) within SESAR in the EU [9], the Civil Unmanned Aircraft System Operation Management System (UOMS) in China [10] and the Low-altitude UTM Development and Demonstration Test for Safe Operation of UAS developed by the Korea Institute of Aviation Safety Technology (KIAST) in South Korea [11].

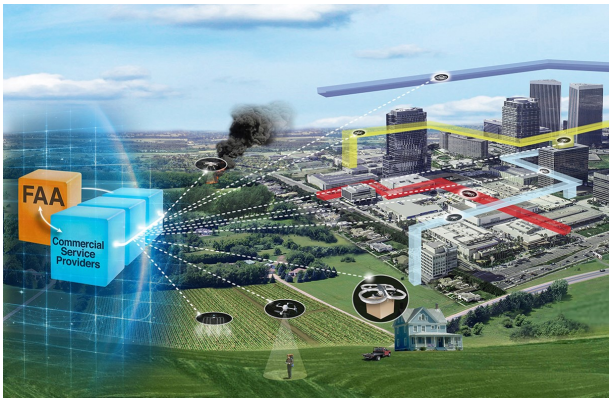


Figure 1.2: NASA's Drone Traffic Management System © NASA

The UTM framework must automatically assign safe, time-stamped corridors to different UAV operators and/or UAV autonomous systems. The integration of such a UTM framework within the existing Air Traffic Management (ATM) systems requires UAVs to operate safely beyond the visual line of sight (BVLOS). Therefore, UAVs must be equipped with autonomous onboard or ground board support systems that assist the UAV to operate safely within the assigned environment. UAS research over the last two decades focused on enhancing UAS automation capabilities, with improvements expected to mature in the next two decades [12]. The study by Sigala *et al.* [12] discusses prospective future UAS capabilities between experts in operations, acquisitions and academia. It concludes that research on UAS capabilities should focus on intelligence, surveillance, and reconnaissance, datalinks and sense and avoid technologies [12].

UAV onboard intelligence is a paramount requirement in the realisation of UTM and ATM integration. The UAV onboard intelligence requirement is more envisaged in indoor applications where the use of Global Positioning Systems (GPS) is restricted [13, 14] and more complex localisation technology is required [15, 16]. Furthermore, the UTM framework capabilities which require customisation may be either less supportive or not available at all since its funding is more restrictive than a national UTM framework. On

the positive side, in indoor applications UAVs are not subject to weather conditions [13] and governmental flying restrictions [17]. The current and prospective use of UAVs in indoor applications vary from industrial setups such as item delivery in different parts within the same factory building [18] or within the same factory area requiring both indoor and outdoor capabilities [19] (Figure 1.3a) to home assistance such as window cleaning [20] (Figure 1.3b).

(a) Spare parts delivery within the same factory

(b) Window cleaning drone

Figure 1.3: UAV Applications: (a) A drone supplying spare parts in ZF which is the first company in Germany to receive official approval for automated UAV flights over factory premises © ZF and (b) A window cleaning UAV that can be used for both commercial and personal use © AERONES

In indoor environments the use of UAVs in specific tasks is mainly determined by the owners and/or users. For UAVs to be considered for specific tasks, their use must positively outweigh the use of other established, conventional systems. One key feature in the intelligence category that makes the UAV option attractive is autonomous, onboard real-time path planning.

Path Planning Definition

Path planning is defined as the process of automatically generating feasible and optimal paths to a predefined goal point, in the presence of static and dynamic environmental and model constraints and uncertainties [21].

This functionality allows UAVs to require minimal human intervention once its working environment and goals are defined, reducing operational costs. It would allow making more efficient use of the upper parts of the floor area which are otherwise rarely used, reducing traffic on the floor area thus increasing the working potential there. In fact, real-time path planning is considered as a desirable feature [22], a requirement [23] and even paramount [24] for real-time autonomous manoeuvring of vehicles let alone UAVs in real, dynamic environments [25]. In conclusion, autonomous and robust path planning is fundamental for UAVs to be considered for indoor applications in industrial, commercial and home applications.

1.1.3. HISTORY OF PATH PLANNING

The need for autonomous path planning emerged with the introduction of robotics in industrial repetitive applications. As the drive for enhanced industrial automation increased to cater for more complex tasks, the GNC capabilities improved over the last few decades shifting from 2-Dimensional (2D) setups [26] to 3-Dimensional (3D) ones [27]. The use of robotics was extended to outside factory floors and finally to homes with for example cleaning robots since 2001 [28]. In such transition, path planning algorithms evolved to operate in the presence of only fixed obstacle [29, 30] to dynamic obstacle environments [31, 32] with and without *a priori* knowledge of the environment. The use of these path planning algorithms can be extended from robotics to UAVs, but UAVs must often operate within a wider spectrum of time variant and time invariant constraints and uncertainties [21].

Path planning algorithms for autonomous vehicles can be broadly categorised into three main categories:

1. Graph-based or Grid-based algorithms,
2. Sampling-based algorithms, and
3. Interpolation algorithms. [21]

Graph-based or grid-based algorithms were first introduced by the Dijkstra algorithm in 1959 [33]. Since then the A* algorithm and its numerous variants were developed to enhance the path planning performance and applicability of graph-based algorithms. These algorithms define the state-space into an occupancy grid and define obstacles as inaccessible grid positions. Then algorithms search the available grid points giving a solution if a path exist [34].

Sampling-based methods were first introduced through the Probabilistic Roadmap Method (PRM) [35] followed by the Rapidly-Exploring Random Tree (RRT) [36] and their respective numerous variants by creating cyclic and acyclic random graphs, respectively. These algorithms select a non-structural finite number of points in the configuration space and create connections between these points [37, 38].

The path constructed by both graph-based and sampling-based methods requires path optimisation to cater for vehicle, mission and environmental kinematic and dynamic constraints [34, 39]. Interpolation methods have been developed as a post-path planning stage, to improve and/or customise trajectory based on the application.

Since the development of the Dijkstra algorithm, path planning algorithms have improved in terms of their capabilities towards optimisation, flexibility and versatility. However, a number of challenges still remain for UAVs to be safely operated within indoor environments in the presence of time variant constraints and uncertainties.

1.2. CHALLENGES

Although the use of UAVs in industrial, commercial and personal applications has increased over the past decades, their potential is far from reached. The relatively slow integration into everyday life can be attributed to a number of challenges that have not been fully tackled and are hindering the use of small UAVs in indoor environments.

This thesis focuses on path planning challenges in indoor, obstacle-rich environments with no UTM availability except for target point assignment. Thus, apart from knowing 'where to go', the UAV is assumed to operate using only onboard facilities.

1.2.1. CHALLENGE 1

Challenge 1

Construct in *real-time*, non-colliding paths from the current UAV position to a target position in initially unknown environments using only onboard UAV resources in the presence of static obstacles.

In unknown environments, real-time path planning is essential as the obstacle sensing systems have a limited range and the target position may not be within the sensing range of the sensing system at the start of the mission. In such situation, the path planner is continuously exploring the environment which is assumed fixed. Besides, some sensing suites like camera-based systems, limit the Field of View (FOV) to less than 360^o in all 3 dimensions, reducing further the situational awareness of the path planning algorithm.

A path planner is considered to be real-time if the time required to generate a path is smaller than the time required to traverse the path [40–42]. This time depends upon UAV speed and onboard computational resources. Speed is governed by both mission requirements and UAV speed limitations while onboard computational resources only depend on the UAV specifications. The bottleneck for real-time path planning is mainly onboard computational resources, as small UAVs can typically achieve a speed higher than required in indoor environments with typical associated payloads.

Real-time path planning in unknown indoor environments requires accurate sensing technologies to provide the path planning algorithm with adequate real-time situation awareness. In conjunction, the computational resources need to be used to control the UAV actuator systems so that the UAV can reach the final or intermediate goal point. Since computational resources need to be shared between the sensing, actuator and path planning systems in real-time, this emphasises the need for increasing onboard computational capacity and the use of computationally efficient systems.

1.2.2. CHALLENGE 2

Challenge 2

Construct in *real-time*, non-colliding paths from the current UAV position to a target position using only onboard UAV resources in the presence of both *static* and *dynamic* moving obstacles.

Challenge 1 only considered obstacles within the environment as static for the duration of the flight. In indoor environments, the UAVs must share space with other machines and people, besides fixed stationary obstacles. Usually the trajectories of moving obstacles are unknown and difficult to predict especially when humans are involved.

Furthermore, stationary obstacles like furniture may be moved from one place to another during the flight.

The ability of the path planning algorithm to operate in time-variant environments requires the path planning algorithm to continuously update the generated path based on new information, to evade collisions with obstacles. This can be achieved when real-time path planning is possible and the UAV can move with the same or larger speed with respect to the moving obstacles. Without this ability, UAVs will be restricted to unoccupied indoor environments and/or dedicated spaces with a separate space for each moving obstacle to operate, limiting the potential amount of traffic in operation at the same time.

1.2.3. CHALLENGE 3

Challenge 3

Construct in *real-time*, non-colliding paths from the current UAV position to a goal position using only onboard UAV resources in the presence of both *static* and *dynamic* obstacles in the presence of *uncertainties*.

No sensing technology can provide state information accurately without uncertainty. Over the decades uncertainty estimation and bounds have been improved, but uncertainties can never be completely eliminated. Especially for small UAVs, their onboard sensing systems are such that the level of uncertainty cannot be neglected. If not considered, this may lead to collisions, jeopardising the robustness and consequently their use in indoor environments. Uncertainty is present within the sensing of the environment and within UAVs' internal capabilities such as the algorithms involved in the position, orientation and manoeuvring systems.

This challenge requires the path planning system to model uncertainties in both the UAV model and the environment when constructing paths to goal position. Again, this process must be performed in real-time, using onboard computational resources since the environment may change every time the sensing and UAV state estimation systems generate new readings.

The explained challenges portray the requirements of a path planning algorithm guiding a UAV in indoor environments. Although different researchers have considered this niche, a gap for further improvement exists between the current state-of-the-art and the robustness and safety requirements necessary to operate small UAVs in indoor environments in close proximity to people.

Different researchers have developed and implemented a wide range of path planning algorithms originating from either graph-based or sampling-based rationales without directly comparing both rationales in view of real-time 3D UAV path planning in the presence of both static and dynamic obstacles. Furthermore, the uncertainty aspect was neglected in the majority of the studies. When uncertainty was considered, the research focused on uncertainty identification and modelling and not on the different effects of individual uncertainty sources and their combination on path planning performance of graph-based and sampling-based methods. These research gaps will be investigated in this thesis with the ultimate aim of assisting UAV designers in using the ideal path plan-

ning configuration for indoor applications.

1.3. RESEARCH GOAL AND RESEARCH QUESTIONS

1.3.1. RESEARCH GOAL

The challenges put forward in Section 1.2 motivate the research goal of this thesis.

Research Goal

Assess the performance of state-of-the-art path planning algorithms in the context of UAVs operating in *3D real-time*, *dynamic* indoor environments in the presence of *uncertainty* and identify a customised UAV path planning configuration based on the application.

The research goal can be divided into two main tasks:

1. Performance assessment of state-of-the-art path planning algorithms in dynamic and uncertain environments; and

2. Use the assessment outcomes to guide future UAV designers to select the best path planning configuration available for their specific application.

To address the research goal, the following research questions were formulated.

1.3.2. RESEARCH QUESTION 1

Research Question 1

What is the state-of-the-art in the field of path planning for UAVs in 3D, and how do these algorithms compare?

To investigate the potential of different path planning algorithms, the current state-of-the-art in all fields of robotics needs to be investigated. This will highlight the potential path planning performance, robustness and applicability to the field of 3D UAV path planning. The majority of path planning algorithms were developed for 2D applications even when applied in 3D environments by assuming one dimension to be constant. Although some algorithms were adapted to handle 3D environments, others require further work to handle the additional dimension. This can result in a higher deterioration in performance with respect to other algorithms.

Furthermore, to fairly assess the prospective algorithms, a 3D environment with obstacles simulating typical indoor environments needs to be constructed. In typical indoor environments various objects with different sizes exist that can be estimated using polyhedrons. The most promising algorithms must be tested on the same platform with different shapes and positions, in the most obstacle-dense environment which the prospective indoor UAV is expected to handle.

1

1.3.3. RESEARCH QUESTION 2

Research Question 2

Can the selected path planning algorithms be applied in real-time environments using the computational capabilities onboard small UAVs?

To fully mitigate Challenge 1 described in Section 1.2.1, the most prominent path planning algorithms of Research Question 1 need to be tested in real-time scenarios. Challenge 1 is mitigated if both Research Questions 1 and 2 are fully tackled.

To answer Research Question 1, the environment can be static and fully known prior the initiation of the flight, implying that path planning can be done offline without any time restrictions. Through Research Question 2, all path planning computation, sensing and environmental modelling and actuator controls must be computed onboard and in real-time.

The same scenarios developed to answer Research Question 1 can be used to tackle Research Question 2. But in Research Question 2 only part of the environment within the sensing distance, determined by the sensing systems, is available, in 3D, to the path planning algorithm. In such a situation, the path planning algorithm can only, if possible, construct a non-colliding path to an intermediate goal point within the sensing range of the sensing system, approaching the final goal position as the UAV moves through the constructed path segments in the direction of this position.

1.3.4. RESEARCH QUESTION 3

Research Question 3

What is the effect on path planning performance if static obstacles are replaced with dynamic obstacles?

This research question directly links with Challenge 2. To tackle Research Questions 1 and 2, obstacles within the environment were assumed to remain invariant in position, and orientation throughout the total flight time. As discussed in Section 1.1 in real indoor environments this is rarely the case and therefore the selected real-time path planning algorithms shall be tested in dynamic environments in which obstacles change in position and orientation.

The inclusion of dynamic environments is external to the path planning algorithm but it can affect the path that the UAV will traverse. While in static environments the generated path from the current UAV position to intermediate or final goal points will remain constant from one iteration to another, with the inclusion of dynamic obstacles the path planning algorithm needs to continuously update the previously generated path so as to avoid collisions.

1.3.5. RESEARCH QUESTION 4

Research Question 4

What uncertainties affect 3D path planning of UAVs? How can these uncertainties be modelled?

This research question addresses Challenge 3 described in Section 1.2.3. Answering this research question requires investigating whether different uncertainty sources influence 3D path planning performance of UAVs operating in indoor environments. These uncertainty sources, in both the UAV model and environment, must be identified, estimated and modelled such that their affects can be incorporated into path planning constraints. The fidelity with which these uncertainties can be predicted will then ultimately determine the usefulness of each potential path planning algorithm.

1.3.6. RESEARCH QUESTION 5

Research Question 5

How can uncertainties be mitigated to ensure collision-free 3D path planning of UAVs in real-time in the presence of dynamic obstacles?

This research question is a continuation of Research Question 4 and in conjunction with it addresses Challenge 3 described in Section 1.2.3. Once uncertainty sources are identified, estimated and modelled for small UAVs in typical indoor environments, the developed 3D real-time path planning algorithms can be assessed in the presence of dynamic obstacles and uncertainties. This research question must answer whether the real-time path planning algorithms considered allow the UAV to safely operate within an indoor environment. By answering this research question the proposed overarching thesis research goal will be achieved.

Each research question proposed builds on the previous one to ultimately reach the final research goal. In the process of assessing the path planning performance (first part of research goal) of each algorithm under review, the response of each method with respect to each additional complicating feature (dynamic obstacles, uncertainty) can be independently analysed, highlighting the characteristics of each method. This information can be used to help future UAV designers in selecting the best configuration based on their application, satisfying the second part of the research goal.

1.4. RESEARCH SCOPE

The research scope of this dissertation is to achieve the research goal set in Section 1.3.1 by answering Research Questions 1 to 5 explained in Section 1.3.2 - Section 1.3.6. To fairly test these algorithms, a common test platform is required. All modelling and simulations are performed on the same MATLAB version (R2014b) using the same personal computer (PC) an Intel Xeon ES-1650, 3.2GHz. In this section some of the main assumptions are discussed.

1.4.1. RESEARCH QUESTION 1: 3D UAV PATH PLANNING

In relation to Research Question 1, a literature review of the different path planning algorithms that can potentially be applied to 3D UAV path planning will yield a list of potential algorithms and their variants. The two algorithms, and the most promising variants of each that best fit the application at hand will be assessed.

The focus of this work is to assist UAV designers by outlining guidelines in configuring the ideal path planning algorithm for a UAV operating in an indoor environment. The modelling and analysis of the sensing and control capabilities and constraints of a specific UAV that will be used in the specific indoor environment is application-specific. For this reason and the fact that only small UAVs will be considered, the UAV is approximated by a point-mass model with generic sensing and control capabilities and constraints will be considered. In this regard, for the scope of all research questions unless otherwise specified, it will be assumed that:

1. The UAV can turn in all 3 dimensions while remaining at the same position;
2. The UAV can move at the same constant speed defined prior mission initiation;
3. The UAV occupies just one point in space with infinitesimally small volume;
4. The UAV movement does not affect the position of obstacles in its vicinity;
5. The UAV is not affected by external factors (example: obstacles and crosswind);
6. The UAV can stop in mid-air and wait;
7. The UAV sensing systems can provide a spherical 360° sensing in all directions;
8. The UAV can accurately follow straight line segments;
9. The UAV has no dominant/preferred orientation; and
10. The sensing and control algorithms' computational time is dependent upon the type of UAV framework required for the specific application. This framework differs from one UAV to another and therefore the sensing and control algorithms' computational time is not considered for the scope of this work and only the path planning time is considered.

The environment is designed beforehand, for all tests considered and the scenario loading computational time is neglected in the path planning computational time. Also it is assumed that the movement time of the UAV from one path node to the next is independent of the path planning time, in line with assumption Item 10.

Ideally, the path planning performance of the algorithms shall be tested in a large number of different scenarios so as to fully assess their performance. Owing to space and time limitations, a representative set of worst case scenarios was designed and modelled. These consisted of both simple cubes and vertical and horizontal planes with small openings through which the UAV needs to pass to reach the goal position. A dimensionless environment is defined as a 1x1x1 box with obstacles occupying a predefined volume or area. This dimensionless approach is considered for other parameters such as path length, making tests easily transferable to real test scenarios by scaling.

1.4.2. RESEARCH QUESTION 2: REAL-TIME UAV PATH PLANNING

The shift from Research Question 1 to 2 mainly affects the UAV environmental visibility and introduces computational constraints. For the scope of this research question a sphere with radius equal to the sensing range of the UAV is considered. Obstacles within this sphere are known with certainty while other obstacles are unknown and can become visible only when the UAV moves in their direction. All other UAV assumptions explained in Section 1.4.1 remain relevant.

For the scope of Research Question 1, UAV movement time was not considered as path planning was generated offline. But, to simulate real-time path planning, it is required that the path planning computational time is less than or equal to the time needed by the UAV to move from the current position to a new position. As explained in Section 1.4.1 the environment creation and loading for each iteration is neglected in computational time calculation. In real situations, environmental sensing, path planning and UAV control all use the limited computational power of the UAV.

Since the environment remains static, the same environment as in Section 1.4.1 is considered. Since for the scope of Research Question 1, paths are constructed offline, the UAV speed is irrelevant for the scope of path planning. However, in real-time situations, this is not the case as speed will determine the maximum computational time for the UAV to traverse from the current UAV position to new UAV position. Modular speed shall be defined in view of state-of-the-art small UAVs proposed for indoor environments.

1.4.3. RESEARCH QUESTION 3: REAL-TIME PATH PLANNING OF UAVS IN DYNAMIC ENVIRONMENTS

Research Question 3 introduces dynamic obstacles to the real-time 3D UAV path planning in indoor environments problem. Dynamic obstacles within an indoor environment can be estimated by symmetrical shapes. For the scope of this work, a set of scenarios consisting of cubes and V-shaped obstacle shapes together with static planes with windows discussed in Section 1.4.1 is considered. The obstacle density is selected also in view of real indoor environments within which the UAV is expected to operate, although some extreme scenarios are also considered to assess the robustness of the algorithms.

Both obstacle movement and orientation are considered in the dynamic environment modelling. The random obstacle movement speed is smaller than or equal to the speed of the UAV. This assumption is required as otherwise situations can exist in which the UAV can never evade the obstacle. Apart from the magnitude restriction, obstacle movement direction and orientation random changes are not restricted.

The dynamic environment is created offline so as not to affect the path planning computational time. During the path construction from the current UAV position to the intermediate goal point, the environment is assumed to be static with obstacles remaining stationary. In reality, sensing systems will provide discontinuous time-stamped screenshots of the environment and in-between these screenshots the planner has to rely on the last shot. This assumption is only valid if the time of construction of path segments is small enough, such that obstacles do not move into the constructed path segment. This is difficult to estimate since the time-varying speed and orientation of obstacles are random and unknown to the planner.

1.4.4. RESEARCH QUESTION 4: UNCERTAINTY IDENTIFICATION AND MODELLING

Research Question 4 deals with the effect of uncertainty present within UAVs operating in indoor environments in real-time. This requires a second literature analysis to identify and model uncertainty sources that might also effect path planning performance. Modelling of uncertainty sources must be integrated within the development of the test platform up to Research Question 3. For the scope of this work, uncertainties are only considered within the UAV model and the environment (perceived through UAV onboard sensing systems only) within which it is expected to operate. Other uncertainties such as communication with user(s) and/or other interface technologies will not be considered.

1.4.5. RESEARCH QUESTION 5: UNCERTAINTY INTEGRATION

Research Question 5 builds on the previous question as it involves integrating uncertainty sources within the real-time 3D path planning algorithms framework. A number of uncertainties in UAV and obstacle environment will be analysed for a set of uncertainty values. Uncertainty is included by adding a bounded, randomised offset to the actual respective parameter. The effect of each uncertainty source will be analysed independently and collectively with dynamic obstacles for both path planning rationales so as to identify how real-time path planning algorithms can safely operate.

In the next section, the main contributions of this dissertation will be defined, explained and analysed.

1.5. MAIN CONTRIBUTIONS

The main contributions of this dissertation can be categorised into: path planning algorithms, real-time path planning, dynamic obstacle modelling and uncertainty identification and modelling.

1.5.1. CONTRIBUTIONS TO PATH PLANNING ALGORITHMS

- Literature review of different graph-based and sampling-based path planning methods that can be applied for 3D UAV path planning in indoor environments;
- Direct comparison and analysis of the A* and RRT algorithms, and their variants. The A* and RRT algorithms represent the state-of-the-art for the graph-based and sampling-based rationales;
- Novel adaptions and improvements to the standard A* and RRT algorithms, to offer a fairer comparison and to mitigate shortcomings in the A* and RRT path planning algorithms;
- Analysis of the effects of path smoothing, using the same smoothing algorithm, on the path planning performance for both the A* and RRT algorithms and their variants; and
- Direct comparison between the A* and RRT algorithms on the same real-time platform in different scenarios with static and dynamic obstacles with and without uncertainties.

1.5.2. CONTRIBUTIONS TO REAL-TIME PATH PLANNING

- A real-time path planning literature review confirmed the need for real-time path planning for UAVs operating in indoor environments;
- Modelling of environment sensing with limited range and UAV movement emulating real-time behaviour;
- Effects on UAV parameter definition between offline and real-time 3D UAV path planning; and
- Relational guidelines between each UAV parameter and path planning performance for both A* and RRT algorithms.

1.5.3. CONTRIBUTIONS TO DYNAMIC OBSTACLE MODELLING

- Analysis of the different obstacle shapes and sizes present within typical indoor environments;
- Modelling of different 2D and 3D dynamic obstacles randomly in view of computational limitations; and
- Identification of common challenges that UAV path planning algorithms can encounter and how these can be mitigated in real-time.

1.5.4. CONTRIBUTIONS TO UNCERTAINTY IDENTIFICATION AND MODELLING

- Understanding of the need of incorporating uncertainty in path planning of a UAV operating in indoor environments. Uncertainty is sometimes neglected although literature and this work show that uncertainty effects are not negligible;
- Identification of the main uncertainty sources and their respective ranges present in a UAV and its sensing and communication systems operating in an indoor environment;
- Modelling of uncertainties and their incorporation into a modular real-time path planning algorithm; and
- Identification of the deterioration level, if any, of individual uncertainty sources both independently and holistically on 3D path planning of UAVs in indoor environments.

1.6. THESIS OUTLINE

The main body of this dissertation is based on a collection of peer-reviewed journal and conference papers. Because of this, some overlap is present in the respective introductions relating to the general motivation of UAV path planning and the definition of the A* and RRT algorithms. Each chapter builds upon the previous one, but was written independently and therefore can also be read individually.

1

Each chapter starts with an abstract, followed by an introduction to the goal targeting the particular research question. Figure 1.4 illustrates the dissertation outline, with each chapter depending and evolving from the knowledge gained in the previous chapters with the aim of reaching the ultimate goal of this dissertation.

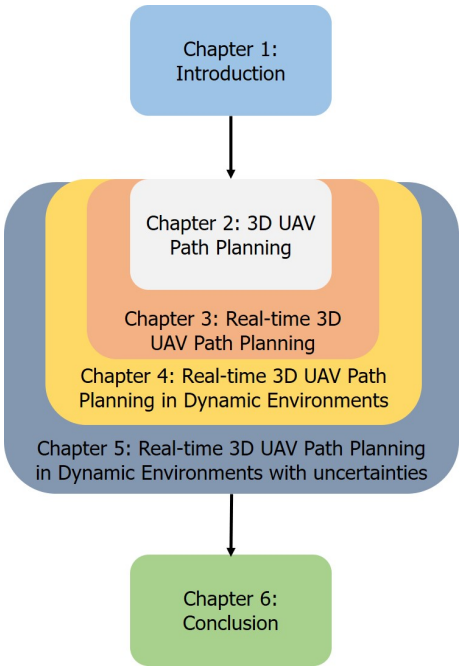


Figure 1.4: Dissertation Outline

Chapter 2 will present a thorough literature review of the different path planning rationales and their variants that can potentially be applied in the field of 3D UAV path planning in indoor environments. The two state-of-the-art rationales and their most promising variants will be implemented, tested and their path planning performance analysed. In this process, inherent shortcomings of both rationales will be mitigated to enhance path planning performance. This chapter is intended to answer Research Question 1 derived from Challenge 1.

Chapter 3 builds upon the knowledge gained in Chapter 2 and develops the two most promising path planning rationales in view of real-time path planning. This chapter presents a detailed literature review of the approaches considered in real-time path planning whilst highlighting the need of real-time consideration. It proposes, develops, tests and implements real-time path planning with static obstacles analysing the effects of different parameters on path planning performance. This chapter aims to answer Research Question 2 also derived from Challenge 1.

In Chapter 4, obstacles are changed from static to dynamic. As in the previous chapters, this chapter initiates with a literature review highlighting the need of dynamic obstacle consideration in indoor environments and presents the state-of-the-art in dy-

namic obstacle modelling. Typical dynamic obstacles are modelled and their effects on real-time path planning performance are assessed. This chapter aims to answer Research Question 3 derived from Challenge 2.

Chapter 5 introduces uncertainty in the 3D real-time path planning problem. This chapter identifies and explains the need for uncertainty considerations, the sources of uncertainty and their quantification in view of UAVs operating in indoor environments. The relevant uncertainties within the research scope are identified, modelled and integrated with the 3D real-time environment developed in Chapter 4. The chapter will conclude with an analysis of the effect of path planning performance of each uncertainty source independently and concurrently that is expected to help future indoor UAV manufacturers in the design of more robust indoor UAVs. Through the work of this chapter, Research Questions 4 and 5 derived from Challenge 3 are expected to be answered. Consequently, the research goal of this dissertation will be reached.

Chapter 6 presents the final conclusions, discussions and future recommendations of this dissertation. Figure 1.5 graphically summarises the links between the challenges, research questions and chapters using the same colours for clarity.

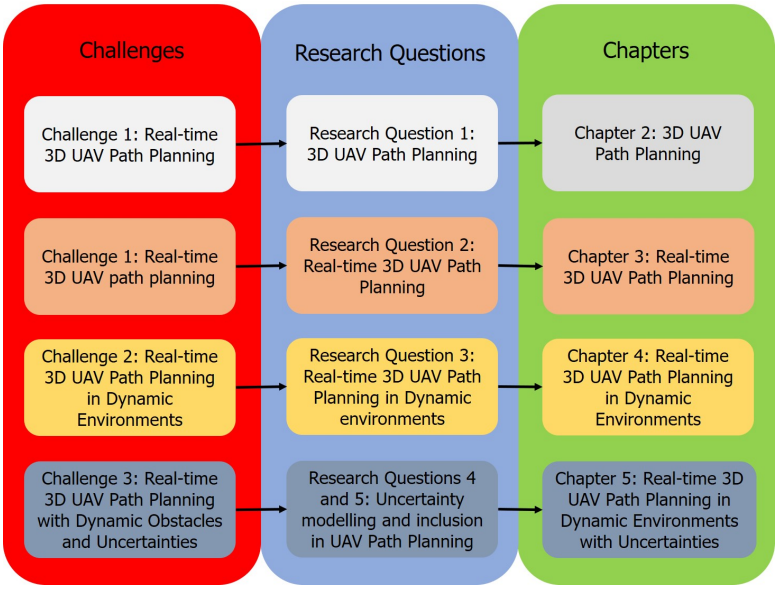


Figure 1.5: An illustration showing the mapping of Challenges to Research Questions and from Research Questions to Chapters

REFERENCES

[1] International Civil Aviation Organisation (ICAO), "Global Air Traffic Management Operational Concept", *Doc. 9854*, AN/458, Ed. 1, pp. 1-82, 2005.

[2] International Civil Aviation Organisation (ICAO), "Unmanned Aircraft Systems (UAS)", *Cir. 328*, AN/190, pp. 1-54, 2011.

[3] Sharp, T., 'The First Hot-Air Balloon Flight', Available: https://www.space.com/16595-montgolfiers-first-balloon-flight.html.

[4] Dave, O., 'UAVs: Info, History, Timeline', Available: https://www.theflightbay.com/uav/.

[5] O' Donnell, S., 'A Short History of Unmanned Aerial Vehicles', Available: https://consortiq.com/en-gb/media-centre/blog/short-history-unmanned-aerial-vehicles-uavs.

[6] Federal Aviation Administration (FAA), 'UAS by the Numbers', Available: https://www.faa.gov/uas/resources/ by_the_numbers/.

[7] National Aeronautics and Space Administration (NASA), 'What is Unmanned Aircraft Systems Traffic Management?', Available: https://www.nasa.gov/ames/utm/.

[8] National Aeronautics and Space Administration (NASA), 'FAQs: NASA's Drone Traffic Management Research in Reno and Corpus Christi', Available: https://www.nasa.gov/feature/ames/faqs-nasa-s-drone-traffic-management-research-in-reno-and-corpus-christi/.

[9] Single European Sky ATM Research Joint Undertaking (SESAR), 'Concept of Operations for European UTM Systems (CORUS)', Available: https://www.sesarju.eu/projects/corus.

[10] Civil UAS Operation Management System (UOMS), 'UOMS in China', Available: https://rpas-regulations.com/wp-content/uploads/2018/06/1.2-Day1_0910-1010_CAAC-SRI_Zhang-Jianping_UOMS-_EN.pdf.

[11] Korea Institute of Aviation Safety Technology (KIAST), 'Korea Institute of Aviation Safety Technology', Available: https://www.kiast.or.kr/en/index.do.

[12] Sigala, A. and Langhals, B., "Applications of Unmanned Aerial Systems (UAS): A Delphi Study Projecting Future UAS Missions and Relevant Challenges", *Drones*, Vol. 4, No. 8, pp. 1-15, 2020.

[13] Maghazei, O. and Netland, T., "Drones in manufacturing: exploring opportunities for research and practice", *Journal of Manufacturing Technology Management*, (fourthcoming), 2019.

[14] Ducard, G. and D'Andrea, R., "Autonomous quadrotor flight using a vision system and accommodating frames misalignment", *International Symposium on Industrial embedded systems*, Lausanne, Switzerland, 8-10 Jul., 2009, pp. 261-2646.

[15] Khosiawan, Y. and Nielsen, I., "A system of UAV application in indoor environment", *Production & Manufacturing Research*, Vol. 4, No. 1, pp. 2-22, 2016.

[16] Ledergerber, A., Hamer, M. and D'Andrea, R. "A robot self-localization system using one-way ultra-wide band communication", *IEEE/RSJ International Conference in Intelligent Robots and Systems (IROS)*, Hamburg, Germany, 28 Sep.-2 Oct., 2015, pp. 3131-3137.

[17] Floreano, D. and Wood, R.J., "Science, technology and the future of small autonomous drones", *Nature*, Vol. 521, No. 7553, pp. 460-466, 2015.

[18] Czentye, J. Dóka, J., Nagy, A., Toka, L., Sonkoly, B. and Szabó, R. "Controlling Drones from 5G Networks", *Proceedings of the ACM SIGCOMM conference on Posters and Demos*, Budapest Hungary, Aug., 2018, pp. 120-122.

[19] ZF Friedrichshafen AG, 'ZF is first in Germany to fly drones over plant premises', Available: https://press.zf.com/press/en/releases/release_3006.html.

[20] Pardell, R., 'AGCFDS: Automated Glass Cleaning Flying Drone System', US 2017/0210470 A1, 2017.

[21] Zammit, C. and van Kampen, E. J., "Comparison between A* and RRT Algorithms for UAV Path Planning", *AIAA Guidance, Navigation and Control Conference*, AIAA SciTech Forum, Kissimmee, FL, 8-12 Jan., 2018.

[22] Sujit, P. B., and Ghose, D., "Search by UAVs with Flight Time Constraints using Game Theoretical Models," *AIAA Guidance, Navigation and Control Conference*, San Francisco, CA, 15-19 Aug. 2005, pp. 1-11.

[23] Bethke, B., Bertuccelli, L. How, J. P., "Experimental Demonstration of MDP-Based Planning with Model Uncertainty," *AIAA Guidance, Navigation and Control Conference*, Honolulu, HI, 18-21 Aug. 2008, pp. 1-22.

[24] Bollino, K., Lewis, L. R., Sekhavat, P. and Ross, I. M., "Pseudospectral Optimal Control: A Clear Road for Autonomous Intelligent Path Planning," *AIAA Infotech Aerospace Conference and Exhibit*, Rohnert Park, CA, 7-10 May 2007, pp. 1-14.

[25] Zammit, C. and van Kampen, E. J., "Comparison of A* and RRT in real-time 3D path planning of UAVs", *AIAA Guidance, Navigation and Control Conference*, AIAA SciTech Forum, Kissimmee, FL, 6-10 Jan., 2020.

[26] Tisdale, J., Kim, Zuwhan Kim Zuwhan and Hedrick, J., "Autonomous UAV path planning and estimation," *IEEE Robotics & Automation Magazine*, Vol. 16, No. 2, 2009, pp. 35-42.

[27] Amin, J. N., Boskovic, J. D. and Mehra, R. K, "A Fast and Efficient Approach to Path Planning for Unmanned Vehicles", *AIAA Guidance, Navigation and Control Conference*, Keystone, CO, 21-24 Aug., 2006, pp. 1-9.

[28] Electrolux ,'The Trilobite 2.0', Available: http://trilobite.electrolux.com/node217.asp.

[29] Dai, R. and Cochran, J., "Path Planning and State Estimation for Unmanned Aerial Vehicles in Hostile Environments", *Journal of Guidance, Control, and Dynamics*, Vol. 33, No. 2, 2010, pp. 595-601.

[30] Foo, J. L., Knutzon, J., Kalivarapu, V., Oliver, J. and Winer, E., "Path Planning of Unmanned Aerial Vehicles using B-Splines and Particle Swarm Optimization", *Journal of Aerospace Computing, Information, and Communication*, Vol. 6, No. 4, 2009, pp. 271-290.

[31] Bollino, K. P. and Lewis, L. R., "Collision-Free Multi-UAV Optimal Path Planning and Cooperative Control for Tactical Applications", *AIAA Guidance, Navigation and Control Conference*, Honolulu, HI, 18-21 Aug., 2008, pp. 1-18.

[32] Sun, X., Cai, C. and Shen, X., "A New Cloud Model Based Human-Machine Cooperative Path Planning Method", *Journal of Intelligent & Robotic Systems*, Vol. 79, 2014, pp. 3-19.

[33] Dijkstra, E. W., "A Note on Two Problems in Connexion with Graphs", *Math. 1*, Pp. 269-271, 1959.

[34] Gonzàlez, D., P*èrez*, J., Milanès, V., and Nashashibi, F., "A Review of Motion Planning Techniques for Automated Vehicles", *IEEE Transactions on Intelligent Transportation Systems*, Vol. 17, No. 4, pp. 1135-1145, 2016.

[35] Kavraki, L. E., Švestka, P., Latombe, J. C. and Overmars, M. H. "Probabilistic roadmaps for path planning in high-dimensional configuration spaces", *IEEE Transactions on Robotics and Automation*, Vol. 12, No. 14, pp. 566-580, 1996.

[36] LaValle S. M. "Probabilistic roadmaps for path planning in high-dimensional configuration spaces", *IEEE Transactions on Robotics and Automation*, Vol. 12, No. 14, pp. 566-580, 1996.

[37] Ghandi, S. and Masehian, E., "Review and taxonomies of assembly and disassembly path planning problems and approaches", *CAD Computer Aided Design*, Vol. 67-68, No. October, pp. 58-86, 2015.

[38] Short, A., Pan, Z., Larkin, Z. and van Duin, S. "Recent Progress on Sampling Based Dynamic Motion Planning Algorithms", *IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, Alberta, Canada, Jul. 2016, pp. 1305-1311.

[39] Labakhua, L., Nunes, U., Rodrigues, R. and Leite, F. S. "Smooth trajectory planning for fully automated passengers vehicles: spline and clothoid based methods and its simulation", *Informatics in Control Automation and Robotics*, Springer 2008, pp. 169-182.

[40] Chakrabarty, A. and Langelaan, J. W., "Energy maps for long-range path planning for small-and micro-uavs", *AIAA Guidance, Navigation and Control Conference*, Chicago, IL, 10-13 Aug., 2009, pp. 1-13.

[41] Benenson, R. Petti, S., Fraichard, T. and Parent, M., "Integrating Perception and Planning for Autonomous Navigation of Urban Vehicles," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China, 9-15 Oct. 2006, pp. 98-104.

[42] Yao, P., Wang, H. and Su, Z., "Real-time path planning of unmanned aerial vehicle for target tracking and obstacle avoidance in complex dynamic environment," *Aerospace Science and Technology*, Vol. 47, pp. 269-279, 2015.

2

COMPARISON BETWEEN A* AND RRT ALGORITHMS FOR 3D PATH PLANNING

The first research question addresses the current state-of-the-art path planning algorithms and how these algorithms compare will be addressed in this chapter. A literature review of the different path planning algorithms that can be potentially applied for 3D UAV path planning in indoor environment will be presented in this chapter. For this review, two state-of-the-art rationales and their most promising variants will be implemented, tested and their path planning performance analysed. In this process, inherent shortcomings of both rationales will be mitigated to enhance path planning performance.

This chapter is based on two conference papers published as:
Zammit, C. and van Kampen, E., "Comparison between A* and RRT Algorithms forUAV Path Planning", *Proceedings of AIAA Guidance, Navigation and Control*, Kissimmee, FL, 8-12 Jan., 2018, AIAA 2018-1846.

Zammit, C. and van Kampen, E., "Advancements for A* and RRT in 3D path planning of UAVs", *Proceedings of AIAA Guidance, Navigation and Control*, San Diego, CA, 7-11 Jan., 2019, AIAA 2019-0920.

The contents of this chapter are accepted for publication:
Zammit, C. and van Kampen, E., "Comparison between A* and RRT Algorithms for 3D UAV Path Planning", Journal of Unmanned Systems.

This chapter aims to present a comparative analysis of the most utilised graph-based and sampling-based algorithm and their variants, in view of 3D UAV path planning in complex indoor environment. The findings of this analysis outline the usability of the methods and can assist future UAV path planning designers to select the best algorithm with the best parameter configuration in relation to the specific application. An extensive literature review of graph-based and sampling-based methods and their variants is first presented. The most utilised algorithms which are the A* for graph-based methods and Rapidly-Exploring Random Tree (RRT) for the sampling-based methods, are defined. A set of variants are also developed to mitigate inherent shortcomings in the standard algorithms. All algorithms are then tested in the same scenarios and analysed using the same performance measures. The A* algorithm generates shorter paths with respect to the RRT algorithm. The A* algorithm only explores volumes required for path generation while the RRT algorithms explore the space evenly. The A* algorithm exhibits an oscillatory behaviour at different resolutions for the same scenario that is attenuated with the novel A* ripple reduction algorithm. The Multiple RRT generated longer unsmoothed paths in shorter planning times but required more smoothing over RRT. This work presents a novel comparison between graph-based and sampling-based algorithms in 3D path planning of UAVs. Furthermore, this work addresses shortcomings in both A* and RRT standard algorithms by developing a novel A* ripple reduction algorithm, a novel RRT variant and a specifically designed smoothing algorithm.

2.1. INTRODUCTION

Since the first industrial revolution, automation has been integrated into everyday life ranging from basic traffic lights to autonomous navigation of submarines. The introduction of these systems into military, industrial, commercial and private use requires such systems to be robust and reliable to ensure a safe operation. These autonomous systems which can vary in scope, size and shape, are equipped with different sensory systems requiring various levels of intelligence depending upon their application. Advancements in control and navigation technologies and cost-reduction in hardware have made it possible for autonomous systems to operate safely and efficiently in a range of applications. The ever-increasing availability of autonomous systems will require more complex path planning systems so that all these systems can work efficiently without hindrance and possibly enhance the operation of one another.

One aspect of autonomous systems is path planning. Path planning algorithms construct optimal paths to a preset goal point in the presence of time invariant and time variant environmental and vehicular constraints and uncertainties [1]. Path planning initiated as a 2D problem to generate paths for ground vehicles [2–4] or to approximate 3D path planning problems with 2D by assuming that 1 dimension is constant [5–9]. Eventually, with the introduction of complex 3D environmental and model constraints, research shifted its focus to 3D path planning methods [9–13]. The path planning system in 2D and 3D environments must deal with a set of constraints and requirements including static [14–16] and moving obstacles [17–19], moving targets[20, 21], weather [22, 23], no go zones [24, 25], communication [27–29], fuel [5, 25] and other model specific constraints [26, 30]. Some of these constraints, such as static obstacles and no go zones,

can be accurately defined prior mission initiation. But in real situations the majority of constraints can only be modelled with uncertainty, primarily weather and dynamic obstacles, although fuel consumption rate and vehicular state also incorporate an element of uncertainty. Besides, in real situations, constraints can change or even new ones pop-up and consequently the path planning system needs to adapt in real-time to ensure safe guidance to the final goal position [24].

To ensure safe and efficient path planning, numerous algorithms were proposed which can be segmented into two main approaches: ***Graph-based and Sampling-based algorithms***. Graph-based methods define the environmental space by a set of grid points. Points residing on obstacles are defined as inaccessible grid points. The algorithm searches through the accessible grid points to construct a path from start to goal position if such a path is possible[31]. Graph-based methods guarantee a path from start to goal only for an adequate grid resolution [32]. Sampling-based methods select a random number of points in the environmental space and creates connections between them [32, 33]. Sampling-based algorithms are simple, efficient and probabilistically complete, i.e. they guarantee a solution at infinite time [32].

This paper provides a literature review of the current state-of-the-art path planning algorithms in the graph-based and sampling-based categories. The two most researched path planning methods, namely the A* (A star) and the Rapidly-Exploring Random Tree (RRT) from the graph-based and sampling-based categories respectively, will be described, tested and analysed in view of 3D UAV path planning. Furthermore, this paper aims to reduce the resultant path length ripple effects at different resolutions for the A* algorithm and the efficiency of the smoothing algorithm since in particular runs the reduction in path length is less than 0.1% even after numerous smoothing iteration attempts.

In this study, the performance parameters are the path length and computational time. Path length determines the time of traversal of a path and consequently the time the vehicle takes to arrive at the goal point. Fuel capacity and fuel efficiency determine the flight range. This is a primary constraint to full autonomy. This implies that path length will determine the effective range of the UAV. Ideally, the path planning time is as low as possible especially in dynamic environments with moving obstacles, since during path planning the environmental situation may significantly vary. Path planning time will impinge on the responsiveness of the path following algorithm to mitigate with both agent and environmental changes. Computational power onboard UAVs may also be very limited and it must be shared with other computationally expensive algorithms and sensing technologies. In this regard, the computational burden of path planning algorithms shall be as low as possible. The main contribution of this paper is an analysis of the A* and RRT algorithms in view of their suitability for 3D UAV path planning.

This chapter will be organised as follows. Section 2.2 defines the standard A* and RRT algorithms, followed by the state-of-the-art graph-based and sampling-based methods in Section 2.3. Section 2.4 defines the A* ripple reduction (A^*_R) algorithm, RRT variants, the improved smoothing algorithm, vehicle model definition and the testing scenarios with their associated obstacle avoidance constraints. In Section 2.5, the results are analysed followed by Section 2.6, which summarises this paper by outlining the main strengths and weaknesses of the A* and RRT algorithms in view of real-time 3D UAV path planning.

2.2. THE A* AND RRT ALGORITHMS

2.2.1. THE A* ALGORITHM

The development of the A* algorithm started in 1964 with the invention of the A1 algorithm by Nils Nilsson that aimed to increase the speed of the Dijkstra's algorithm. Then in 1967, Bertram Raphael enhanced performance by developing the A2 algorithm but failed to show optimality. In 1968, Peter E.Hart added some minor amendments to A2 to proof its optimality, naming the new algorithm A* [34].

The A* algorithm constructs an optimal path based on an evaluation function $f(n)$ which calculates the actual cost of the path passing through generic node n, initiating from the starting point x_{init} to the goal node of n in M-Dimensional space such that $x_{init} \in \mathbb{R}^M$ and $x_{goal} \in \mathbb{R}^M$ [35, 36]. As described by the below formula, $f(n)$ is the sum of the actual costs from x_{init} to a node n ($g(n)$) and from n to the goal point of n ($h(n)$), where $f, g, h: \in \mathbb{R}^M \rightarrow \mathbb{R}$:

$$f(n) = g(n) + h(n) \tag{2.1}$$

As the cost of the evaluation function $f(n)$ is a function of resolution, $g(n)$ and $h(n)$ can only be estimated. The estimated costs are denoted by $\hat{g}(n)$ and $\hat{h}(n)$, respectively, where the estimated evaluation function denoted by $\hat{f}(n)$ equates to:

$$\hat{f}(n) = \hat{g}(n) + \hat{h}(n) \tag{2.2}$$

As the resolution $\rightarrow \infty$, the evaluation function $\hat{f}(n) \rightarrow f(n)$.

Algorithm 1 defines and Figure 2.1 shows the rationale behind the A* Algorithm [35]. The A* algorithm starts by defining the start and goal nodes, x_{init} and x_{goal}, respectively. Then, $\hat{f}(x_{init})$ is evaluated for all possible *"Open"* nodes $n_i \in \mathbb{R}^M$ and the node with the smallest cost $c_i \in \mathbb{R}$ is assigned to n. *"Open"* nodes include all nodes n_i except for nodes that reside on an obstacle and nodes already forming part of the optimal path. These are referred to as *"Closed"* nodes. No path exists if $\hat{f}(x_{init})$ is $empty$ implying that the cost to all *"Open"* nodes is $null$. The A* algorithm continues by generating successive nodes using the successor operator (Γ). Through this operator, a set of successor nodes $n_i \in \mathbb{R}^M$ and their associated cost $c_i \in \mathbb{R}$ for each node n, are calculated based on a predefined heuristic graph movement, such that $\mathbb{R}^M \times \mathbb{R} \rightarrow \mathbb{R}^M$, provided the current node is not an element of x_{goal} otherwise the path from start to goal nodes is found [35].

2.2.2. RAPIDLY-EXPLORING RANDOM TREE (RRT) ALGORITHM

The standard RRT algorithm grows a tree from the start point, by randomly planting seeds (nodes) in the configuration space, consequently growing tree branches creating a feasible path [37]. This kinodynmic path planning algorithm, was developed by Steven M. LaValle and James J. Kuffner between 1998 and 2001 [37, 38].

Algorithm 2 defines and Figure 2.2 illustrates the rationale behind the RRT algorithm. The following will describe the parameters that are used to describe the standard RRT algorithm. $\tau \in (\mathbb{R}^M, \mathbb{R}^M \times \mathbb{R})$ refers to tree nodes and their edges with the first element at the starting node. X_{free} refers to unoccupied points in the environment space. $x_{init} \in \mathbb{R}^M$: start node, $x_{goal} \in \mathbb{R}^M$: goal node, $x_{rand} \in \mathbb{R}^M$: random node selected to construct new branches, $x_{near} \in \mathbb{R}^M$: nearest node to x_{rand}, $l_{near} \in \mathbb{R}^M$: nearest point on edge to

1: Assign x_{init} as an *open* node
2: Calculate $\hat{f}(x_{init})$ from the start node x_{init} to all *open* nodes n_i
3: Select the node n with the smallest $\hat{f}(x_{init})$
4: **If** $\hat{f}(x_{init})$ == *empty*
5: No path is possible
6: **End**
7: **While** $n \notin x_{goal}$
8: Mark node n as *closed*
9: Apply the successor operator Γ
10: Re-calculate $\hat{f}$ function for successor nodes of n and mark these nodes as *open*
11: **If** $\hat{f}$ is *empty*
12: No path is possible
13: **Else If** $\hat{f}(n_i)$ is now smaller than when n_i was *closed*
14: Remark successor *closed* nodes of n as *open*
15: **End**
16:**End**
17:Parent node n is *closed* (Path Found) [35]

Algorithm 1: A* Algorithm

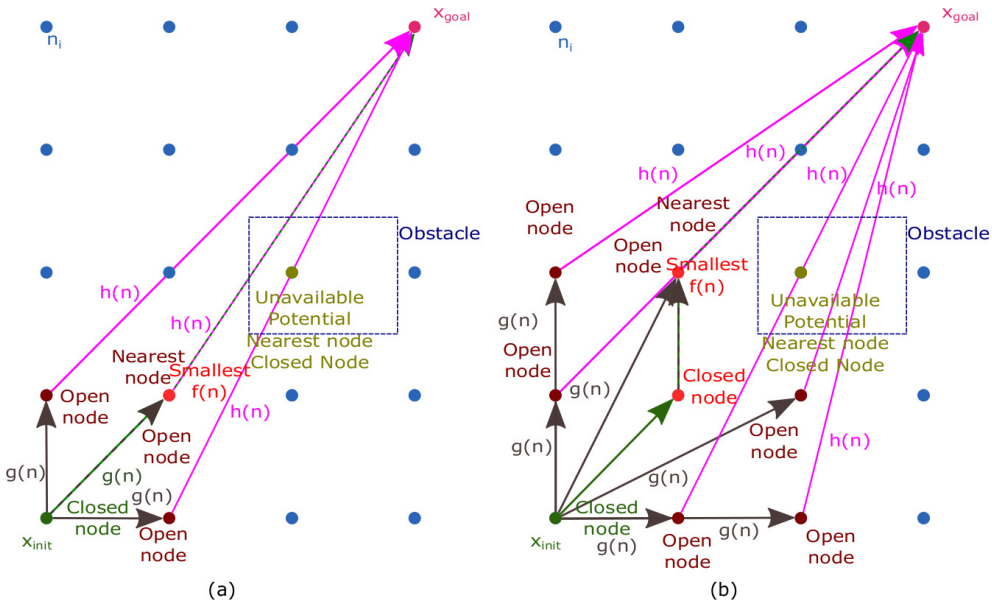


Figure 2.1: Two planning iterations of the A* Algorithm: (a) 1st Iteration and (b) 2nd Iteration

x_{rand}, $x_{edge} \in \mathbb{R}^M$: nearest node on edge, $x_{new} \in \mathbb{R}^M$: new tree node D distance from x_{near} or l_{near} equal to the preset tree branch length and K: predefined maximum number of iterates to construct a tree from start to goal.

1: Assign x_{init} to first node of tree τ
2: **While** $i < K$ **AND** $x_{goal} \notin \tau$
3: Randomly generate $x_{rand} \in X_{free}$
4: Find the nearest node x_{near} and edge l_{near} to x_{rand}.
5: **If** Distance (x_{near} to x_{rand}) > Distance(l_{near} to x_{rand}), **then**
6: Calculate x_{new}, a distance D from x_{near} to x_{rand}.
7: **If** Line connecting x_{near} to x_{new} does not result into a collision **then**
8: Add new node x_{new} to τ
9: **End**
10: **Else** Re-calculate x_{new}, a distance D from the nearest node on l_{near}, x_{edge} to x_{rand}
11: **If** Line connecting x_{edge} to x_{new} does not result into a collision **then**
12: Add new node x_{new} to τ
13: **End**
14: **End**
15: **If** Distance from (x_{new} to x_{goal}) < D **AND** line x_{new} to x_{goal} is free **then**
16: $x_{goal} \in \tau$
17: **End**
18: $i++$;
19: **End**
20: **If** $x_{goal} \in \tau$ then path is found
21: **Else** No path found
22: **End**

Algorithm 2: Pseudo Code of the RRT Algorithm

2.3. LITERATURE REVIEW

The main scope of this literature review is to identify potential path planning algorithms for 3D UAV path planning of small UAVs operating in indoor environments using limited onboard computational resources. Current literature reviews discuss and compare path planning algorithms either generically or in view of a specific application with a specific 2D or 3D robot. This differs with the scope of this work that presents a literature review in view of 3D UAV applications that is independent of the UAV model and focuses primarily on path planning performance. In this regard, this section will focus on the implementation, utilisation, advantages and disadvantages of graph-based and sampling-based path planning approaches and their variants in line with the scope. This review will conclude with a direct comparison of graph-based and sampling-based approaches in a common scenario.

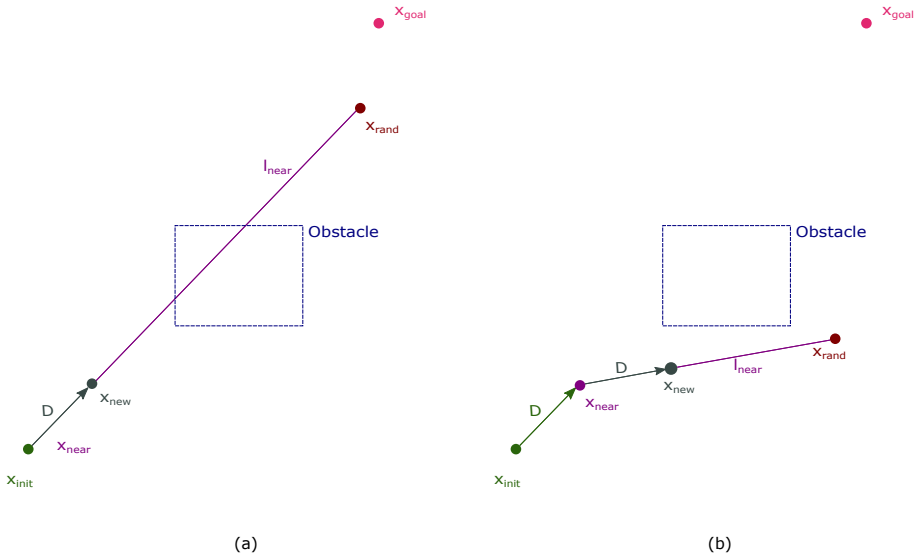


Figure 2.2: Two planning iterations of the RRT Algorithm: (a) 1st Iteration and (b) 2nd Iteration

2.3.1. GRAPH-BASED APPROACHES

Graph-based methods have been extensively employed for path planning in different engineering problems due to their relative simplicity. A pioneer on graph-based search algorithms was the Dijkstra algorithm [39]. The A* algorithm, an advancement over the Dijkstra's algorithm, combines the benefit of heuristic methods with the Dijkstra's algorithm [40, 42].

DIJKSTRA ALGORITHM

Dijkstra's algorithm searches all possible options to find the minimum cost paths from two points [39]. The Continuous Dijkstra, a variant of the original Dijkstra algorithm, finds the shortest path at boundaries through an analogy with Snell's Law of Refraction [43]. An improved Dijkstra algorithm uses the Fibonacci technique to sequentially select ideal path planning positions resulting in faster, smaller weight paths [44]. The Dijkstra's algorithm was used for path planning in urban environments [45], in multi-agent scenarios [46] and for autonomous driving in the DARPA challenge [47, 48].

A* ALGORITHM AND ITS VARIANTS

The A* algorithm is computationally efficient and guarantees the shortest possible path [40]. Path planning in both grid and visibility graphs can be computed using the A* algorithm. It was successfully implemented even for multi agent systems in dynamic environments [41]. A number of A* variants were developed to handle dynamic graph changes [42]. One variant, the Dynamic A* (D*) only updates new nodes, reducing computational demand [49]. Similarly, the focused D*, reduces computational demand by heuristically searching previous search results [50]. Also, the Lifelong Planning A* (LPA*), an incremental A* variant, makes use of invariant parts from previous searches to reduce

search time [51]. An advancement of the LPA is the D* Lite that uses one requirement to compare path planning priorities. The computational demand of D* lite is efficient at least as D* [52]. Another variant, the field D*, re-plans using linear-interpolation techniques to efficiently construct smooth paths [53]. The L* algorithm, a faster variant than A*, ensures linear computational complexity and can be applied in global path planning situations [54].

The Theta* is another variant of A*. Two variants of Theta* exist namely: the Basic Theta* and the Angle-Propagation Theta*[55]. The Basic Theta* is relatively computationally efficient although it does not ensure that an optimal path is constructed [55]. Oppositely, the Angle-Propagation Theta* generates slower, more complex and longer paths with respect to the Basic Theta* although it generates better worst-case complexity per vertex [55]. The Phi*, a variant of the standard Theta*, re-plans by searching in all angles using only nodes within line-of-sight of their respective parents [56]. Overall it requires more calculation time than the basic Theta* [56]. This time can be reduced by using the Incremental Phi* which uses a pre-processing stage [56] developed for the Differential A* algorithm [57]. Another variant, the Lazy Theta*, developed by Nash *et al.* [58] reduces the number of line-of-sight checks effectively reducing planning time. This algorithm was used to operate a UAV in real-time outdoor environments [59].

Another variant, the Block A*, first smooths predefined short paths available in a look-up table and then starts the standard A* search [60]. This approach is consequently faster than both A* and Theta* but owing that path construction is dependent upon the look-up table, this approach does not guarantee a solution [56]. Another fast approach is the Jump Point Search (JPS) which exploits symmetries but neglects neighbouring cells and does not search in every direction [61, 62].

The idea of using previous information to reduce computational time is the basis of other A* variants including Differential A* [57] and Fringe-Saving A* [64]. These methods initiate by re-planning iterations in the largest static area by comparing previous planning iterations [63]. Moreover, incremental heuristic algorithms, such as the Generalized Adaptive A* [65] and Path- and Tree-Adaptive A* [66], also use knowledge acquired from previous searches to enhance heuristics ultimately reducing computational demand. Such heuristics are a function of cost edges [63].

Anytime and truncated incremental techniques were also integrated with A* and its variants to improve performance. The integration of Anytime methods on A* and D* is realised in the Anytime Repairing A* (ARA*), Anytime Tree Restoring A* (ATRA*) and Anytime D* (AD*). These variants construct sub-optimal paths within a predefined short time frame [4]. Similarly, Truncated Incremental Search methods were integrated with LPA* (TLPA*), D* (TD*) and D* Lite (TD* Lite). These techniques utilise information from previous iterations to focus the search expansion based on sub-optimal target bounds [67]. These variants improve performance at the expense of increasing computational demand and therefore these methods suffer in high-dimensional environments [53].

This review shows that the standard A* algorithm has been extensively applied in various fields through the evolution of its variants and its integration with other techniques. Its extensive use is derived from the low computational and optimal path benefits that this algorithm can construct.

2.3.2. SAMPLING-BASED APPROACHES

Sampling-based methods are extensively used in various fields due to their relative low computational requirements and guarantee of convergence [32]. The most commonly used sampling-based methods are the Probabilistic Roadmap Method (PRM) [68] and the Rapidly-Exploring Random Tree (RRT) [38]. These methods construct cyclic and acyclic random trees respectively [32].

PROBABILISTIC ROADMAP METHOD (PRM)

The PRM algorithm is made up of two phases: the learning and the query phase. In the learning phase, a local planner connects sample points to form a roadmap while the query phase make use of the previously constructed roadmap to search a feasible path from start to goal point using a graph search algorithm [68].

To improve the performance of the standard PRM method, a set of variants was introduced. The PRM* improves path optimality by making use of a variant distance constraint [69]. To ease computational demand, the Lazy PRM initiates by neglecting all obstacles in the environmental space. Consequently, the algorithm constructs a path from start to goal and then checks for collision. Nodes and edges resulting in a collision are eliminated from the roadmap [70]. Similarly, the Dynamic Roadmap (DRM) method generates a roadmap by mapping the workspace maps to roadmap edges assuming an obstacle-free space as in the Lazy PRM. During online re-planning edges residing on obstacles are neglected [71]. The cell-based PRM (CPRM) method segments the environmental space into cells, assigning high priority to cells close to both straight path segments and disconnected components. For faster re-planning, searches are biased to solution areas [72]. In this regard, Pomarlan *et al.* [73] allocates higher costs for edges residing near colliding edges diverging the planner away from obstacle zones.

More PRM variants include the Reactive Deformation Roadmap (RDR) which generates a roadmap attracted by time-variant sub-targets and repelled by obstacles. Subtargets and their respective reactive links are added and removed to ensure roadmap connectivity [74]. The Flexible Anytime Dynamic PRM (FADPRM) selects desirable areas by environment segregation. An A* search continuously improves the solution through priority heuristics focused in the vicinity of frontier edges [75]. The Elastic Roadmap variant connects roadmap points in a dynamic environment through feedback control [76]. The Grid Path Planning Roadmap Planning (GPRM) and the Particle Probabilistic Roadmap (PPRM) are another two variants with the former producing a grid environment and the latter utilising a probabilistic approach [77].

RAPIDLY-EXPLORING RANDOM TREE (RRT)

The Rapidly-Exploring Random Tree (RRT) [38] constructs a unidirectional search tree starting from the start node by randomly generating and connecting states not residing on obstacles until one tree branch extends to the goal node. A distinguished advantage of the RRT with respect to the PRM is the consideration of kinematic and dynamic constraints in the path planning process [38]. Paths constructed using the standard RRT are computationally efficient even in complex situations although they suffer from path optimality [78–80]. The RRT trees grow evenly in free spaces unless restrictions or potentials are included in the random generation of nodes [81]. This allows the RRT algorithm to be used in non-holonomic and restricted environments. In this regard, the

standard RRT algorithm was successfully implemented to construct collision-free paths in the presence of static obstacles [16].

The lack of optimality is one of the major drawbacks of the standard RRT. To tackle this drawback, the RRT* [82] was developed. RRT* exhibits asymptotic optimality by introducing a minimum length cost, although it lacks in ensuring a local optimum at the start node [69]. During sampling the RRT* neglects the configuration-cost function and considers only the quality of the path when introducing or removing nodes. Due to its asymptotic optimality, RRT* is slow in achieving an optimal path in complex high-dimensional environments [83]. The RRT*-Smart variant was developed to improve convergence with respect to RRT*, by constructing quasi optimum or optimum paths in a smaller time frame [84]. Another RRT* variant was developed to add probabilistic guarantees of completeness and optimality by constructing tree edges based on vehicle control inputs[85].

The bidirectional or RRT-Connect grows two trees, one starting from the start and the other from the goal node [86]. The Artificial Field algorithm was integrated with the RRT-Connect algorithm to optimise path length [87] and with the RRT* algorithm to optimise convergence speed [88]. The Execution Extended RRT (ERRT) variant efficiently re-plans to improve path quality by using both cost priority search and previous nodes [89]. Also, Martin *et al.* [90] proposed the integration of evolutionary algorithms with bidirectional RRT creating the bi-directional Rapidly Exploring Evolutionary Trees (RET). By searching in the vicinity of neighbouring nodes through balanced Spatial KD-Trees and using *apriori* environmental knowledge, results show an improvement in time variant environments [89, 90]. Gros *et al.* concludes that bidirectional trees are inherently less flexible in cluttered environments than unidirectional search tree algorithms [11].

Another RRT variant neglects the connection to the nearest node rule and connects a new tree node to all obstacle-free nodes [91]. Re-planning is done using an obstacle-free distance cost function in a predefined time and provided that no collisions exist prior re-planning. Just as for the cell-based PRM, the cell-based RRT uses probabilistic techniques and decomposes the environment for efficient path planning [92]. A set of decomposition granularities were assessed and results show that biasing the search in the vicinity of the goal improves success rate and reduces path length [92]. By using previously constructed path segments and applying intelligent decomposition techniques based on the environmental situation, performance was improved [92].

The Transition-based RRT (T-RRT) variant adds nodes to low cost areas to focus the search to these regions using a Metropolis-like transition test [79]. By eliminating the path quality requirement in the construction of edges, path optimality is negatively affected [79]. An advancement over T-RRT, the Transition-based RRT* (T-RRT*) combines low-cost area biasing of the T-RRT algorithm with the path quality constraint of the RRT* algorithm [79]. In T-RRT* different physical such as road limits and environmental including obstacles restrictions were used to formulate biases for efficient tree growth [2, 93, 94]. Similarly, Shang *et al.* [95] used previous planning knowledge to bias the search in a high-complexity 3D environment.

As with graph-based paradigms, Anytime methods integrated with RRT algorithms [96–98] use previous searches to efficiently construct a path that is optimal, converging and feasible. In this regard, the Anytime T-RRT (AT-RRT) uses the Anytime paradigm to T-

RRT for path re-planning to converge the path to optimum [79]. An Anytime integration to RRT is the RRT-Roadmap (RRM) which compromises between exploration and path quality based on RRT* [99]. This meta-approach amalgamates short-cutting with path hybridization [100]. The Dynamic RRT (DRRT) neglects child nodes of invalid tree nodes and re-plans from goal to current node [101]. Furthermore, the Anytime Dynamic RRT (AD-RRT), exploits the strengths of both techniques [102]. A further enhancement, the multipartite RRT (MP-RRT) designed for unknown or dynamic environments, samples and re-plans as in ERRT, removing invalid nodes as in DRRT [103].

Due to their benefits, RRT algorithms are also applied in real-time path planning applications. In this regard, the RRT^X a real-time asymptotic optimum re-planning algorithm continuously constructs non-colliding and optimal paths in time-varying environments. The path from start to goal is re-planned by updating search tree when a potential collision is detected [104]. The Partial Motion Planner (PMP) dedicates a fixed time window for both path planning and traversal. Similarly to RRT^X, this real-time path planner, constructs current to goal node path segments of predefined length. Then the environmental model is updated and the re-planning process restarts [105]. Another real-time planner, the Closed Loop RRT (CL-RRT), re-plans a tree from the current position in a predetermined time window using a closed-loop feedback method. This algorithm can be applied even for non-linear controllers and agent models with unstable dynamics. The stabilisation controller associated with CL-RRT improves prediction accuracy and rejects disturbances acting on the vehicles [2].

The Greedy Incremental Path-Directed (GRIP) algorithm constructs a path prior movement by growing a tree from current point in predetermined time by considering only kinodynamic constraints [106]. During movement, the algorithm uses free nodes found in previous planning stages to update the tree. The search is greedy biased provided that state-space exploration is probabilistically ensured [106]. Similarly, Taheri *et al.* [107] developed a Fuzzy Greedy RRT (FG-RRT) path planning algorithm that controls the tree edge propagation in configuration space, resulting in lower path planning time and complexity with respect to the greedy RRT algorithm.

Reconfigurable Random Forests (RRF) algorithm applies the underlying principles of MP-RRT and separates and grows disconnected trees, with the aim of interconnecting them when trees grow [108]. To mitigate with moving obstacles the Lazy Reconfigurable Forest (LRF) keeps a forest of trees. As opposed to DRRT which validates the entire forest, LRF validates only path edges [109]. Similarly, Multiple RRTs (MRRT) algorithm simultaneously grows a set of trees starting from predetermined points including the start and goal points. The ultimate goal is to improve path planning and exploration [81]. The Multiple Incremental RRTs (MIRRT) variant uses incremental paradigms to keep trees constructed in previous iterations that may be used in future path planning [81].

The evolution of PRMs to RRTs to numerous sampling-based subcategories that even integrate with other techniques show the maturity and flexibility of sampling-based techniques. This literature review showed that these methods were applied for path planning in 2D and 3D in both static and dynamic environments using different vehicle model. Therefore, sampling-based methods can be considered as candidates for feasible path planning using a wide spectrum of vehicular systems.

2.3.3. COMPARING APPROACHES

Studies compared path planning performance of inherent graph-based with sampling-based techniques. In this regard, Tsai *et al.* [110] firstly implemented the RRT algorithm for 3D path planning of multirotor Umannned Aerial Vehicles (UAVs) and the Dijkstra algorithm for path length reduction. In a second implementation using the same system, 3D path planning was achieved using the RRT-Connect algorithm. Path length was reduced through an A* variant that only made use of the flight path angle without grid discretization. In both cases Bézier Curves were employed for 3D path smoothing. Tsai *et al.* [110] noted an improvement of computational demand and path efficiency by using RRT-Connect instead of the standard RRT, the amended A* instead of the Dijkstra algorithm and the addition of path smoothing through the Bézier Curves [110].

Similarly, Rao *et al.* [111] compared the RRT-Connect and A* path planning algorithms for Autonomous Underwater Vehicles (AUVs) to generate feasible energy-optimal 2D paths biased by ocean currents. An Iterative k-Nearest RRT (IkRRT) considered k-neighbours to adjust heuristics required to set up the Voronoi-biased RRT-Connect tree [112]. Rao *et al.* [111] concluded that both RRT-Connect and A* algorithms are potential candidates for this application. RRT methods found it difficult to find solutions in view of Voronoi-bias although the planner was able to avoid shallow regions. Oppositely, the A* methods performed well in strong current situations [111].

Graph-based and sampling-based paradigms employ different methodologies to construct a path from start to goal. Both have their inherent advantages and disadvantages which make each method ideal for different applications. Mainly, graph-based methods discretise the environment, reducing computational cost but limiting the possibility of available nodes based on the resolution that is selected. Oppositely, the non-discretised sampling-nature of the other method, may be computationally expensive in a relatively large obstacle-rich environment but will gradually converge to a solution (if possible) provided there are no time restrictions. In conclusion, it is adequate to analyse both rationales to best select the most promising method for the considered scenarios.

2.4. PATH PLANNING ALGORITHM ENHANCEMENTS AND TEST ENVIRONMENT

2.4.1. INTRODUCTION

The main aim of this part is to describe the A* and RRT variants that are implemented to provide a better comparison between graph-based and sampling-based paradigms. Moreover, to investigate the effects of path smoothing and for fair comparison, the same smoothing algorithm is applied to both A* and RRT variants. Finally, the test environment constructed to assess the performance of each path planning algorithm is described.

2.4.2. THE A* RIPPLE REDUCTION ALGORITHM (A^*_R)

The A*'s algorithm graph-based paradigm described in Section 2.2.1 introduces situations where a small increase in resolution results in a much longer or shorter unsmoothed path and vice-versa. This effect is explained in our previous work [113].

To attenuate this, all free graph points are shifted by a random distance varying be-

tween zero and half the distance between adjacent grid points. Obstacle planes are not shifted with the same distance. This creates a different grid layout for the same situation. This techniques will ultimately average out large variations in path length segments created by the discretision nature of the A* algorithm.

In A*, each situation is repeated 100 times in each scenario and resolution yielding the same path in all 100 tests. In A^*_R, a different random shift is added for each situation (scenario and resolution) creating 100 different paths. The mean path length for these 100 paths will effectively reduce the path length ripple. Refer to [113] for further details.

2.4.3. THE RRT ALGORITHM WITHOUT STEP SIZE CONSTRAINT

In this variant, the RRT algorithm's tree branch length defined in Section 2.2.2 is limited by the *step size*. In an alternative approach, the *step size* constraint is neglected and tree branches are connected between the nearest and randomly generated nodes through straight lines only if a collision-free path connecting these two points exists. The path following algorithm shall segment the path into trajectories in view of vehicle's kinematic and dynamic constraints. This algorithm is intended to offer a comparison between the standard RRT and Multiple RRT (MRRT) algorithms.

2.4.4. MULTIPLE RAPIDLY-EXPLORING RANDOM TREE (MRRT)

The MRRT algorithm constructs a predetermined number of trees that are simultaneously expanded. Tree seeds are placed at start and goal positions with the others placed at random or preset positions [81]. Trees grow per iteration in the direction of the randomly-generated node as in RRT and ultimately interconnect neglecting *step size* constraints. Similar to the RRT, this algorithm terminates when a single tree connects the start and goal nodes.

2.4.5. SMOOTHING ALGORITHM

A smoothing algorithm is applied to all the path planning algorithms considered in this study. This post path planning algorithm randomly selects two path points and randomly selects a point from each line connecting the selected node to their respective next path point. Path points in between the latter two points are neglected if a collision-free interconnection is possible. Path planning test results show that after 100 smoothing iterations the path length improvement is less than 5%. Based on this analysis, the iterations are set to 1000.

Path planning analysis of the A*-based and RRT-based algorithms show that A* methods require fewer smoothing iterations to reach a point where path length reduction is negligible when compared with RRT-based methods. For fair comparison the smoothing iteration stopping condition must be set at a point where the prospective improvement of future iterations is negligible for the problem scope. An indication of the prospective improvement of future iterations can be provided from the path length reduction of the previous set of smoothing iterations. In fact, analysis shows that path length asymptotically approaches a minimum with the number of iterates. This rationale is used in the development of a variant of the smoothing algorithm.

This new smoothing variant is governed by the path length reduction improvement over a preset value of iterations, defined by $excess$ and set to 20. The smoothing algo-

rithm is required to compute at least *excess* number of iterations. If the smoothing algorithm has computed more than *excess* and improvement is below a predefined value (set at 1%) the smoothing process is finished otherwise the smoothing algorithm computes a new iteration. The smoothing process also stops when the smoothing iteration reaches the maximum preset value (set at 1000). For further details of this algorithm and discussion about the empirical definition of the described parameters, please refer to [113].

2.4.6. VEHICLE MODEL DEFINITION

The main scope of this study is to assess, compare and possibly enhance the performance of graph-based and sampling-based algorithms in static 3D environments. A generic approach is considered in both environmental and vehicular model definitions. As both methods and their variants were applied to wide spectrum of vehicles, we considered the simplest and most generic type of model, namely the point model, to attenuate or possibly eliminate the vehicle model's kinematic and dynamic effects and their estimations on the path planning performance for both algorithms.

The point model considered assumed that:

1. The vehicle can rotate any angle in all 3 dimensions on itself without changing position.

2. The vehicle has no orientation and is symmetrical in all dimensions.

The path planning algorithm generates a set of points that, when interconnected, connects the start and goal positions. The interconnection of these points can be done by any line or curve depending upon the kinematic and dynamic capabilities of the vehicle. The vehicle dependent trajectories shall then be constructed by an associated path following algorithm using path points generated from the path planning algorithm. Ultimately, the path following algorithm will be integrated with the path planning algorithm, as in [114]. For the scope of analysis we assumed that path segments connecting consecutive points are linked by straight lines.

Furthermore, in path construction it is assumed that the point model can reside a buffer distance from the nearest point on obstacle planes without colliding with them. If a prospective path point or a point on the line intersecting two consecutive path points reside within a smaller distance than the described buffer distance, the path planner will invalidate the last created node and the path planner re-plans. This obstacle avoidance rationale is applied to all algorithms including the post processing smoothing algorithms. This distance is empirically defined as half the distance between grid positions in A*. To offer a fair comparison the same distance is considered for RRT-based methods. Depending upon the kinematic and dynamic constraints and vehicle dimensions of the considered vehicular system, this buffer distance can be increased or decreased appropriately. Also, a generic unit cube is considered in the definition of the environment. This allows the tests to be scaled to any environment and consequently adapt the buffer distance between vehicular path and obstacles.

2.4.7. EXPERIMENTAL SCENARIOS

A unity cube with generic units is considered as the environmental space. The centre of the cube is at the origin with limits from -0.5 to 0.5 in all 3 dimensions. The normalisation of the environmental space allows the testing environment to be exported to real-life environmental spaces. The test scenarios that will be considered in this study where developed by by Clifton *et al.* [115]. These are available online at [116]. The start and goal points are set at [0,-0.5,0] and [0,0.5,0], respectively, for all tests. The following obstacle scenarios, illustrated in Figure 2.3 are considered:

1. Scenario 1: Two obstacle planes on the X-Z axis with 0.2x0.2 square openings;

2. Scenario 2: Three obstacle planes on the X-Z axis with 0.2x0.2 square openings and two obstacle planes in the X-Y planes with no openings; and

3. Scenario 3: Five obstacle planes in the X-Z axis with 0.2x0.2 square openings and two obstacle planes in the X-Y planes with no openings.

The considered control parameters are the resolution for A*, the step size per iterate for RRT and the number of seeds per axis for MRRT. The resolution is varied between 11 and 29 in steps of 2 for A* and consequently the step size per iterate for RRT is varied between $\frac{1}{10}$ and $\frac{1}{28}$ in steps of $\frac{1}{10+(2i)}$ where i=0,1,...,9. The number of seeds per axis for MRRT is varied between 2 and 20 in steps of 2.

The same random sequence generator is considered for RRT for unbiased comparison. The following formula, developed in [116], is used to evaluate the maximum tree number (N) as a function of the number of seeds per axis (K):

$$N = 3(K^3) + 2 \tag{2.3}$$

The minimum number of trees is 2, one at the start and another at the goal nodes. The multiplication and power factors of 3 are a result of the three dimensional environment the path planning algorithms will operate in.

Each test for all algorithms is repeated for 100 times. Path length before and after smoothing and computational time are the performance measures that are considered. Owing to the generic environment, the subsequent results can be extrapolated to assess real applications. [1].

2.5. RESULTS

2.5.1. INTRODUCTION

This section aims to identify the best configurations for the realisation of 3D path planning. Therefore, this section will present, analyse, discuss and compare the path planning performance of the A* and RRT algorithms, their variants and the smoothing algorithm. This will help identify the strengths and weaknesses of each configuration in 3D path planning in both generic and specific environments with associated mission constraints.

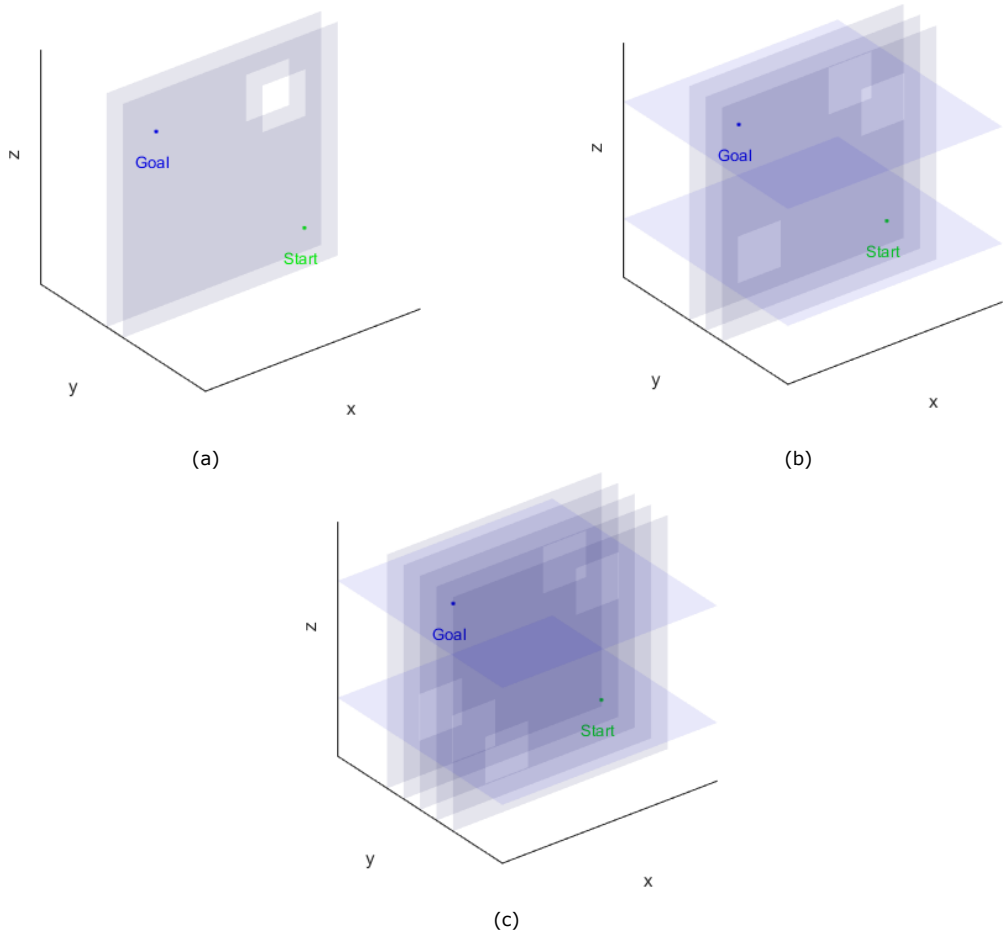


Figure 2.3: The different scenarios: (a) Scenario 1 (b) Scenario 2 and (c) Scenario 3

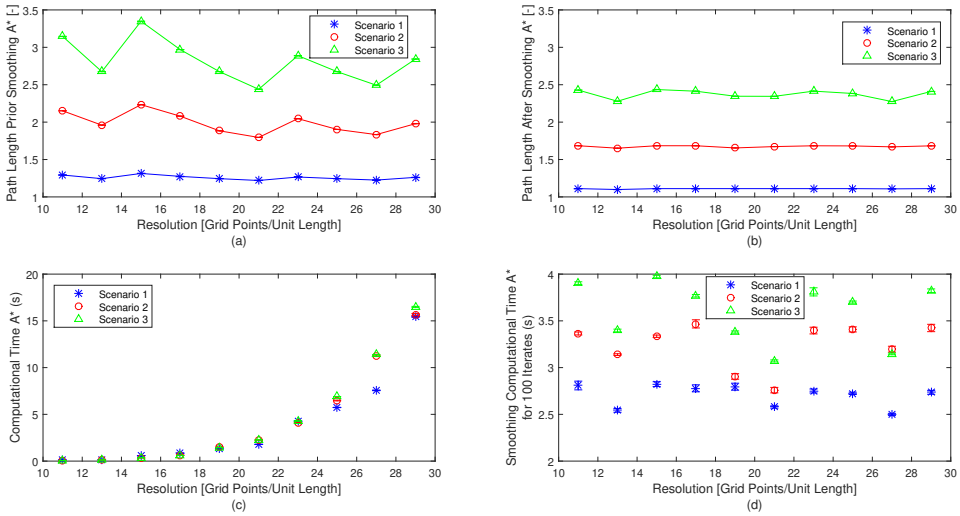


Figure 2.4: Results for A* with 95% Confidence Interval (CI): (a) Non-smoothed path length, (b) Smoothed path length, (c) Planning Time and (d) Cumulative Smoothing Time for 100 Iterations

2.5.2. A* ALGORITHM

The A* path planner constructs a path from start to goal points in all scenarios and resolutions. This may imply that using the A* algorithm a solution will always be guaranteed. But as remarked in literature [40], this is not the case as a solution depends also upon the selected resolution. From Figure 2.4 (a) it can be deduced that path length prior smoothing is a function of scenario complexity. In this regard, the path lengths for Scenarios 2 and 3 are 50% and 100% longer with respect to Scenario 1, since the path planner must pass through three and five windows in opposite obstacle plane extremes instead of two windows respectively, as can be deduced from Figure 2.3.

The direct relationship between path length and scenario complexity is further confirmed in the path length after smoothing results illustrated in Figure 2.4 (b). The ripple effect in path length is reduced at smoothing stage since path segment length and grid position restrictions are eliminated. A comparison between the mean smoothed and unsmoothed path lengths show a 12%, 16% and 17% decrease for Scenarios 1 to 3 respectively, with respect to the unsmoothed path lengths. Tests show that increasing the number of smoothing iterates beyond 1000 increases computational time proportionally without any significant improvement in path length. Moreover, through smoothing iterate increase, the number of path points increase consequently increasing path navigation complexity and ultimately reducing navigation performance.

From Figure 2.4 (c) it can be concluded that path planning time is a function of resolution and scenario difficulty, with resolution having the major dependence. Resolution is directly proportional with the number of grid points and therefore the more

[1] An Intel Xeon ES-1650, 3.2GHz is used for testing

time is required by the A* path planner to generate a path. Similarly, scenario complexity increases path planning time as more options need to be considered for a non-colliding path to be computed. The path construction time is much larger than the path smoothing time and therefore the latter effect on computational time is negligible. Scenario complexity also effects path smoothing time as can be deduced from Figure 2.4 (d).

2.5.3. A* RIPPLE REDUCTION ALGORITHM (A^*_R)

As illustrated in Figure 2.4 (a) the variance limits at 95% confidence interval for the unsmoothed path length is 0. This results since each test for the same resolution and scenario is repeated for 100 times using the same start and goal positions. Therefore all 100 runs will yield the same path. Oppositely, with the introduction of A^*_R, each run is different leading to a non-zero 95% confidence interval as illustrated in Figure 2.5. An asymptotic relationship between path length and resolution can be deduced from Figure 2.5 (a). From a theoretical perspective, as the resolution increases the distance between graph nodes reduces, increasing the number of graph points in the same environmental space and therefore more optimised paths can be constructed. This confirms the conclusion drawn from the tests that path length and resolution have an inversely proportional relationship.

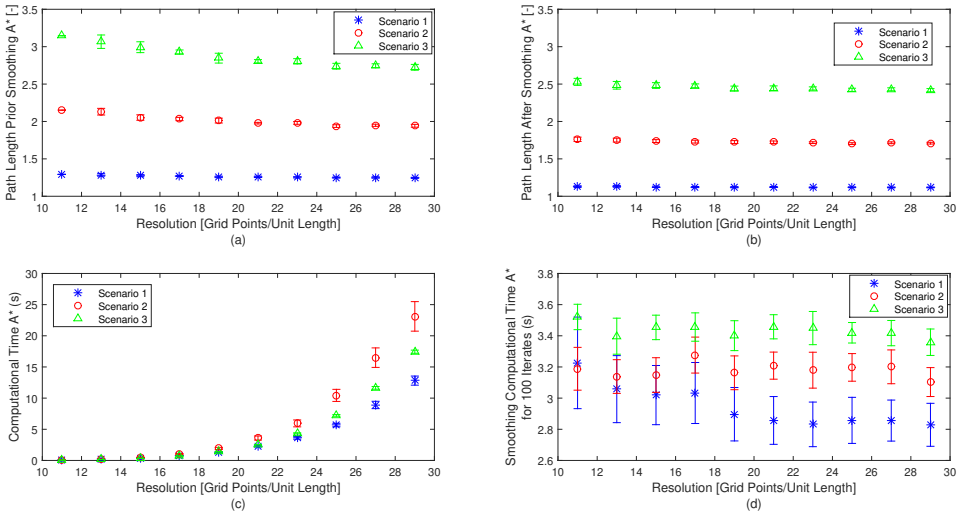


Figure 2.5: Results for A* Ripple Reduction Algorithm (A^*_R) with 95% CI: (a) Non-smoothed path length, (b) Smoothed path length, (c) Planning Time and (d) Cumulative Smoothing Time for 100 Iterations

An asymptotic behaviour in confidence interval with increase in resolution is also exhibited for the smoothed path length (Figure 2.5 (b)) since with increase in resolution the path length variance in possible path options reduces. Similar to path generation time results of A*, the confidence interval for path generation time increases exponentially as

for A*. In terms of confidence interval, the resolution has a minimal effect on smoothing time. The smoothing algorithm is not restricted by grid positions and therefore is independent of resolution.

A path length ripple reduction of 46%, 46% and 48% in terms of standard deviation for Scenarios 1 to 3 respectively, is introduced by the A^*_R algorithm with respect to the standard A* algorithm. This result is confirmed using a different random seed generator. This result confirms that path length ripple exhibited by the A* algorithm is the result of a plane residing exactly on a plane of grid positions with the only available grid points residing at the respective obstacle windows. Using the A^*_R algorithm that normalises such iterations, a fairer A* performance response results.

With the addition of the smoothing algorithm the path length ripple is further reduced by 3 to 4 times for all scenarios but never reaches 0. Results show that the ripple reduction in amplitude due to the smoothing algorithm is higher for the standard A* with respect to A^*_R, since in A* the ripple is higher. This behaviour shows that the smoothing algorithm's path length reduction reduces as the path reaches its optimal length.

A marginal mean increase of 1%, 2% and 2% for Scenarios 1 to 3 respectively, results in unsmoothed path length for A^*_R with respect to A* algorithms. This increase is a consequence of obstacle modelling that defines a buffer of half the distance between nodes. Without this buffer, planes not residing exactly on nodes will be neglected, but this buffer can create situations where one obstacle plane is modelled by 2 obstacle planes.

The mean smoothed path length shows 1%, 3% and 4% increase for Scenarios 1 to 3 respectively, for the A^*_R with respect to A*. This increase correlates with the path length increase in the unsmoothed path length of both A* and A^*_R algorithms. The path planning time is 5% shorter and 49% and 4% longer for the A^*_R algorithm with respect to the A* algorithm. The major increase in path planning time for Scenario 2 is a consequence of the obstacle grid point estimation as explained earlier in [113]. The path smoothing time is 9% longer, 2% shorter and 5% shorter for the A^*_R algorithm with respect to the A* algorithm. This marginal difference is a consequence of the environmental shifting and the associate possibility of different paths. Further details are provided in [113].

In conclusion, it can be stated that the A^*_R algorithm reduces path length ripple with respect to the A* algorithm. This reduction is not achieved at the expense of longer path length or computational time. These results are confirmed through tests conducted on different complexity 3D scenarios.

2.5.4. RAPIDLY-EXPLORING RANDOM TREES (RRT)

Figure 2.6 shows that the RRT algorithm finds a path solution in all scenarios and for all considered step sizes. This supports the notion that the RRT algorithm is probabilistically complete [32]. Figure 2.6 (a) shows that the path length prior smoothing is mainly independent of the step size for all scenarios. The step size determines the tree branch length. Therefore the zig-zagging amplitude is directly proportional with the step size. The sampling-based nature of the RRT algorithm in constructing a path using standard length tree branches shows the algorithm's lack of optimality emphasised in literature [32]. The smoothing algorithm reduces the mean path length by 46%, 50% and 49% for scenarios 1 to 3 respectively.

Overall, the A* algorithm constructed shorter unsmoothed paths with respect to

RRT for all scenarios. Although both A* and RRT algorithms are restricted in movement, A* can only pass through predefined grid positions while RRT can use any non-obstacle space, provided that this is distant by the step size constraint from the parent node. Therefore path length results cannot be compared directly. Similarly for both algorithms, the path construction time exhibits an inverse relationship with resolution or step size since the number of path segments for equal volume increases with increase in resolution and decrease in step size. Although the smoothing time for both algorithms consumes only a small fraction of the total path generation time with resolution and step size constraints neglected, scenario complexity also increases smoothing time.

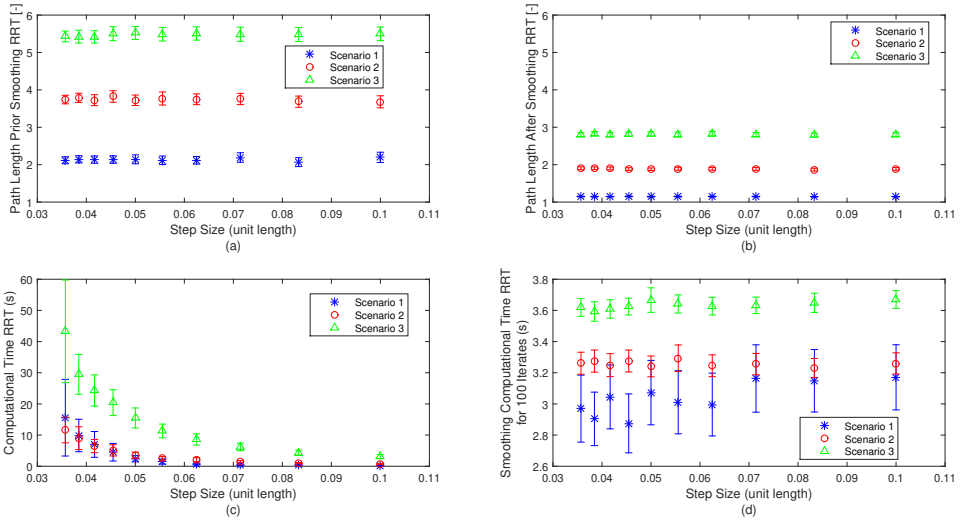


Figure 2.6: Results for RRT with 95% CI: (a) Non-smoothed path length, (b) Smoothed path length, (c) Planning Time and (d) Cumulative Smoothing Time for 100 Iterations

2.5.5. RAPIDLY-EXPLORING RANDOM TREES (RRT) WITHOUT STEP SIZE

Table 2.1 tabulates the mean results for all three scenarios for 100 iterations with no limits on tree branch length. The unsmoothed path length is 17%, 11% and 4% longer for Scenarios 1 to 3 respectively, for the unconstrained RRT variant with respect to the standard RRT. The difference reduced significantly to 1%, 0.4% and <0.1%, for Scenarios 1 to 3 respectively when the smoothed path lengths of the variant and standard RRT are compared, with the latter remaining the shorter. **Therefore it can be concluded that path length is deteriorated by removing constraints, with the major deterioration resulting in low obstacle density scenarios as long zig-zagging path segments are constructed which would have been attenuated if tree branch lengths are constrained.**

Path planning time for this RRT variant is reduced by an average of 450, 7 and 10 times with respect to the standard RRT for Scenarios 1 to 3, respectively. This result shows that the tree branch length constraint limits the propagation to the goal position

Table 2.1: RRT without step size constraints

Parameter	Scenario 1	Scenario 2	Scenario 3
Non-smoothed path length [-]	1.83	3.38	5.27
Smoothed path length [-]	1.13	1.88	2.81
Path planning time (s)	0.01	0.67	1.64
Average smoothing time per iterate (ms)	35.97	46.57	51.69

especially in simple scenarios where only a small amount of manoeuvres are acquired to reach the goal. An increase of 18%, 43% and 42% in smoothing time is exhibited for the RRT without step size constraint with respect to the standard RRT. A slight increase in smoothing time is exhibited for the unconstraint RRT in comparison with the constraint RRT mainly due to the zig-zagging in the low obstacle density paths. Path planning time for A* is also larger than for this RRT variant since the latter is not limited by grid positions. The difference is more emphasised at low resolutions in simple scenarios. In addition, this RRT variant's smoothing time is less than half of A* for the same scenarios.

2.5.6. MULTIPLE RAPIDLY-EXPLORING RANDOM TREES (MRRT)

Figure 2.7 shows that a non–colliding path from start to goal is constructed for all scenarios. Results show that scenario difficulty is the main factor affecting path length while this parameter is not affected by the number of seeds per axis. **MRRT constructs longer unsmoothed paths when compared with the unconstrained RRT for all scenarios.** In MRRT, trees can be interconnected with no length restrictions. These lack-of-length restrictions result in zig-zagging paths prior smoothing, increasing in amplitude at low complexity scenarios, similar to the unconstrained RRT results.

The smoothed path length for MRRT remains longer than that for RRT for each respective scenario, although the difference in mean path length between the two methods is reduced through the smoothing algorithm. For A*, the mean unsmoothed path length is less than half with respect to MRRT for all scenarios. As in other algorithms discussed in this study, the path planning time for MRRT depends on scenario difficulty. Furthermore, the planning time is directly proportional with seeds-per-axis value.

Overall, the MRRT algorithm constructed a path in less time than both A* and RRT with the major difference exhibited in low complexity scenarios. This results since the MRRT tree propagation and interconnection is unrestricted as opposed to both A* and RRT which are restricted in path points and tree branch length, respectively. The unconstrained RRT constructed the path in multiple less times as opposed to MRRT. To construct a path, the MRRT requires more nodes than both RRTs, resulting in a longer smoothing time. In comparison with A*, the MRRT required less smoothing time owing that A* constructs almost optimal paths and therefore the smoothing algorithm requires more time to find path segments which can improve the path.

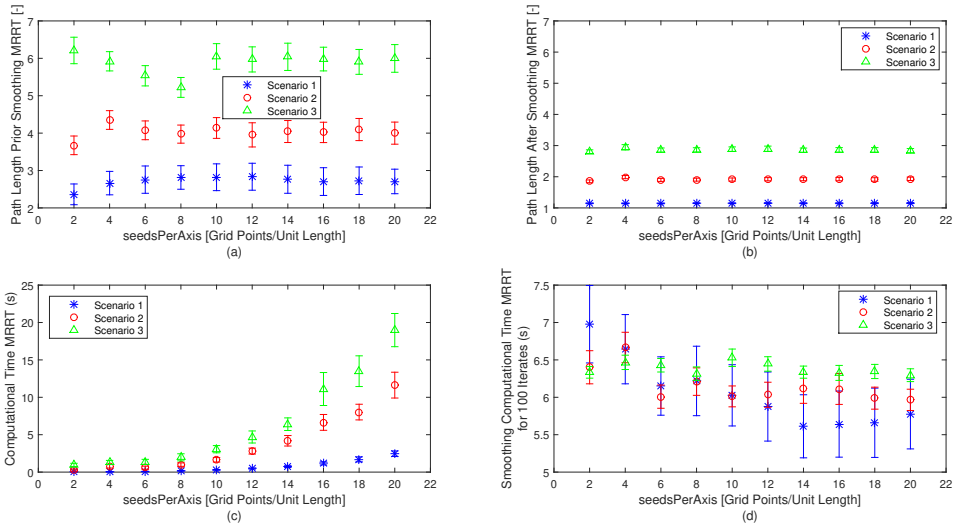


Figure 2.7: Results for MRRT with 95% CI: (a) Non-smoothed path length, (b) Smoothed path length, (c) Planning Time and (d) Cumulative Smoothing Time for 100 Iterations

2.5.7. NEW SMOOTHING ALGORITHM

The new smoothing algorithm is computationally more efficient than the older smoothing algorithm version. For the A* algorithm, results show that the lower the scenario complexity, the more smoothing iterations are required since the stopping condition (<1% reduction in last 20 iterations) is reached. This results since in low obstacle density scenarios, the environmental space is primarily obstacle free as opposed to high obstacle density scenarios where path improvement is more restricted. Also, grid point restrictions are neglected by the smoothing algorithms, therefore path planning restriction due resolution are eliminated. For complex situations, such as Scenario 3, the minimum of 20 iterations is too low as sometimes, owing to scenario complexity, it is too difficult to find a set of points in 20 iterations on the path segments to construct a shorter overall path. For more results and further analysis please refer to [113].

The new smoothing algorithm requires more time for RRT with respect to A* for all step sizes/resolutions and scenarios under review. Step size constraints minimally affect the number of smoothing iterations since the stopping condition is reached. This results for the same reason explained for A*. Smoothing results for RRT in simple scenarios show a reduction of between 13-19 times with a minor increase in smoothed path length. Results show that in complex scenarios the minimum of 20 smoothing iterations is too low and shall be increased as is it difficult to find shorter non-colliding path segments, explained for A*. The smoothed paths resulting from the A* algorithm outperform those from the RRT algorithm in terms of both length and smoothed time since the unsmoothed path generated by the A* is better than that for RRT. The elimination of the tree branch length in RRT has no effect on smoothed path length and time.

For MRRT, scenario complexity increases the number of smoothing iterates until the

stopping condition is reached. This is attributed to the lack of path optimality constructed by the RRT-based algorithms. MRRT's seeds per axis parameter have no effect on the smoothing algorithm's stopping condition since it neglects nodal and path segment length constraints. Through the new smoothing algorithm smoothing time has decreased by 22,17 and 24 times for Scenarios 1 to 3 respectively, with respect to the original smoothing algorithm.

In conclusion, results show that a path length reduction of 20% - 50% can be achieved at a small percentage of the path construction time. Furthermore this analysis highlights the direct relationship between smoothing iterations and path length. This can guide in custom tuning the amount of path length reduction based on the application requirements and computational power availability.

2.5.8. CONCLUSION

This section discussed the path planning performance of the different path planning algorithms and their variants presented in Section 2.2 and Section 2.4 and outlined the main outcomes derived from the tests presented in this section. In summary, it was concluded that:

1. All algorithms constructed a path in all scenarios considered.
2. The A^*_R algorithm is able to reduce the ripple in path length introduced by the original A* without compromising on path length and time.
3. The unsmoothed path length for A* is shorter than that for RRT in all scenario complexities.
4. The RRT without step size constraints reduced the path generation time by multiple oders of magnitude with respect to the standard RRT at the expense of increasing path ripple.
5. The MRRT algorithm constructed longer path in less path planning time with respect to the standard A* and RRT.
6. The new smoothing algorithm presents an improvement (by multiple times) in terms of computational time over the original smoothing algorithm.

Table 2.2 summarises the results presented in this section by ranking from 1 to 5 all the considered path planning algorithms in terms of path length and path planning time.

2.6. CONCLUSION

This work analyses the path planning performance of the A* and RRT based algorithms with an associated smoothing algorithm in different complexity 3D environments. In line with literature, the A* algorithm constructed more optimal paths than RRT. The A* algorithm searches volumes in the line of sight of the goal while the RRT algorithms searches evenly throughout the environment. The paths constructed by the A* algorithm are shorter, also after smoothing, and constructed in less time than RRT. The A* algorithm exhibits path length ripples as resolution is varied for the same scenario. Through

Table 2.2: Ranking of path planning algorithm in terms of path planning performance

Parameter	Path Length	Planning Time	Ripple Amplitude
A*	1	3	2
A_R^*	2	4	1
RRT	3	5	3
RRT without step size	5	1	5
MRRT	4	2	4

A* ripple reduction algorithm, a 46% to 48% path ripple reduction is recorded for all situations considered with respect to the A* algorithm. This improvement is achieved without increasing path length and planning time. The unconstrained RRT variant reduced path planning time especially in low obstacle density scenarios. Moreover, the evenly-distributed MRRT generated longer unsmoothed paths in shorter planning times but required more smoothing over RRT for all considered scenarios. The new smoothing algorithm, developed to eliminate unnecessary smoothing iterates, shows a 90% reduction in smoothing time for all algorithms. This smoothing time reduction is achieved with a path length increase of less than 10% for A* and between 25% and 45% for all RRT algorithms.

In addition to path length optimality and non-colliding paths, the path planning time is also an important factor especially in time-varying environments. In such situations, the path must be continuously checked, optimised and if a potential collision is possible, re-planned using only available and limited onboard resources. This work highlights A*'s potential as an online 3D path planning algorithm in dynamic environments owing to its optimality and low computational demand. A*'s environmental discretisation nature allows it to be tuned based on mission requirements and agent onboard resources. RRT-based methods can construct efficient paths in both evenly and focused exploration situations. Their performance in these situations can outperform A*'s, in terms of path length and time, if tree propagation nodes are empirically selected based on obstacles' current and prospective future states. Finally, the new smoothing algorithm further improves path construction in all algorithms by taking <1% of the total path planning time.

Furthermore, the inclusion of both kinematic and dynamic agent models, sensor limitations, time-invariant and time-varying obstacle positions, speed and orientation in addition to uncertainty in agent, sensor and environment will further determine the A*-based and RRT-based algorithms' applicability in static and dynamic 3D environments for different vehicular systems including UAVs and AUVs.

REFERENCES

[1] C. Zammit and E. J. van Kampen, Comparison between A* and RRT Algorithms for UAV Path Planning, in *Proc. AIAA Guidance, Navigation and Control Conf. AIAA SciTech Forum*, (Kissimmee, FL, 2018), pp. 1–23.

[2] Y. Kuwata, S. Karaman, J. Teo, E. Frazzoli, J. How and G. Fiore, Real-Time Motion

Planning With Applications to Autonomous Urban Driving, *IEEE Trans. Control Systems Tech.* **17**(5) (2009) 1105–1118.

[3] A. Nash and S. Koenig, Any-Angle Path Planning, *Artificial Intelligence (AI) Magazine*, **34**(4) (2013) 9–31.

[4] M. Likhachev, D. Ferguson, G. Gordon, A. Stentz and S. Thrun, Anytime search in dynamic graph, *Artificial Intelligence*, **172**(14) (2008) 1613—1643.

[5] P. B. Sujit and D. Ghose, Search by UAVs with Flight Time Constraints using Game Theoretical Models, in *AIAA Guidance, Navigation and Control Conf.*, (San Francisco, CA, 2005), pp. 1–11.

[6] J.-W. Lee, B. Walker and K. Cohen, Path Planning of Unmanned Aerial Vehicles in a Dynamic Environment, in *Infotech@Aerospace*, (St. Louis, Missuori, 2011), pp. 1–19.

[7] L. Blackmore, Robust Path Planning and Feedback Design Under Stochastic Uncertainty, in *AIAA Guidance, Navigation and Control Conf. and Exh.*, (Honolulu, HI, 2008), pp. 1–12.

[8] A.R. Babaei and M. Mortazavi, Fast trajectory planning based on in-flight waypoints for unmanned aerial vehicles, *Aircraft Engineering and Aerospace Technology*, **82**(2) (2010) 107–115.

[9] M. D. Zollar, R. G. Cobb and D. J. Grymin, Optimal SUAS path planning in three-dimensional constrained environments, *Unmanned systems*, **7(2)** (2019) 105–118.

[10] A. Chakrabarty and J. W. Langelaan, Energy maps for long-range path planning for small-and micro-uavs, in *AIAA Guidance, Navigation and Control Conf.*, (Chicago, IL, 2009), pp. 1–13.

[11] M. Gros, A. Schöttl and W. Fichter, Spline and OBB-based Path-Planning for Small UAVs with the Finite Receding-Horizon Incremental-Sampling Tree Algorithm, in *AIAA Guidance, Navigation and Control Conf.*, (Boston, MA, 2013), pp. 1–17.

[12] J. N. Amin, J. D. Boskovic and R. K. Mehra, A Fast and Efficient Approach to Path Planning for Unmanned Vehicles, in *AIAA Guidance, Navigation and Control Conf.*, (Keystone, CO, 2006), pp. 1–9.

[13] Y. Huang, J. Chen, H. Wang and G. Su, A method of 3D path planning for solar-powered UAV with fixed target and solar tracking, *J. Aerospace Science and Technology*, **92** (2019) 831–838.

[14] J. L. Foo, J. Knutzon, V. Kalivarapu, J. Oliver and E. Winer, Path Planning of Unmanned Aerial Vehicles using B-Splines and Particle Swarm Optimization, *J. Aerospace Computing, Information, and Communication* **6**(4) (2009) 271–290.

[15] R. Dai and J. Cochran, Path Planning and State Estimation for Unmanned Aerial Vehicles in Hostile Environments, *J. Guidance, Control, and Dynamics*, **33**(2) (2010) 595–601.

[16] Y. Dong, Y. Zhang and J. Ai, Experimental Test of Unmanned Ground Vehicle Delivering Goods Using RRT Path Planning Algorithm, *Unmanned Systems*, **5**(1) (2017) 45–57.

[17] K. P. Bollino and L. R. Lewis, Collision-Free Multi-UAV Optimal Path Planning and Cooperative Control for Tactical Applications, in *AIAA Guidance, Navigation and Control Conf.*, (Honolulu, HI, 2008), pp. 1–18.

[18] X. Sun, C. Cai and X. Shen, A New Cloud Model Based Human-Machine Cooperative Path Planning Method, *J. Intelligent & Robotic Systems*, **79** (2014) 3–19.

[19] E. L. D. Angelis, F. Giulietti, G. Pipeleers, G. Rossetti, and R. Van Parys, Optimal autonomous multirotor motion planning in an obstructed environment, *J. Aerospace Science and Technology*, **87** (2019) 379–388.

[20] Z. Zhen, Y. Chen, L. Wen and B. Han, An intelligent cooperative mission planning scheme of UAV swarm in uncertain dynamic environment, *J. Aerospace Science and Technology*, **100** (2020) 1–16.

[21] C. Hu, Z. Zhang, N. Yang, Y.-S. Shin and A. Tsourdos, Fuzzy multiobjective cooperative surveillance of multiple UAVs based on distributed predictive control for unknown ground moving target in urban environment, *J. Aerospace Science and Technology*, **84** (2019) 329–338.

[22] S. Park, H.-L. Choi, N. Roy and J. How, Learning Covariance Dynamics for Path Planning of UAV Sensors in a Large-Scale Dynamic Environment, in *AIAA Guidance, Navigation and Control Conf.*, (Chicago, IL, 2009), pp. 1–18.

[23] C. Crispin and A. Sobester, An Intelligent, Heuristic Path Planner for Multiple Agent Unmanned Air Systems, in *AIAA Information Technology at Aerospace*, (Kissemmee, FL, 2015), pp. 1–13.

[24] J. D. Boskovic, N. Knoebel, N. Moshtagh and G. L. Larson, Collaborative Mission Planning & Autonomous Control Technology (CoMPACT) System Employing Swarms of UAVs, in *AIAA Guidance, Navigation and Control Conf.*, (Chicago, IL, 2009), pp. 1–24.

[25] P. P. Wu, D. Campbell and T. Merz, Multi-Objective Four-Dimensional Vehicle Motion Planning in Large Dynamic Environments, *IEEE Trans. on Systems, Man and Cybernetics- Part B Cybernatics*, **41**(3) (2011) 621–634.

[26] L. Babel, Flight path optimization with application to in-flight replanning to changing destinations, *Aircraft Engineering and Aerospace Tech.*, **90**(8) (2018) 1192–1202.

[27] S. Luke, K. Sullivan, L. Panait and G. C. Balan, Tunably Decentralized Algorithms for Cooperative Target Observation, in *4th Inter. Joint Conf. Autonomous Agents and Multiagent Systems*, (Utrecht, The Netherlands, 2005), pp. 1–24.

[28] P. B. Sujit and D. Ghose, Two-agent cooperative search using game models with endurance-time constraints, *Engineering Optimization*, **42**(7) (2010) 617–639.

[29] C. Sabo, D. Kingston and K. Cohen, A Formulation and Heuristic Approach to Task Allocation and Routing of UAVs under Limited Communication, *Unmanned Systems*, **2**(1) (2014) 1–17.

[30] A. Ryan and J. K. Hedrick, A mode-switching path planner for UAV-assisted search and rescue, in *Proc. 44th IEEE Conf. Decision and Control*, (Seville, Spain, 2005), pp. 1471–1476.

[31] D. Gonzàlez, J. Pèrez, V. Milanès and F. Nashashibi, A Review of Motion Planning Techniques for Automated Vehicles, *IEEE Trans. on Intelligent Transportation Systems*, **17**(4) (2016) 1135-1145.

[32] S. Ghandi and E. Masehian, Review and taxonomies of assembly and disassembly path planning problems and approaches, *CAD Computer Aided Design*, **67–68** (2015) 58–86.

[33] A. Short, Z. Pan, Z. Larkin and S. van Duin, Recent Progress on Sampling Based Dynamic Motion Planning Algorithms, in *IEEE Int. Conf. Advanced Intelligent Mechatronics (AIM)*, (Alberta, Canada, 2016), pp. 1305–1311.

[34] S. V. Konakalla, A Star Algorithm, in *Indiana State University*, Technical Report, 2014.

[35] P. E. Hart, N. J. Nilsson and B. Raphael, A Formal Basis for the Heuristic Determination of Minimum Cost Paths, *IEEE Trans. on Systems Science and Cybernetics*, **4**(3) (1968) 100–107.

[36] F. H. Tseng, T. T. Liang, C. H. Lee, L. D. Chou and H. Chao, A Star Search Algorithm for Civil UAV Path Planning with 3G Communication, in *10th Int. Conf. Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP)*, (Kitakyushu, Japan, 2014), pp. 942–945.

[37] S. M. Lavalle and J. J. Kuffner, Randomized kinodynamic planning, *Int. J. Robotics Research*, **20**(3) (2001) 378–400.

[38] S. M. LaValle, Rapidly-exploring random trees: a new tool for path planning, *Iowa State University*, Technical Report, 98–11, 1998.

[39] E. W. Dijkstra, A Note on Two Problems in Connexion with Graphs, *Math. 1* (1959) 269–271.

[40] M. B. Sathyaraj, L. C. Jain, A. Finn and S. Drake, A Multiple UAVs path planning algorithms: a comparative study, *Fuzzy Optimization and Decision Making*, **7**(3) (2008) 257–267.

[41] J. Gibson, T. Schuler, L. McGuire, D. M. Lofaro and D. Sofge, Swarm and Multi-agent Time-based A* Path Planning for Lighter–Than–Air Systems, *Unmanned Systems*, **8**(3) (2020) 253–260.

[42] M. Lan, S. Lai, T. H. Lee and B. M. Chen, A Survey of Motion and Task Planning Techniques for Unmanned Multicopter Systems, *Unmanned Systems*, **9**(2) (2021) 165–198.

[43] J. Mitchell and C. Papadimitriou, The Weighted Region Problem: Finding Shortest Paths Through a Weighted Planar Subdivision, *J. Ass. Computing Machinery (ACM)*, **38**(1) (1991) 18–73.

[44] S.-L. Guo, J. Duan, Y. Zhu, X.-C. Li and T.-W. Chen, Improved Dijkstra Algorithm Based on Fibonacci Heap for Solving the Shortest Path Problem with Specified Nodes, in *Proc. Int. Conf. on Computer Science and Artificial Intelligence*, eds. W.-J. Chang, (Guilin, China, 2017), pp. 52–61.

[45] Q. Li, Z. Zeng, B. Yang and T. Zhang, Hierarchical route planning based on taxi gps-trajectories, in *17*th *Int. Conf. Geoinformatics*, (Fairfax, VA, 2009), pp. 1–5.

[46] R. Kala, and K. Warwick, Multi-level planning for semi-autonomous vehicles in traffic scenarios based on separation maximization, *J. Intelligent and Robotic Systems*, **72**(3–4) (2013) 559–590.

[47] J. Bohren, T. Foote, J. Keller, A. Kushleyev, D. Lee, A. Stewart, P. Vernaza, J. Derenick, J. Spletzer and B. Satterfield, Little ben: The ben franklin racing team' s entry in the 2007 darpa urban challenge, *J. Field Robotics*, **25**(9) (2008) 598–614.

[48] A. Bacha, C. Bauman, R. Faruque, M. Fleming, C. Terwelp, C. Rein- holtz, D. Hong, A. Wicks, T. Alberi, D. Anderson, Odin: Team victortango' s entry in the darpa urban challenge, *J. Field Robotics*, **25**(8) (2008) 467–492.

[49] A. Stentz, Optimal and efficient path planning for unknown and dynamic environments, *Carnegie Mellon Robotics Institute*, Technical Report, CMU-RI-TR-93-20, 1993.

[50] A. Stentz, The focused D* algorithm for real-time replanning, in *Proc. Int. Joint Conf. Artificial Intelligence*, (Quebec, Canada, 1995), pp. 1652–1659.

[51] S. Koenig and M. Likhachev, Incremental A*, in *Proc. Neural Info. Process. Systems*, (Vancouver, Canada, 2001), pp. 1–8.

[52] S. Koenig and M. Likhachev, D* Lite, in *Proc. AIAA Conf. on Artificial Intelligence*, (Indianapolis, IN, 2002), pp. 476–483.

[53] D. Ferguson and A. Stentz, Using interpolation to improve path planning: The field D* algorithm, *J. Field Robotics*, **23**(2) (2006) 79–101.

[54] A. Niewola and A. Podsędkowski, L* Algorithm—A Linear Computational Complexity Graph Searching Algorithm for Path Planning, *J. Intelligent & Robotic Systems*, **91** (2017) 425–444.

[55] K. Daniel, A. Nash, S. Koenig and A. Felner, Theta^*: Any-angle path planning on grids, *J. Artificial Intelligence*, **39** (2010) 533–579.

[56] A. Nash, S. Koenig and M. Likhachev, Incremental Phi*: Incremental Any-Angle Path Planning on Grids, in *Proc. 21st Int. Joint Conf. on Artificial Intelligence (IJCAI)*, (Pasadena, CA, 2009), pp. 1824–1830.

[57] K. Trovato and L. Dorst, Differential A*, *IEEE Trans. Knowledge and Data Engineering*, **14**(6) (2002) 1218—1229.

[58] A. Nash, S. Koenig and C. Tovey, Lazy Theta*: Any-Angle Path Planning and Path Length Analysis in 3D, in *Proc. 3rd Annual Symp. Combinatorial Search Intelligence (SOCS)*, (Atlanta, GA, 2010), pp. 153–154.

[59] K. Trovato, R. Marín, M. Popović, I. Maza and A. Viguria, Efficient Lazy Theta* Path Planning over a Sparse Grid to Explore Large 3D Volumes with a Multirotor UAV, *Sensors*, **19**(1) (2019) 174.

[60] P. Yap, N. Burch, R. Holte and J. Schaeffer, Block A*: Database-driven search with applications in any-angle path-planning, in *Proc. Conf. of Artificial Intelligence (AAAI)*, (San Francisco, CA, 2011), pp. 120–125.

[61] D. Harabor and A. Grastien, Online Graph Pruning for Path finding On Grid Maps, in *Proc. Conf. of Artificial Intelligence (AAAI)*, (San Francisco, CA, 2011), pp. 114–119.

[62] F. Duchoň, A. Babineca, M. Kajana, P. Beňo, M. Floreka, T. Ficoa and L. Jurišica, Path planning with modified A star algorithm for a mobile robot Františšek, *Modelling of Mechanical and Mechatronic Systems (MMaMS) Procedia Engineering* **96** (2014) 59–69.

[63] K. Gochev, A. Safonova and M. Likhachev, Anytime Tree-restoring A* Graph Search, *Proc. 7th Annual Symposium on Combinatorial Search SoCS*, (Prague, Czech Republic, 2014), pp. 80–88.

[64] X. Sun, W. Yeoh and S. Koenig, Dynamic fringe-saving A*, in *Proc. 8th Int. Conf. on Autonomous Agents and Multiagent systems (AAMAS)*, (Budapest, Hungary, 2009), pp. 891–898.

[65] X. Sun, S. Koenig and W. Yeoh, Generalized adaptive A*, in *Proc. 7th Int. Conf. on Autonomous Agents and Multiagent systems (AAMAS)*, (Estoril, Portugal, 2008), pp. 469–476.

[66] C. Hernández, X. Sun, S. Koenig and P. Meseguer, Tree adaptive A*, in *Proc. 10th Int. Conf. on Autonomous Agents and Multiagent systems (AAMAS)*, (Taipei, Taiwan, 2011), pp. 123–130.

[67] S. Ainea and M. Likhachevb, Truncated incremental search, *Artificial Intelligence*, **234** (2016) 49–77.

[68] L. E. Kavraki, P. Švestka, J. C. Latombe and M. H. Overmars, Probabilistic roadmaps for path planning in high-dimensional configuration spaces, *IEEE Trans. Robotics and Automation*, **12**(14) (1996) 566–580.

[69] S. Karaman and E. Frazzoli, Sampling-Based Algorithms for Optimal Motion Planning, *Int. J. Robotics Research*, **30**(7) (2011) 846–894.

[70] R. Bohlin and L. E. Kavraki, Path planning using lazy PRM, *Proc. IEEE Int. Conf. on Robotics and Automation*, **1** (2000) 521–528.

[71] K. P. Leven and S. Hutchinson, Toward real-time path planning in changing environments, *Algorithmic and Computational Robotics: New Directions: 4th Int. Workshop on the Algorithmic Foundations of Robotics*, (Hanover, NH, 2000), pp. 363–376.

[72] K. Klasing, D. Wollherr and M. Buss, Cell-based probabilistic roadmaps (cprm) for efficient path planning in large environments, in *Proc. Int. Conf. on Advanced Robotics*, (Daegu, South Korea, 2007).

[73] M. Pomarlan, and I. A. Sucan, Motion planning for manipulators in dynamically changing environments using real-time mapping of free workspace, in *IEEE 14th Int. Symp. Computational Intelligence and Informatics (CINTI)*, (Budapest, Hungary, 2013), pp. 483—487.

[74] A. Gayle, M. Sud, C. Lin and D. Manocha, Reactive deformation roadmaps: motion planning of multiple robots in dynamic environments, in *Int. Conf. on Intelligent Robots and Systems (IROS)*, (San Diego, CA, 2007), pp. 3777–3783.

[75] K. Belghith, F. Kabanza, L. Hartman and R. Nkambou, Any-time dynamic path-planning with flexible probabilistic roadmaps, in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, (Orlando, FL, 2013), pp. 2372–2377.

[76] B. Y. Yang and O. Brock, Elastic roadmaps motion generation for autonomous mobile manipulation, *Autonomous Robots*, **28**(1) (2010) 113–130.

[77] A. Dias, T. Fernandes, J. Almeida, A. Martins and E. Silva, 3D path planning methods for unmanned aerial vehicles in search and rescue scenarios, in *Proc. 20th Int. Conf. on CLAWAR*, (Porto, Portugal, 2017), pp. 213–220.

[78] S. M. LaValle and J. J. Kuffner, Randomized kinodynamic planning, in *Pro. IEEE Int. Conf. on Robotics and Automation*, (Detroit, MI, 1999), pp. 473–479.

[79] D. Devaurs, T. Siméon, and J. Cortés, Optimal Path Planning in Complex Cost Spaces With Sampling-Based Algorithms, *IEEE Trans. on Automation Science and Engineering*, Institute of Electrical and Electronics Engineers, **13**(2) (2015) 415–424.

[80] R. Geraerts and M. Overmars, Creating high-quality paths for motion planning, *Int. J. Robotics Research*, **26**(8) (2007) 845–863.

[81] E. G. Tsardoulias, A. Iliakopoulou, A. Kargakos and L. Petrou, A Review of Global Path Planning Methods for Occupancy Grid Maps Regardless of Obstacle Density, *J. Intelligent and Robotic Systems*, **84** (2016) 829—858.

[82] S. Karaman and E. Frazzoli, Optimal kinodynamic motion planning using incremental sampling-based methods, in *Proc. 49th IEEE Conf. Decision and Control*, (Atlanta, GA, 2010), pp. 7681–7687.

[83] D. Devaurs, T. Siméon and J. Cortés, Enhancing the transition-based RRT to deal with complex cost spaces, in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, (Orlando, FL, 2013), pp. 4561–4568.

[84] J. Nasir, F. Islam, U. Malik, Y. Ayaz, O. Hasan, M. Khan and M. S. Muhammad, DRRT*-SMART: a rapid convergence implementation of RRT*, *J. Advanced Robotic Systems*, **10**(7) (2013) 1–12.

[85] Z. Zhang, R. Du and R. V. Cowlagi, Randomized sampling-based trajectory optimization for UAVs to satisfy linear temporal logic specifications, *J. Aerospace Science and Technology*, **96** (2020) 1–9.

[86] J. J. Kuffner and S. M. LaValle, Rrt-connect: An efficient approach to single-query path planning, in *Proc. IEEE Int. Conf. on Robotics and Automation*, (San Francisco, CA 2000), pp. 521–528.

[87] D. Zhang, Y. Xu and X. Yao, An Improved Path Planning Algorithm for Unmanned Aerial Vehicle Based on RRT-Connect, in *Proc. 37th Chinese Control Conf. (CCC)*, (Wuhan, China, 2018), pp. 4854–4858.

[88] P. Pharpatara, B. Herisse and Y. Bestaoui, 3D trajectory planning of aerial vehicles using RRT*, *IEEE Trans. Control Syst. Technol.*, **25**(3) (2017) 1116—1123.

[89] J. Bruce and M. Veloso, Real-time Randomized Path Planning for Robot Navigation, in *IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, (Lausanne, Switzerland, 2002), pp. 2383–2388.

[90] S. R. Martin, S. E. Wright and J. W. Sheppard, Offline and Online Evolutionary Bi-Directional RRT Algorithms for Efficient Re-Planning in Dynamic Environments, in *Proc. IEEE Int. Conf. on Automation Science and Engineering*, (Scottsdale, AZ, 2007), pp. 1131–1136.

[91] E. Frazzoli, M. A. Dahleh and E. Feron, Real-time motion planning for agile autonomous vehicles, *J. Guidance, Control, and Dynamics*, **25**(1) (2002) 116–129.

[92] J. Guitton, J. L. Farges and R. Chatila, Cell-RRT: Decomposing the environment for better plan, *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, (St. Louis, MO), pp. 5776–5781.

[93] D. Braid, A. Broggi and G. Schmiedel, The TerraMax autonomous vehicle, *J. Field Robotics*, **23**(5) (2016) 693–708.

[94] R.-H. Ryu, D. Ogay, D. Bulavintsev, H. Kim and J.-S. Park, Development and Experiences of an Autonomous Vehicle for High-Speed Navigation and Obstacle Avoidance, *Frontiers of Intelligent Autonomous Systems*, **466** (2013) 105–116.

[95] W. Shang, J. Liu, R. Ning and M. Liu, Computational path planner for product assembly in complex environments, *Chinese J. Mechanical Engineering*, **26**(2) (2013) 282–292.

[96] D. Ferguson and A. Stentz, Anytime RRTs, in *IEEE Int. Conf. Intelligent Robots and Systems (IROS)*, (Beijing, China, 2006), pp. 5369–5375.

[97] Q. Zhu, Y. Wu, G. Wu and X. Wang, An improved Anytime RRTs algorithm, in *Int. Conf. Artificial Intelligence and Computational Intelligence*, (Shanghai, China, 2009), pp. 268–272.

[98] Y. Abbasi-Yadkori, J. Modayil and C. Szepesvári, Extending rapidly-exploring random trees for asymptotically optimal Anytime motion planning, in *IEEE Int. Conf. on Intelligent Robots and Systems (IROS)*, (Shanghai, China, 2010), pp. 127–132.

[99] R. Alterovitz, S. Patil and A. Derbakova, Rapidly-exploring roadmaps: weighing exploration vs. refinement in optimal motion planning, in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, (Shanghai, China, 2011), pp. 3706–3712.

[100] R. Luna, I. Şucan, M. Moll and L. Kavraki, Anytime solution optimization for sampling-based motion planning, in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, (Orlando, FL, 2013), pp. 5068–5074.

[101] D. Ferguson, N. Karla and A. Stentz, Replanning with RRT, in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, (Orlando, FL, 2006), pp. 1243–1248.

[102] D. Ferguson and A. Stentz, Anytime, dynamic planning in high-dimensional search spaces, in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, (Pasadena, CA,. 2007), pp. 1310–1315.

[103] M. Zucker, J. Kuffner and M. Branicky, Multipartite rrts for rapid replanning in dynamic environments, in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, (Pasadena, CA, 2007), pp. 1603–1609.

[104] M. Otte and E. Frazzoli, RRT^X: Asymptotically Optimal Single-Query Sampling-Based Motion Planning with Quick Replanning, *Int. J. Robotics Research*,**29**(7) (2015) 797–822.

[105] R. Benenson, S. Petti, T. Fraichard and M. Parent, Integrating perception and planning for autonomous navigation of urban vehicles, in *IEEE Int. Conf. on Intelligent Robots and Systems (IROS)*, (Beijing, China, 2006), pp. 98–104.

[106] K. E. Bekris and L. E. Kavraki, Greedy but safe replanning under kinodynamic constraints, in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, (Pasadena, CA, 2007), pp. 704–710.

[107] E. Taheri, M. H. Ferdowsi and M. Danesh, Fuzzy Greedy RRT path planning algorithm in a complex configuration space, *Int. J. Control, Automation and Systems*, **16**(6) (2018) 3026–3035.

[108] T. Y. Li and Y. C. Shie, An incremental learning approach to motion planning with roadmap management, in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, (Washington, DC, 2002), pp. 3411–3416.

[109] R. Gayle, K. R. Klingler and P. G. Xavier, Lazy reconfiguration forest (lrf)-an approach for motion planning with multiple tasks in dynamic environments, in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, (Pasadena, CA, 2007), pp. 1316–1323.

[110] Y. Tsai, C. Lee, C. Lin and C. Huang, Development of Flight Path Planning for Multirotor Aerial Vehicles, *Aerospace 2015*, **2** (2015) 171–188.

[111] D. Roa and S. Williams, Large-scale path planning for Underwater Gliders in ocean currents, in *Australasian Conf. on Robotics and Automation (ACRA)*, (Sydney, Australia, 2009), pp. 1–9.

[112] R. Simmons and C. Urmson, Approaches for heuristically biasing RRT growth, in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, (Las Vegas, NV, 2003), pp. 1178–1183.

[113] C. Zammit and E. J. van Kampen, Advancements for A* and RRT in 3D path planning of UAVs, in *AIAA Guidance, Navigation and Control Conf.*, (San Diego, CA, 2019), pp. 1–17.

[114] S. Lee, A. Cho and C. Kee, Integrated waypoint path generation and following of an unmanned aerial vehicle, *Aircraft Engineering and Aerospace Technology*, **28**(5) (2010) 296–304.

[115] M. Clifton, G. Paul, N. Kwok, D. Liu and D. Wang, Evaluating Performance of Multiple RRTs, in *IEEE Conf. on Mechatronic and Embedded Systems and Application*, (Bejing, China, 2009), pp. 564–569.

[116] G. Paul, Multiple Rapidly-exploring Random Tree (RRT), *MATHWORKS*, [Online Database], https://www.mathworks.com/matlabcentral/fileexchange/21443-multiple-rapidly-exploring-random-tree-rrt-?requestedDomain=www.mathworks.com [retreived 30 October, 2016].

3

Comparison of A* and RRT in Real-time 3D Path Planning of UAVs

This Chapter builds upon the analysis carried out in the previous chapter to answer Research Question 2, derived to tackle Challenge 1, that queries whether path planning algorithms can be applied in real-time. From the analysis in Chapter 2, the two most promising path planning rationales are selected in view of real-time path planning. Furthermore, this chapter presents a detailed literature review of the approaches considered in real-time path planning whilst highlighting the need of real-time consideration. This chapter proposes, develops, tests and implements real-time path planning with static obstacles analysing the effects of different parameters on path planning performance.

The contents of this chapter have been published as:
Zammit, C. and van Kampen, E., "Comparison of A* and RRT in real-time 3D path planning of UAVs", *Proceedings of AIAA Guidance, Navigation and Control*, Orlando, FL, 6-10 Jan., 2020, AIAA-2020-0861.

A summary of this chapter will be submitted as:
Zammit, C. and van Kampen, E., "A Comparative Analysis of the A* and RRT Algorithms in Real-time 3D UAV path planning", *Journal of Aerospace Science and Technology*

Unmanned Aerial Vehicles (UAVs) are being integrated into a wide range of military, industrial and commercial applications. Some of these applications require faultless autonomous systems to coordinate, guide, navigate and control different UAVs of different sizes, designed for different purposes with different capabilities. In this regard, path planning algorithms are developed to furnish UAVs with collision-free paths. The paramount path planning algorithms are the A* and the RRT algorithms, a graph-based and a sampling-based algorithm respectively. These algorithms shall ideally operate in real-time to supply the UAV navigation system with valid, obstacle-free paths in view of changes in the environment or other external or user-defined restrictions. Owing to this need, this paper developed a real-time algorithm to assess the performance of the A* and RRT algorithms with an associated smoothing algorithm using 3D obstacle environments of different complexity using time and sensory range constraints, emulating navigational restrictions experienced by UAVs operating in real-time. Results show that the A* outperforms the RRT algorithm in both path length and computational time for all scenarios considered, with difference increasing with scenario complexity. Furthermore, this work investigates the relation between path planning performance and user-defined, system-defined and internal parameters. This analysis can help determine the best configuration for UAV designers, in view of application requirements and constraints. In fact, results show that both path planning algorithms can achieve a success rate close to 100%, if parameters, such as speed, computational power and sensory range are attentively selected based on the analysis of the consequences of the selected parameter values on path planning performance.

3.1. INTRODUCTION

Unmanned Aerial Vehicle (UAVs) are potential candidates for a wide range of applications in both civil and military setups. In these scenarios, different UAVs require varying levels of autonomy, reliability and efficiency based on the assigned task. To reach a goal or set of goals, UAVs are equipped with sensory, processing and actuator systems with different accuracy, redundancy, preciseness, latency, reliability and computational power.

Real-time, efficient and reliable paths are fundamental to ensure that the UAV autonomously reaches the goal safely and in due time. Path planning is the process of automatically generating feasible and optimal 2D [1, 2] or 3D [3, 4] paths. Different UAVs have different levels of path planning autonomy varying from solely human-controlled [5] and shared human-controlled [6] systems to fully autonomous goal-oriented systems [7, 8] designed for different applications. Some of these applications include: agricultural remote sensing [9], ground vehicle tracking [10], traffic surveillance [11], package delivery [12, 13], medicinal delivery in remote areas [14] and ambulance drone [15]. Moreover, UAVs can be utilised in situations where the mission is too difficult or too dangerous for human pilots such as monitor critical structures in natural disasters, search and rescue and monitor weather inside a storm [16].

The path planning algorithms utilise sensory, processing and actuator systems to generate paths in view of different kinematic, dynamic [17, 18] and environmental [19, 20] time invariant and time-varying constraints. Once a path is generated through the

path planning algorithm, a path following algorithm will generate the control parameters for the UAV to follow the generated path. Although these control parameters can be generated offline, real-time path planning allows the path following algorithm to amend navigational instructions in view of unpredictable and/or uncertain model and environmental changes. Currently, UAV systems incorporate advanced control algorithms that allow UAVs to manoeuvre in cluttered environments if the control algorithm is provided with accurate, timely and efficient real-time state information. This goal-driven, autonomous UAV can be realised via a real-time path planning algorithm governed by state-of-the-art path following and control systems.

Even in indoor applications, UAVs are expected to operate in a time-varying environment. A UAV may encounter obstacles moving in an unpredictable way, such as persons, door and window openings and other UAVs as well as internal unpredictable events such as fuel limitations, loss of movement in 1 or more Degrees of Freedom (DoF) and loss of partial or complete sensory information. In these real-life situations the control, path following and planning algorithms must operate safely and efficiently irrespective of any shortcomings in the sensory and actuator systems [21]. To cater for these situations real-time control and path planning are a must.

The contribution of this paper is to compare the A* with the ripple reduction enhancement and the RRT algorithm in view of real-time 3D UAV path planning performance as a function of UAV parameters. An extensive literature review of the state-of-the-art path planning algorithms, presented in [22], concluded that A* and RRT are the most utilised graph-based and sampling-based path planning methods, respectively. Further analysis in [22, 23] showed that these algorithms and their variants are key candidates for 3D UAV path planning. Therefore, these two algorithms will be assessed for performance mainly in terms of computational time, success rate and path length. The outcomes of this paper will help determine the ideal path planning configuration and UAV selection, in view of mission requirements and constraints.

The paper will be organised as follows. Section 3.2 will present the state-of-the-art in real-time path planning. Section 3.3 provides a brief resume of the considered path planning algorithms (A* and RRT) and the smoothing algorithm defined in depth in our previous work [22, 23]. Section 3.4 will formulate the real-time path problem and consequently define the theoretical aspect of the developed algorithm designed to assess the appropriateness of the considered path planning algorithms in real-time applications. Section 3.5 will define the experimental scenario with the associated arbitrary-defined parameters. The following section (Section 3.6), will present, analyse and assess the results in view of 3D UAV path planning in real-time. The paper will conclude by Section 3.7 which based on the benefits and shortcomings will rate the appropriateness of the implemented algorithm for 3D UAV path planning in real-time.

3.2. REAL-TIME PATH PLANNING LITERATURE REVIEW

3.2.1. INTRODUCTION

Real-time path planning is considered as a desirable feature [24], a requirement [25] and paramount [26] for real-time autonomous manoeuvring of vehicles let alone UAVs in real, dynamic environments. Real-time path planners are requested to generate paths

in the presence of other cooperative or non-cooperative UAVs, unexpected UAV damage, altering model constraints and definitions, and in view of uncertainties in the environment in which they operate [3, 24, 25]. Furthermore, the path planner must make optimal use of available resources such as computational power and fuel [27, 28].

A path planner is considered to be real-time if the time required to generate a path is smaller than the time to traverse the path [3, 29, 30]. For Short *et. al.* [31] a realistic path planner must react in synchronisation with information update from the sensory systems. Furthermore, such path planner must generate a path even with restricted global information [21].

As computational time is the bottleneck of real-time path planning, Karaman and Frazzoli [32], remarked that computational time per iterate shall be bounded. In certain iterates, longer computational times may be required due to fewer path solutions. If not tackled these will increase the overall computational time making the path planning algorithm unsuitable for real-time path planning. Sub-bounds will attenuate this situation even if a solution is not found in that iterate, but a minor movement on the previously generated path may yield a path possibly in less time.

Real-time path planning of UAVs requires high fidelity modelling of the environment. Simultaneous Localisation and Mapping (SLAM) algorithms can be utilised to generate maps for realistic environments to facilitate real-time path planning. Such algorithms can be utilised to mitigate absence or partial absence of GPS information, discontinuity of sensor information and noise [33].

This literature review will be segmented into three main path planning categories: Optimisation algorithms, graph-based methods and sampling-based methods. Their application in 3D real-time path planning will be analysed and assessed.

3.2.2. OPTIMISATION ALGORITHMS

In both real-time and offline applications there is no guarantee of convergence to the goal let alone in a predetermined time either because no path exists, the environment is not known enough or the path planning algorithm intrinsically cannot generate the path in the particular situation [27]. In such situations, a trade-off between computational time and optimality was considered by Frazzoli [27] for a finite-state automation method to compute trajectories for multiple UAVs in safety-critical, high performance vehicles with complex dynamics.

Disturbances from external torques initiating through for example wind shears, sensor inaccuracies and parameter uncertainties will further increase the path planning and following demands of complex systems such as UAVs [26]. Real-time applications of optimal control theory showed that feedback control will enhance performance in such complex nonlinear systems [34, 35]. Furthermore, through sensor fusion, the accuracy and responsiveness of the sensing system will be improved [26]. Gong *et. al.* [35] and Bollino *et. al.* [26] utilised a pseudospectral method for optimal control and path planning of complex systems, respectively. Simulation results of this method show that although pseudospectral methods are mainly utilised for path following providing optimal control instructions, they can be utilised for autonomous path planning [26].

A single optimisation method cannot simultaneously guarantee target tracking and obstacle avoidance [30], especially in 3D UAV path planning environments. Yao *et. al.*

[30] proposed a combination of improved Lyapunov Guidance Vector Field (LGVF), the Interfered Fluid Dynamical System (IFDS) and the strategy of varying receding-horizon optimisation based on Model Predictive Control (MPC) to generate 3D paths for a UAV in dynamic environments under constraints. This hybrid algorithm was proposed since although MPC were successfully applied to generate suboptimal paths in real-time [36, 37] and to generate paths in 2D scenarios, the computational efficiency and smoothness will deteriorate in 3D environments [30].

Furthermore, Roberge *et. al.* [38], compared Particle Swarm Optimisation (PSO) and Genetic algorithms (GA) in real-time for the automatic path planning of fixed-wing UAVs in complex 3D environments. Both algorithms generated feasible and quasi-optimal trajectories in view of vehicle dynamics. Execution time was reduced through single-program, multiple-data on an 8 core processor. Although both algorithms generated a path within 10s (a preset path planning time), GA produced statistically better trajectories with respect to PSO in terms of distance travelled, average altitude and danger zones avoidance. Roberge *et. al.* [38] remarked that both algorithms can generate a path for cruising the fastest fixed wing UAV available at the time of writing.

In dynamic real-world populated environments and considering the disturbances and eventualities mentioned above, 10s is too long to react and generate a new non-colliding path. Solving complex optimisation problems arising from these scenarios will result in high computational load [39].

3.2.3. GRAPH-BASED METHODS

Real-time path planning of complex dynamic environments still remains a challenge even for 2D environment, let alone 3D [40]. Kuwata *et. al.* [40] remarked that it is very difficult to model nonlinear dynamics especially for demanding manoeuvres when using generic graph-based path planning methods. This is because generic graph-based methods such as the original A* assume the environment to be static [41]. Similarly, Singh *et. al.* [33], remarked that although graph-based methods are effective in configured environments, it is unsuitable for real-time path planning in large or complex environments. Local approaches of such graph-based methods can stall in local minima [33].

Although as remarked by Kuwata *et. al.* [40], graph-based methods are not optimal for real-time path planning, the differential A* [42], based on the A* algorithm (a graph-based method), was developed. More efficient results in the majority of cases were achieved when compared to A*, proposing the algorithm as a candidate for real-time dynamic, re-planning [42].

Fernandes *et. al.* [41], extended the A* algorithm in cell decomposition, considering both position and orientation in the computation of the path. This approach made it possible to reduce the computational time while maintaining the same configuration space [41].

A parallel non-deterministic adaptation of the Dijkstra algorithm, a pioneer graph-based method, was applied to generate an energy efficient, global re-planner for real-time UAV rescue operations in dynamic environments with an area of $200m^2$ utilising a map discretization of $1m^2$ [43]. Results show tens of multiple times improvement over the sequential Dijkstra algorithm with an average path cost error of less than 1.2% [43].

3.2.4. SAMPLING-BASED METHODS

Sampling-based methods are applicable to general dynamical models. Their incremental nature makes them inherent for use in real-time applications whilst guaranteeing a solution. Furthermore, sampling-based methods do not require enumeration of constraints allowing trajectory-wise checking of complex constraints [44, 45].

Advancements in sampling-based methods have proposed these algorithms for real-time path planning in dynamic and unknown environments [31]. Re-planning is essential in these situations as the environment is only partially known at one point in time, revealing more detail in the direction of the goal with every vehicle movement. This can also create situations, in which a previous non-colliding path may lead to a collision with more details in the environment. Therefore, the planner must react in real-time to mitigate these situations whilst the UAV is moving. According to Kunz *et. al.* [46], this reaction time shall not exceed 200ms to mimic human reaction.

Since according to Kuwata *et. al.* [45] the standard RRT is not able to generate safe and feasible paths in the presence of uncertainty in real-time for 2D applications, the latter proposed a closed-loop prediction in the framework of RRT, with a low-level Proportional-Integral speed controller and a pure pursuit steering controller to manoeuvre the vehicle based on real-time path planning information. Real-time execution requires reusing information from previous iterations [47, 48]. Otherwise only a sparse tree will be created when compared to the reuse approach in which computational resources are used to add new improved branches to the existing tree [45]. A non-colliding path is retained as long as possible to firstly limit *"no path"* situations and secondly to grant the necessary time for the path planning algorithm to generate efficient trajectories [40]. This closed loop RRT technique shows that real-time path planning is the bottleneck even in 2D environments.

3.2.5. CONCLUSION

This review first highlighted that 3D path planning algorithms shall successfully and efficiently operate in real-time with restricted computational power, lack of onboard resources, sensor low responsiveness and inaccuracy and environmental uncertainties. Optimisation algorithms are prospective candidates for 3D UAV path planning in real-time. Although results are promising, this approach requires knowledge of UAV dynamics and non-linearities. Graph-based methods are intrinsically designed for static environments although amendments to the basic algorithms can extend their application to real-time situations due to their relative computational simplicity. Sampling-based methods and their adaptations are also worth considering especially if re-usability of previous non-colliding paths or branches are retained. Furthermore, hybrid approaches combining different algorithms with different strengths can be utilised to mitigate inherent shortcomings of one of the three discussed approaches.

3.3. THE A*, RRT AND SMOOTHING ALGORITHMS

Graph-based methods divide the working space into an occupancy grid with obstacles defined as inaccessible grid points [44, 49]. Such methods do not offer a guarantee of solution [50]. Oppositely, sampling-based methods create a path by connecting un-

evenly selected points from the configuration space [31, 44]. The A* and RRT are the most utilised algorithms for the graph-based and sampling-based methods, respectively [22]. Although a number of A* and RRT variants were developed to mitigate with inherent shortcomings of both algorithms, the standard A* and RRT algorithms are selected in view of their maturity and relatively lower complexity and computational demand which are essential for real-time implementation. The A* and RRT path planning algorithms construct path segments which consist of a set of discrete points stored in order of traversal that when interconnected generates a path from current position to an intermediate or final goal point. To optimise the path a smoothing algorithm was developed and applied to both algorithms, in our previous work [22, 23].

3.3.1. THE A* ALGORITHM

The standard A* algorithm constructs an optimal path based on an evaluation function $f(n)$ that calculates the actual cost of an optimal path constrained to pass through n, from a point x_{init} to the goal node of n, $x_{goal} \in \mathbb{R}^M$ [51, 52]. n is any node and x_{init} is the starting node in the M-Dimensional available space such that n, $x_{init} \in \mathbb{R}^M$. This evaluation function $f(n)$ is the summation of the actual cost from x_{init} to a node, n ($g(n)$), and the actual cost from n to the goal point of n, ($h(n)$), where f, g, h: $\in \mathbb{R}^M \rightarrow \mathbb{R}$:

$$f(n) = g(n) + h(n) \tag{3.1}$$

The A* algorithm computes these evaluation functions ($f(n)$) to all possible obstacle-free nodes and selects the node n with the smallest cost.

3.3.2. THE RRT ALGORITHM

The RRT algorithm grows trees of feasible trajectories by randomly planting a number of seeds. Seeds are only considered if they lie on an obstacle free point. A point a predefined distance from the nearest seed is selected if the direct path to the latter does not collide with an obstacle. Ultimately a tree that interconnects the start and goal points will define a feasible path [53–55]. Paths generated by RRT are not optimal [56, 57] as opposed to A* which can generate more optimal paths [22].

3.3.3. THE SMOOTHING ALGORITHM

The smoothing algorithm randomly selects two path points and consequently defines two points on the lines connecting these path points with their respective next path point. If an interconnection is possible without collision with obstacles then intermediate points between these two path points are eliminated.

Reference is made to Chapter 2 and our previous work [22] for a more in depth understanding of the working principle for the A*, RRT and smoothing algorithms. In addition, a detailed literature review of the current state-of-the-art in graph-based and sampling-based methods is available.

3.4. THE REAL-TIME ALGORITHM

Both path planning algorithms were successfully implemented in static offline applications. Furthermore, the RRT without step-size constraints and the Multiple Rapidly-

Exploring Random Tree (MRRT) algorithms were also considered in our previous works [22, 23]. The characteristics, strengths and weaknesses of these algorithms were identified, assessed and analysed through the use of different experimental scenarios in view of 3D UAV path planning [22, 23].

As concluded in Section 3.2, real-time path planning is a must for UAV's to autonomously reach a goal, especially in the presence of static and dynamic obstacles [24–26]. Therefore, a testing framework is set up to assess and possibly improve the performance of the most utilised graph-based and sampling-based method.

3.4.1. PROBLEM STATEMENT

Consider a cubic airspace $W \subseteq \mathbb{R}^3$ with a number of obstacles ($O_o \subseteq \mathbb{R}^2, o = 1,2,...,n$). Obstacles size, position and orientation are assumed to remain constant through the duration of the mission. The UAV speed is assumed constant. The UAV sensing system is assumed to have a limited range and a 360^o field-of-view (FOV). In order to reach the final goal node, the real-time path planning algorithm must define an intermediate obstacle-free goal point (d_{int_goal}) within the sensing range, since outside this spherical zone, the environment is unknown.

Figure 3.1 graphically illustrates the problem a real-time path planning algorithm with limited range is expected to solve. The UAV sensory system has a limited range (green-dotted circle) into which prospective new intermediate goal points can be selected. It is assumed that obstacles in this area are known (Olive green dots). In this illustration it is assumed that the UAV is equipped with a 360^o FOV sensory system but the principle can be applied to smaller FOV sensory systems provided that the final goal position is known.

In real-life path planning, the UAV path generation system must generate and/or update the existing path to goal ideally every few milliseconds, irrespective of changes in the UAV position and obstacles. Such approach demands high update rate. Such rate is beyond the computational power available onboard state-of-the-art UAVs. Therefore, the real-time path planning algorithm only updates at predetermined time intervals derived from the nominal speed of the UAV. Although it is assumed that the UAV actuator system will move the UAV a predetermined distance defined by the preset time step in a certain direction set by the generated path, if variations exists due to internal (speed controller) or external factors (weather), the real-time path planning system can make use of updated positional information when updating the path in the subsequent iterate.

As explained in Section 3.2, the real-time path planning algorithm must construct a non-colliding path segment within the time needed to traverse the same path segment. A maximum time ($t_{iterate\_max}$) to generate a path to a goal, within the sensing range ($goal\_int$), depends on UAV speed, computational power and the distance to be travelled by the UAV in one planning iteration. For path planning to be successful, the UAV must reach the final goal position without never exceeding $t_{iterate\_max}$ in all planning iterations and never colliding with obstacles. The path planning performance of the algorithms and the effects of different parameters on path planning performance is assessed in terms of path length from start to goal, planning time and success rates.

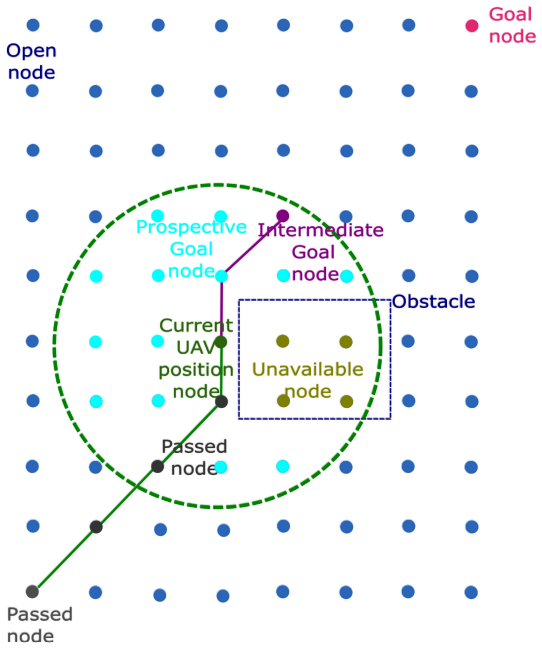


Figure 3.1: Real-time path planning with finite limited look-ahead distance

3.4.2. PARAMETER DEFINITION AND INITIATION

Firstly, the real-time algorithm initiates by defining the start ($start$) and goal ($goal$) positions and the considered resolution (res). In RRT, since the randomly generated point can reside anywhere within an environmental space, the equivalent of resolution is the step size which denotes the distance that the current path point can move in the direction of the generated path point. This will be denoted by d_{step_RRT}. For clarity res will be considered in the definition of other parameters.

In real-time the UAV path planning system has a finite time to generate a feasible path to the final goal as otherwise the UAV must either stop intermittently in mid-air (if a quadrotor is considered) or the path planning algorithm becomes unfeasible. As cited in Section 3.2, a real-time path planner must generate a path segment in less than the time to traverse it [3, 29, 30]. For a real-time path planner stopping in mid-air is not an option. Therefore the maximum time between iterates needs to be defined based on the following two assumptions. The UAV moves at a constant nominal speed of v_{UAV} in a cubic environmental space of length d_{env_space} in all 3D dimensions. Based on this work space the time to generate a path between iterates $t_{iterate\_max}$ is:

$$t_{iterate\_max}(s) = \frac{60 \times 60 \times d_{env\_space}}{(res-1) \times v_{UAV}} \tag{3.2}$$

This equation is derived from $time = \frac{distance}{speed}$. The distance moved in 1 iterate equates to the step size $\frac{1}{(res-1)}$ multiplied by the total size of one axial dimension of

the environmental space d_{env_space}. Speed is measured in [-]/hr therefore it must be changed to [-]/s hence the inclusion of the 60×60 term.

The time to generate a path between iterates $t_{iterate\_max}$ is directly related to the available computational power. Analysis of this parameter can determine whether a specific computational power is enough for a specific situation and/or conversely what computational power is required to ensure that a path segment can be constructed in the allocated time for the same situation.

Another constant parameter closely related to $t_{iterate\_max}$ is the distance covered by the UAV in every iterate (d_{s_step}). This value is a function of the UAV speed and computational power assuming no external factors such as weather are affecting the UAV. Based on the above rationale the distance covered in $t_{iterate\_max}$ shall be $\leq d_{s\_step}$. This modular unit value is arbitrary chosen although it can be varied based on the fidelity and confidence of the UAV sensory and actuator systems and environmental model.

Besides the maximum time to generate a path segment between iterates $t_{iterate\_max}$, the maximum time allocated to reach the goal point from the start position $t_{path\_gen\_max}$ is set based on the application's requirements. This upper time limit is included as situations can arise in which the UAV will venture around obstacles without actually getting closer to the goal. Moreover, UAV fuel autonomy is finite. This value shall denote double the time required to traverse the diagonal distance between two extreme points of the environmental space.

Moreover, d_{int_goal} denotes the distance between the current UAV position and the prospective intermediate goal point. This value is a function of the range of the sensory system which shall ensure that in d_{int_goal} all obstacles are know with certainty. In certain instances the resultant intermediate goal position may reside on an obstacle. In such situations, a new intermediate goal point must be defined. As the maximum look-ahead distance is defined by d_{int_goal}, the new intermediate goal point must reside nearer to the current UAV position. Therefore a distance reduction factor, $d_{factor} < 1$ (set at 0.9) is defined. Finally, lim defines the 3D boundaries of the working environment which may vary due to the shifting introduced by the A* ripple reduction algorithm Section 2.5.3.

After these constants are defined the A* ripple reduction algorithm is applied in case the A* algorithm is under review. An in depth definition of this algorithm is available in our previous work [23]. The new 3D limits introduced by this algorithm are assigned to lim. The current UAV position s_{cur} is set to $start$. In the case of the A* algorithm, the $start$ location will vary by a maximum of half the distance between grid positions, in all 3 dimensions, due to the shifting introduced by the A* ripple reduction algorithm. Furthermore, the path possibility flag ($flag_{path}$), signals whether a path could be created in future iterates (=1) or not as either a path has been created or no possibility of a path exist (=0). Furthermore, the total path computational time $time$ is set to 0s.

The real-time application of path planning algorithms initiates with path planning to an intermediate goal point, followed by the smoothing of the planned path. Then the UAV is moved to a new position on the smoothed path. This new UAV position will be fed into the path planning algorithm to plan a new path. Figure 3.2 graphically illustrates this process. This process continues provided the following conditions are satisfied:

1. Distance between the current UAV position and the final goal point (d_{int_s-to-g})

is larger than the distance between three consecutive grid points (two step sizes). For the A* algorithm to construct a path at least the start and goal points must be separated by one node as otherwise a trivial path having only the start and goal node results. This requirement is therefore included to eliminate this instance, converging to a solution faster. This requirement is considered for A* but to offer a fair comparison it is also included for RRT; and

2. Computational time since the start of the path generation process ($time$) has not exceeded $t_{path\_gen\_max}$; and
3. Allotted time to generate an intermediate path between the current UAV position and the intermediate goal point $t_{iterate\_max}$ has not been exceeded; and
4. A path is possible $flag_{path} == 1$.

In case these conditions are not satisfied, either the goal has been or cannot be reached. In either case, the iterate is stopped.

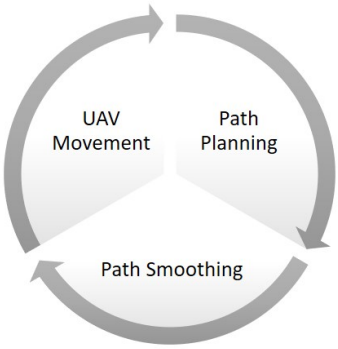


Figure 3.2: High level illustration of the real-time application of the path planning algorithms

3.4.3. THE MOVE FUNCTION

If the current UAV position has moved beyond the start position, the UAV shall be moved a predetermined distance in the direction of the previously smoothed path. The move function defined in Algorithm 3 considers all the situations in defining feasible next iterates' UAV position. Table 3.1 defines the parameters considered in the formulation of the move function algorithm. Besides the constants defined earlier, environmental parameters namely the size of the environmental space and the size and position of obstacles in the environmental space are imported into the Move function. In A*, as opposed to RRT, the size and position of obstacles may vary slightly due to the discretisation of the environmental space.

The move function initiates by checking that the distance to the intermediate goal point from the current UAV position is larger than the distance the UAV is expected to move per iteration as otherwise no path is possible or the UAV has reached the goal point. The algorithm adds up the distances of consecutive nodes on the previously smoothed

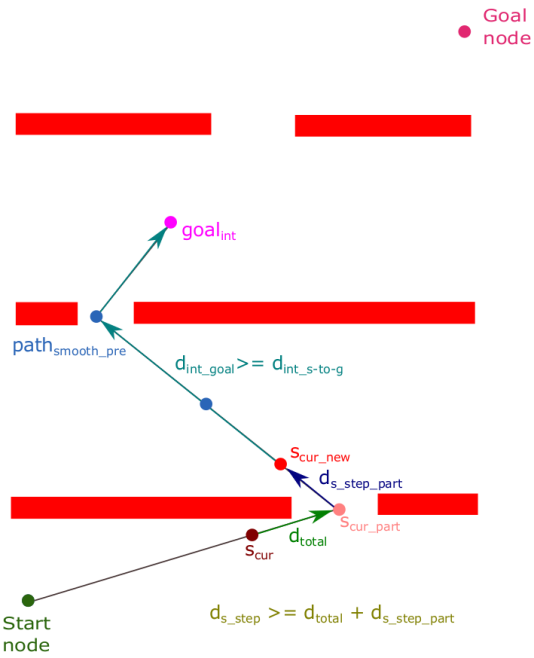


Figure 3.3: Illustration of the move function

path until the expected UAV distance moved per iterate is exceeded or the previously smoothed path has been fully traversed. In the former case, the UAV position in the next iteration will reside in the segment connecting the node to the previous node in the smoothed path constructed in the previous iterate when the distance addition stops. In case the whole path is traversed, either the prospective UAV position is nearer to the goal by less than the expected UAV distance moved per iterate or no path has been created. Figure 3.3 graphically illustrates this rationale in case where the UAV position in the next iteration resides between the 2nd and 3rd node of the previously smoothed path (first node of the smoothed path is the UAV current position).

A situation can exist when the prospective UAV position resides within an unsafe distance of $\pm\frac{0.5}{(res-1)}$ from the obstacle planes. This buffer is included due to the environmental discretisation of the A* algorithm. Theoretically, the UAV position in the next iteration should never reside on an obstacle since the UAV position in the next iteration is a point on an obstacle free path. But, if this buffer is not included situations exist in which the UAV would be placed in a position less than half the distance from obstacle planes. This will potentially yield a new starting position on an obstacle plane after the discretisation of the UAV position and the intermediate goal points in the next iteration.

In case, the UAV position in the next iterate is within a distance of $\pm\frac{0.5}{(res-1)}$ from the obstacle planes then the predetermined distance moved by the UAV in one iterate is multiplied by a predetermined distance factor between 0 and 1. This process is repeated each time reducing the distance moved by the UAV in the specific iterate, until an obstacle-free UAV position for the next iterate is found.

Table 3.1: Parameter Definition for the Move Function

Parameter	Description
s_{cur}	Current UAV position
env_{para}	Environmental parameters namely the size of the environmental space and the size and position of obstacles in the environmental space
d_{s_step}	Distance covered by the UAV in every iterate
$path_{smooth}$	Constructed and smoothed path in the current iteration
$path_{smooth\_pre}$	Constructed path in the previous iteration
d_{int_s-to-g}	Distance between the current UAV position and the final goal point
i	Iterate count
d_{total}	Distance from s_{cur} to a previous smoothed path point in $path_{smooth\_pre}(i)$
$d_{s\_step\_part}$	Distance to be moved by the UAV $(d_{s\_step} - d_{total})$ from $path_{smooth\_pre}(i)$
$flag_{small}$	A Boolean parameter that denotes whether the distance between d_{s_step} is smaller than $path_{smooth\_pre}(1)$ and $path_{smooth\_pre}(2)$
$d_{int\_(i)-to-(i+1)}$	Distance between the smoothed path points i^{th} to $(i+1)^{th}$
n_{path_points}	Number of points in the previously generated smoothed path
d_{mov}	Distance the UAV will move from the current position. The new point will be the current UAV position at the next iteration
s_{cur_new}	Next iterate UAV's current UAV position
d_{factor}	Distance reduction factor

$s_{cur} = move(env_{para}, d_{s\_step}, path_{smooth\_pre}, goal, d_{int\_s-to-g})$

01: **if** $d_{int\_s-to-g} > d_{s\_step}$
02: $i = 1; d_{total} = 0; d_{s\_step\_part} = d_{s\_step}; flag_{small} = 0$
03: Find $d_{int\_(i)-to-(i+1)}$ and n_{path_points}
04: **while** $(d_{total} < d_{s\_step})$ **and** $(i < n_{path\_points})$
05: **if** $i > 1$ **then** find $d_{int\_(i)-to-(i+1)}$ **end**
06: $d_{s\_step\_part} = d_{s\_step} - d_{total}$
07: $d_{total} = d_{int\_(i)-to-(i+1)} + d_{total}; i++; flag_{small} = 1$
08: **end**
09: **if** $flag_{small} == 1$ **then** $i--; d_{mov} = d_{s\_step\_part}$
else $d_{mov} = d_{s\_step}$ **end**
10: Define s_{cur_new}, d_{mov} from $path_{smooth\_pre}(i)$
11: **while** s_{cur_new} is on obstacle (check env_{para})
12: $d_{mov} = d_{mov} \times d_{factor}$
13: Define s_{cur_new}, d_{mov} from $path_{smooth\_pre}(i)$
14: **end**
15: **else if** $path_{smooth}$ is empty **then** $s_{cur} = NaN$
16: **else if** $path_{smooth\_pre}(n_{path\_points}) == goal$ **then** $s_{cur} = goal$
17: **else** $s_{cur} = path_{smooth\_pre}(n_{path\_points})$ **end**
18: **end**
19: **end**

Algorithm 3: Move Function

If the UAV is nearer to the intermediate goal point by less than the distance to be moved by the UAV per iterate, either no path has been created since either UAV position and/or intermediate goal point are on an obstacle or no path is possible that can connect the UAV to the intermediate goal point. In these cases, the UAV position in the next iterate is assigned to NaN since the algorithm need to stop for the particular test case. In such case the algorithm is not successful in generating a feasible path. In case the last point in the previously generated smoothed path is the goal node, then the UAV position in the next iterate is assigned to the goal node and then the iterate is stopped in the main algorithm as a path to goal has been found. Otherwise, the UAV position in the next iterate is assigned to the last point in the smoothed path constructed in the previous iterate. The resultant UAV position is input into the main real-time algorithm to generate a smoothed path in the current iteration. After the UAV position for the next iterate has been defined the new distance between the latter and the intermediate goal point is re-calculated.

3.4.4. MAIN REAL-TIME ALGORITHM

Provided that the UAV position in the current iterate is defined and does not reside on the goal node, the new intermediate goal point is defined as the final goal point ($goal$). This is only possible if the distance between the current UAV position and the final goal point is nearer by less than the predefined distance between the current UAV position and a prospective intermediate goal point. Otherwise a new intermediate goal point a predefined distance from the current UAV position is defined in the direction of the final goal. In case, that this new intermediate goal point resides on an obstacle, the distance from the current UAV position and the prospective intermediate goal point is reduced by a distance factor, similar to the next iterate UAV position calculation. This process is repeated each time reducing the previous UAV current position to intermediate goal distance by the distance factor until a new obstacle-free intermediate goal point is found provided that the distance between the current UAV and the intermediate goal point is larger than the distance the UAV is expected to move in one iterate. If the latter condition is not considered then a new intermediate goal point may reside nearer to the current UAV position by less than 1 step ahead possibly even in the same position as the current UAV position. This will effectively imply that the UAV will move by less than 1 step. In such cases, no feasible path can be constructed.

The intermediate goal point can be selected in any direction from the current UAV position only if it is assumed that the sensory system has a 360^o field-of-view. With this approach, the UAV will waste time and resources in exploring areas in the vicinity of the start location, limiting progress towards the goal and increasing the risk of collision or attack from obstacles and enemy, respectively. Therefore, new goal points shall be selected in the direction of the goal point unless the new intermediate goal point resides on an obstacle.

Once the current UAV position and a valid intermediate goal point are defined, the A* or RRT algorithm is applied to generate a feasible path between the current UAV position and the intermediate goal point. If a path is not generated, then the While loop of line 04 in Algorithm 4 is terminated. The intermediate time and the total time are noted irrespective of whether a path has been created or not for analysis purposes. Then the

previous UAV position is set to the new generated UAV position.

Finally, if the *while* loop at line 04 is terminated with the UAV reaching the goal position, then the algorithm is successful otherwise the UAV is not able to reach the goal either because a path is not possible or the time allocated is not enough. Algorithm 4 and the associated parameter definition Table 3.2, defines the rationale explained in this sub-section.

Table 3.2: Parameter Definition for the Real-time Algorithm

Parameter	Description
res	Resolution
$t_{iterate\_max}$	Maximum time to generate a path between iterates
$t_{path\_gen\_max}$	Maximum time allocated to reach the goal point from the start position
d_{int_goal}	Distance between the current UAV position and the prospective intermediate goal point
lim	Limits of 3D boundaries
$flag_{path}$	Path possibility flag
$time$	Time counter
$time_{iterate}$	Iterate time
$iterate$	Iterate number
$goal_{int}$	Intermediate goal point in the next iterate

3.4.5. CONCLUSION

The previous subsection presented an algorithm for the real-time implementation of the A* and RRT algorithms applicable to a 3D environment. This generic real-time algorithm is designed to assess the applicability of both path planning algorithms with respect to time and sensory constraints. The UAV must be supplied with timely obstacle free path segments whilst it is moving in a previously unknown environment with the aim of reaching the final goal in the minimum time without being in a situation of standing still waiting for new path segments to follow. The success of either or both A* and RRT algorithms will depend upon the considered environment and most importantly the UAV and its onboard systems. In the next section, the constants defined in Table 3.3 will be correlated with the parameters of real UAVs utilised for indoor applications. Moreover, the experimental scenarios considered will be defined.

3.5. PARAMETER DEFINITION AND EXPERIMENTAL SCENARIO DEFINITION

3.5.1. REAL-TIME ALGORITHM PARAMETER ASSIGNMENT

In this section the different parameters associated with the implementation of the real-time path planning algorithm are defined. In this regard, Table 3.3 defines the assigned values (Maximum, Minimum and Nominal values) for each parameter considered in the

01: Define $start$, $goal$, res, $t_{iterate\_max}$, d_{s_step}, $t_{path\_gen\_max}$, d_{int_goal}, d_{factor} and lim.
02: Apply the A* ripple reduction algorithm if A* is considered [23]. Update lim.
03: $s_{cur} = start$; $flag_{path} = 1$; $time = 0$
04: **while** $d_{int\_s-to-g} > \frac{2}{(res-1)}$ **and** $time < t_{path\_gen\_max}$ **and** $time_{iterate} < t_{iterate\_max}$ **and** $flag_{path} == 1$ **then**
05: Re-set and start $time_{iterate}$
06: **if** $s_{cur} != start$ **then**
07: $s_{cur\_new} = move(env_{para}, d_{s\_step}, path_{smooth\_pre}, goal, d_{int\_s-to-g})$.
08: **if** $s_{cur\_new} == NaN$ **then** $flag_{path} = 0$ **end**
09: **end**
10: **while** $(flag_{path} == 1)$ **or** $s_{cur} == goal$
11: Update d_{int_s-to-g}
12: **if** $d_{int\_s-to-g} < d_{int\_goal}$ **then** $goal_{int} = goal$
13: **else** $goal_{int}$ is d_{int_goal} from s_{cur_new} in the direction of $goal$ **END**
14: $iterate = 1$; $d_{s-to-int\_goal} = goal_{int}$
15: **while** $goal_{int}$ is on obstacle **and** $d_{s-to-int\_goal} > d_{s\_step}$
16: $goal_{int}$ is $d_{int\_goal} \times d_{factor}{}^{iterate}$ from s_{cur_new} in the direction of $goal$.
17: $iterate = iterate + 1$; Re-calculate $d_{s-to-int\_goal}$
18: **end**
19: Apply the A* or RRT algorithms to define $path_{smooth}$ using as input arguments the obstacle scenario, res, s_{cur_new}, $goal_{int}$, $path_{smooth\_pre}$ and lim (refer [22])
20: **if** $path_{smooth} == NULL$ **then** $flag_{path} = 0$ **end**
21: Stop $time_{iterate}$; $time = time + time_{iterate}$; $s_{cur} = s_{cur\_new}$.
22: **end**
23: **end**
24: **if** $d_{int\_s-to-g} < \frac{2}{(res-1)}$ **and** $flag_{path} == 1$ **then UAV reached** $goal$.
25: **else** $goal$ **could not be reached end**
26: **end**

Algorithm 4: Real-time Algorithm

real-time path planning algorithm explained in Section 3.4. Some of the parameters are inter-related and are derived from UAV path planning literature. The resolution (for A*) (step size (for RRT), reciprocal of resolution) is set at 21. This nominal value is selected based on the analysis of our previous work [22, 23].

Literature suggest that a 4s to 5s look-ahead is required for relatively low speed UAVs <50km/hr [58] and more than 20s look-ahead for high speed UAVs >500km/hr [30]. Sensor ranges define look-ahead distance. The UAV speed will determine how fast this distance can be covered. Therefore sensor ranges are a function of the UAV speeds. In low speed application the range will be in the region of 10m [58, 59] although large obstacle such as bushes and poles can be identified within a 20m to 25m range [58]. Oppositely, in high speed application such range is extended to a few km [30]. The sensor update rate is defined by researchers in UAV obstacle avoidance algorithms between 10Hz-25Hz [6, 16, 45, 58]. Although the environment is continuously changing, for collision avoidance the obstacle speed must be smaller than UAV speed. This requirement is included since the UAV must be capable of moving at a higher speed than all the obstacles in the environment as otherwise the UAV cannot move away from obstacles that can be uncooperative agents. In view of this requirement it is assumed that the environment is refreshed between one step iterate and the next and in the mean time the environment is static.

The environmental space in low speed applications is a cube in the range of 250-350m [9, 59] increasing to 10-25km for high speed applications [26, 30]. Obstacle sizes are intuitively defined based on the speed and environmental space. In fact obstacles are defined by Yu *et. al.* [60] in a range of $16m \times 10 \times 100m$ for medium-low speeds of 40km/hr in a $700m \times 700m$ square. Similarly, Call [16] defined an obstacle of $60m \times 60m \times 60m$ for a speed of 58km/hr in a $500m \times 500m \times 500m$ cube environment. The obstacles in high speed environment are set to a few kilometres in all axes [30].

In our analysis of UAV path planning for indoor applications it is assumed that the UAV will have a speed between 5km/h and 50km/h with a nominal value of 5km/hr. The look-ahead distance limited by sensor range is varied from 0.108[-] to 0.3[-], for resolutions between 29[points/-] to 11[points/-]. This translates to a look-ahead circle (as illustrated in Figure 3.1) of 3 times the distance moved by the UAV in $t_{iterate\_max}$ (d_{s_step}). Theoretically, the look-ahead distance must be greater or equal to $d_{s\_step\_min}$ but some buffer must be considered as otherwise the UAV can be placed in a point where the immediate vicinity is unknown. When the UAV moves to this point, an obstacle can reside just outside the known area and if it is moving towards the UAV a collision can result. Ideally $d_{int\_goal} > 0.01[-]$ to allow for this buffer. If this parameter is not under analysis it will be set to 0.2[-] irrespective of the resolution considered.

The maximum time to generate a path segment ($t_{iterate\_max}$) is dependent upon the time for the UAV to move d_{s_step} and UAV speed v_{UAV}. This parameter is being assessed to analyse the effect of different processing power on path planning performance. This time will vary between 4.29s (highest UAV speed, with lowest d_{s_step}) and 36s (lowest UAV speed, with highest d_{s_step}). If this parameter is not under analysis it is set to $\frac{d_{s\_step}}{v_{UAV}}$.

Based on the above literature, for low to medium speeds in a $500m \times 500m \times 500m$ environmental space, d_{s_step}, the distance moved by the UAV in one iterate, is set to the distance between three consequent graph points translating into a distance of 35.71m

for a resolution of 29 per dimension [points/500m] until a distance of 300m for a resolution of 11 per dimension [points/500m]. If a generic cubic environment of $1 \times 1 \times 1$ is considered, the speed range translates from 0.01[-/s] to 0.1[-/s] in steps of 0.01[-/s] and d_{s_step} translates to 0.07143[-] to 1[-] for resolutions of 29 to 11[points/-] per dimension.

The distance considered shall always be greater than the distance between two graph points especially in A*, since after the discretisation of the environment or when the new prospective UAV position resides on an obstacle and the distance reduction factor is applied, the UAV future position may map to the current grid point and the UAV would be blocked in the same position yielding no path. This further confirms the selection of the nominal d_{s_step} to the distance between three consecutive graph points.

The maximum time to generate the whole path ($t_{path\_gen\_max}$) is set in a range of 45s to 360s based upon factor of 10 × factor of $t_{iterate\_max}$. The distance factor reduction applied in cases where the new intermediate goal point and/or the new UAV position reside on an obstacle is set in a range of 0.5 to 0.95 and 0.8 if this parameter is fixed.

Table 3.3: Real-time algorithm parameter under analysis

Parameter	Minimum	Maximum	Nominal Value	Units
External Parameters				
UAV Speed (v_{UAV})	0.01	0.1	0.03	[-/s]
Sensor Range (d_{int_goal})	$d_{s\_step\_min} = 0.07143$	0.3	0.2	[-]
Computational Power ($t_{iterate\_max}$)	$\frac{d_{s\_step\_min} \times 60 \times 60}{u_{UAV\_max} \times 1000} = 4.29$	$\frac{d_{s\_step\_max} \times 60 \times 60}{u_{UAV\_max} \times 1000} = 36$	$\frac{d_{s\_step} \times 60 \times 60}{v_{UAV} \times 1000}$	s
Internal Parameters				
Distance to travel per iterate (d_{s_step})	$\frac{2}{res_{max}-1} = 0.07143$	$\frac{u_{UAV\_max} \times 1000 \times t_{iterate\_max}}{60 \times 60} = 1$	$\frac{2}{res-1}$	[-]
Maximum time to generate path ($t_{path\_gen\_max}$)	$10 \times t_{iterate\_max}(min) = 45$	$10 \times t_{iterate\_max}(max) = 360$	$10 \times t_{iterate\_max}$	s
Distance reduction factor (d_{factor})	0.5	0.95	0.8	[-]

Based on this analysis, it can be concluded that the main parameters of interest to decide which UAV is required for a specific situation are: UAV speed, sensor range and computational power of the UAV. Other parameters including resolution, distance to travel per iterate, maximum time to generate a path and the distance reduction factor are equally important parameters in relation to the real implementation of the real-time

path planning algorithm. The path length, path planning time and success rate are dictated by the mission requirements. Therefore, in line with the scope of this work, the UAV parameters will be investigated independently to outline the relationship between these parameters and path planning performance. In this regard, parameter values are generically defined as in Table 3.3 to facilitate the usability of the recommendations and conclusions of this work.

3.5.2. EXPERIMENTAL SCENARIOS

The experimental space is defined as a cube with dimension of 1x1x1 in generic units with centre (0,0,0) as in our previous work [22, 23]. The distance between two surfaces of the cube can vary by up to $\pm\frac{0.5}{(res-1)}$ in case of the application of the A* ripple reduction algorithm. The generic environmental space is normalised to arbitrary units so that the test scenarios can be scaled to any environmental space. These test scenarios were originally developed by Clifton *et. al.* [61] and made available online in [62]. For all tests the start and goal nodes are defined at [0,-0.5,0] and [0,0.5,0], respectively. In case the A* ripple reduction algorithm is applied, the associated shifting is added to the start and goal points. The same three different obstacle scenarios, defined in Chapter 2 and illustrated in Figure 3.4 are considered:

1. Scenario 1: Two obstacle planes in the Y-Z axis with 0.2x0.2 square openings;
2. Scenario 2: Three obstacle planes in the Y-Z axis with 0.2x0.2 square openings and two obstacle planes in the X-Y planes with no openings; and
3. Scenario 3: Five obstacle planes in the Y-Z axis with 0.2x0.2 square openings and two obstacle planes in the X-Y planes with no openings.

3.6. RESULTS

The aim of this analysis is to assess the applicability of 3D UAV path planning in real-time. This analysis will highlight the effects of the parameters mentioned in Table 3.3 to help identify the best configuration for different UAV real-time path planning applications.

The real-time path planning algorithm defined in Section 3.4 is implemented using both the A* and RRT algorithms in the experimental scenarios defined in Section 3.5.2 with the parameters defined in Section 3.5.1. Each section will analyse one parameter separately keeping all other parameters constant so that its effect on path planning performance in terms of path length computational time and success rate will be assessed independently.

All tests are performed using a desktop computer equipped with an Intel Xeon ES-1650, 3.2GHz. The same processor is used to test both path planning algorithms, implying that a fair comparison can be made between the two algorithms. The processing power in the mentioned processor is significantly larger than standard UAV onboard processors. In this setup, the processor that is used is not totally dedicated to the MATLAB code but must compute additional background tasks associated with Windows and other programs. Moreover, in this work, a set of parameters tabulated in Section 3.5.1 are

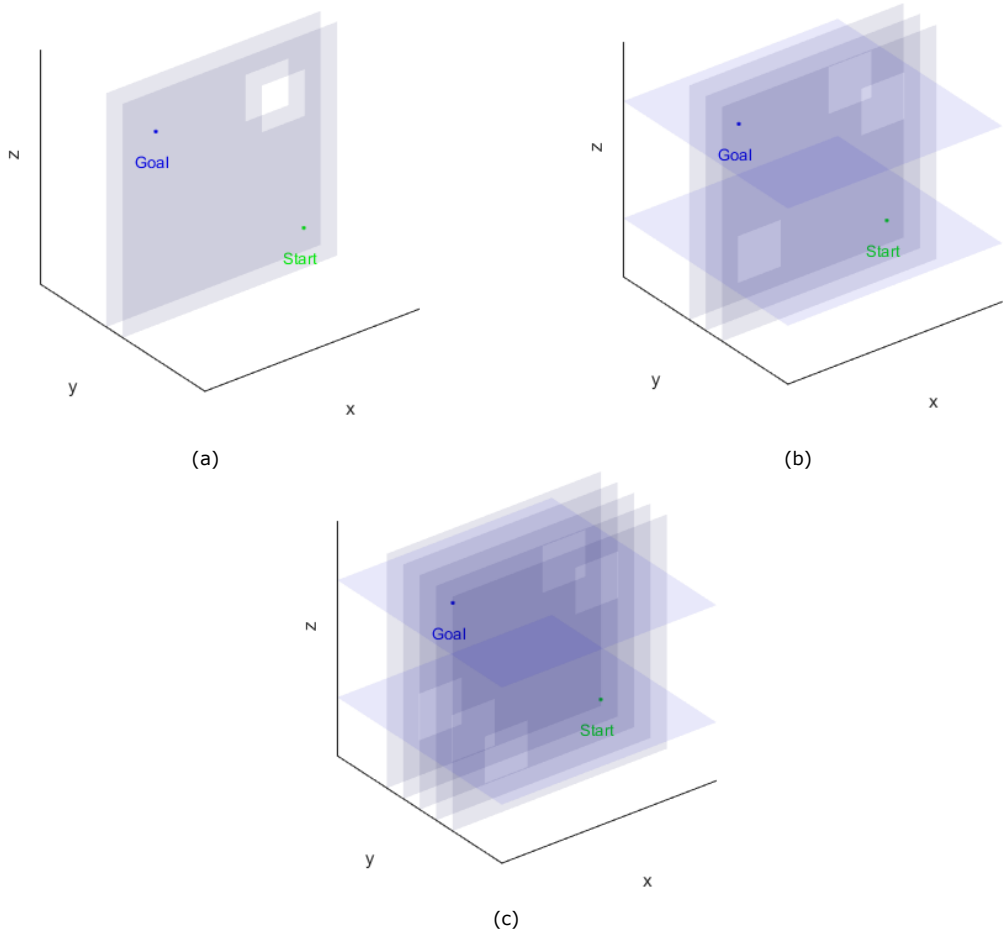


Figure 3.4: Obstacle scenarios: (a) First Scenario (b) Second Scenario and (c) Third Scenario modified from [22], consisting of obstacle planes in the X-Z with windows as openings and X-Y planes for scenarios 2 and 3 with no openings.

generically defined in view of the mission constraints and processing power available. If a different processor is available and/or different mission constraints are requested these parameters can be tuned on the basis of the recommendations of this work to ensure real-time path planning.

3.6.1. SPEED (v_{UAV})

Speed is one of the variable parameters that is considered for analysis. Speed is varied between 0.01[-/s] and 0.1[-/s] in steps of 0.01[-/s] while all the other parameters defined in Table 3.3 are set at their nominal value. For each considered speed for both A* and RRT algorithms the test is performed 100 times and the mean and 95% confidence interval illustrated in Figure 3.5. Since both algorithms are not able to generate the path in all considered scenarios, either because the time to generate a path between iterates or the total time is exceeded, a bar graph showing the distribution of successful and unsuccessful paths is illustrated in Figure 3.5 (c) and (f) for A* and RRT respectively. The unsuccessful runs are not considered in the path length vs. speed and time vs. speed plots.

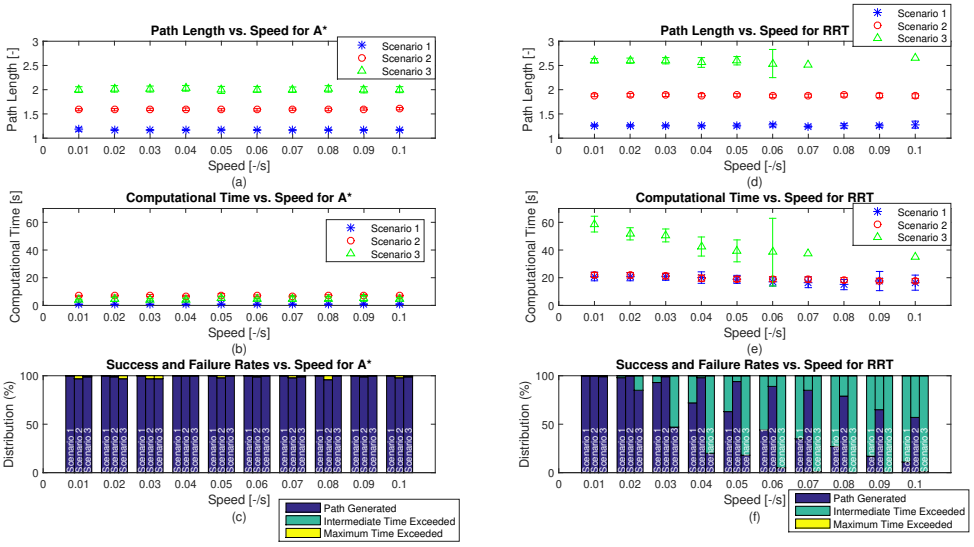


Figure 3.5: Performance parameters vs. speed: (a) Path Length for A*, (b) Computational Time for A*, (c) Success and Failure rates for A*, (d) Path Length for RRT, (e) Computational Time for RRT and (f) Success and Failure rates for RRT for 100 iterates for each considered situation (speed and scenario) with 95% confidence interval. ($res = 21, d_{step\_RRT} = 0.01[-], d_{s\_step} = 0.01[-], d_{int\_goal} = 0.02[-], d_{factor} = 0.8$ and $t_{iterate\_max}, t_{path\_gen\_max}$ are a function of the UAV speed).

The mean path length and standard deviation for A* remains almost constant for all considered UAV speeds. Similarly for RRT, the mean path length remains constant for all considered speeds although the success rate reduces with increase in speed for all scenarios. The standard deviation remains constant except for the high-end speeds of Scenario 3. This is attributed to the fact that as scenario complexity and speed increase

the algorithm is less successful (refer to Figure 3.5 (f)) and therefore the mean and standard deviation are measured on a smaller sample made up of the best performance runs. In fact in this case, the amount of successful runs decreases, reducing the sample size to less than 10 with no successful runs at 0.09[-/s] and only 1 successful run for 0.08[-/s] and 0.1[-/s].

The path length depends mainly on Scenario complexity for both the A* and RRT algorithms as a longer path is required to traverse through obstacle plane windows residing in alternating sides of obstacle planes [22, 23]. Furthermore, the mean path length for A* is smaller than RRT, confirming, theory that the A* algorithm is more optimal than RRT [44]. The difference in path length between the A* and RRT algorithms increases further as the scenario complexity increases. A* generated paths are more optimal and the path length reduction by the smoothing algorithm is less than that for RRT. In complex scenarios the improvement introduced by the smoothing algorithm is limited since the elimination of an intermediate node is more likely to generate colliding paths and therefore the oscillatory RRT-generated paths will be reduced by a lesser margin with respect to A* for the same situation.

The time to generate a path for A* is independent of the speed for all scenarios. This result confirms theory, since the algorithm will utilise the same amount of time to generate a path, as it considers approximately the same intermediate start and goal nodes, irrespective of the time required by the UAV to travel to the next position, provided that this time is longer than the time to generate the intermediate path. As complexity increases the mean time increases to approximately 7 times and 4 times with respect to Scenario 1, for Scenarios 2 and 3 respectively, while the standard deviation increases proportionally. The mean path planning time for Scenario 3 is shorter than Scenario 2 although Scenario 3 is more complex. This results since the success rate of Scenario 2 is larger than that of Scenario 3 and therefore only the easiest instances are successful in Scenario 3 leading to a better mean path planning time than Scenario 2. For the considered scenarios, as the complexity increases the time increases, as it takes longer to find a non-colliding path and an intermediate start and goal nodes.

For RRT the computational time reduces with increase in UAV speed for all scenarios. This time reduction is attributed to the analysis described earlier that, the best performing runs for the same situation are considered, as other runs did not generate a path in the allocated time. In fact the drop in computational time is in line with the drop in successful runs (refer to Figure 3.5 (f)). The drop rate in successful runs for Scenario 2 with increase in speed is lower than the drop rate for other scenarios. This lower drop rate behaviour is exhibited in the computational time for Scenario 2 with respect to the other scenarios.

Scenario 2 is the most successful scenario for RRT although it is more complex than Scenario 1. From our previous results [22, 23], the difference in the path generation time between Scenario 1 and 2 using the RRT algorithm is close to zero for all considered step sizes. Furthermore, in simple scenarios with non-optimal resulting paths such as in RRT the smoothing iterates required until the stopping condition is reached is larger than that of the same path in an obstacle-rich environment since it is more likely in the latter environment for a smoothed path segment to collide with an obstacle. The increase in computational time due to smoothing increases the time to generate an intermediate

path, exceeding the maximum intermediate time allocation. Since for Scenario 2, the possibility of smoothing is lower than Scenario 1 while the difference in time to generate a path between intermediate nodes is similar in both cases then this explains the result that Scenario 2 is more successful than Scenario 1. Although more iterates are required for a longer Scenario 2 path than Scenario 1, the bottleneck in RRT is the maximum intermediate time and not the total time. For Scenario 3 the situation is different. From our previous work [22, 23], a longer computational time (more than 5 times) was required to generate a complete path due to the condensed obstacle space and very limited path options. This had a major effect at high speeds as a longer path must be generated in the same time with increase in speed.

The computational time for A* when compared to RRT is shorter by multiple times for all scenarios. It can be concluded for the considered resolution/step size that the computational time for Scenarios 1 and 2 is similarly increasing by a factor of 3 in Scenario 3 for RRT with respect to A* [22, 23]. During path construction A* considers points between the current UAV position and the intermediate goal point. On the other hand, the RRT algorithm can produce seeds anywhere in the available space, although the tree branch length is limited by d_{step_RRT}. This implies that the considered area in A* is much smaller than RRT. Therefore, the allocated time must be increased by multiple times for RRT with respect to A*. The difference in computational time for Scenarios 1 and 2 increases by a factor of about 5 for RRT with respect to A*, as the whole environmental space is 5 times the look-ahead distance. Similarly, for Scenario 3 the difference is even larger since this 5 times factor is multiplied by the 3 times factor difference in computational time between Scenarios 1 and 2 with respect to Scenario 3 described earlier. Furthermore, the RRT algorithm is less optimal than the A* and therefore more smoothing iterates are required until less than 1% reduction results over the past 20 smoothing iterates.

The A* algorithm is able to generate a path in the allocated maximum intermediate and total time in a range between 96% to 100% with an average of 99%. The stopping condition resulting in unsuccessful cases is triggered when the maximum time to generate the path is exceeded. Not the same can be deduced for RRT. As the speed increased the success rate drops. For Scenario 2 the success rate drops to a minimum of 57% at 0.1[-/s], although the algorithm is able to generate a path in at least 94% of the cases from low speeds up to 0.06[-/s]. The stopping condition triggered for all unsuccessful situations is always that the maximum allocated intermediate time is exceeded. This is mainly attributed to the lower optimality of the RRT algorithm with respect to the A* algorithm and the fact described earlier that the RRT algorithm considers all environmental space when generating an intermediate path. For Scenario 1 the success rate drop starts even at low speeds of 0.02[-/s] increasing up to 11% at the highest speed. A sharper drop is experienced in Scenario 3 where a 47% success rate is noted at a speed of 0.03[-/s], dropping further to 5% at 0.06[-/s], reaching 0 successful runs at 0.09[-/s].

This result shows that A* can be applied for all speeds considered at a resolution of 21 and maximum and intermediate time allocation, while RRT can only be successfully applied for low speeds not exceeding 0.02[-/s] for Scenarios 1 and 2 and less than 0.01[-/s] for Scenario 3 for a step size of 0.05[-] for the same time constraints as A*, unless the intermediate step size is increased. This situation does not eliminate the RRT

algorithm from consideration as the amount of smoothing can be reduced, reducing the intermediate time to generate a path, ultimately increasing the success rate.

3.6.2. LOOK-AHEAD DISTANCE (d_{int_goal})

The look-ahead distance, defined as the distance between the current UAV position and a prospective intermediate goal point, is another parameter that affects the performance of the path planning algorithms. This distance is mainly dependent upon the precision and accuracy of sensory systems available on-board a UAV and thus can vary between different UAV models. The look-ahead distance parameter is varied between the distance travelled by the UAV per iterate to 0.3[-] which for the considered resolution/step size is 3 times the distance moved by the UAV in every step. In the former case the planner will define a path that will be totally traversed by the UAV in the next iterate while in the latter only a third of the planned path will be traversed. In practice, the minimum value considered can result in situations where the intermediate goal node will reside exactly in the vicinity of an obstacle resulting in a collision and no solution in the next iterate as no look-ahead will be considered.

Therefore, the look-ahead distance (d_{int_goal}) is varied from 0.1[-] to 0.3[-] in steps of 0.02[-] while the other parameters defined in Table 3.3 are set to their nominal value. For each considered distance for both A* and RRT algorithms the test is performed 100 times and the mean with a 95% confidence interval is illustrated in Figure 3.6. A similar bar graph as in the previous results representing the successful and unsuccessful runs is also illustrated in Figure 3.6 (c) and (f) for the A* and RRT algorithms, respectively.

The results in Figure 3.6 show that the path length for A* is mainly dependent upon the scenario difficulty. Another interesting point is that, as the look-ahead distance increases, the path length reduces for simple scenarios and slightly increases for complex scenarios. For Scenario 1, an 8% difference in path length between the lowest and highest look-ahead distance with respect to the mean path length results. This difference is attributed to the fact that the longer the look-ahead distance, the lower the variation from the shortest line connecting the start and goal points when considering also that in this scenario only an upper right turn (refer to Figure 3.4 (a)) is required to pass through plane windows. On the other hand, for Scenario 2 and 3, the lower path length results at the lowest look-ahead distance and as the look-ahead distance increases to $3 \times d_{s\_step}$. For obstacle-rich scenarios like Scenarios 2 and 3, a maximum occurs at $1.5 \times d_{s\_step}$. This results since in complex scenarios the intermediate goal point positions are very limited especially in Scenario 3 and such a look-ahead distance can offset the intermediate goal point, which in the next iterate can change drastically. If this look-ahead distance is reduced to d_{s_step}, then the next UAV position will be the current intermediate goal point leading to very low overshoot in complex scenarios. For Scenario 3 this drop in path length is more evident in Scenario 2 due to the drop in success rate for Scenario 3 at low look-ahead distances. For A* in Scenario 2, a drop in success rate is exhibited at the path length maximum. Otherwise a close to 100% success rate results for Scenario 2 just as for RRT. Furthermore, as the look-ahead increases from $1.5\times \rightarrow 3 \times d_{s\_step}$, the overshoot is limited as compared to the lower look-ahead distance cases due to the reason described for Scenario 1. The path length reduction rate as the look-ahead distance increases from $1.5\times \rightarrow 3 \times d_{s\_step}$ is lower for Scenario 2 and 3 with respect to Scenario 1

due to the limited free space.

As deduced in the above analysis A* produced shorter paths with respect to RRT for all scenarios considered and the difference in length increases with an increase in scenario complexity. In RRT the path length at the highest look-ahead distance is 16% shorter with respect to the path length at the lowest look-ahead distance. This behaviour is also exhibited in A* but with only 8% difference for the same scenario. In RRT intermediate goal points can reside on any obstacle-free point on the connecting line from the current UAV position to the final goal point. This advantage over A*, which is limited by grid positions, is bigger as the look-ahead distance increases, because the intermediate goal point will reside on the line connecting the start and goal positions, limiting overshoot especially for simple scenarios. For Scenario 2, a lower drop of 6% in path length is recorded due to a denser obstacle environment with increase in look-ahead distance. Furthermore, the same maximum occurs at $1.5 \times d_{s\_step}$ for the same reasons described for A*. Similar to Scenario 3, the same maximum is exhibited but the drop at low look-ahead distance is lower with respect to A*, confirming that the drop in success rate for A* in Scenario 3 is responsible for a reduction factor in path length at low look-ahead distances.

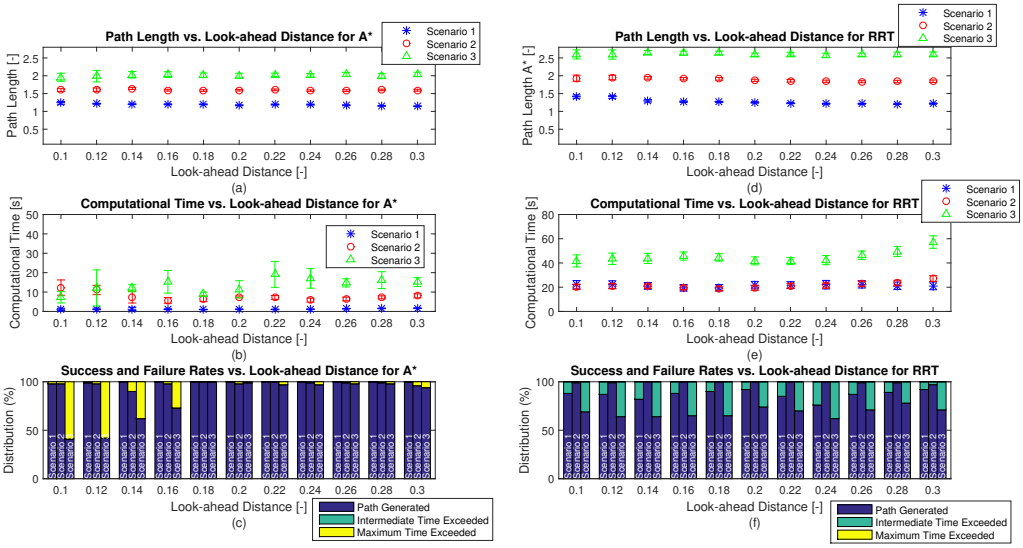


Figure 3.6: Performance parameters vs. Look-ahead distance (d_{int_goal}): (a) Path Length for A*, (b) Computational Time for A*, (c) Success and Failure rates for A*, (d) Path Length for RRT, (e) Computational Time for RRT and (f) Success and Failure rates for RRT for 100 iterates for each considered situation (speed and scenario) with 95% confidence interval. ($res = 21, d_{step\_RRT} = 0.1[-], v_{UAV} = 0.03[-/s], d_{s\_step} = 0.1[-], t_{iterate\_max} = 12s, t_{path\_gen\_max} = 120s$ and $d_{factor} = 0.8$).

The computational time for both A* and RRT algorithms mainly depends upon the scenario complexity although RRT Scenarios 1 and 2 yield almost the same result, as more smoothing iterates are required in Scenario 1 than in Scenario 2, because Scenario 2 limits smoothing due to the obstacle-rich environment. The difference in computa-

tional time between A* and RRT is the same as described earlier, with A* outperforming RRT in all scenarios. The variation in computational time with respect to look-ahead distance can be approximated by a parabola for all scenarios in both path planning algorithms. For A* a local minimum in computational time is exhibited between 80m and 90m for all scenarios, whilst for RRT the lowest computational time is exhibited at 90m, 90m and 110m for Scenarios 1 to 3, respectively.

The success rate for A* is 100% for 0.18[-] in all scenarios. For Scenario 1, the success rate remains 100% for look-ahead distance larger than 0.14[-] dropping by 2% for the lowest considered resolution. For Scenario 2, a 90% success rate results at a look-ahead distance of 0.14[-] with all other situations exhibiting a success rate of 95% or more. This occurs at the maximum of path length for this particular scenario. For Scenario 3, a successful rate of more than 92% is exhibited for look-ahead distances greater than 0.16[-]. For lower look-ahead distances the success rate drops to 40%. The maximum time is exceeded in all unsuccessful runs. For RRT in the first Scenario the success rate varies between 74% at 0.2[-] (d_{int_goal}) to 92% (d_{int_goal}). For Scenario 2, the success rate varies from 97% to 100%. For Scenario 3, the success rate varies from 78% at 0.3[-] (d_{int_goal}) to 61% at 0.24[-] (d_{int_goal}). The maximum allowable intermediate time is exceeded in all unsuccessful runs. **In conclusion, between 0.18[-] and 0.2[-] both the A* and RRT algorithms exhibited the best results.**

From the path length, computational time and success rate results it can be concluded that the optimal look-ahead distance for both algorithms for the considered scenarios, resolution/step size and speed shall be $1.8\times \rightarrow 2\times$ **the distance moved per iterate. In other words, the planner shall ideally define an obstacle-free position two steps from the current position with knowledge of the environment within this distance.**

3.6.3. MAXIMUM INTERMEDIATE TIME ($t_{iterate\_max}$)

The maximum intermediate time is the maximum time allocated for a path to be generated from the current UAV position to an intermediate goal point. As defined in Table 3.3, it is mainly dependent upon speed and the distance moved between iterates d_{s_step}. By varying this parameter, the computational power onboard the UAV is indirectly analysed to ultimately determine the required computational power for the particular situation. From the results in Section 3.6.1 and Section 3.6.2, it can be concluded that this parameter is the bottleneck, especially for RRT in complex scenarios. Therefore, to further assess the effect of this parameter on performance it is varied between 0.006[-] and 0.024[-] in steps of 0.002[-]. These values are defined in line with the minimum and maximum values defined in Table 3.3 assuming a constant speed of 0.03[-/s] and the distance moved per iterate of double the distance between grid position or step size for A* and RRT, respectively, that is 0.024[-]. Figure 3.7 illustrates the mean test results for 100 runs with a 95% confidence interval.

Results of Figure 3.7 (a) and (d) confirm the results described earlier that the path length is mainly dependent upon the scenario complexity and that the A* algorithm outperformed the RRT algorithm with the difference increasing with scenario complexity. For Figure 3.7 (b) and (e), the same conclusions as above can be drawn, with the A* outperforming the RRT in all scenarios considered with the difference increasing for

Scenario 3.

The main scope of this analysis is to define the lowest intermediate time that shall ensure that the UAV has time to generate an intermediate path in real time for all considered scenarios. The maximum intermediate time is defined based on real UAV parameters for a nominal speed and using an off-the-shelf processor. This parameter is directly proportional to the distance moved per iterate and inversely proportional to the UAV speed. Therefore when increasing the maximum allowable intermediate time either the UAV speed is decreased or the distance moved per iterate is decreased or both. The effect of each of these parameters on path planning performance was already described in their respective sub-sections. Usually the UAV speed is defined by application and therefore its range is restricted. The lower limit of the distance moved per iterate is limited by the resolution/step size and if resolution increases, so that the former parameter decreases, the computational time will increase due to larger resolution for A*. The limitation for RRT, a sampling-based algorithm is not that direct, although the shorter the step size the more computational demand is required to generate a path for the same distance.

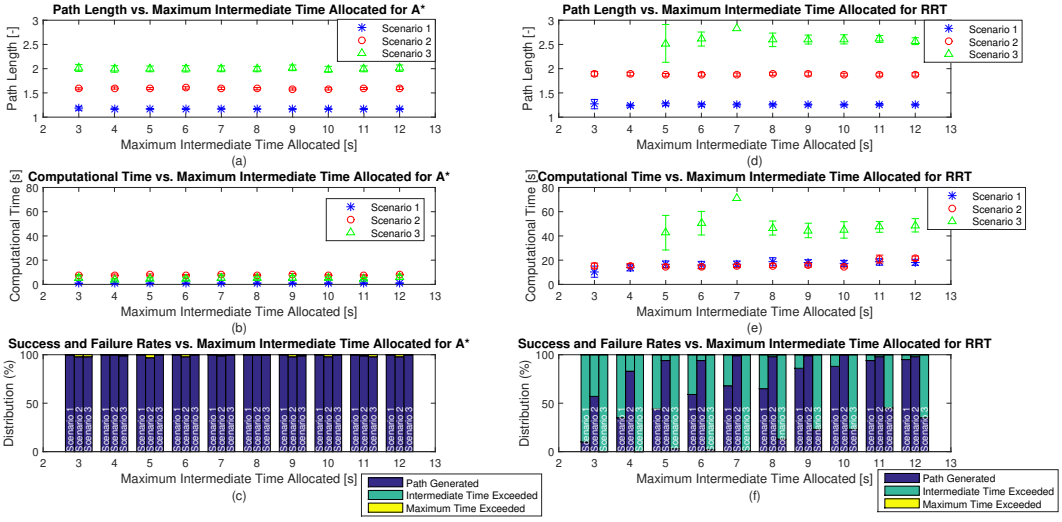


Figure 3.7: Performance parameters vs. Maximum Intermediate Time Allocated ($t_{iterate\_max}$): (a) Path Length for A*, (b) Computational Time for A*, (c) Success and Failure rates for A*, (d) Path Length for RRT, (e) Computational Time for RRT and (f) Success and Failure rates for RRT for 100 iterates for each considered situation (speed and scenario) with 95% confidence interval. ($res = 21, d_{step\_RRT} = 0.1[-], v_{UAV} = 0.03[-/s], d_{s\_step} = 0.1[-], d_{int\_goal} = 0.2[-], d_{factor} = 0.8$ and $t_{path\_gen\_max}$ is a function ($\times 10$) of the maximum intermediate time allocated).

For the A*, for the considered resolution/step size, speed, distance travelled per iterate and look-ahead distance, the planner is never limited by the allocated intermediate time. The low percentage of unsuccessful runs ($\leq 3\%$) occurs because the maximum time to generate the path is exceeded and not because the intermediate time is

not enough. Therefore, the defined maximum intermediate time limit can be further reduced from 12s for all scenarios, possibly allowing the UAV to increase its speed and/or in resolution.

Not the same can be concluded for RRT that is able to generate the path in less than 35% for the maximum considered intermediate time allocation in Scenario 3. For Scenario 1, at least 11s are required to achieve a success rate greater than 90% while for Scenario 2, at least 4s are required to achieve a 90% success rate. For the considered speed of 0.03[-/s] and $d_{s\_step} = 0.1[-]$ the maximum intermediate computational time is 12s. This shows that for Scenario 3, the intermediate time must be increased by multiple times, to achieve a success rate of 100%. For Scenario 1, this limit must be increased to a higher value by either decreasing the speed of decreasing the RRT step size with the latter leading to increase of computational time. For Scenario 2, the limit under analysis can remain the same for the considered speed, distance moved per iterate and RRT step size.

3.6.4. DISTANCE TO TRAVEL PER ITERATE (d_{s_step})

The distance to travel per iterate is a function of the UAV speed as the distance the UAV travels in a pre-defined time dictates the maximum allowable time to generate a path. The nominal value of this parameter is set to double the distance between grid positions (A*) or double the step size (RRT). To assess the effect of this parameter on performance, it is varied between this nominal value and the look ahead distance (d_{int_goal}). Therefore, the distance to travel per iterate is varied from 0.1[-] to 0.2[-] in steps of 0.01[-] while the other parameters listed in Table 3.3 are set to their nominal value. The longer the distance to travel per iterate the more intermediate time is allocated for the path planning and smoothing algorithms. For each considered distance for both A* and RRT algorithms the test is performed 100 times and the mean with a 95% confidence interval is illustrated in Figure 3.8. A similar bar graph representing the successful and unsuccessful runs is also illustrated in Figure 3.8 (c) and (f) for A* and RRT, respectively.

The mean and standard deviation in path length for both the A* and RRT algorithms is independent of distance to travel per iterate (d_{s_step}) but depends primarily on scenario complexity. The range of d_{s_step} considered is equivalent to the distance of three consecutive grid positions or $2 \times d_{step\_RRT}$. This relatively small variation and the relatively small windows through which the planner must pass shall have minimal effect on the path length, especially in obstacle-rich environments where the path options are limited, since d_{s_step} only dictates how the path is to be traversed not how it is constructed, as confirmed by the results of Figure 3.8 (a) and (d). Furthermore, the mean path length for A* is shorter than RRT since A* is more optimal, with the difference increasing with scenario complexity as deduced in Section 3.6.1.

For both algorithms the computational time is independent of d_{s_step} although as d_{s_step} approaches its minimum value (distance between 3 consecutive grid points for A* and equivalent for RRT), the computational time decreases. This minor improvement results as the path planning task will be divided into a larger set of simpler sub-tasks. This result is more evident for Scenario 3 of the RRT algorithm since the success rate at low d_{s_step} drops and therefore, as described in Section 3.6.1, the best performance results are considered in the mean and standard deviation calculations. As deduced from

literature [56, 57], and in line with the conclusions drawn in the speed analysis, the RRT algorithm takes 4-10 times longer to compute a path with respect to A* algorithm for the same conditions.

The A* algorithm is able to generate a path in all situations for Scenario 1 and in at least 97% and 98% for Scenarios 2 and 3. The maximum computational time is the stopping condition for unsuccessful runs. Not the same can be concluded for RRT as the success rate drops especially for Scenario 3 due to insufficient intermediate time as d_{s_step} approaches its minimum value. A maximum minor drop of 5% results for Scenario 1 while the RRT algorithm is 100% successful for Scenario 2. The result shows that the RRT is more computationally intensive since, as d_{s_step} approach its minimum value and hence the allowable computational time is lowest (path planning algorithm needs to generate a path in the time required to traverse double the step size d_{step_RRT}), the success rate drops.

The bottleneck in the considered path planners is the computational time, especially for the RRT algorithm. The analysis carried out in this sub-section shows that the distance traversed by the UAV in each iterate will not affect the validity and optimality of the generated path and is independent of the computational time required to generate the path.

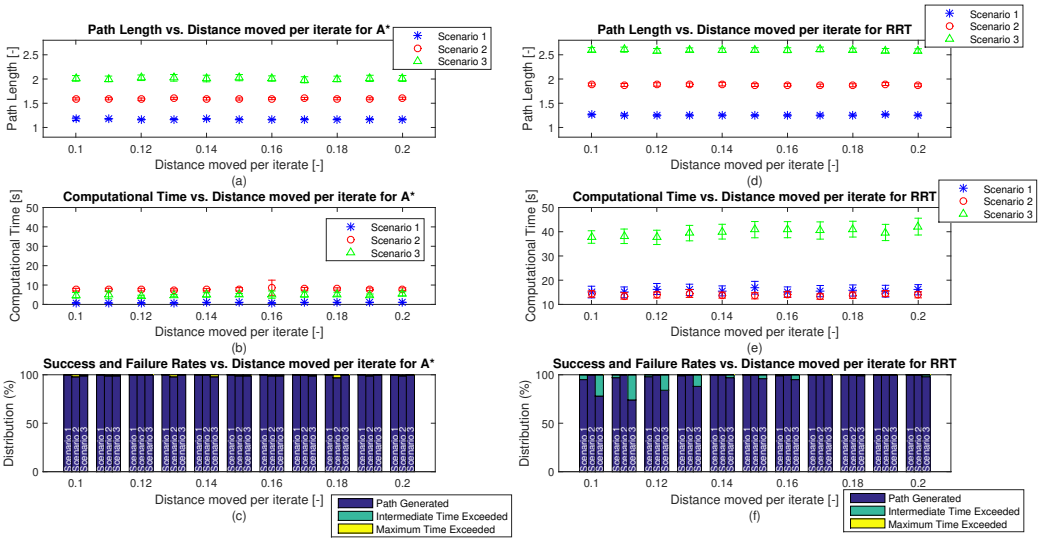


Figure 3.8: Performance parameters vs. Distance to travel per iterate (d_{s_step}): (a) Path Length for A*, (b) Computational Time for A*, (c) Success and Failure rates for A*, (d) Path Length for RRT, (e) Computational Time for RRT and (f) Success and Failure rates for RRT for 100 iterates for each considered situation (speed and scenario) with 95% confidence interval. ($res = 21, d_{step\_RRT} = 0.1[-], v_{UAV} = 0.03[-/s], d_{int\_goal} = 0.2[-], d_{factor} = 0.8$ and $t_{iterate\_max}, t_{path\_gen\_max}$ are a function of the distance to travel per iterate).

From a practical point of view, increasing d_{s_step}, will allow more time for the planner to generate a path improving the success rate without deteriorating the path length. Therefore the longer d_{s_step} the better. But the latter parameter is limited by the look-

ahead distance which is governed by the sensory systems and provided that all obstacles residing between the current and future UAV positions do not cross the path in the time the UAV is traversing.

3.6.5. MAXIMUM TIME TO GENERATE A PATH ($t_{path\_gen\_max}$)

This parameter is a function of the length of the path from start to goal. In the test analysis in Section 3.6, this parameter is arbitrarily defined as 10× the maximum intermediate time, irrespective of the scenario difficulty. For this considered nominal value, only a small percentage of runs for Scenario 3 are stopped due to this maximum time limitation. Although this parameter can be defined as a function of the number of intermediate path generations, the path planning algorithm may waste time venturing around prior to arriving to the goal. Moreover, the allocated time to reach a goal may be defined by the application. Therefore to define the best value for each Scenario this parameter is varied between 2× and 20× the maximum intermediate time ($t_{iterate\_max}$) in steps of 2. This implies that the maximum total time to generate a path is varied between 24 and 240 seconds. All the other parameters listed in Table 3.3 are set to their nominal value. Figure 3.9 illustrates the mean test results in terms of path length, computational time and success rate for 100 runs with a 95% confidence interval for both A* and RRT algorithms.

Results of Figure 3.9 (a) and (d) confirm that the path length is mainly dependent upon the scenario difficulty with the A* algorithm outperforming the RRT algorithm in all scenarios with the percentage difference in path length increasing with scenario difficulty. Similarly, as can be deduced from Figure 3.9 (b) and (e) the computational time is lower for A* with respect to RRT, for the respective scenarios, with the major difference of multiple times present for Scenario 3.

The main scope of this analysis is to define an adequate maximum time for the UAV to reach the goal point in view of a real time implementation. This parameter is dependent on a number of unrelated or inter-related and bounded or unbounded parameters namely scenario complexity, UAV speed, allocated, computational power, allocated intermediate time and the distance moved per iterate, assuming that the sensory system update rate is much higher than the allocated intermediate time. Therefore this parameter must be chosen in view of these relationships and other external constraints such as a user defined maximum mission duration.

For the A* algorithm, Scenario 1 is successful in all considered instances of maximum time allocation. This implies that for simple scenarios the A* algorithm can generate multiple path segment in $1 \times t_{iterate\_max}$, planning the path to goal in less than 1 second. For Scenario 2, 5% of runs are unsuccessful since the total time is exceeded prior to finding a solution. From Figure 3.9 (b), the upper bound in computational time never exceeded 8s. Therefore it can be concluded that the A* will not be able to reach the goal at some instances, as a collision or no possible path is possible, irrespective of the allocated total time which in the lowest case is 2× the highest computational time recorded. Similarly the same conclusions can be drawn for Scenario 3 although the failure rate is only 2% maximum. Although these values are low, A* could not guarantee a 100% solution for complex scenarios. This confirms theory that claims that A* does not offer a guarantee of solution [50].

For the RRT algorithm, the success rate improved as the maximum time to generate the path increases from 24s to 240s. For Scenarios 1, 2 and 3 the success rate improved from 84% to 95%, from 83% to 100% and from 0% to 72%, respectively. These results confirm that the RRT algorithm is computational intensive especially for complex scenarios. Furthermore, if the allocated time is greater or equal to 36s, the maximum intermediate time limitation is triggered and not the parameter under review. In fact, for Scenario 1 and 3 an improvement is present as the maximum time to generate the path increases, but 5% and 28% of the runs, respectively remain unsuccessful and does not decrease further unless the intermediate time is increased further. For Scenario 2, the maximum intermediate time constraint is not limiting so the success rate increased to 100% as the total allocated time to generate a path increased to more than 36s.

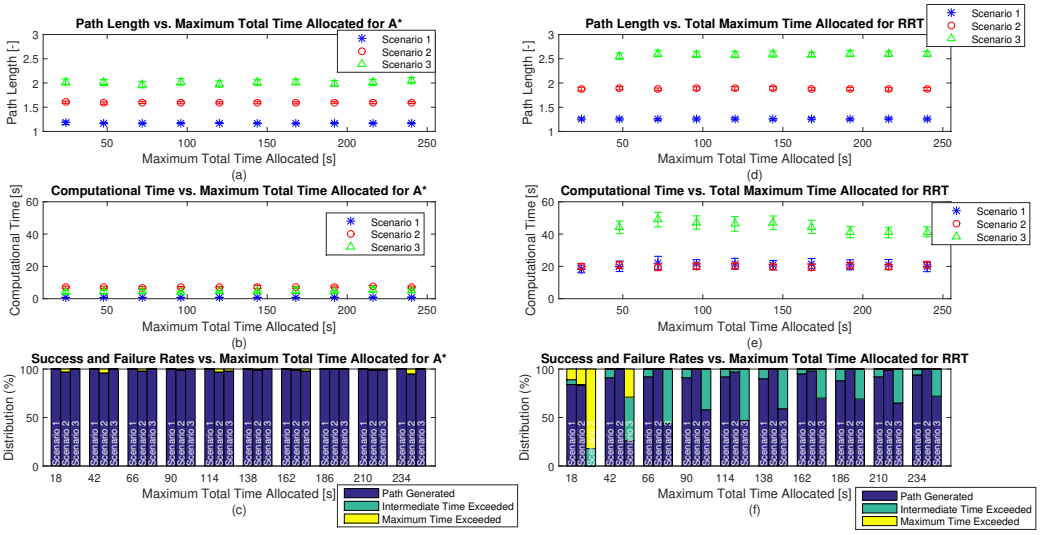


Figure 3.9: Performance parameters vs. Maximum Total Time Allocated ($t_{path\_gen\_max}$): (a) Path Length for A*, (b) Computational Time for A*, (c) Success and Failure rates for A*, (d) Path Length for RRT, (e) Computational Time for RRT and (f) Success and Failure rates for RRT for 100 iterates for each considered situation (speed and scenario) with 95% confidence interval. ($res = 21, d_{step\_RRT} = 0.1[-], v_{UAV} = 0.03[-/s], d_{s\_step} = 0.1[-], d_{int\_goal} = 0.2[-], d_{factor} = 0.8$ and $t_{iterate\_max} = 12s$).

In conclusion although the maximum allowable time to reach the goal is increased, neglecting real-time constraints, a solution cannot be guaranteed for all scenarios for both algorithms, although a minimum of 95% success rate is recorded for A* for all situations considered and the maximum time exceeded condition is never triggered beyond $4 \times t_{iterate\_max}$ for all scenarios in RRT. Furthermore it can be concluded that for both A* and RRT at $6 \times t_{iterate\_max}$ and above, the maximum allowable time to reach the goal has minimal to low effect on performance for the considered parameters listed in the caption of Figure 3.9.

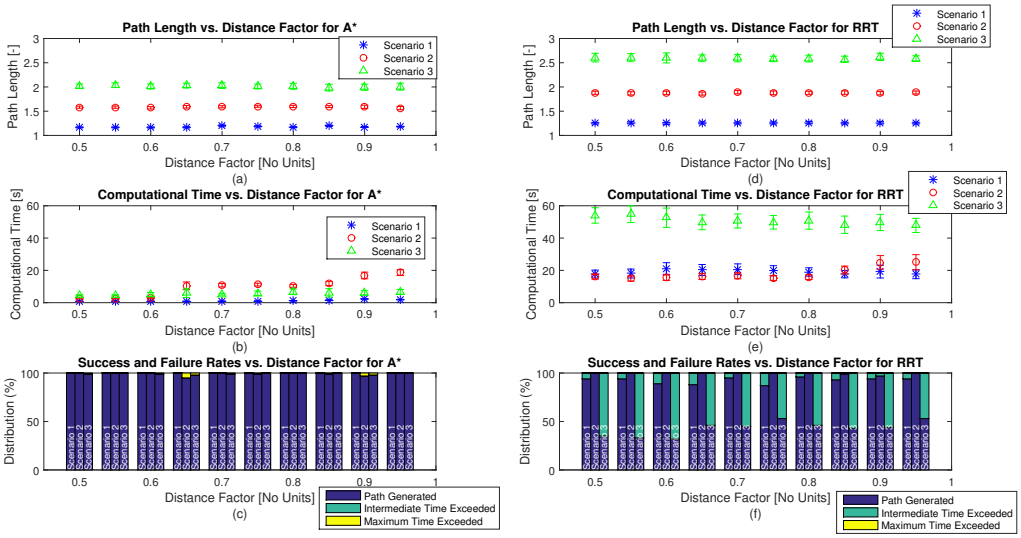


Figure 3.10: Performance parameters vs. Distance Factor (d_{factor}): (a) Path Length for A*, (b) Computational Time for A*, (c) Success and Failure rates for A*, (d) Path Length for RRT, (e) Computational Time for RRT and (f) Success and Failure rates for RRT for 100 iterates for each considered situation (speed and scenario) with 95% confidence interval. ($res = 21, d_{step\_RRT} = 0.1[-], v_{UAV} = 0.03[-/s], d_{s\_step} = 0.1[-], d_{int\_goal} = 0.2[-]$, $t_{iterate\_max} = 12s$ and $t_{path\_gen\_max} = 120s$).

3.6.6. DISTANCE REDUCTION FACTOR (d_{factor})

The distance reduction factor is a parameter utilised to re-evaluate the distance from the current UAV position to the next UAV position and from the current UAV position to an intermediate goal position as both may reside on an obstacle. When the next UAV position is defined, this position is not on an obstacle, as this point is selected on the path from the current UAV position to an intermediate goal point. But when the ripple reduction algorithm is applied, the current UAV position may reside on an obstacle. When the intermediate goal node is defined based on a look-ahead distance from the current UAV position, such a point can reside on an obstacle. The excessive reduction of both the distance moved by the UAV in one iterate and the look-ahead distance can lead to longer computational time to generate a path as shorter path segments are generated and/or situations in which the UAV and the goal point will remain in the same position. On the other hand, too low reductions will also increase computational time as each reduction needs to be checked each time. Therefore, to best define this value this parameter is varied between 0.5 to 0.95 in steps of 0.05 and applied to tests on the same test environment as in the previous sub-sections and setting the other parameters described in Table 3.3 to their nominal value. Figure 3.10 illustrate the mean path length, computational time and success rate test results for both A* and RRT algorithms.

Results in Figure 3.10 (a) and (d) show that the path length is mainly dependent upon the scenario complexity and independent of the distance factor parameter for both algorithms. This implies that irrespective of the reduction in length of path segments, in

the case where the new intermediate goal point or current UAV position resides on an obstacle, the path length shall not increase.

As the distance factor approaches unity the computational time shall increase as the algorithm is required to consider more points in the line connecting the current and future UAV position and/or the current and future intermediate goal points. For all scenarios in A*, an exponential increase in mean computational time with multiple times increase in confidence interval is exhibited at different values of distance factor, namely 0.85, 0.65 and 0.6 for Scenarios 1 to 3, respectively, with Scenario 2 exhibiting the largest ($> \times 3$) increase. These results impose the need to best define this parameter based on the different parameters considered and on the scenario in which the algorithm is required to operate. Furthermore, Figure 3.10 (c) shows that at certain instances of distance factor: 0.55, 0.8 and 0.95 the A* algorithm is 100% successful. **Therefore, this show that A* can offer a high success rate of 98% or better for the parameters considered if the distance factor is appropriately selected.**

For the RRT algorithm, the computational time is unaffected by the distance factor as can be deduced from Figure 3.10 (e). As the RRT algorithm is a sampling based algorithm, the environment is not quantised and therefore the possibility of the new UAV position residing on an obstacle after the movement function is computed is not applicable if a fixed obstacle environment is assumed. Not the same can be concluded for the A* algorithm. Only the possibility that the new intermediate goal position will reside on an obstacle remains. As obstacles are planes with 2 dimensions a slight deviation on the line connecting the current UAV position to the final goal position will eliminate this conflict. **Therefore in the case of RRT only one step will be required irrespective if the distance deduction (through the d_{factor} parameter) will be enough to produce an obstacle-free intermediate goal point. Therefore, ideally a 0.95 factor shall be considered not to deviate too much from the predefined d_{int_goal} distance.**

3.6.7. CONCLUSION

This section analysed the results of the developed path planning algorithm defined in Section 3.4 in terms of UAV speed, distance to travel per iterate, distance between current UAV position and prospective intermediate goal point, maximum intermediate time, maximum total time and distance factor for both algorithms in all 3 defined scenarios. Results show that:

1. A* is less affected by increase in speed with respect to RRT.
2. The longer the distance moved per iterate, which is limited by look-ahead distance, the higher the success rate.
3. The ideal look-ahead distance shall be $1.8\times \rightarrow 2\times$ the distance moved per iterate.
4. The A* algorithm is never limited by the maximum intermediate time allocation while RRT is able to generate the path in less than 35% of the cases.
5. For both A* and RRT $6\times$ the maximum total time is enough.
6. Some runs remain unsuccessful even if the maximum total time is increased.

7. A 0.95 distance reduction factor results in the best performance for both algorithms.

These summarised results in conjunction with others show the strengths and weaknesses of both algorithms as each of the above mentioned parameters is varied keeping other parameters constant. Table 3.4 summarises the results presented in this section by showing the effect of each UAV parameter on path planning performance. A +- sign shows that the effect of the UAV parameter on path planning performance measure is minimal and a + or - notation imply that the UAV parameter and path planning performance measure are directly or inversely proportional, respectively. The additional sign shows that the UAV parameter highly effects the path planning performance measure. For example, an increase in UAV speed for RRT will highly deteriorate the success rates. Finally, through this analysis guidelines for the definition of parameter values, according to mission requirements and constraints, are set out. These empirical values can be scaled in view of user-defined and external demands such as UAV speed.

Table 3.4: Relational table between UAV parameters and path planning performance

Parameter	Path Length		Planning Time		Success rate	
	A*	RRT	A*	RRT	A*	RRT
External Parameters						
UAV Speed (v_{UAV})	+-	+-	+-	+-	+-	- -
Sensor Range (d_{int_goal})	-	-	+	+	++	+-
Computational Power ($t_{iterate\_max}$)	+-	+-	+-	+	+-	++
Internal Parameters						
Distance to travel per iterate (d_{s_step})	+-	+-	+	+	+-	+
Maximum time to generate a path ($t_{path\_gen\_max}$)	+-	+-	+-	+-	+-	+
Distance reduction factor (d_{factor})	+-	+-	-	-	+-	+

3.7. CONCLUSION AND FUTURE WORK

The aim of this paper is to develop a platform to assess the validity of the two most utilised path planning algorithms (A* and RRT) in view of 3D UAV path planning in real-time. Literature highlights the importance of real-time path planning for autonomous 2D systems [24–26]. UAVs have to manoeuvre in complex, dynamic environments and therefore the need for real-time path planning is a must. Both path planning algorithms are tested with a common smoothing algorithm in 3 different scenarios with varying complexity. Real-time path planning is governed by a set of user-defined parameters

such as speed and time to reach the goal node, system-defined parameters such as look-ahead distance and computational power and internal constants such as resolution/step size and the distance reduction factor. The effect of the salient parameters on performance is assessed and analysed.

Results show that the A* algorithm outperforms the RRT algorithm in both path length and computational time for all scenarios considered, with the difference increasing with scenario complexity. Also the A* is successful in more than 90% of all tests for all scenarios considered, provided the look-ahead distance is at least double the distance moved per iterate. Oppositely, the RRT algorithm resulted in a lower success rate owing primarily to the longer computational time required to produce paths from the current UAV position to an intermediate goal point. The analysis presented in Section 3.6 outlines the best empirical values for each considered parameter if such parameter is not restricted by user demands or hardware limitations. To conclude, this analysis showed that both algorithms can be applied in real-time with a success rate up to 90% for all scenarios considered. It is up to the designer of the real-time 3D UAV path planning system to decide the best configuration for the requested task/s based on the analysis of Section 3.6.

This work can be utilised in the future to configure a real UAV for autonomous 3D UAV movement in indoor obstacle-rich environments emulating the considered scenarios. The implementation can be extended to outdoor environments where other factors such as wind and rain may influence the dynamics and sensory systems onboard the UAV. Furthermore, the performance of the real-time A* and RRT algorithm with moving obstacles is a research area that requires further assessment of the robustness of the developed real-time algorithm. The future trajectory, size and speed of such moving obstacles may be known, known with a certain degree of uncertainty or totally unknown to the path planning algorithm.

REFERENCES

[1] Kim, M.-H., Baik, H. and Lee, S., "Response Threshold Model Based UAV Search Planning and Task Allocation," *Journal of Intelligent & Robotic Systems*, Vol. 75, No. 3-4, 2014, pp. 625-640.

[2] Dai, R. and Cochran, J., "Path Planning and State Estimation for Unmanned Aerial Vehicles in Hostile Environments", *Journal of Guidance, Control, and Dynamics*, Vol. 33, No. 2, 2010, pp. 595-601.

[3] Chakrabarty, A. and Langelaan, J. W., "Energy maps for long-range path planning for small-and micro-uavs", *AIAA Guidance, Navigation and Control Conference*, Chicago, IL, 10-13 Aug., 2009, pp. 1-13.

[4] Amin, J. N., Boskovic, J. D. and Mehra, R. K, "A Fast and Efficient Approach to Path Planning for Unmanned Vehicles", *AIAA Guidance, Navigation and Control Conference*, Keystone, CO, 21-24 Aug., 2006, pp. 1-9.

[5] Gurdan, D., Stumpf, J., Achtelik, M., Doth, K-M, Hirzinger, G., Rus, D. "Energy-efficient autonomous four-rotor flying robot controlled at 1 kHz," *IEEE International Conference on Robotics and Automation*, Roma, Italia, 10-14 April, 2007, pp. 361-366.

[6] Franchi, A., Secchi, C., Ryll, M., Bulthoff, H. and Giordano, P (2012). Shared Control: Balancing Autonomy and Human Assistance with a Group of Quadrotor UAVs. *IEEE Robotics & Automation Magazine* [online] Available at: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6290692 [Accessed 26 Aug. 2018].

[7] Dittrich, J. S., Adolf, F.-M., Langer, A. and Thielecke, F. "Mission Planning for Small UAV Systems in Uncertain Environments,", *2nd European Micro Aerial Vehicle Conference*, Braunschweig, Germany, 25-26 July, 2006.

[8] Wang, G., Chu, H. E. and Mirjalili, S., "Three-dimensional path planning for UCAV using an improved bat algorithm," *Aerospace Science and Technology*, Vol. 49, pp. 231-238, 2016.

[9] Barrientos, A., Colorado, J., Martinez, A., Rossi, C., Sanz, D. and Valente, "Aerial Remote Sensing in Agriculture: A Practical Approach to Area Coverage and Path Planning for Fleets of Mini Aerial Robots", *Journal of Field Robotics*, Vol. 28, pp. 667-689, 2011.

[10] Lee, J., Huang, R., Vaughn, A., Xiao, X., Hedrick, J. K., Zennaro, M. and Sengupta, R. 'Strategies of path-planning for a UAV to track a ground vehicle", *AINS Conference*, 2003, pp. 1-7.

[11] Puri, A. 'A Survey of Unmanned Aerial Vehicles (UAV) for Traffic Surveillance", *Department of Computer Science and Engineering, University of South Florida*, 2005.

[12] Mayerowitz, S. "Amazon.com sees delivery drones as future (Update)", *Phys.org, The Associated Press*, [Online Database], https://phys.org/news/2013-12-amazon-unveils-futuristic-mini-drone-delivery.html, [retrieved 09 September, 2018].

[13] Wright, T. "In Rural Virginia, a Drone Makes the First Legal U.S. Package Delivery", *Air & Space Smithsonian*, [Online Database], https://www.airspacemag.com/daily-planet/rural-virginia-drone-makes-first-legal-us-package-delivery-180956053/?no-ist, [retrieved 09 September, 2018].

[14] Nakashima, R. "Drone company demos how blood air-drops will work in Rwanda", *AP News*, [Online Database], https://apnews.com/e5336fa71af347db99e11cd69ab16054/drone-company-demos-how-blood-air-drops-will-work-rwanda, [retrieved 09 September, 2018].

[15] Nakashima, TU Delft. "TU Delft ambulance drone drastically increases the chances of surviving cardiac arrest", *TU Delft*, [Online Database], https://www.tudelft.nl/2014/tu-delft/ambulance-drone-tu-delft-vergroot-overlevingskans-bij-\hartstilstand-drastisch/, [retrieved 09 September, 2018].

[16] Call, B. (2006). *Obstacle avoidance for unmanned air vehicles*. Master Dissertation. Brigham Young University.

[17] Boskovic, J. D., Knoebel, N., Moshtagh, N. and Larson, G.L., "Collaborative Mission Planning & Autonomous Control Technology (CoMPACT) System Employing Swarms of UAVs", *AIAA Guidance, Navigation and Control Conference*, Chicago, IL, 10-13 Aug., 2009, pp. 1-24.

[18] Ryan, A. and Hedrick, J. K., "A mode-switching path planner for UAV-assisted search and rescue", *Proceedings of the 44th IEEE Conference on Decision and Control*, Seville, Spain, 12-15 Aug. 2005, pp. 1471-1476.

[19] Park, S., Choi, H.-L., Roy, N., and How, J., "Learning Covariance Dynamics for Path Planning of UAV Sensors in a Large-Scale Dynamic Environment", *AIAA Guidance, Navigation and Control Conference*, Chicago, IL, 10-13 Aug., 2009, pp. 1-18.

[20] Crispin, C. and Sobester, A., "An Intelligent , Heuristic Path Planner for Multiple Agent Unmanned Air Systems", *AIAA Information Technology at Aerospace*, Kissemmee, FL, 5-9 Jan., 2015, pp. 1-13.

[21] Bollino, K. P. and Ryan Lewis, L., "Collision-free Multi-UAV Optimal Path Planning and Cooperative Control for Tactical Applications", *AIAA Guidance, Navigation and Control Conference*, Honolulu, HI, 21-24 Aug., 2008, pp. 1-18.

[22] Zammit, C. and van Kampen, E. J., "Comparison between A* and RRT Algorithms for UAV Path Planning", *AIAA Guidance, Navigation and Control Conference*, AIAA SciTech Forum, Kissimmee, FL, 8-12 Jan., 2018.

[23] Zammit, C. and van Kampen, E. J., "Advancements for A* and RRT in 3D path planning of UAVs", *AIAA Guidance, Navigation and Control Conference*, AIAA SciTech Forum, San Diego, CA, 7-11 Jan., 2019.

[24] Sujit, P. B., and Ghose, D., "Search by UAVs with Flight Time Constraints using Game Theoretical Models," *AIAA Guidance, Navigation and Control Conference*, San Francisco, CA, 15-19 Aug. 2005, pp. 1-11.

[25] Bethke, B., Bertuccelli, L. How, J. P., "Experimental Demonstration of MDP-Based Planning with Model Uncertainty," *AIAA Guidance, Navigation and Control Conference*, Honolulu, HI, 18-21 Aug. 2008, pp. 1-22.

[26] Bollino, K., Lewis, L. R., Sekhavat, P. and Ross, I. M., "Pseudospectral Optimal Control: A Clear Road for Autonomous Intelligent Path Planning," *AIAA Infotech Aerospace Conference and Exhibit*, Rohnert Park, CA, 7-10 May 2007, pp. 1-14.

[27] Frazzoli, E., "Maneuver-Based Motion Planning and Coordination for Multiple UAVs," *Proceedings of the AIAA/IEEE Digital Avionics Systems Conference*, Portsmouth, VA, 20-23 May 2002, pp. 1-11.

[28] Huang, Y., Chen, J., Wang, H. and Su, G., "A method of 3D path planning for solar-powered UAV with fixed target and solar tracking," *Aerospace Science and Technology*, Vol. 92, pp. 831-838, 2019.

[29] Benenson, R. Petti, S., Fraichard, T. and Parent, M., "Integrating Perception and Planning for Autonomous Navigation of Urban Vehicles," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China, 9-15 Oct. 2006, pp. 98-104.

[30] Yao, P., Wang, H. and Su, Z., "Real-time path planning of unmanned aerial vehicle for target tracking and obstacle avoidance in complex dynamic environment," *Aerospace Science and Technology*, Vol. 47, pp. 269-279, 2015.

[31] Short, A., Pan, Z., Larkin, Z. and van Duin, S. "Recent Progress on Sampling Based Dynamic Motion Planning Algorithms", *IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, Alberta, Canada, Jul. 2016, pp. 1305-1311.

[32] Karaman, S. and Frazzoli, E., "Sampling-based algorithms for optimal motion planning", *The International Journal of Robotics Research*, Vol. 30, No. 7, pp. 846-894, 2011.

[33] Singh, Y. and Sharma, S., "Optimal path planning of unmanned surface vehicles," *Indian Journal of Geo-Marine Sciences*, Vol. 47, No. 7, pp. 1325-1334, 2018.

[34] Ross, I. M., "A Unified Computational Framework for Real-Time Optimal Control," *Proceedings of the 42nd IEEE Conference on Decision and Control*, Maui, Hawaii, Dec. 2003, pp. 2210-2215.

[35] Gong, Q., Kang, W. and Ross, I.M., "A Pseudospectral Method for the Optimal Control of Constrained Feedback Linearizable Systems," *IEEE Transactions on Automatic Control*, Vol. 51, No. 7, pp. 1115-1129, 2006.

[36] Gao, X., Ren, J. and Chen, D., "Developing an effective algorithm for dynamic UAV path planning with incomplete threat information," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, Vol. 226, No. 4, pp. 413-421, 2012.

[37] Peng, X. and Xu, D., "Intelligent online path planning for UAVs in adversarial environments," *International Journal of Advanced Robotic Systems*, Vol. 9, pp. 1-12, 2012.

[38] Roberge, V., Tarbouchi, M. and Labonte, G., "Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning," *IEEE Transactions On Industrial Informatics*, Vol. 9, No. 1, pp. 132-141, 2013.

[39] Lavalle, S. M., "Motion Planning : The Essentials", *IEEE Robotics & Automation Magazine*, Vol. 18, No. 1, pp. 79-89, 2011.

[40] Kuwata, Y., Teo, J., Karaman, S., Fiore, G., Frazzoli, E., and How, J. P., "Motion Planning in Complex Environments using Closed-loop Prediction", *AIAA Guidance, Navigation and Control Conference and Exhibit*, Honolulu, HI, 18-21 Aug. 2008, pp. 1-22.

[41] Fernandes, E. Costa, P., Lima, J. and Veiga, G. "Towards an orientation enhanced astar algorithm for robotic navigation", *IEEE International Conference in Industrial Technology (ICIT)*, Amman, Jordan, 12-15 May, pp. 3320-3325, 2015.

[42] Trovato, K. and Dorst, L. "Differential A*", *IEEE Transaction on Knowledge and Data Engineering*, Vol. 14, No. 6, pp. 1218—1229, 2002.

[43] Palossi, D., Furci, M., Naldi, R., Marongiu, A., Marconi, L., Benini, L. "An energy-efficient parallel algorithm for real-time near-optimal UAV path planning", *Proceedings of the ACM International Conference on Computing Frontiers*, Como, Italy, 16-18 May, pp. 392-397, 2016.

[44] Ghandi, S. and Masehian, E., "Review and taxonomies of assembly and disassembly path planning problems and approaches", *CAD Computer Aided Design*, Vol. 67-68, No. October, pp. 58-86, 2015.

[45] Kuwata, Y., Teo, J., Fiore, G., Karaman, S., Frazzoli, E., and How, J. P., "Real-Time Motion Planning With Applications to Autonomous Urban Driving", *IEEE Transactions on Control Systems Technology*, Vol. 17, No. 5, pp. 1105-1118, 2009.

[46] Kunz, T. Reiser, U., Stilman, M. and Verl, A. "Real-time path planning for a robot arm in changing environments," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Taipei, Taiwan, 8-14 Oct. 2010, pp. 5906-5911.

[47] Stentz, A., "Optimal and Efficient Path Planning for Partially Known Environments", *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, San Diego, CA, 8-13 May, 1994, pp. 3310-3317.

[48] Petti, S. and Fraichard, T., "Safe Motion Planning in Dynamic Environments", *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Edmonton, Alta., Canada, 2-6 Aug., 2005, pp. 2210-2215.

[49] Gonz*à*lez, D., P*èrez*, J., Milan*è*s, V., and Nashashibi, F., "A Review of Motion Planning Techniques for Automated Vehicles", *IEEE Transactions on Intelligent Transportation Systems*, Vol. 17, No. 4, pp. 1135-1145, 2016.

[50] Sathyaraj, M. B., Jain, L. C., Finn, A. and Drake, S., "A Multiple UAVs path planning algorithms: a comparative study", *Fuzzy Optimization and Decision Making*, Vol. 7, No. 3, 2008, pp. 257-267.

[51] Hart, P. E., Nilsson, N. J. and Raphael, B., "A Formal Basis for the Heuristic Determination of Minimum Cost Paths", *IEEE Transactions on Systems Science and Cybernetics*, Vol. 4, No. 3, pp. 100-107, 1968.

[52] Tseng, F. H., Liang, T. T. and Lee, C. H. Chou, L. D. and Chao, H., "A Star Search Algorithm for Civil UAV Path Planning with 3G Communication", *10th International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP)*, Kitakyushu, Japan, 27-29 Aug. 2014, pp. 942-945.

[53] Lavalle S. M. and Kuffner J. J., "Randomized kinodynamic planning", *International Journal of Robotics Research*, Vol. 20, No. 3, pp. 378-400, 2001.

[54] LaValle S. M. "Probabilistic roadmaps for path planning in high-dimensional configuration spaces", *IEEE Transactions on Robotics and Automation*, Vol. 12, No. 14, pp. 566-580, 1996.

[55] LaValle, S. M. and Kuffner, J. J. "Randomized kinodynamic planning", *Proceedings of the IEEE International Conference on Robotics and Automation*, Detroit, MI, 10-15 May 1999, pp. 473-479.

[56] Devaurs, D., Siméon, T. and Cortés, J. "Optimal Path Planning in Complex Cost Spaces With Sampling-Based Algorithms", *IEEE Transactions on Automation Science and Engineering*, Institute of Electrical and Electronics Engineers, 2015.

[57] Geraerts, R. and Overmars, M. "Creating high-quality paths for motion planning", *International Journal of Robotics Research*, Vol. 26, No. 8, pp. 845-863, 2007.

[58] Hrabar, S., "3D Path Planning and Stereo-based Obstacle Avoidance for Rotorcraft UAVs", *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nice, France, 22-26 Sep. 2008, pp. 807-814.

[59] Ferguson, D. and Stentz, A. "Using interpolation to improve path planning: The field D* algorithm", *Journal of Field Robotics*, Vol. 23, No. 2, pp. 79-101, 2006.

[60] Yu, H., Beard, R. W. and Byrne, J. "Vision-based Navigation Frame Mapping and Path Planning for Micro Air Vehicles", *AIAA Guidance, Navigation, and Control Conference*, Chicago, Illinios, 11-13 Aug. 2009, pp. 1-10.

[61] Clifton, M., Paul, G., Kwok, N., Liu, D. and Wang, D. "Evaluating Performance of Multiple RRTs", *IEEE conference on Mechatronic and Embedded Systems and Application*, Bejing, China, 12-15 Oct. 2009, pp. 564-569.

[62] Paul, G. "Multiple Rapidly-exploring Random Tree (RRT)", *MATHWORKS*, [Online Database], https://www.mathworks.com/matlabcentral/fileexchange/21443-multiple-rapidly-exploring-random-tree-rrt-?requestedDomain=www.mathworks.com, [retreived 30 October, 2016].

4

3D Real-time Path Planning of UAVs in Dynamic Environments

Once a real-time path planning algorithm is constructed, tested and analysed in the previous chapter, Research Question 3, formulated to tackle Challenge 2, that queries the effect on performance if the static environment is changed into a dynamic one will be addressed in this chapter. This chapter initiates with a literature review highlighting the need of dynamic obstacle consideration in indoor environments and presents the state-of-the-art in dynamic obstacle modelling. Typical dynamic obstacles are modelled and four different scenarios with different difficulties are constructed. Using these scenarios, the effect on real-time path planning performance is assessed.

The contents of this chapter have been published as:
Zammit, C. and van Kampen, E., "3D real-time path planning of UAVs in dynamic environments", *Proceedings of AIAA Guidance, Navigation and Control*, Nashville, TN, 11-15 Jan., 2021, AIAA 2021-1955.

Part of this chapter will be submitted to:
Zammit, C. and van Kampen, E., "Real-time 3D UAV path planning in dynamic environments in the presence of uncertainty", *Journal of Guidance, Control and Dynamics*

Unmanned Aerial Vehicles (UAVs) are taking active roles in personal, commercial, industrial and military applications due to their efficiency, availability and low-cost. UAVs must operate safely and in real-time in both static and dynamic environments. An extensive literature review, defines the dynamic environment term, the need for dynamic path planning and reviews different solutions. This paper presents a 3D real-time path planning algorithm to assess the performance of the A* and RRT algorithms. Four test scenarios with varying difficulty are constructed consisting of V-obstacles, cubes and 2D planes moving at time-varying speed, direction and orientation. Two rationales to either wait or move further in the direction of the goal when an intermediate goal point is not available are considered. Results show that for both A* and RRT the moving variant case performs better especially in complex scenarios. RRT performs better in simple scenarios and complex scenarios at low speed while A* performs better at high speeds in complex scenarios. A success rate of over 95% is recorded for three scenarios for all considered speeds and for both algorithms.

4.1. INTRODUCTION

The ever increasing availability of different Unmanned Aerial Vehicles (UAVs) for a wide range of personal, commercial, industrial and military applications, is increasing the need for robust, reliable and autonomous guidance, navigation and control systems that must operate in real-time even within obstacle-dense environments. The environment in which a UAV is operating may contain fixed and/or moving obstacles that may vary in size, speed and orientation. Furthermore, these characteristics can change as the UAV is navigating to reach a pre-defined goal. Path planning algorithms are therefore required to navigate the UAV in such time-varying environment with the available computational, sensory and fuel resources available online.

In real indoor and outdoor environments, the UAV path planning algorithm must generate or update a path in real time to reach an intermediate or final goal position. These systems mainly rely on the real-time data of onboard sensory systems with their associated inaccuracies, latency and uncertainty to update obstacle sizes and positions, intermediate goal and current UAV positions.

In dynamic environments, the time varying obstacles properties may be known or unknown beforehand. In the latter case, the UAV can only rely on the real-time data of sensory systems. But in the former, the path planning algorithm can have apriori knowledge of future obstacle positions, for example automatically operated doors or gates. Furthermore, in the same environment in which the path planning algorithm is guiding the assigned UAV, other UAVs may be operating. Their current and future assigned path maybe fixed as their tasks would be very specific. For example, in a hospital environment delivering blood samples from the laboratory to the consultants' office is normal practice. In such cases a preassigned path maybe available or not to the path planning algorithm apriori or in real-time. Oppositely other UAVs operating in the same environment may operate independently or belong to enemies.

In our previous work [1, 2] and previous Chapters of this dissertation fixed obstacle 2D planes with tight windows were considered to assess the performance of the most utilised algorithms in the graph-based and sampling-based categories namely the A* and Rapidly-Exploring Random Tree (RRT) algorithms which will be briefly explained

in Section 4.3. The next natural step owing to the positive performance results of both algorithms is to assess their validity in the dynamic environments described earlier. A valid path planning system shall be able to ensure that the path to a goal position is *always* obstacle free. In this light, the path planning algorithm must be integrated within a real-time or online algorithm. Such system shall re-assess already re-assigned paths and if necessary re-evaluate to ensure that a path from the current UAV position to a goal position is obstacle free based on real-time environmental awareness whilst the UAV is traversing the respective path segment, if and only if a path exists. Such an assessment method was developed in our previous work [3] and will be briefly explained in Section 4.3.

The aim of this paper is to assess the performance and consequently the validity of the A* and RRT algorithms for real-time 3D UAV path planning in dynamic environments. For the scope of this paper a single agent path planning system with no *a priori* knowledge of the obstacles paths, including other UAVs, shall be considered. The path length, computational time and success rates will be the performance measures considered to assess the usability of both the A* and RRT algorithms in dynamic environments. In the context of this work, a dynamic environment will contain both fixed and moving obstacles with the latter having either a time-variant size, position, speed or orientation or any combination of these possibilities. Furthermore, a real-time path planner is required to generate a path in the same or less time than the time required by the UAV to traverse the same path [4–6].

The paper will be organised as follows. Section 4.2 will present the state-of-the-art in path planning in the presence of dynamic obstacles. Section 4.3 provides a brief resume of the A* and RRT algorithms, the smoothing algorithm (a post-path generation algorithm applied to the RRT algorithm) and the real-time path planning algorithm all extensively defined in our previous works [1–3]. Section 4.4 will define the obstacle generation algorithm based on pre-defined static and dynamic obstacle characteristics. Section 4.5 will describe the amendments to the developed real-time algorithm developed in our previous work [3]. This will be followed by Section 4.6 which will present and analyse the results based on pre-defined performance measures in view of real-time 3D UAV path planning. Finally, this paper will conclude (Section 4.7) with a resume of the performance results highlighting stengths and weaknesses of both algorithms and integrated system's applicability to real 3D UAV path planning in dynamic environments for both algorithms. Furthermore, Section 4.7 identifies future work that can be conducted to implement this system in a real UAV environment.

4.2. PATH PLANNING IN DYNAMIC ENVIRONMENTS REVIEW

4.2.1. INTRODUCTION

Autonomous path planning is the process of automatically generating a feasible path to the final goal point even in the presence of static and dynamic constraints, obstacles and threats (COT) [1, 7]. Therefore, a sound path planner must always guarantee that the UAV will reach the goal node and remain at the goal point unless given instructions otherwise in the presence of COTs and irrespective of sensing and control uncertainties [8]. Some path planners focused on path optimisation in an obstacle–free environment [9–11] but

in real situations the environment incorporates different COTs [12–14]. This review will analyse dynamic environmental modelling with its various obstacle definitions in the context of real–time 3D UAV path planning. It is not the scope of this work to review path planning of UAVs. For an extensive literature review on different path planning solutions refer to Zammit *et al.* [1].

4.2.2. DYNAMIC ENVIRONMENT DEFINITION

A dynamic environment may include a vast spectrum of static and time–varying constraints, obstacles and threats which can 'pop–up' whilst the UAV is navigating through such an environment [12, 15, 16]. These include wind, fuel, altitude constraints, flight profile requirements, UAV kinematic and dynamic holonomic and non-holonomic constraints, no fly zones, buildings, vehicles, changing weather conditions, control failures, moving targets, addition/removal of COTs, surface to air missiles, tanks, lane markings, irrigation system, other aircraft or UAVs and quality, loss or delay in communication between agents in a swarm setup [7, 15–25]. Not all COTs incorporate the same level of complexity. Goerzen *et al.* [8] defined obstacle–complexity in terms of obstacles edges and vertices, number of obstacles in a specific area and memory storage required for obstacle representation and acknowledged that is a commonly used metric. Fujimura [26] identified three categories of dynamic environments: Moving obstacles in a known environment, Static obstacles in an unknown environment and Moving obstacles in an unknown environment. A combination of two or even all situations requires also dynamic motion planning.

4.2.3. THE NEED FOR DYNAMIC PATH PLANNING

Path planning can be segmented into two main categories: global and local path planning [27–29]. A global planner generates a low-resolution, long range offline path based on a high fidelity situational awareness. This path planning strategy is inadequate for a time–varying environment or in situations where obstacles, threats or constraints are unknown *apriori*. In these situations the path is optimised before the execution of a path [27, 30]. Oppositely, a local planner creates higher resolution paths in a smaller environmental space based on real–time information from on-board sensors. This reactive path planning approach is ideal for dynamic environments [27, 30]. A hybrid approach is suggested to outweigh the weakness of one method with the strengths of the other [30–33].

A review on path planning over a span of 15 years (2000–2015) presented by Mac *et al.* [27] showed that in 49% of the path planning projects, the obstacles and targets were considered to be static while in only 18% of the projects considered a combination of static and dynamic obstacles. 11% considered a dynamic target with 9% of which considered adaptive UAV velocity [27]. So although different studies considered dynamic COTs, the majority considered the agent to move at constant speed [27, 34]. This highlights the need for path planning in dynamic environments, although path planning in dynamic environments is still considered a challenging aspect for researchers [27].

In a dynamic and/or unknown environment, an autonomous path planning algorithm must react and re–plan in real–time or over a predefined time window, utilising local onboard system information, a collision free path to goal based on new previously unknown COT without the assistance of an operator or guidance ground systems

[8, 12, 15, 16, 21, 24, 35, 36]. This *obvious* requirement [12], is a must for realistic path planning applications [37]. Such path planners are referred to as reactive path planners [8, 35, 36, 38]. According to *Goerzen et al.* [8] such planners are ideal to mitigate rapid time–variant environment but suffer in global planning problems finding difficulty in reaching the final goal position and therefore lacking in path length optimisation. Although the future position of obstacles can be estimated in certain situations, such planner is required to ensure non–collision in a given time window if these obstacles change state in an unexpected manner [39].

Ensuring safety in an unknown dynamic environment for a UAV operating in real–time is an important requirement [24]. Kuwata *et al.* [24] defined safety as a state into which a vehicle can remain indefinitely without colliding with static and moving obstacles or breaching constraints, assuming a constant heading by moving obstacles. In terms of safety, both control and obstacle avoidance systems must prove that they can reach the goal node in all possible vehicular and environmental variations [14]. Without these capabilities the use of unmanned vehicular systems will be highly restricted to time–invariant, *a priori* defined environments [14].

For realistic path planning the UAV must be equipped with an advanced sensory system for environmental sensing that shall be able to detect new COTs as they appear in the scenario space [12, 13, 16]. The sampling rate of such system is inversely proportional to the distance at which COTs must be detected [25]. Furthermore, the path planning system must be provided with appropriate COT representation and mapping through mathematical models that incorporate kinematic and dynamic characteristics of obstacles with minimal computational demand and a fast update rate to immediately detect changes in the environment in harmonisation with the onboard environmental characteristics database [21, 34, 35, 40]. This stresses the need for dynamic resolution that will be able to automatically adjust to handle different COTs at different distances from the onboard sensing system so as to make optimal use of computational resources while accurately defining obstacles eliminating the risk of leaving small passages between obstacles available [41]. Moreover, COT representation shall make use of approximations to mitigate memory and computational power limitations [8]. The control algorithm must also react simultaneously with the path planning algorithm to avert collisions, stressing the need for an online implementation [36]. These requirements highlight the difficulty for a UAV to operate in unknown areas. In fact, Dittrich *et al.* [12] remarked that the best solutions have yet to be found.

4.2.4. ENVIRONMENTAL ASSUMPTIONS

It is assumed that the environmental space is made up of two disjoint subsets of either free space or obstacle space [8]. Different research works consider either that the environmental characteristics including obstacle geometry and locations are known *a priori* [15, 42] or oppositely no *a priori* knowledge of the environment is provided to the path planning algorithm [13, 24, 43] or a combination of both of the above [16, 35, 44].

In the first case, COT information is provided as the exploration area expands due to moving target [15]. In the second case, the path planner can initially assume an obstacle–free environment with obstacles known only within a finite receding–horizon, re–planning a new non–colliding path when a new COT is detected from sensory sys-

tems that violates the previously generated path [13, 21, 24, 43]. This option is ideal for surveillance, moving targets and high time–variant environments [24]. In the third case, paths are generated offline based on known environmental COT, triggering reactive re–planning in the eventuality of randomly–generated COT pop–ups that will result in a collision or non–optimal paths if the offline generated path is not updated [12, 16, 35, 41, 44–46]. In high time–variant dynamic environments the planner has to re–plan frequently demanding more computational power consequently limiting the ability of the path planning algorithm to be applied online.

4.2.5. CONSTRAINTS, OBSTACLE AND THREATS (COT) SENSING AND MODELLING SYSTEMS

Computational resources are very limited in real–time implementations and their efficient use is key for realistic implementations [35]. Therefore, COT modelling for dynamic re-planning in dynamic environments require efficient storage, fast addition and deletion and fast and efficient collision checks [40]. These requirements depend also on obstacle density which is dependent upon the environment into which the UAV will operate. For example, Koenig *et al.* [47] considered an obstacle density of 30%. Amin *et al.* [35] remarked that COT representations for unmanned vehicles in 2006 were in the majority not suitable for real–time implementations such as [48, 49] and suggested that systems developed for the video gaming industry shall be considered.

Obstacle representation formats can be classified into 3 main categories: (1) Raw Data (e.g. vertices-edges sets); (2) Bounding Volumes (e.g. Oriented Bounding Boxes (OBB) and their variants) and (3) Spatial Partitioning (e.g. Grids, Quadtrees and Octrees) [35, 50].

RAW DATA

Stereo vision was used to build 3D occupancy maps that allowed dynamic re–planning to be realised [51]. Dittrich *et al.* [12] utilised also a stereo camera based system to construct polygonal lines or circles with a minimal safety distance from each detected obstacle. Besides stereo vision, Scherer *et al.* [52] and Shim *et al.* [53] utilised laser range finders for obstacle detection and re-planning. Owing to the weight of laser range finders, Hrabar [51] remarked that stereo vision offers the best solution for small UAVs although such obstacle detection system shall be capable of defining absolute range measurements.

Similarly, a depth map can be constructed using computer vision algorithms to define the time to collision estimates [43]. Consequently, an obstacle map can be built in local frame of reference as a depth map can provide the range to the obstacle derived from the airspeed and bearing to obstacle from the UAV current position [43]. Furthermore, Ya *et al.* [19] constructed a 500x500 cell environment utilising a variable probability function (0% to 50%) that randomly places obstacles within each cell.

BOUNDING VOLUMES

Gottschalk *et al.* [54] modelled COTs using OBBs for fast and reliable dynamic collision identification in a 2D environment. Similarly, Gros *et al.* [21] utilised the same obstacle modelling technique with time-invariant COTs of fixed radius of 150m. Results in a high density COT environment showed that real–time can be achieved with a Finite Receding-Horizon Incremental-Sampling Tree (RHIST). Similarly, Ögren *et al.* [55] considered a

constant obstacle margin of 15m was considered although in sparse environments this margin was increased to 50m.

Foo *et al.* [7], defined a threat zone as a sphere for above ground and a hemisphere for a ground vehicle of user–defined radius surrounding an obstacle. A radius of 20,000ft. was arbitrary selected for such threats. The radius tolerance was reduced by a factor of 4 for non-enemy UAVs [7]. Similarly, Srikanthakumar *et al.* [14] considered spherical obstacles with a safe radius greater than the actual radius by 25% and a variable influence range with a repulsive potential to divert the UAV from obstacles. Also, collision cones are constructed for every obstacle which is also surrounded by a safety ball of 5m–20m radius [36]. Similarly, Ok *et al.* [56] defined a low-level planner that adds repulsive forces to obstacles governed by a higher level Generalized Vonoroi Diagram to construct a collision–free path in the presence of uncertainty.

Bollino *et al.* [15, 57] defined obstacles by simple shapes including squares, diamonds, circles, ellipses and rectangles using the p-norm formulation. Similarly, to Foo *et al.* [7] and Srikanthakumar *et al.* [14], Bollino *et al.* defined a distance–dependent buffer to mitigate uncertainty in obstacle modelling [15]. Such buffer increases with increase in distance to the obstacle from the UAV so as to emulate the uncertainty increases as the UAV moves away from obstacles [15, 58]. Similarly, Likhachev *et al.* [44] considered a buffer zone around obstacles with high costs that required the planner to stop and conduct 90 degree turns to avoid obstacles.

Adolf *et al.* [13] considered an incremental heuristic path planner for *a priori* known COTs and a reactive anytime path planning algorithm for incremental obstacle pop–ups which were defined by 3D non–convex polyhedral surfaces. A 3D voxel grid was used to index and update polygonal pop-up COTs. The helicopter model was considered to be holonomic and the 3D path planning algorithm exhibited exponential runtime complexity [13]. Consequently, anytime planners may lack in constantly generating new global paths whenever the environment changes [44]. Similarly, Lee *et al.* [16] designed static obstacles using polygons, moving obstacles using rectangular no–fly zones and pop–up threats as circles with the latter designed using the Markov Chain method. Through a Model Predictive Control method the agent was able to manoeuvre in a 9 pop–up threat environment with 5 targets and 5 static obstacles. Also a first order Markov Chain was developed in [45, 59] to generate the sequence of pop–ups.

SPATIAL PARTITIONING

Visibility maps build lines with all possible vertices from the agent position by considering that the shortest path touches polygonal obstacles at their vertices [8]. The Edge–Sampled Visibility Graph, a variant of the primary method, defines a minimum edge length to build a visibility graph by assigning multiple vertices along edges of polyhedral obstacles [8]. Kim *et al.* [60] proposed the Quantized Visibility Graph (QVG) to model polygon shaped obstacles. In a hybrid approach, the Freeway Method used bounding cylinders around obstacles to build a map of lines. Results show that it is not limited to 2D but is incomplete and non–optimal [8]. Oppositely, the Silhoutte Method is complete for any obstacle geometry with any dimension [8].

The visibility method in conjunction with a sparse uniform space sampling algorithm was developed by Tsardoulias *et al.* [61] to model COTs. Sampling–based methods

do not require explicit construction of obstacles when opposed to more deterministic approaches reducing computational time [62]. Results showed that the A* algorithm for 3D path planning in combination with visibility graph for polygonal obstacles modelling did not guarantee the shortest paths [63]. Moreover, Visibility and Voronoi diagrams can only generate shortest paths with optimal clearances in low–dimensional spaces with only polygonal obstacles [64, 65]. Voronoi diagrams were mainly used for static obstacles although Roos *et al.* [66] added bounds to Voronoi channels to mitigate dynamic COTs in 2D environments. Such approach can be extended for 3D environments [67].

A Quadtree data structure was utilised by Amin *et al.* [35] for obstacle representation. This was integrated with a modified RRT algorithm for path planning. Another variant is the grid map decomposition technique that offline defines obstacle–free rectangles and replaces all nodes within with edges so as to improve path optimisation [68].

4.2.6. SOLUTIONS

Numerous researchers have tried to use classical approaches such as classical algorithms such as cell decomposition, potential field, sampling–based and sub-goal networks to achieve real–time path planning that can then be applied for dynamic re-planning [27, 69–71].

Graph–based approaches such as the Sparse A* Search path planning algorithm is also a potential candidate for path planning in dynamic environments [72–74]. To mitigate the static nature of the A* algorithm, a mechanism named the Virtual Force was proposed for dynamic re–planning [75]. Likhachev *et al.* [44] successfully implemented an A* based anytime algorithm in a time–variant obstacle where targets are randomly generated. The trajectory planner was tested in a 500mx500m area with a constant speed of 5m/s. Trajectories included both parking, reverse manoeuvring in a dense environment.

Sampling–based methods were also proposed by a number of researchers for dynamic re-planning. Such methods are considered due to their asymptotic optimality, efficiency although they cannot guarantee an optimal solution [27, 76, 77]. Hsu *et al.* [39] developed an online Probabilistic Roadmap (PRM) that generates a new path in a predefined time window when obstacle motion differs from estimated during execution. Similarly, Otte *et al.* [78] developed an asymptotically optimal re–planning algorithm (RRT^X) that updates a goal rooted tree when new obstacles are detected. During re–planning the environment is assumed static.

Heuristic methods, such as Artificial Neural Network, Fuzzy Logic and Nature–inspired Methods can also generate optimal path planning in uncertain, partially unknown and dynamic environments [27]. However, these methods require a learning phase and high computational demand. The latter is highly limited in real–time applications especially for small UAVs [27].

Different researchers model various different static and dynamic environments with different degrees of complexity. Hrabar [51] defined a 40mx10mx6m environmental space furnished with three 4m poles, a 1.5m wall and a rectangular obstacle. It was concluded from this study that on average 0.15s were required to re-plan a new non-colliding path when new obstacles were detected. Bollino *et al.* [57] considered a 32 obstacle 2D environment into which the planner was allowed to travel backwards, solving problems that otherwise would lead to no solutions. U-turns and backwards movements were also

considered in [25].

A different approach is the velocity obstacle concept that assumes known velocities of obstacles and considers a range of possible agent velocities based upon a predefined maximum acceleration to generate collision paths offline [12, 79, 80]. This concept assigns discretises velocities and associated costs based on the vicinity to nearby velocity obstacles [12, 79]. Similarly, Ögren *et al.* [55] considered a variable agent speed with a maximum acceleration of $30 m/s^2$ and a maximum speed of 100m/s with moving obstacles at constant direction and speed of 30m/s. Results show that by increasing horizon lengths performance is enhanced for both static and moving obstacles. Others developed the Directional Priority Sequential Selection [81] and Predictive Trajectory Planning algorithms [38] for 2D reactive trajectory planning in dynamic environments.

In the majority of the studies constant speed scenarios are considered [34]. For inspection purposes, UAV typically fly at low speed, lower than 5m/s or just hover in situations when they are acquiring images [13, 51]. Such situation differs from high altitude problems as the agent is operating close to ground [12].

The pre–defined time window requirement of reactive obstacle avoidance was defined by Chawla *et al.* [36] at 4s to 8s using a partial integrated guidance and control approach using a real six–DOF model that executed in a single loop. It must be pointed out that such parameter increases significantly from 2D to 3D. In fact, path planning in 2D requires polynomial time while in 3D the solution is NP–hard for polygonal and polyhedral obstacles respectively [82, 83].

Bohren *et al.* [25] developed a sensory system that is able to provide a sensing range varying from 4 to 60m using 90 degree field of view sensors at a rate of 10Hz. Results in a 300x300m map showed that vehicles were detected out to 60m with an accuracy of 1m/s and ground points within a 30m range in good conditions and depending on ground reflectivity. Furthermore, Benjamin *et al.* [84] implemented wireless communication to provide the agent with real–time 2D obstacle information for real–time obstacle avoidance in marine Unmanned Surface Vehicles (USVs). In the same field, Larson *et al.* [42] developed real–time obstacle avoidance using the projected area method.

Different performance measures are used to assess the validity of different path planning algorithms. In our previous work, path length and computational time were considered [1–3]. Besides these two parameters in dynamic environment the clearance criterion i.e. the minimum distance from the UAV to the COT was considered as a performance measure [14, 36]. Simulations show that at least five times the radius of a surrounding ball around the obstacle is required for safe operation [36]. Furthermore, another performance measure is the deviation to the goal point for the mission to be successful. In this regard, Chawla *et al.* [36] considered a maximum deviation of 0.5m radius around the goal position.

4.2.7. CONCLUSION

This extensive literature review initiated with an overview of what different researchers considered as a dynamic environment. Such environment constituted time–invariant and time–variant COTs that are either known, partially known or unknown to the path planner *a priori*. The need to generate a reactive real–time path in such an environment was discussed in view of realistic UAV applications. This highlighted the importance of

the UAV to be equipped with all the reliable sensory systems and adequate computational power. Then the review discussed the different COT sensing and modelling systems considered by different researchers to best represent the dynamic environment in view of computational demand limitations and efficient path planning. Finally, the most promising path planning solutions proposed by different researchers in 2D/3D dynamic environments in different dynamic environments emulating reality even with USVs and robots as agents were reviewed and discussed. This review will form the basis of our 3D real–time path planning in a dynamic environment that will be discussed in the next sections.

4.3. A*, RRT, SMOOTHING AND REAL–TIME ALGORITHMS

4.3.1. INTRODUCTION

This section will briefly describe the most utilised graph–based and sampling–based methods, namely the A* and RRT algorithms. Then the smoothing algorithm employed to smoothen the non–optimal path generated by the RRT algorithm will be explained. This section will conclude with a resume of the implementation of both path planning algorithms emulating real–time situations.

4.3.2. THE A* ALGORITHM

Graph–based methods define the state space into an occupancy grid defining obstacles residing in grid points as inaccessible points. Based on the free grid points, the graph–based algorithms check whether a path connecting the start and goal position exists [85]. Graph–based algorithms only offer a guarantee of solution if an adequate resolution is selected [77].

The standard A* algorithm uses a heuristic evaluation function ($f(n)$) to determine the cost of neighbouring grid points [22]. This evaluation function sums the cost from the current position to a prospective future position and the cost from the latter to its goal node [22, 86]. For a detailed explanation of this algorithm refer to our previous work [1–3].

4.3.3. THE RAPIDLY–EXPLORING RANDOM TREE (RRT) ALGORITHM

Sampling–based methods generate a path by connecting unevenly selected obstacle–free points in the configuration space [37, 77]. As opposed to graph–based methods, such algorithms can generate a path within infinite time provided that a path exists [77].

The standard Rapidly–Exploring Random Tree (RRT) constructs a unidirectional tree by randomly selecting obstacle–free points. A new tree branch is defined a predefined distance from the nearest point on the tree if a direct path to the latter does not collide with an obstacle. A path is formed when one of the tree branches reaches the goal node and another connects to the start point [87–89]. Such algorithm is efficient in complex high–dimensional environments although the non-optimal path generated by this algorithm may require smoothing [64, 89, 90]. As for A* refer to our previous work for a more detailed explanation of the RRT algorithm and its variants [1–3].

4.3.4. THE SMOOTHING ALGORITHM

The smoothing algorithm randomly selects two path points and then randomly defines two points on the path segment connecting the formerly selected points and their respective next path points. If the interconnection of the latter two points results in an obstacle–free line, then all points between these two points are neglected from the path. This process is repeated until the percentage path length reduction over the last 20 iterations is less than 1%. Please refer to our previous work for a more detailed explanation and for assessing the algorithm's effect on path planning performance [2, 3].

The smoothing algorithm is developed to target the non–optimality of paths generated by the RRT algorithm. Oppositely, the A* algorithm generates the shortest available path based on the considered resolution. Therefore the smoothing algorithm can only improve the path by eliminating the grid and assumes that each point in the free space can be used to smoothen the path. From our previous work [1, 2] it was concluded that the improvement is marginal. Furthermore, the implementation of the A* algorithm in real–time situations can lead to non–colliding smoothed path points which are very near to obstacles or reside on obstacles due to different frame of reference in the next iterate when the start point is moved further into the path. Such situation can result in a collision.

4.3.5. THE REAL–TIME ALGORITHM

As remarked earlier, real–time path planning is fundamental for a UAV to reach the final goal position in a dynamic environment [57, 91, 92]. In this light, an algorithm to emulate real–time behaviour was developed in our previous work [3].

In a nutshell, this algorithm defines an obstacle–free intermediate goal point in the direction of the final goal a predefined distance from the current UAV position based upon the sensory system's range and Field of View (FOV). An intermediate path, if possible, is generated by the A* or the RRT algorithm from the current position to the intermediate goal point. Consequently, the UAV's new position is defined a predefined distance along the generated path. This distance is selected based upon the UAV speed and maximum allocated intermediate time and assuming that actuator systems are defined with high fidelity and the UAV is not affected by external factors. Owing that the time needed for the UAV to travel this arbitrary distance is very low it was assumed that the environment will remain static in this time frame. The algorithm has a two layer time limitation one to generate an intermediate path and another for the total duration of the mission. Review our previous work [3] for a thorough explanation, parameter definition and assignment and performance results with both A* and RRT algorithms.

4.3.6. CONCLUSION

This section provided an overview of the algorithms that are utilised to generate a real–time path in a dynamic environment. The A* and RRT algorithms formed the backbone of the path planning algorithms. The smoothing algorithm is considered to mitigate the non–optimality of the RRT algorithm. Finally, the real–time algorithm provided a method to assess the performance of both path planning algorithms in situations derived from real life UAV applications.

4.4. THE OBSTACLE GENERATION ALGORITHM

4.4.1. INTRODUCTION

The scope of this algorithm is to serve as a generic method where different obstacles derived from real–life situations but not only are modelled with custom user–defined fixed and time–variant characteristics to assess the validity of the holistic real–time path planning algorithm. Although in this case the A* and RRT algorithms are considered, this algorithm is modular enough to test any path planning algorithm using either the previously developed real–time algorithm or any other algorithm.

The initial and future environments are estimated before the initiation of the real–time path planning tests. In real–time path planning the environment is estimated by the sensing and modelling system which is independent of the path planning system. The computational demand required by the sensing and modelling system is beyond the remit of this work and may vary depending upon the software and hardware utilised. The maximum computational time to generate an intermediate path was selected in view of the UAV speed, resolution and environmental size.

4.4.2. THEORETICAL RATIONALE

As derived from literature, in real–life situations every obstacle can be approximated by a regular shape [7, 12, 14, 21]. The algorithm initiates with retrieving from a predefined file the shape and size of each obstacle that will be present initially and at a future time in the environment. The obstacle position in the obstacle characteristic file was initially set at the middle of the environmental space. Each shape is modelled through a finite number of planes which are interconnected to form closed or open shapes. This shape is then replicated for a predetermined number of times and each copy is randomly placed in the environmental space. In case that also rotation is considered, a rotation by a random value different for all 3 dimensions about a random line is performed for each replica.

The next step is to differently shift and if requested rotate each of the generated replicas by a random distance in a random direction for a finite number of times. In real terms, this implies variable speed, roll, pitch and yaw obstacles. Each time all obstacles within the environmental space are shifted implies that a predetermined amount of time has elapsed. This time is equal to the time required by the UAV to traverse from the current point to a new point on the intermediate path. Each environmental space is allocated a time stamp to harmonise with the real–time algorithm. Obstacles are allowed to move in and out of the environmental space and to combine into one another so as to emulate real–life situations. The distance moved by obstacles is required to be less than the distance to be covered by the UAV in the same time as otherwise the UAV will not be able to avoid moving obstacles. This requirement is essential for randomly moving obstacles whilst for non-random motion, a model can be estimated and even fast moving obstacles can be avoided.

4.4.3. IMPLEMENTATION

In the implementation three shapes were considered, namely a cube, the V obstacle and 2D planes with small windows. The characteristics of each are tabulated in Table 4.1. Modular dimensions with no units are considered to offer a direct scaling with respect to

the environment volume. Four different scenarios with increasing complexity are considered, namely:

- Scenario 1. 10 cubes of 0.1x0.1x0.1 with no rotation;
- Scenario 2. 10 cubes the same size as Scenario 1 but with random rotation at definition stage and with changing independent rotation with time iterates;
- Scenario 3. 10 V obstacles constructed by adjoining one side of each of the two planes with an angle of 53° between the planes. Each plane has a size of 0.1x0.112 and is randomly rotated as in Scenario 2. This plane size was considered so that it fits exactly into the considered cube; and
- Scenario 4. Two planes on the Y–Z axis separated by 0.4 each with a window of 0.2x0.2 as well as the obstacles in Scenarios 2 and 3 combined.

Figure 4.1 illustrates each scenario for a random time iterate. The positions of each obstacle for each scenario will change every iterate as described earlier.

Table 4.1: Obstacle shapes

Shape	Size	Number of Planes	Closed/Open
Cube	0.05x0.05	6	Closed
V obstacle	0.05x0.05	2; (90^o with each other)	Open
Plane with window	1x1	1; window (0.05x0.05)	Open

4.4.4. CONCLUSION

This section provided an overview of the rationale and implementation of the dynamic obstacle generation algorithm. This modular algorithm was designed to emulate real–life situations and the user can design the environment based on the already considered obstacles or new ones. Furthermore, the number, characteristics, rotation and movement can be individually defined in a time–variant nature. In flight pop-ups and real–time obstacle elimination can also be modelled.

4.5. ENHANCEMENTS TO THE REAL–TIME PATH PLANNING ALGORITHM

The real–time algorithm thoroughly explained in [3] was adapted to integrate the obstacle generation algorithm described in Section 4.4. Table 4.2 adapted from our previous work [3] tabulates the nominal parameter values that are considered as constant for the scope of this paper. Each parameter is defined in a modular way based on the rationale described in our previous work [3]. All parameter assignments considered a 1x1x1 environment. Furthermore, for A* the environment and start and goal points were shifted by a random distance between 0 and half the distance between grid points to eliminate path length ripple as thoroughly explained in [2].

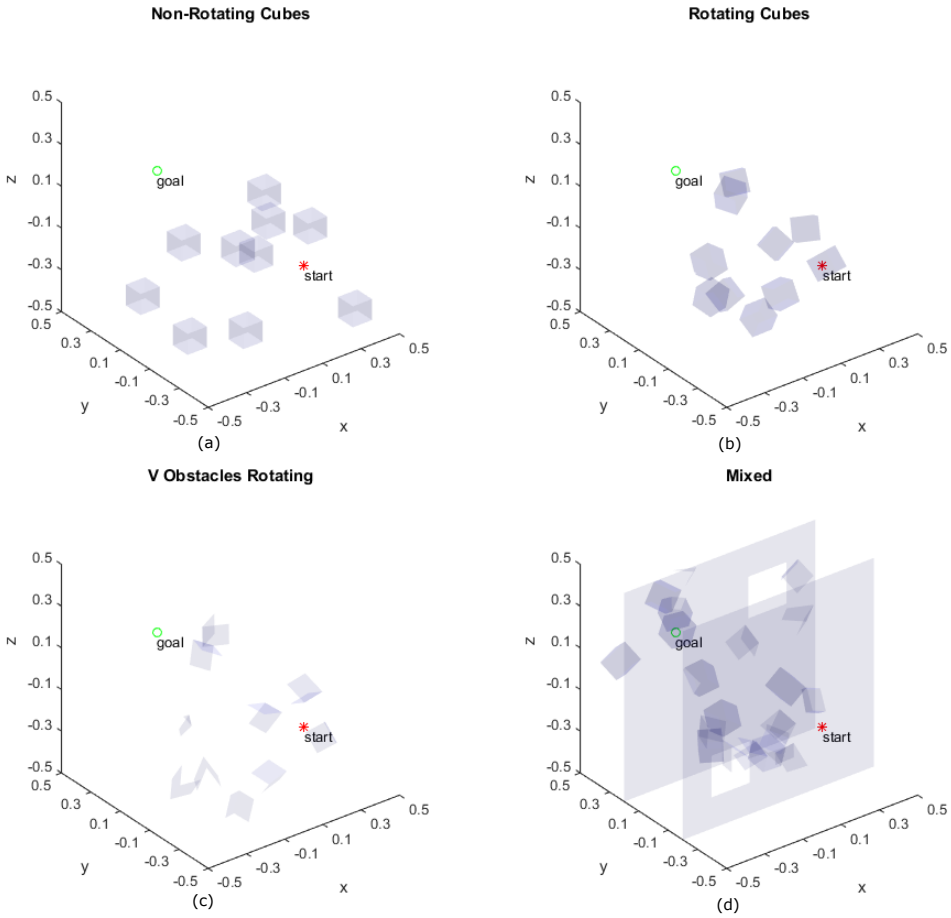


Figure 4.1: Environmental scenarios: (a) Non-rotating cubes scenario, (b) Rotating cube scenarios, (c) V obstacle rotating scenario and (d) Mixed scenario. These scenarios incorporate cubes, V obstacles and obstacle planes in Y-Z with windows as openings.

The resolution is defined as the amount of grid points per dimension. The RRT is a sampling–based method and therefore every point in the available space is available. The limitation is the distance the planner can move during intermediate path construction in the generated path direction. The distance to travel per iterate and the distance between the current UAV position and an intermediate goal point are denoted by d_{s_step} and d_{int_goal}, respectively. The maximum time to generate an intermediate path and the maximum time to reach the goal from start position are denoted by $t_{iterate\_max}$ and $t_{path\_gen\_max}$, respectively. Finally, the distance reduction factor (d_{factor}) is considered to reduce the distance to the intermediate goal point and the new UAV position as the latter may reside on an obstacle.

The UAV speed remains constant during the traversal of the path but is varied between one test and another for both algorithms and all scenarios. For the analysis this modular parameter is varied between 0.01[-]/s and 0.1[-]/s in steps of 0.01[-]/s, where [-] represent modular distance units. A 5000 × 5000 × 5000 distance unit environment was considered. As described in [3], these values were determined based on the nominal speeds for exploration situations in a nominal environment.

Table 4.2: Real–time algorithm parameter definition

Parameter	Nominal Value	Units
Resolution (res)	21	[-]
Step size RRT (d_{step_RRT})	$\frac{1}{21-1} = 0.05$	[-]
Distance to travel per iterate (d_{s_step})	$\frac{2}{res-1}$	[-]
Distance between current UAV position and prospective new intermediate goal point (d_{int_goal})	0.4 and 0.6 for Mixed case scenario	[-]
Maximum time to generate path segment ($t_{iterate\_max}$)	$\frac{d_{s\_step} \times 60 \times 60}{100 \times v_{UAV}}$	s
Maximum time to generate path ($t_{path\_gen\_max}$)	$10 \times t_{iterate\_max}$	s
Distance reduction factor (d_{factor})	0.8	[-]

Once the above parameters are defined, the environment is generated *apriori* for a predefined number of times based on the expected amount of iterates required which varies significantly for different path planning algorithms, environmental scenarios and random sequence that is different for different iterates but initiates using the same random seed for both A* and RRT tests. Therefore, the number of environmental generations was defined with a large margin.

Unless the intermediate and total path generation time is less than $t_{iterate\_max}$ and $t_{path\_gen\_max}$, respectively the real–time algorithm can continue searching for a path to intermediate goal otherwise no path is possible. In a nutshell, during this searching process, the algorithm loads the respective environment, assigns a new intermediate goal point, generates a path from the current UAV position to goal if possible and moves the UAV to a new current position on the generated path.

The new intermediate goal point is determined by d_{int_goal} in the direction of the final goal provided that the selected point does not reside on an obstacle otherwise d_{int_goal} is reduced by d_{factor}. If although an intermediate goal point is available but

a path cannot be constructed, the algorithm reduces d_{int_goal} by d_{factor} to increases the chances of generating a path. This process is repeated until the intermediate time exceeds the maximum allowable intermediate time. Another option that is considered is to wait at the current UAV position until the intermediate goal point is available. This waiting process is halted if no solution to the intermediate goal point results in the allowable maximum intermediate time. Both solutions will be assessed and the results discussed in Section 4.6.

A similar approach was considered in defining a new UAV position based on the constructed path using either the A* or RRT algorithms to the intermediate goal point. Although the intermediate path was obstacle free when constructed, in the next iteration an obstacle may have moved in the path line. Therefore, the algorithm must re–check that the path line d_{s_step} distance from the current UAV position has remained obstacle–free. Otherwise, the algorithm will need to move the UAV $d_{s\_step} \times d_{factor}$ distance on the path. In case, the path segment in consideration remained non–obstacle–free the expression $d_{s\_step} \times d_{factor}$ is further re–multiplied by d_{factor}. This process is repeated until a non–colliding path segment to new UAV position is found or the maximum intermediate time has been exceeded. Only in the case of A*, a movement less than the resolution can yield a no movement whatsoever, resulting in a waiting phase for the UAV.

In contrast with our previous implementation [3] that only considered 2D planes, this implementation considered also open and closed 3D obstacles. 3D obstacles present a new situation that makes it more difficult to determine whether the obstacle is closed and therefore points within the obstacle are unavailable or opened from one part or another of the 3D obstacle, implying that points within are freely available. This issue was mitigated by checking that each point is not smaller than the maximum and not larger than the minimum of coordinate in each plane for every dimension. In this case, the point will reside inside the closed 3D obstacle. Besides this, for A*, a safety margin of half the distance between grid positions was also considered.

4.6. RESULTS

4.6.1. INTRODUCTION

The whole algorithm described in the previous sections was implemented in MATLAB and simulations were computed using an Intel Xeon ES–1650, 3.2GHz. The path length, computational time and success rate are the performance measures considered. Each constant parameter tabulated in Table 4.2 was assigned the values tabulated in Table 4.2 whilst the UAV speed was varied as described in Section 4.5.

4.6.2. A* RESULTS

Figure 4.2 illustrates the performance results of the A* algorithm for the two cases described earlier that consider two contrasting rationales when a path to the intermediate goal point is not available. In the first case, the real–time path planning system waits in its current UAV position until a path to the intermediate goal point is available (A* waiting). In the second case, the real–time path planning system defines a new intermediate goal point a shorter distance (governed by d_{factor}) towards the assigned UAV position (A* moving).

The path lengths for A* waiting and A* moving are similar for the first three scenarios for all speeds considered although the mean of the A* waiting instances results in a longer length of 0.6%, 0.1% and 0.3% as compared to A* moving for the first three scenarios, respectively. This shows that by moving nearer to the obstacle there is the possibility of finding shorter paths for the same environment. Another interesting point is that the path length for the cube with no rotation is always less than the rotation case for the moving algorithm. Oppositely, the waiting algorithm shows situations were the two lengths are equal or even the rotating case is shorter with respect to the non–rotating case. By nearing in the vicinity of obstacles, the planner can make optimal use of the non–rotating factor creating a shorter path. By waiting for the obstacle to clear for the prospective intermediate goal position, being a rotating or a non–rotating obstacle will make lesser of a difference in terms of path length.

Overall, the shortest path length was recorded for the non–rotating cubes followed by the rotating cube and the V–obstacle cases. Rotating objects virtually occupy a larger volume than their actual size as opposed to non–rotating equivalent objects resulting in longer paths as their effect on the generated path can be larger once their orientation and position changes. Although the V–obstacle, being an open obstacle with 2, 2D planes occupies a lower volume with respect to the cube cases, it may result in shorter paths, but this is not the case as confirmed from the results. The definition of the V–obstacles in a graph–based environment is dependent upon the resolution. For the considered resolution, inside the V–obstacle the distance between the planes is at some parts (more than 50%) lower than the distance between grid positions (a buffer of half the distance between grid positions is considered for all planes). Also, the V–obstacle is rotating with each plane larger than each side of the obstacle. These two factors combined consequently increase the path length with respect to the cube cases. Speed for both cases considered had no effect on the path length. This is attributed to the fact that irrespective of speed the planner allocates the shortest sub–path. With higher speeds the UAV will travel this sub–path faster consequently not effecting path length at all.

For the Mixed cases, the waiting algorithm resulted in 0% success rate for the majority of the speeds considered with a maximum of 2% at 4 other different speeds. So although from the successful runs it can be deduced that the path length increases by approximately 1.5 times with respect to the first 3 scenarios, statistically the successful sample is low to draw conclusions. From the moving case results, with an average success rate of over 50%, the same conclusion as for the waiting case can be drawn. Owing to the difficulty and lack of free space in the mixed scenario case, the planner has to traverse a larger portion of the environmental space to reach the goal position resulting in 1.5 times path length with respect to the other scenarios. With increase in speed a drop in path length is recorded for the moving situation. This does not imply that at higher speeds the path length decreases but results since the higher the speed the lower the success rate due to less intermediate time allocation. Therefore the best performing iterations as speed increases are considered in the path length analysis since the others lead to the maximum intermediate time violation.

The computational time for A* moving for all scenarios for all speed considered is longer by 2.31, 2.27, 6.32 and 7.19 times with respect to A* waiting for the non–rotating, rotating, V–obstacle and Mixed cases respectively. The speed has no effect on compu-

tational time for both A* waiting and A* moving for the reasons described earlier. If the mixed case scenarios are neglected due to low success rate for the A* waiting algorithm, results show that by waiting for the obstacle to clear the UAV will reach to a solution faster. So if computational demand is the bottleneck and the scenario complexity is in line with the first three scenarios, it is not worth the risk of nearing in the vicinity of an obstacle as the success rate is in line with A* moving.

The maximum intermediate time allocated for the first three scenarios is absolutely more than enough to generate an intermediate path. The lowest computational time is recorded for the V–obstacle. It is only 8.0% and 21.8% of the non–rotating cube case for the A* waiting and moving respectively. The V–obstacle scenario constitutes the lowest obstacle restricted case since each obstacle consists of only two planes, therefore it is easier to define the obstacle nodes in the graph space and check for collisions. The obstacle avoidance sub–routines are the most computationally demanding parts of the algorithm. Therefore, the inclusion of 3D obstacles will increase the computational burden since besides a larger number of planes from 20 (V–obstacle) to 60 (cubes) the algorithm need to model and check for collisions inside the cubes. Also from Figure 4.2 it can be deduced that obstacle rotation increases computational time by 5.1% and 3.3% for A* waiting and moving, respectively. This is attributed to the rationale described earlier that with rotation the unavailable space changes in orientation between one iterate and another causing more adaptions in the construction of the new intermediate path. For the mixed case in the A* moving algorithm, a larger mean and variance is recorded for the low speed situations. This is attributed to the higher success rate for low speed situations due to the higher maximum intermediate time allocation.

For A* waiting the average success rate is 100%, 99.8%, 97.4% and 0.6% while for A* moving the success rate is 99.7%, 100%, 98.7% and 66.2% for Scenarios 1 to 4, respectively. The results show that A* moving outperforms A* waiting in the mixed case scenario with an almost equal success rate close to 100% for the other scenarios. The difference in success rate between the variants increases with the complexity of the scenario. This can be attributed to the fact that the A* moving algorithm takes more risks in approaching an obstacle with a greater chance of finding an available grid position behind the obstacle while the A* waiting case is more vigilant and waits until a path is available within its safe zone. Although the difference in success rate for the first three scenarios is minimal this does not imply that by adding rotation the chances of finding a path to goal remain the same. Results of success rate only show that the allocated maximum intermediate and total times are more than required for both. So the difference is not visible. The rotating factor success rate reduction cannot be neglected if more stringent times are applied, as is deduced in the computational time analysis.

Furthermore, the success rate is independent of speed for both variants for all scenarios except for the mixed case in A* moving. With lower speeds the planner is allocated more time to find a path and therefore the chance of finding one is higher. This trend is not shown in all cases except for the A* moving in the mixed case since the success rate is saturated. For the A* moving mixed case a drop in success rate is visible mainly due to the lower maximum intermediate time allocation. The main cause of unsuccessful runs in both A* waiting and moving is the lack of intermediate time. A* moving success rate results also in some total time violations for the mixed case as the path length is

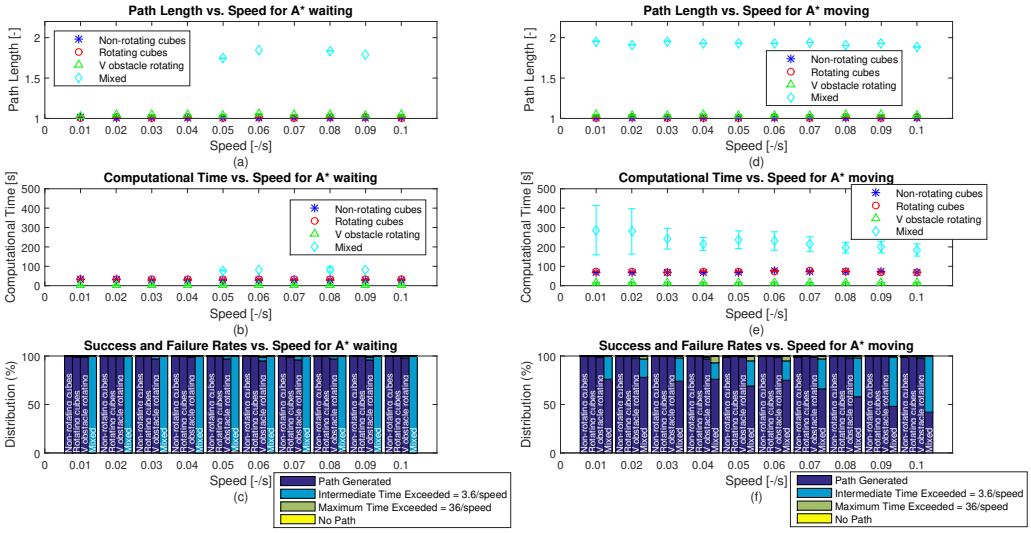


Figure 4.2: Performance parameters vs. speed: (a) Path Length for A* waiting, (b) Computational Time for A* waiting, (c) Success and Failure rates for A* waiting, (d) Path Length for A* moving, (e) Computational Time for A* moving and (f) Success and Failure rates for A* moving for 100 iterates for each considered situation (speed and scenario) with 95% confidence interval. ($res = 21, d_{int\_goal} = 0.2, d_{factor} = 0.8$ and $t_{iterate\_max}, t_{path\_gen\_max}$ are a function of the UAV speed).

longest for this scenario. This violation is never triggered in A* waiting. Oppositely, a No path violation is recorded for A* waiting. A No path violation is recorded in 2 instances out of 4000 where the planner waited but while waiting the obstacle moved towards her causing a collision. This rare situation is very dangerous especially in non-combat applications. So the use of A* waiting should be used with caution to eliminate this possibility. Otherwise it would be better to risk rather than get crashed into by an enemy or moving obstacle.

In conclusion, the path length and computational time for both A* waiting and A* moving are similar. The overall success rate is only equal for the first three scenarios but better by 65.6% for A* moving with respect to A* waiting for Scenario 4. This shows that there exists a better chance of reaching the final goal if the planner identifies a closer intermediate goal point that will allow the UAV to extend its visual line of sight at the expense of being nearer to the obstacle. As the speed of the UAV is assumed to be equal or higher than that of an obstacle in the environmental space, the UAV will always be safe to fly away from the obstacle. Speed has negligible effect on path length and computational time for the considered speed range. Finally, obstacle density and rotation have adverse affect on path length, computational time and success rate.

4.6.3. RRT RESULTS

Figure 4.3 illustrates the performance results for the RRT algorithm for the waiting and moving cases for the scenarios described in the A* algorithm analysis. The lowest path

length was exhibited for the cube case with no rotation, followed by the rotating cube and V–obstacle scenarios with the mixed case exhibiting a multiple times larger path length due to the multiple times higher scenario complexity with respect to the other scenarios. Although the V–obstacle occupies less space than the cube, its two planes are larger than each plane of the cube. Furthermore, since a path can be constructed multiple times faster for the V–obstacle scenario than for both cube cases due to lower obstacle density, the constructed path will not be as optimal since the requirement in our path construction algorithm is to construct a non–colliding path in the minimum time possible and the amount of restrictions to tree propagation in the V–obstacle case will be less with respect to the cube cases. Although the smoothing algorithm will reduce sharp turns in the path it will not achieve the same result as if the path was smoother from the beginning.

The results in terms of mean and standard deviation for both RRT variants are equal for all speeds considered for all scenarios except for the mixed case scenarios. The relatively large path length difference for the moving configuration in the mixed case is attributed to the higher obstacle density. Results show that path length is independent of speed for all scenarios. This implies that irrespective of speed the path length is invariant although the path followed may differ depending on the random seed sequence. Rotation adds to path length as can be deduced from the non–rotating and rotating scenarios. The addition is less than 0.1% for both RRT waiting and moving configurations. These conclusions were also derived in the A* case.

The computational time for RRT waiting is 3.0%, 2.4% and 1.2% times longer with respect to RRT moving for non–rotating cubes, rotating cubes and V–obstacle rotating, respectively. For the mixed scenario, comparison cannot be made since the success rate for the waiting configuration is 0% for all tests. Results for the first three scenarios show that in terms of computational time, the RRT moving option is better than the RRT waiting option for all cases. As for A*, the lowest computational time is recorded for the V–obstacle case, 5.0 and 4.9 times lower than the second best performing scenario i.e. non–rotating cube case for RRT waiting and moving, respectively. By introducing rotation to the cubes the computational time increased by 32.3% and 33.0% for RRT waiting and moving, respectively. The difference in computational time between the mixed case and the other scenarios is multiple times longer for the moving case.

The low obstacle density of the V–obstacle case (open obstacle) is advantageous for the RRT planner as the environment is almost free except for two planes per V–obstacle. Computationally it is easier to check for collisions in tree branches, path segments and new intermediate and UAV positions if obstacles are made of only 2D planes rather than closed obstacles. In closed obstacles, besides checking for collisions with obstacle surfaces, the collision avoidance sub–algorithm must check if any part of the path segment or the point lies inside the 3D obstacle. This explains the lower computational time of the V-obstacle case with respect to the non–rotating cubes. Zooming in on Figure 4.3 (b) and (e), shows that as the speed increases the computational time remains constant for all scenarios except for the Mixed case Scenario in the moving case configuration. Only the maximum intermediate and total times are a function of speed. The maximum intermediate and total times only determine the success rate. So independent of speed the planner is required to construct a path a constant distance from the current UAV position

to an intermediate goal point and determine the new UAV position also a constant distance from the current UAV position on the constructed path segment. This explains why for the different speeds considered the computational time is invariant. For the mixed case in the moving configuration the reduction in computational time with increase in speed is a consequence of the drop in success rate due to lower maximum intermediate time allocation which is inversely proportional with speed. The computational time to generate an intermediate path is mainly dependent upon scenario difficulty as can be deduced from Figure 4.3. Therefore, it can be concluded that computational time is independent of speed for both RRT variants.

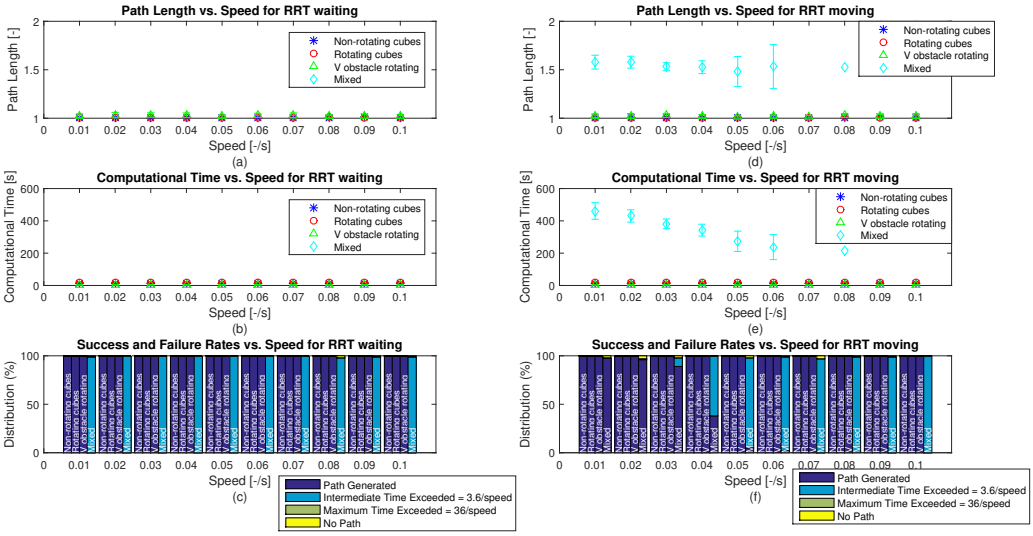


Figure 4.3: Performance parameters vs. speed: (a) Path Length for RRT waiting, (b) Computational Time for RRT waiting, (c) Success and Failure rates for RRT waiting, (d) Path Length for RRT moving, (e) Computational Time for RRT moving and (f) Success and Failure rates for RRT moving for 100 iterates for each considered situation (speed and scenario) with 95% confidence interval. ($d_{s\_step} = 0.05, d_{int\_goal} = 0.1, d_{factor} = 0.8$ and $t_{iterate\_max}, t_{path\_gen\_max}$ are a function of the UAV speed).

Figure 4.3 (c) and (f) illustrate the success rate for RRT waiting and moving, respectively. The figure shows a 100% success rate for both configurations for the cube with and without rotation and the V–obstacle scenarios. The results show the robustness of both RRT variants and the look–ahead distance algorithm that is able to mitigate the effect of moving obstacles. For the mixed case, the RRT waiting case showed no successful runs while for the moving case a success rate of 98% was achieved for low speed reducing to 0 as the speed increases. This is attributed to the fact that in the waiting case the planner waits until the intermediate goal point a predetermined distance from the current UAV position is free from obstacles and a path to this intermediate goal point is possible. The maximum intermediate time which is equal to the waiting time is then not enough for the obstacle to move away from the intermediate goal position. For the moving case, the planner reduces this distance by the distance reduction factor until an intermediate

path to an intermediate goal point is possible. This effect is only visible for the mixed case since in this situation the environment is crowded with obstacles and so the above mentioned condition that an intermediate path is not possible is more frequent.

As described earlier, speed defines the maximum allowable intermediate and total time for the planner to generate a path. For the first three scenarios, time was never the bottleneck and the planner was able to construct the path in a fraction of the maximum intermediate and total time. The planner in all situations will require a range of times to generate intermediate paths as obstacles' initial and time–variant positions change during path construction from one run to another although the start and goal points, environmental space and amount, size and shape of obstacles is constant for a particular scenario. Therefore as speed increases and therefore the maximum allowable intermediate and maximum time reduces a higher percentage of runs for the Mixed case will exceed the maximum allowable intermediate time at one instance, resulting in a maximum intermediate time violation. In fact, the majority of unsuccessful runs results due to insufficient intermediate time. This explains the drop in success rate with increase in speed.

Another aspect is that the RRT waiting case resulted in less No Path situations than RRT moving. This is attributed to the fact that in the moving case the UAV is nearer to moving obstacles than in the waiting case that waits until a path a predetermined distance from the current UAV position is possible. This increases the risk of collision as the obstacles can move into the new or current UAV position for the moving case with respect to the waiting case.

In conclusion, results of both A* and RRT show that exploring further into the vicinity of an obstacle increases the chances of finding a viable path $vis-a-vis$ waiting at the current position until a path to an intermediate point is possible. The moving tactic increases the risk of collision due to a number of factors possibly sensor inaccuracy and unknown future obstacle state. On the other hand, waiting allows the UAV to stay at a safer position from the obstacle in its vicinity, but the UAV cannot stop forever and staying stationary will make the UAV an easy prey. Therefore, for the scope of comparison between A* and RRT both moving cases will be considered as for both A* and RRT the moving variant resulted in better success rates.

4.6.4. A* VS. RRT

Figure 4.4 illustrates the path length, computational time and success rate for the A* and RRT algorithms for the moving variant. In terms of path length the RRT algorithm constructed shorter paths with respect to the A* algorithm, although the mean difference is less than 1% for the cube cases increasing to 3.8% for the V-obstacle cases. For the mixed case, the success rate at speed higher than 0.06[-]/s is 1% or less. If we consider the path length for speeds from 0.01[-]/s to 0.05[-]/s the RRT algorithm constructed paths 23.3% shorter with respect to A*. In terms of variance, RRT also exhibits lower variance for all scenarios for all speeds with respect to A*, implying that the constructed path is less affected by the movement of obstacles. The shorter path length and larger variance range is mainly attributed to the graph–based nature of the A* algorithm. In A*, obstacles are modelled as unavailable grid positions with a buffer equal to half the distance between grid positions as otherwise during re–assignment a node which resides a distance lower

than the buffer from an obstacle may be unavailable in the next iterate even if the environment remains static both due to quantisation and also due to the ripple reduction algorithm [2]. The smoothing algorithm which was only applied for the RRT algorithm cannot be applied to the A* algorithm since this will near the path to a distance less than the buffer distance to the obstacles possibly resulting in collisions in the next iteration. Refer to [1–3] for further details. Furthermore, for both algorithms the path length is independent of speed as speed only effects the maximum allocated time to construct the path.

The computational time characteristics illustrated in Figure 4.4 (b) and (e) for A* and RRT, respectively show that a longer mean computational time with a higher variance is recorded for A* with respect to RRT. For both algorithms, the V–obstacle results in the shortest path followed by the cube without and with rotation and the mixed case. Also for both algorithms speed has no effect on performance. The mean computational time for A* is 6.6, 5.1 and 7.0 times longer with respect to RRT for the first three scenarios respectively and 17.9% shorter for A* with respect to RRT for speeds from 0.01[-]/s to 0.05[-]/s as success rate is very low for RRT at larger speeds. The variance is similar for the first three scenarios but is larger for A* in the Mixed case with respect to RRT for small speeds. This increase in variance is the result of the quantisation of the graph–based nature of the A* algorithm.

From the computational time results it can be concluded that the RRT algorithm performed better than the A* for the first three scenarios deteriorating at a faster rate than A* as the obstacle complexity increases. In our previous analysis [1, 2] it was shown that the A* algorithm was faster than RRT in finding a goal offline in the presence of 2D obstacles. With the inclusion of 3D obstacles the computational demand to check whether each node is within or residing at a distance smaller than half the distance between grid positions has increased significantly. For RRT the difference in computational demand is less as the planner is not required to model the environment within the sensory system field of view but only check for collision once tree nodes and branches are to be constructed. Since the obstacle density in the first three scenarios is low this explains why the RRT performed better than A*. For the RRT algorithm in the obstacle–rich Mixed case scenario, the chance of collisions with obstacles is much larger and therefore more time is required to find a sub–path. For A*, the difference is minimal as in all cases the planner needs to check each node in the environment.

Figure 4.4 (c) and (f) show that the success rate is almost 100% for A* and 100% for RRT for the first three scenarios for all speeds considered for both A* and RRT algorithms. From the results it can be concluded that RRT will be able to generate a path in low complexity environments while A* may not be able to offer this guarantee. The computational response shown in Figure 4.4 (b) and (e) shows that the allocated time is more than enough for the majority of the situations and the non successful runs are a consequence that no path could be constructed with the considered range and resolution in A* in few situations for the first three scenarios irrespective of the allowable time.

For the Mixed case a drop in success rate is shown as speed increases with the major drop exhibited for RRT. The Mixed case is at least three times more obstacle dense than the other three scenarios. As scenario complexity increases the computational requirements for RRT increases at a higher rate with respect to A* resulting in maximum

intermediate time violations. This is attributed to the fact that A* must model all obstacles within its range and FOV irrespective of the complexity of the environment while the RRT only checks for collisions as described earlier.

Therefore it can be concluded that the RRT is more vulnerable to increase in obstacle complexity with respect to RRT. The RRT algorithm is able to achieve a success rate of 95% or better for the lowest three speeds considered. This high success rate is never recorded for A*. From this it can be concluded that A* will not achieve the same level of success rate as RRT if the latter is given appropriate time to construct sub–paths. More computational time can be achieved by reducing the speed and/or increasing the computational processing power. Another point is that the RRT issued No Path violations for certain mixed case scenarios while A* never issued this violation. This implies that a violation occurred since the UAV cannot move as no path is possible to an intermediate goal point or that an obstacle collided into the UAV. This situation is very dangerous as it could lead to the loss of the vehicle. For A* this situation never resulted mainly due to the buffer distance considered for quantisation purposes and the UAV although not reaching the path never collided or is at risk of colliding.

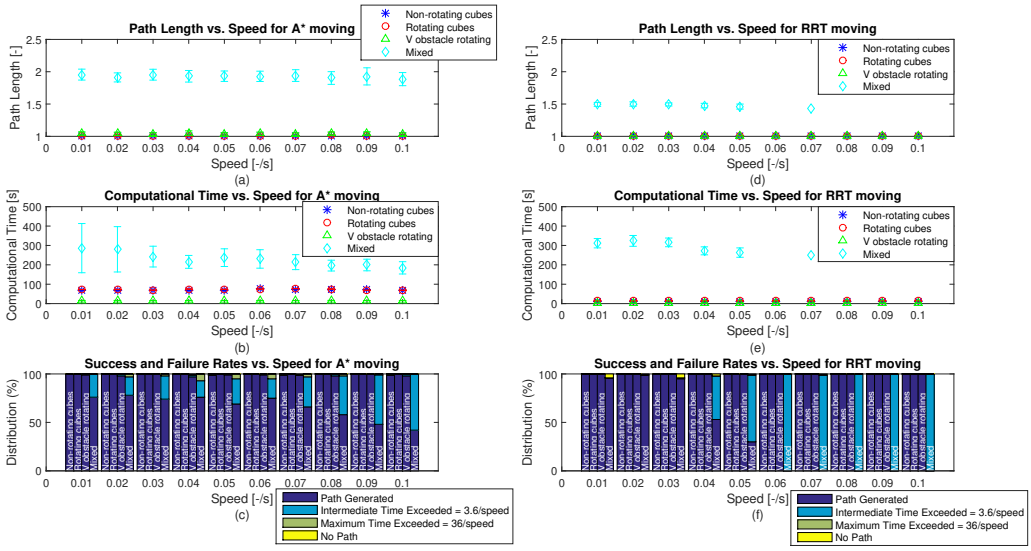


Figure 4.4: Performance parameters vs. speed: (a) Path Length for A*, (b) Computational Time for A*, (c) Success and Failure rates for A*, (d) Path Length for RRT, (e) Computational Time for RRT and (f) Success and Failure rates for RRT for 100 iterates for each considered situation (speed and scenario) with 95% confidence interval. ($res = 21, d_{s\_step} = 0.05, d_{int\_goal} = 0.4$ (for Scenarios 1,2 and 3) and $d_{int\_goal} = 0.6$ (for Scenario 4), $d_{factor} = 0.8$ and $t_{iterate\_max}, t_{path\_gen\_max}$ are a function of the UAV speed).

4.6.5. CONCLUSION

Throughout this section, the performance results of the A* and RRT algorithms using both waiting and moving variants were illustrated, analysed and compared to select the best option in view of 3D UAV path planning with moving obstacles. Results showed

that:

1. The A* moving variant produced better results with respect to A* waiting variant while similar results were recorded for RRT for both variants
2. RRT results in better or equal success rate for all scenarios at all speeds with respect to A*, except for high speeds in the Mixed case scenario.
3. Results show that RRT is better suited for 3D real–time applications in dynamic obstacle–rich environments provided that appropriate time is allocated.
4. The A* algorithm cannot be excluded from dynamic environments especially if safety is a paramount requirement and the computational power in complex environments is limited. In such case the A* would lead to better results.

Table 4.3 summarises the results presented in this section by showing the effect of the waiting and moving rationale on path planning performance.

Table 4.3: Relational table between UAV parameters and path planning performance

Parameter	Path Length	Planning Time	Success rate
A* Waiting vs. A* Moving	Minor difference Moving better	Major difference Waiting better	Major difference Moving better
RRT Waiting vs. RRT Moving	No difference	Minor difference Moving better	Major difference Moving better
A* vs. RRT	Minor difference RRT better	Major difference RRT better	Minor difference A* better

4.7. CONCLUSION AND FUTURE WORK

This paper implemented 3D real–time path planning A* and RRT algorithms in the presence of dynamic constraints, obstacles and threats. Literature confirms that dynamic path planning is essential as UAVs are sometimes required to operate in time-variant environments. In this regard, 4 different scenarios were considered with increasing difficulty consisting of V–obstacles, enclosed cubes and 2D planes moving at different time–varying speeds, direction and orientation. Only, obstacle characteristics a look–ahead distance away from the current UAV position are known with certainty by the planner with updates being provided as the UAV moves. This was done to emulate sensory systems onboard UAVs. The planner was time–limited both in the construction of individual path segments and in the overall path as the UAV is not allowed to stop and is required to always have available the next position before moving from the current position. Two rationales were considered and implemented for both A* and RRT planners in case an intermediate goal point was not available, namely waiting until the defined intermediate goal point becomes available or moving to the intermediate goal point nearer to the

current UAV position consequently increasing the chance of moving further towards the final goal. Path length, computational time and success rate were considered as the performance measures. These measures were assessed with UAV speed which was varied between 0.01 and 0.1[-/s].

Results show that the moving option yielded better overall results in terms of path length, computational time and success rate for A* and RRT with respect to the waiting option especially for the Mixed case which recorded an almost 0% success rate for the waiting case for both path planning algorithms. UAV speed determines the maximum intermediate and total time, reducing as speed increases. Results show that as speed increases success rate drops due to lack of path planning time for the Mixed case scenarios in both A* and RRT as in these situations the required path planning time is near the maximum while for the other scenarios the allocated time is much more than required. Overall, both A* and RRT produced similar results for the first three scenarios with RRT recording slightly better results in path length, computational time and success rate. For the Mixed case, the RRT algorithm performed better at low speeds but worse than A* for high speeds. Also, A* was only limited with time while the RRT resulted in No Path situations that can potentially lead to collisions. The results show that both algorithms can be applied in low obstacle density environments in real–time in the presence of moving obstacles. For complex scenarios the RRT is better suited if time is not limited while the A* algorithm is less susceptible to time constraints. Finally, this work shows that 3D real–time path planning in low and high obstacle density, moving obstacle environments is possible.

Future enhancement shall include the analysis of the effects of other parameters on the performance of path planners with both waiting and moving rationale. These parameters shall include, resolution and step–size in case of A* and RRT respectively, look–ahead distance, distance to travel per iterate, intermediate and total time and the distance factor which defines the reduction factor of look–ahead distance and distance to travel per iterate in case the intermediate goal point or the new UAV position resides on an obstacle. This analysis will flag the best configuration for both algorithms in the considered test scenarios which offer a range of path construction difficulties. Furthermore, real sensory systems sense the environment within a certain degree of uncertainty that changes with a number of factors including: distance to the object, shape and light levels. This uncertainty may have an effect on the performance of the 3D real–time path planners as paths are generated only with a certain degree of non–collision. This can generate unexpected collisions with obstacles that can ultimately result in the loss of the UAV.

REFERENCES

[1] Zammit, C. and van Kampen, E. J., "Comparison between A* and RRT Algorithms for UAV Path Planning", *AIAA Guidance, Navigation and Control Conference*, AIAA SciTech Forum, Kissimmee, FL, 8–12 Jan., 2018.

[2] Zammit, C. and van Kampen, E. J., "Advancements for A* and RRT in 3D path planning of UAVs", *AIAA Guidance, Navigation and Control Conference*, AIAA SciTech Forum, San Diego, CA, 7–11 Jan., 2019.

[3] Zammit, C. and van Kampen, E. J., "Comparison of A* and RRT in real–time 3D path planning of UAVs", *AIAA Guidance, Navigation and Control Conference*, AIAA SciTech Forum, Orlando, FL, 6–10 Jan., 2020.

[4] Chakrabarty, A. and Langelaan, J. W., "Energy maps for long–range path planning for small-and micro–uavs", *AIAA Guidance, Navigation and Control Conference*, Chicago, IL, 10–13 Aug., 2009, pp. 1–13.

[5] Benenson, R. Petti, S., Fraichard, T. and Parent, M., "Integrating Perception and Planning for Autonomous Navigation of Urban Vehicles," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China, 9–15 Oct. 2006, pp. 98–104.

[6] Yao, P., Wang, H. and Su, Z., "Real-time path planning of unmanned aerial vehicle for target tracking and obstacle avoidance in complex dynamic environment," *Aerospace Science and Technology*, Vol. 47, pp. 269–279, 2015.

[7] Foo, J. L., Knutzon, J., Kalivarapu, V., Oliver, J. and Winer, E., "Path Planning of Unmanned Aerial Vehicles using B–Splines and Particle Swarm Optimization", *Journal of Aerospace Computing, Information, and Communication*, Vol. 6, No. 4, 2009, pp. 271–290.

[8] Goerzen, C., Kong, Z. and Mettler, B. "A Survey of Motion Planning Algorithms from the Perspective of Autonomous UAV Guidance", *Journal of Intelligent & Robotic Systems*, Vol. 57, pp. 65—100, 2010.

[9] Luke, S., Sullivan, K. and Balan, G. C., "Tunably Decentralized Algorithms for Cooperative Target Observation", George Mason University, GMU–CS–TR–2004–1, 2004.

[10] Maza, I., and Ollero, A., "Multiple UAV cooperative searciteching operation using polygon area decomposition and efficient coverage algorithms", *Jistributed autonomous robotic systems*, in R. Alami, R. Chatila, and H. Asama (Eds.), Vol. 6, Tokyo, Japan: Springer, pp. 221—230, 2012.

[11] Jiao, Y.-S.,Wang, X.-M., Chen, H. and Li, Y., "Multiple Research on the coverage path planning of UAVs for polygon areas", *5th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, Taichung, Taiwan, 15–17 Jun., 2010, pp. 1467—1472.

[12] Dittrich, J. S., Adolf, F.-M., Langer, A. and Thielecke, F. "Mission Planning for Small UAV Systems in Uncertain Environments,", *2nd European Micro Aerial Vehicle Conference*, Braunschweig, Germany, 25-26 July, 2006.

[13] Adolf, F.–M., Andert, F. and Rocha, J. G. F., "Rapid Online Path Planning Onboard A VTOL UAV", *AIAA Infotech, Aerospace Conference*, Atlanta, GA, Apr. 20–22, 2010.

[14] Srikanthakumar, S., Liu, C. and Chen W. H. "Optimization-Based Safety Analysis of Obstacle Avoidance Systems for Unmanned Aerial Vehicles", *Journal of Intelligent Robot Systems*, Berlin, Germany: Springer Science & Business Media, 2012.

[15] Bollino, K. P. and Ryan Lewis, L., "Collision-free Multi-UAV Optimal Path Planning and Cooperative Control for Tactical Applications", *AIAA Guidance, Navigation and Control Conference*, Honolulu, HI, 21–24 Aug., 2008, pp. 1–18.

[16] Lee, J.-W., Walker, B. and Cohen, K. "Path Planning of Unmanned Aerial Vehicles in a Dynamic Environment", *Infotech@Aerospace*, St. Louis, Missuori, 29–31 Mar., pp. 1–19.

[17] Swartzentruber, L., Foo, J. L. and Winer, E., H. "Three-dimensional multi-objective uav path planner using terrain information", *50th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Palm Springs, California, 4–7 May 2009, pp. 1–19.

[18] Wu, P. P., Campbell, D. and Merz, T., "Multi–Objective Four–Dimensional Vehicle Motion Planning in Large Dynamic Environments", *IEEE Transactions on Systems, Man and Cybernetics– Part B Cybernatics*, Vol. 41, No. 3, Jun. 2011, pp. 621–634.

[19] Yap, P., Burch, N., Holte, R. and Schaeffer, J. "Block A*: Database–driven search with applications in any–angle path–planning", *Proceedings of the Conference of Artificial Intelligence (AAAI)*, 2011.

[20] Boskovic, J. D., Knoebel, N., Moshtagh, N. and Larson, G.L., "Collaborative Mission Planning & Autonomous Control Technology (CoMPACT) System Employing Swarms of UAVs", *AIAA Guidance, Navigation and Control Conference*, Chicago, IL, 10–13 Aug., 2009, pp. 1–24.

[21] Gros, M., Schöttl, A., and Fichter, W., "Spline and OBB–based Path–Planning for Small UAVs with the Finite Receding–Horizon Incremental–Sampling Tree Algorithm", *AIAA Guidance, Navigation and Control Conference*, Boston, MA, 19–22 Aug., 2013, pp. 1–17.

[22] Tseng, F. H., Liang, T. T. and Lee, C. H. Chou, L. D. and Chao, H., "A Star Search Algorithm for Civil UAV Path Planning with 3G Communication", *10th International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH–MSP)*, Kitakyushu, Japan, 27–29 Aug. 2014, pp. 942–945.

[23] Barrientos, A., Colorado, J., Martinez, A., Rossi, C., Sanz, D. and Valente, "Aerial Remote Sensing in Agriculture: A Practical Approach to Area Coverage and Path Planning for Fleets of Mini Aerial Robots", *Journal of Field Robotics*, Vol. 28, pp. 667–689, 2011.

[24] Kuwata, Y., Teo, J., Karaman, S., Fiore, G., Frazzoli, E., and How, J. P., "Motion Planning in Complex Environments using Closed-loop Prediction", *AIAA Guidance, Navigation and Control Conference and Exhibit*, Honolulu, HI, 18–21 Aug. 2008, pp. 1–22.

[25] Bohren, J., Foote, T., Keller, J., Kushleyev, A., Lee, D., Stewart, A., Vernaza, P., Derenick, J., Spletzer, J. and Satterfield, B. "Little ben: The ben franklin racing team' s entry in the 2007 darpa urban challenge", *Journal of Field Robotics*, Vol. 25, No. 9, pp. 598–614, 2008.

[26] Fujimura, K. *Motion planning in dynamic environments*, Berlin, Germany: Springer Science & Business Media, 2012.

[27] Mac, T. T., Copot, C., Tran, D. T. and Keyser, R. D. "Heuristic approaches in robot path planning: A survey", *Journal of Robotics and Autonomous Systems*, Vol. 86, Aug. 2016, pp. 13–28.

[28] Qin, Y. Q., Sun, D. B., Li, N. and Cen, Y. G. "Path Planning for mobile robot using the particle swarm optimization with mutation operator", *International conference on Machine learning and Cybernetics*, Shangai, China, 26–29 Aug. 2004, pp. 2473–2478.

[29] Mac, S. P. "Optimal path palnning for mobile robots: A review", *International Journal of Physical Science*, Vol. 7, No. 9, Aug. 2012, pp. 1314–1320.

[30] "A bioinspired neural network for real-time concurrent map building and complete coverage robot navigation in unknown environments," *IEEE Transactions on Neural Networks*, Vol. 19, No. 7, pp. 1279–1298, 2008.

[31] Zhang, H., Butzke, M. and Likhachev, M. "Combining global and local planning with guarantees on completeness", *IEEE International conference on Robotics and Automation*, St. Paul, MN, 14–18 May 2012, pp. 2500–2506.

[32] Wang, L. C., Yong, L. S. and Ang J. M. H. "Hybrid of global path planning and local navigation implemented on a mobile robot in indoor environment", *IEEE International Symposium on Intelligent Control*, Vancouver, British Columbia, Canada, 27–30 Oct. 2012, pp. 821–826.

[33] Bi, Z. and Yimin, Y. "Hierarchical path planning approach for mobile robot navigation under the dynamic environment", *IEEE International Symposium on Industrial Informatics*, Daejeon, Korea, 13–16 Jul. 2008, pp. 372–376.

[34] Singh, Y. and Sharma, S., "Optimal path planning of unmanned surface vehicles," *Indian Journal of Geo-Marine Sciences*, Vol. 47, No. 7, pp. 1325–1334, 2018.

[35] Amin, J. N., Boskovic, J. D. and Mehra, R. K, "A Fast and Efficient Approach to Path Planning for Unmanned Vehicles", *AIAA Guidance, Navigation and Control Conference*, Keystone, CO, 21–24 Aug., 2006, pp. 1–9.

[36] Chawla, C. and Pahdi, R., "Neuro–Adaptive Augmented Dynamic Inversion Based PIGC Design for Reactive Obstacle Avoidance of UAVs", *AIAA Guidance, Navigation, and Control Conference*, Portland, OR, 08–11 Aug., 2011, pp. 1–25.

[37] Short, A., Pan, Z., Larkin, Z. and van Duin, S. "Recent Progress on Sampling Based Dynamic Motion Planning Algorithms", *IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, Alberta, Canada, Jul. 2016, pp. 1305–1311.

[38] Svec, P., Shah, B.C., Bertaska, I.R., Alvarez, J., Sinisterra, A.J., Von Ellenrieder, K., Dhanak, M., and Gupta, S. K. "Dynamics-aware target following for an autonomous surface vehicle operating under COLREGs in civilian traffic", *IEEE International Conference on Intelligent Robots and Systems*, Tokyo, Japan, 3–7 Nov., 2013, pp.3871–3878.

[39] Hsu, D., Kindel, R., Latombe, J.–C., and Rock, S. “Randomized kinodynamic motion planning with moving obstacles," *International Journal of Robotics Research*, Vol. 21, No. 3, pp. 233–255, 2002.

[40] Howlett, Schulein, G. and Mansour, M. H., “A Practical Approach to Obstacle Field Route Planning for Unmanned Rotorcraft”, *60th Annual Forum at the American Helicopter Society*, Baltimore, MD, Jun. 7–10, 2004.

[41] Oomkens, W. “UAV Aviating Automation”, Master Thesis, TU Delft, 2007.

[42] Larson, J., Bruch, M., and Ebken, J., “Autonomous navigation and obstacle avoidance for unmanned surface vehicles,” *Proceedings of SPIE*, Vol. 8, pp. 17–20, 2006.

[43] Yu, H., Beard, R. W. and Byrne, J. “Vision–based Navigation Frame Mapping and Path Planning for Micro Air Vehicles”, *AIAA Guidance, Navigation, and Control Conference*, Chicago, Illinios, 11–13 Aug. 2009, pp. 1–10.

[44] Likhachev, M., Ferguson, D., Gordon, G., Stentz, A. and Thrun, S., “Anytime search in dynamic graph”, *Artificial Intelligence*, Vol. 172, No. 14, pp. 1613—1643, 2008.

[45] Liu, Y., Cruz, J. B. and Sparks, A. G., “Coordinating Networked Uninhabited Air Vehicles for Persistent Area Denial”, *43rd IEEE Conference on Decision and Control*, Paradise Island, Bahamas, Dec., 2004, pp. 3351–3356.

[46] Koenig, S. and Smirnov, Y. “Sensor–based planning with the freespace assumption”, *International Conference on Robotics and Automation (ICRA)*, Albuquerque, New Mexico, 20–25 Apr., 1997, pp. 3540—3545.

[47] Koenig, S. and Likhachev, M., “D* Lite”, *Proceeding of the AIAA Conference on Artificial Intelligence*, Indianapolis, IN, 7–10, Jul. 2002, pp. 476–483.

[48] Guivant, J. and Nebot, E.“Optimization of the Simultaneous Localization and Map Building Algorithm for Real Time Implementation”, *IEEE Transactions of Robotics and Automation*, Vol. 17, No. 3, pp. 242—257, 2001.

[49] Thrun, S., Diel, M. and Hahnel, D. “Scan Alignment and 3-D Surface Modeling with a Helicopter Platform”, *4th International Conference on Field and Service Robotics*, Lake Yamanaka, Japan, 14–16 Jul., 2003, pp.1–6.

[50] Ericson, C. “Real-Time Collision Detection”, *Real–Time Collision Detection*, 1st ed., Vol. 1, Morgan Kaufmann Publishers, San Francisco, CA, 2005.

[51] Hrabar, S., “3D Path Planning and Stereo-based Obstacle Avoidance for Rotorcraft UAVs”, *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nice, France, 22–26 Sep. 2008, pp. 807–814.

[52] Scherer, S., Singh, S., Chamberlain, L., and Saripalli, S.“Flying fast and low among obstacles”, *IEEE International Conference on Robotics and Automation (ICRA)*, Rome, Italy, 11–13 Apr., 2007, pp.2023–2029.

[53] Shim, D., Chung, H., Kim, H. J., and Sastry, S. "Autonomous exploration in unknown urban environments for unmanned aerial vehicle", *AIAA Ground, Navigation and Control Conference*, San Francisco, CA, 15–18 Aug., 2005, pp.1–8.

[54] Gottschalk S., Lin M. C. and Manocha D., "OBB–Tree: A Hierarchical Structure for Rapid Interference Detection," *Proceedings of 23rd ACM Siggraph Structure for Rapid Interference Detection*, ACM, NY, pp. 171–180.

[55] Ögren, P. and Robinson, J. W. C. "Receding Horizon Control of UAVs using Gradual Dense–Sparse Discretizations," *AIAA Guidance, Navigation, and Control Conference*, Toronto, Ontario, Canada, 2–5 Aug. 2010, pp. 1–11.

[56] Ok, K.-C., Ansari, S., Gallagher, B., Sica, W., Dellaert, F., and Stilman, M."Path planning with uncertainty: Voronoi uncertainty fields", *IEEE International Conference on Robotics and Automation (ICRA)*, Karlsruhe, Germany, 6–10 May, 2011, pp.4596–4601.

[57] Bollino, K., Lewis, L. R., Sekhavat, P. and Ross, I. M., "Pseudospectral Optimal Control: A Clear Road for Autonomous Intelligent Path Planning," *AIAA Infotech Aerospace Conference and Exhibit*, Rohnert Park, CA, 7–10 May 2007, pp. 1–14.

[58] Lewis, L.R. " Rapid Motion Planning and Autonomous Obstacle Avoidance for Unmanned Vehicles", Master Thesis, Naval Postgraduate School, Monterey, CA, 2006.

[59] Subramanian, S. K. and Cruz, J. B., "Adaptive Models of Pop-Up Threats for Multi-Agent Persistent Area Denial," *42nd IEEE Conference on Decision and Control*, Maui, Hawaii, 9–12 Dec. 2003, pp. 510–515.

[60] Kim, J., Kim, M. and Kim, D. "Variants of the quantized visibility graph for efficient path planning", *Journal of Advanced Robotics* , Vol. 25, No. 18, pp. 2341—2360, 2011.

[61] Tsardoulias, E. G.,Iliakopoulou, A., Kargakos, A. and Petrou, L. "A Review of Global Path Planning Methods for Occupancy Grid Maps Regardless of Obstacle Density", *Journal of Intelligent & Robotic Systems*, Vol. 84, No. 1–4 pp. 829—858, 2016.

[62] Karaman, S. and Frazzoli, E., "Sampling-based algorithms for optimal motion planning", *The International Journal of Robotics Research*, Vol. 30, No. 7, pp. 846–894, 2011.

[63] Choset, H., Lynch, K., Hutchinson, S., Kantor, G., Burgard, W., Kavraki, L. and Thrun, S., "Principles of Robot Motion: Theory, Algorithms, and Implementations", Massachusetts: MIT Press, 2005.

[64] Devaurs, D., Siméon, T. and Cortés, J. "Optimal Path Planning in Complex Cost Spaces With Sampling–Based Algorithms", *IEEE Transactions on Automation Science and Engineering*, Institute of Electrical and Electronics Engineers, 2015.

[65] Latombe, J.–C., "Robot Motion Planning", *The Springer International Series in Engineering and Computer Science*, Berlin, Germany: Springer Science & Business Media, 1991.

[66] Roos, T. and Noltemeier, H. “Dynamic Voronoi Diagrams in Motion Planning” *Lecture Notes in Computer Science*, Vol. 553, No. 17, Berlin, Germany: Springer Science & Business Media, 1991.

[67] Yang, L., Qi, J., Song, D., Xiao, J., Han, J. and Xia, Y. “Survey of Robot 3D Path Planning Algorithms” *Journal of Control Science and Engineering*, Vol. 2016, No. 5, pp. 1–22, 2016.

[68] Harabor D. and Botea, A. “Breaking Path Symmetries on 4-Connected Grid Maps,” *Proceedings of the Sixth Artificial Intelligence and Interactive Digital Entertainment Conference*, Stanford, CA, 11–13 Oct., 2010.

[69] Lau, B., Sprunk, C. and Burgard, W. “Efficient grid–based spatial representations for robot navigation in dynamic environments” *Robotic Automation Systems*, Vol. 61, pp. 1116–1130, 2013.

[70] Park, B., Choi, J. and Chung, W. K. “An efficient mobile robot path planning using hierarchical roadmap representation in indoor environment” *IEEE International Conference on Robotics and Automation (ICRA)*, St. Paul, MN, 14–18 May, 2012, pp. 180–186.

[71] Aoude, G. S., Luders, B.D., Levine, D. S. and How, J.P. “Decentralized path planning for multi–agent teams in complex environment using rapidly–exploring random trees” *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, 9–13 May, 2011, pp. 4956–4961.

[72] Sun, X., Cai, C. and Shen, X., “A New Cloud Model Based Human–Machine Cooperative Path Planning Method”, *Journal of Intelligent & Robotic Systems*, Vol. 79, 2014, pp. 3–19.

[73] Szczerba, R.J., Galkowski, P., Glicktein, I. and Ternullo, N., “Robust algorithm for real-time route planning”, *IEEE Transactions on Aerospace Electronic Systems*, Vol. 36, 2000, pp. 869–878.

[74] Zheng, C., Xu, F., Hu, X., Sun, F. and Yan, P., “Online route planner for unmanned air vehicle navigation in unknown battlefield environment," *IMACS Multiconference on Computational Engineering in Systems Applications*, Beijing, China, 4–6 Oct. 2006, pp. 814–818.

[75] Dong, Z., Chen, Z., Zhou, R. and Zhang, R. “A Hybrid Approach of Virtual Force and A* Search Algorithm for UAV Path Re-Planning,” *Proceedings of the 6th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, Beijing, 21–23 Jun., 2011, pp. 1140-1145.

[76] Elbanhawi, M. and Simic, M. “Sampling-Based Robot Motion Planning: A Review” *IEEE Access*, Vol. 2, pp. 56–77, 2014.

[77] Ghandi, S. and Masehian, E., “Review and taxonomies of assembly and disassembly path planning problems and approaches”, *CAD Computer Aided Design*, Vol. 67–68, No. October, pp. 58–86, 2015.

[78] Otte, M. and Frazzoli, E. “RRTX: Real-time motion planning/replanning for environments with unpredictable obstacles,” *11th Algorithmic Foundations of Robotics*, Berlin, Germany: Springer Science & Business Media, 2015, pp. 461–478.

[79] Fiorini, P. and Shiller, Z., “Motion Planning in Dynamic Environments Using Velocity Obstacles," *The International Journal of Robotics Research*, Vol. 7, No. 7, 1998, pp. 760–772.

[80] Zhuang, J.Y., Su, Y.M., Liao, Y.L. and Lei, S. “Motion planning of USV based on Marine rules,” *Procedia Engineering*, Vol. 15, pp. 269–276, 2011.

[81] Naeem, W., Irwin, G.W., and Yang, A., “COLREGs-based collision avoidance strategies for unmanned surface vehicles,” *Mechatronics*, Vol. 22, pp. 669–678, 2012.

[82] Canny, J. and Reif, J., “New lower bound techniques for robot motion planning problems," *Proceedings of the Symposium on the Foundations of Computer Science*, Los Angeles, CA, 12–14 Oct. 1987, pp. 49–60.

[83] Canny, J., Donald, B. and Reif, J., “On the complexity of kinodynamic planning," *29th Annual IEEE Symposium on Foundations of Computer Science*, White Plains, NY, 24–26 Oct. 1988, pp. 306–316.

[84] Benjamin, M. R., and Curcio, J. A. “COLREGS-based navigation of autonomous marine vehicles,” *IEEE/OES Autonomous Underwater Vehicles Conference*, Sebasco, ME, 17–18 Jun., 2004, pp. 32–39.

[85] Gonz*à*lez, D., P*è*r*ez*, J., Milan*è*s, V., and Nashashibi, F., “A Review of Motion Planning Techniques for Automated Vehicles”, *IEEE Transactions on Intelligent Transportation Systems*, Vol. 17, No. 4, pp. 1135–1145, 2016.

[86] Hart, P. E., Nilsson, N. J. and Raphael, B., “A Formal Basis for the Heuristic Determination of Minimum Cost Paths”, *IEEE Transactions on Systems Science and Cybernetics*, Vol. 4, No. 3, pp. 100–107, 1968.

[87] LaValle S. M. “Probabilistic roadmaps for path planning in high–dimensional configuration spaces”, *IEEE Transactions on Robotics and Automation*, Vol. 12, No. 14, pp. 566–580, 1996.

[88] Lavalle S. M. and Kuffner J. J., “Randomized kinodynamic planning”, *International Journal of Robotics Research*, Vol. 20, No. 3, pp. 378–400, 2001.

[89] LaValle, S. M. and Kuffner, J. J. “Randomized kinodynamic planning”, *Proceedings of the IEEE International Conference on Robotics and Automation*, Detroit, MI, 10–15 May 1999, pp. 473–479.

[90] Geraerts, R. and Overmars, M. “Creating high–quality paths for motion planning”, *International Journal of Robotics Research*, Vol. 26, No. 8, pp. 845–863, 2007.

[91] Sujit, P. B., and Ghose, D., “Search by UAVs with Flight Time Constraints using Game Theoretical Models,” *AIAA Guidance, Navigation and Control Conference*, San Francisco, CA, 15–19 Aug. 2005, pp. 1–11.

[92] Bethke, B., Bertuccelli, L. How, J. P., "Experimental Demonstration of MDP–Based Planning with Model Uncertainty," *AIAA Guidance, Navigation and Control Conference*, Honolulu, HI, 18–21 Aug. 2008, pp. 1–22.

[93] Sathyaraj, M. B., Jain, L. C., Finn, A. and Drake, S., "A Multiple UAVs path planning algorithms: a comparative study", *Fuzzy Optimization and Decision Making*, Vol. 7, No. 3, 2008, pp. 257–267.

5

3D Real-time Path Planning of UAVs in Dynamic Environments in the Presence of Uncertainty

This chapter addresses Research Questions 4 and 5, queried to tackles Challenge 3, by investigating uncertainty effects in the 3D real–time UAV path planning in indoor environments. This chapter identifies and explains the need for uncertainty considerations, the sources of uncertainty and their quantification in view of UAVs operating in indoor environments. The relevant uncertainties within the research scope are identified, modelled and integrated with the 3D real-time environment developed in Chapter 4. This chapter will conclude with an analysis of the effect of path planning performance of each uncertainty source independently and concurrently.

The contents of this chapter have been published as:
Zammit, C. and van Kampen, E., "3D real-time path planning of UAVs in dynamic environments in the presence of uncertainty", *Proceedings of AIAA Guidance, Navigation and Control*, Nashville, TN, 11-15 Jan., 2021, AIAA 2021-1956.

Part of this chapter will be submitted to:
Zammit, C. and van Kampen, E., "Real-time 3D UAV path planning in dynamic environments in the presence of uncertainty", *Journal of Guidance, Control and Dynamics*

Unmanned Aerial Vehicles (UAVs) are being integrated into all spheres of life varying in a wide range of applications from military to civil. In such applications, UAVs are expected to operate safely in the presence of uncertainties present in the dynamic environment and the UAV itself. Based on literature different uncertainty sources are identified, quantified and modelled using bounding shapes. The UAV model, path planner parameters and four scenarios of different complexity are defined. For analysis uncertainty is varied between 2% and 20% for UAV position and obstacle position and orientation. Results show that both types of uncertainty deteriorate path planning performance of both A* and RRT algorithms for all scenarios considered especially for RRT. RRT results in faster and shorter paths with approximately the same success rate (>95%) as A* for simple scenarios. For complex scenarios A* performs better.

5.1. INTRODUCTION

Unmanned or Uninhabited Aerial Vehicles (UAVs), Unmanned Aircraft or more commonly known as drones are defined by International Civil Aviation Organisation (ICAO) as *"pilotless aircraft, in the sense of Article 8 of the Convention on International Civil Aviation, which is flown without a pilot-in-command on-board and is either remotely and fully controlled from another place (ground, another aircraft, space) or programmed and fully autonomous"* [1]. Moreover, Unmanned Aircraft Systems (UAS) extend beyond the aircraft and incorporate all associated elements necessary to operate efficiently and safely an aircraft without a pilot on board [2].

The idea of UAVs has initiated and progressed concurrently with advances in aviation. In fact, less than 14 years from the first flight of the Wright brothers on 17th December 1903 in North Carolina, on March 1917 A.M. Low launched the unmanned Ruston Proctor AT using compressed air from Salisbury Plain, North England. Over the past 100 years the technology, purposes and use of Unmanned Aerial Vehicles (UAVs) have evolved primarily in view of military demands which have been the main driver.

UAVs of different sizes and shapes are being integrated into all spheres of life. UAVs are being proposed for a wide range of indoor and outdoor applications varying from surveillance, search and rescue and other military to more civil applications including aerial filming and photography, agriculture, postage delivery and leisure flying. The inclusion of UAVs into the civil airspace and high precision autonomous military applications requires robust autonomous guidance, navigation and control (GNC) systems that shall ensure safe navigation in view of constraints and uncertainties present within manned aircraft and associated supporting systems, UAV systems and the environment in which they will operate. Oppositely to UAVs which were economically viable for the mainstream only over the last decade, manned aircraft systems have been in civil operation for a century. As a result, Standard Operating Procedures (SOPs) for both fixed and rotary wing manned aircraft are well defined mainly as a result of post-incident analysis. In fact, although UAV manufacturers are pushing for integration as the commercial prospect is unbounded, ICAO is cautious and is only foreseeing a medium-term integration of remotely-controlled UAVs within non-segregated and aerodrome environments. Moreover, ICAO [2] indicates that fully autonomous UAVs will not be integrated within civil aviation systems in the foreseeable future.

One fundamental difference between a machine and a human is that the latter is accustomed to deal efficiently with uncertainty from early childhood improving his/her skills through adequate training while programming machines to deal with uncertainty in real time is not straight forward. Although certain constraints such as buildings, no-fly zones and particular kinematic constraints can be defined accurately, in reality the absolute majority can only be defined with an element of uncertainty, for example weather, fuel consumption and vehicles' position and state. Furthermore, constraints can pop-up whilst in flight and the path planning algorithm shall be able to mitigate these *a priori* unknown situations [3].

Uncertainty in path planning has been investigated for quite some time. In the beginning uncertainty was considered as a domain of complaint control and uncertainty was mitigated by allowing or requiring the agent to touch [4]. Although contact with an obstacle can be accepted for gripping or manipulators, in UAV path planning applications this must be avoided at all costs [4]. However, different researchers stress the need for robust and generic path planning solutions that can be applied in the presence of uncertainty. Dadkhah *et al.* point out that while efficient algorithms offering solutions to sub-problem exist, general real-time path planning solutions in the presence of uncertainty is still pending [5]. Similarly, Goerzen *et al.* remark that more research is required to deal with different uncertainty sources as till now this field has not been adequately studied [4]. Furthermore, Vanegas *et al.* identified the need for a model and a system that can incorporate sensing uncertainties in the UAV state calculation besides uncertainty in target location [6].

These researchers and others outline the need to investigate the effect of uncertainty in path planning of UAVs both for indoor and outdoor environments. This motivation is key to the aim of this paper. Therefore the aim of this paper is to investigate the effect of uncertainty on the performance of real-time 3D UAV path planning in a dynamic environment. For the scope of this study the most utilised path planning algorithm in the graph-based and sampling-based categories will be assessed namely the A* and the Rapidly-Exploring Random Trees (RRT) algorithms. It will be assumed that the planner has no *a priori* knowledge of obstacle paths and/or future positions. Moreover, the uncertainty if any of different parameters is not provided to the planner at initiation stage and it can change while the UAV is constructing and traversing the path in real-time. For real-time path planning, the planner must generate the path in at least the time required by the UAV to traverse it [7–9]. The path length, computational time and success rate will be the performance measures that will be considered to assess the performance of the A* and RRT algorithms in a 3D real-time environment. The dynamic environments developed in our previous work [10] will be considered.

The paper will be organised as follows. Section 5.2 will present the state-of-the-art in path planning of UAVs in the presence of uncertainty. Section 5.3 provides a brief resume of the A* and RRT algorithms, the smoothing algorithm which is only applied to the RRT algorithm, the real-time path planning algorithm and the dynamic obstacle definition framework all extensively described in our previous work [10–13]. Section 5.4 will define the environmental scenarios, the UAV model and path planner parameters definitions and constraints and uncertainty modelling and quantification rationale in view of 3D real-time path planning algorithm. Section 5.5 will present and analyse the re-

sults in view of real-time 3D UAV path planning requirements. This paper will conclude with Section 5.6 highlighting the main outcomes, strengths and weaknesses of this study whilst pointing out future recommendations.

5.2. Path planning in the Presence of Uncertainty Review

5.2.1. Introduction

This section will first identify the need for path planning in the presence of uncertainty. Then a list of different uncertainty sources and their challenges to the path planning problem will be explained. The following section will deal with the representation and quantification of uncertainties proposed by different studies. This section will conclude with a resume of the performance of different path planning algorithms in the presence of uncertainty. This will help identify the strengths and weaknesses of each method independently and in relation to others.

5.2.2. The Need for Path Planning in the Presence of Uncertainty

A lot of things in this World are uncertain although uncertainty can be quantified to a certain degree. Therefore in an indoor or outdoor environment a UAV path planning system will have to deal with a number of factors some of which incorporate different uncertainties. These include partially known environments, limited payload capacity, limited on-board computational power, differential constraints, environmental disturbances and uncertainty in both state and measurement [5]. It is commonly assumed that unknown space is occupied in path planning in unknown or partially known environments [24].

An effective path planner in the presence of uncertainty must *always* guarantee that the UAV will reach and stop at the goal region without colliding with any obstacle despite uncertainty in sensing and control [19]. To achieve this, the planner must simultaneously consider the projected motion of obstacles and the time-varying uncertainty in their location [27]. In addition, UAV state uncertainty including both perception and dynamics shall also be considered [16, 21].

Different studies tried to incorporate uncertainty modelling within their path planning systems. Although as will be discussed in the next sub-sections different approaches have been made to eliminate or heavily attenuate uncertainty effects on path planning systems Kim *et al.* remark that path planners that require global or extensive local information may not guarantee satisfactory performance in the presence of uncertainty [14]. Therefore, the issue of uncertainty in sensing and control cannot be neglected in real UAV applications [4]. Goerzen *et al.* remark that uncertainty and robustness has not been fully studied. This study considers these two fundamental aspects as paramount in determining the prospect of an algorithm that must be efficient, optimal and robust at the same time.

Cui *et al.* consider UAV dynamic path planning as extremely challenging due to different uncertainties, time variant magnitudes and interaction of these factors and constraints present within UAV utilisation environments [16]. The planner has to deal with errors and imperfections of sensing systems to figure out threats and positional, kine-

matics and dynamics information for state estimation in real-time. Moreover, UAV mission objectives and control modes of operation of UAVs further add to the complexity of the problem [16].

5.2.3. UNCERTAINTY SOURCES

Different uncertainty sources were considered in different studies. LaValle *et al* segmented uncertainty in robotic systems into 4 categories namely: uncertainty in robot sensing, uncertainty in robot predictability, uncertainty in environment sensing and uncertainty in environment predictability [17]. This implies that uncertainty sources are derived from the sensing and prediction derived as well from the sensing of the agent and the environment in which the agent resides. The same rationale can be applied to 3D UAV environments. In this study, uncertainty sources are segmented into 4 main categories: UAV model, UAV sensing system, environmental sensing and prediction, and communication uncertainties:

UNCERTAINTY IN SENSING SYSTEM

- Generic sensor errors [18, 19];
- Uncertainty in attitude information both of UAV and target [5, 16, 21–23];
- Uncertainty in localisation, velocity and acceleration of both UAV and the target [4, 16, 19, 20, 24, 25];
- Initial uncertainty state [16, 20, 21].

UNCERTAINTY IN UAV MODEL

- Uncertainty in system modelling including agent dynamics [5, 26];
- Disturbances from the nominal model [18];
- Uncertainty in system configuration sensing [21, 26];
- Uncertainty in command tracking precision [4, 5];
- Estimation Uncertainty [24];
- Uncertainty in the system error distribution [16].

UNCERTAINTY IN ENVIRONMENT SENSING AND PREDICTION

- Uncertainty in environmental situational awareness such as obstacle locations and threats [4, 5, 19, 25, 26];
- Uncertainty in future environment prediction [26–28];
- External disturbances into the operational environment. For example wind, atmospheric turbulence and rain [5, 6, 19].

UNCERTAINTY IN COMMUNICATION

- Communication errors, delays, packet dropouts,and finite communication ranges [18].

5.2.4. Uncertainty Modelling

The uncertainty sources described in the previous section need to be mathematically modelled for the path planning algorithm to reach the target successfully. As the guidance, navigation and control systems can be designed as modular sub-systems, modularity can also be applied to segregate and individually define uncertainty sources even based on mission requirement [5]. The uncertainty modelling strategy will have a direct influence on the robustness of the path planning system. Uncertainty modelling can be broadly categorised into two main categories: Bounded Shapes and Probabilistic Distributions [24, 29].

Bounded Shapes

Bounded shapes consider worst-case bounds to define the edges of the bounding shape [24]. Bounded shapes were used to define both threats and obstacles but also state definitions. With reference to the latter, robustness in the presence of uncertainty can be ensured by attentively adjusting the safety margin between failure and nominal states [26]. Bounded uncertainties in state estimation and disturbances were modelled in [30, 31] to verify non linear UAV models. Results show elegant formulations of verifiable planning [30, 32]. Chance-constraint optimisation is a method used for path planning in the presence of uncertainty [33, 34]. This method upper bounds the probability of collision at any time instant by a constraint in the optimisation process [24].

Bounded uncertainty can be modelled as time variant and time invariant. In fact, Page *et al.* [35] modelled an extroceptive sensor that allowed a bounded time invariant uncertainty in its sensing zone. Another study developed the use of "safe zones" that reset to zero or quasi-zero the uncertainty bounds until the agent remains in the safe zone [36]. In this case, uncertainty can be time variant as long as the extrapolation of uncertainties is limited by visiting safe zones. Similarly, Larson *et al.* [37] defined a 2D time variant obstacle area to simulate in real-time obstacle dynamic properties including change in speed and direction in unmanned surface vehicles.

Different shapes were considered to estimate uncertainties. Lihua *et al.* [25] estimated GPS position by a cylindrical region centred at the agent position with radius and height equal to the horizontal and vertical position estimation accuracy. Yang *et al.* [38] mitigated control and sensing uncertainties by modelling obstacles as ellipsoids in 2D and a ball shaped regions in 3D. Similarly, ellipse were used to model uncertainty of intermediate points used in the construction of the path to the final goal while a bounding box is defined around the vehicle to provide a safety margin for uncertainties in the vehicle position [21]. This time variant bounding box is inflated by a multiple of the standard deviation as uncertainty propagates during path following [21]. The rate of inflation depends upon the distance from the origin (which is assumed to be known with certainty) and the visual scale as the latter determines the quality of the position estimate [21]. Also, ellipsoids were used by Pepy *et al.* [39] to model state distribution uncertainty of robot position and orientation at the configuration.

Probabilistic Distributions

Uncertainty in this category defines a specific or a set of unbounded distribution functions to estimate uncertainty for different parameters or agent states [24, 30, 32]. Robustness in probabilistic estimations will limit the failure probability depending on the

definition of distributions [26]. Van der Berg *et al.* [40] modelled obstacle uncertainty using a Linear-Quadratic-Gaussian (LQG). Positional uncertainty was also modelled with an independent Gaussian distribution in [41]. In this study, fixed obstacles were modelled using a constant measurement uncertainty. For quasi-static obstacles the centre of position remains at the initial location while for moving obstacles at constant speed the centres will move accordingly [41]. For both quasi-static and moving obstacles the positional uncertainty will grow linearly with time as long as the obstacles are not continuously observed [41, 42]. Florence [24] considers only continuous depth information in the local frame of reference for robustness in the presence of uncertainty in state estimation.

A normal distribution around the mean take-off position was used in [6] to account for initial uncertainty. A bounded normal distribution with mean about the desired heading was considered in the same study to mitigate with motion uncertainty. Similarly, Wen *et al.* [43] estimated the standard deviation of the probabilistic UAV state distributions at each tree node by Linear Quadratic Gaussian Motion Planning (LQG-MP). Uncertainty in threats were estimated by bounded circles with different risk factors. These studies showed that hybrid approaches are also an option that can be considered [43].

5.2.5. UNCERTAINTY QUANTIFICATION AND REDUCTION

Uncertainties for different parameters are linked with other mission, environmental and UAV modelling constraints. This section provides a resume of a sample of implementations with the associated uncertainty values. Rathbun *et al.* [27] limited speed between 21m/s and 34m/s, turn radius to a minimum of 0.18km and 1.52 times the fuel necessary to travel the distance from the initial position to goal. In the same work, the authors concurrently considered a bounded obstacle positional uncertainty of 0.1km, a bounded uncertainty for velocity of 5km/s and an uncertainty growth of 0.001km/s/s. Similarly, Cui *et al.* [16] considered a variable speed UAV with predefined bounds (10m/s to 50m/s) and initial speed of 25m/s in an obstacle free environment. Concurrently the authors considered an initial distance error of 30m, 30m and 2m, respectively in all dimensions, a horizontal velocity error of 2m/s, an angle error of 0.5^o and 10% distance and 0.2% angle uncertainties in the system error distribution. Other work considered a predefined holistic uncertainty. In this regard, Yang *et al.* [38] considered a bounded control uncertainty of 0.5m. As regards to obstacle size quantification Zeng *et al.* [41] represented each obstacle by a circle with a radius of 2× the standard deviation implying a confidence interval of 95.4%.

From the implementations described in this section it is shown that uncertainties in different parameters are defined at initiation stage propagating at predetermined rate as the mission progresses. Some researchers have identified ways how such uncertainty can be reduced. One way is by detecting using sensory vision systems identification tags placed on static non-enemy obstacles. As these are fixed sensory errors can be reset [20]. Others make use of previous information to attenuate uncertainty in already visited areas [6]. Another solution is to shorten the path planning time reducing uncertainty increase at an expense of reducing the path planning success and optimality and/or making the path planning solution more complex to compute [6]. Another ap-

proach is to consider only the latest sensory measurement data with the scope of reducing time propagated uncertainty [44–46].

5.2.6. Path planning Solutions under Uncertainty

Researchers have implemented a number of path planning solutions to mitigate uncertainties in different parameters. The following will outline the different path planning methods that were proposed.

Rapidly-Exploring Random Tree (RRT) Algorithm

The RRT algorithm is one of the most utilised path planning algorithms in the presence of uncertainty. Although this method has been successful implemented to plan in complex real world scenarios such as autonomous driving such as in the DARPA challenge [47], this algorithm does not explicitly incorporate uncertainty [26]. This inherent characteristic has encouraged researchers to apply the RRT algorithm in uncertain environments by adding uncertainty into the planned paths [48–50].

In this regard, the chance constraint RRT, an RRT variant, was developed and employed to handle uncertainty in model and environmental situational awareness in different implementations [24, 34]. This method is implemented to achieve robustness against uncertainties by growing trees of state distributions. However this method requires an accurate vehicle dynamics model to grow a tree of state distributions [26].

Another variant was developed by Yang *et al.* [38]. In this work the authors added a guiding attraction factor to the RRT variant, RRT* to enhance path convergence and smoothness in less computational time in the presence of fixed obstacles with uncertainty in both 2D and 3D environments. Auode *et al.* [51] remarked that RRT-based estimators offer both accuracy and efficiency for long-term prediction of dynamic threats in uncertain and partially unknown environments.

In another approach, Kothari *et al.* [26] generated probabilistically robust paths in partially known environments using the RRT algorithm. Kothari *et al.* [26] stresses about the need to modify the RRT algorithm to an anytime algorithm to arrive to a real time solution. The authors remark that based on the results the computational time increases approximately linearly with the number of obstacles.

The Dynamic Domain Rapidly-Exploring Random Tree (DDRRT) combined with a linear Quadratic Gaussian Motion Planning (LQG-MP) is developed by Wen *et al.* [43] to plan paths under threats and uncertainties. Static threats incorporating uncertainty were modelled using intuitionistic fuzzy sets while dynamic threats were modelled using the pursuit-evasion method. The path was further enhanced by the safety adjustment method and the RRT* variant of RRT. Results show that this system was able to construct online safe paths in uncertain and hostile environments.

Finally, Achtelik *et al.* [21] made use of the Rapidly-Exploring Random Belief trees to evaluate offline multiple path hypothesis in a known map with fixed obstacles so as to select a path exhibiting the motion required to estimate vehicle state. The planner consequently selects paths that minimise uncertainty state estimation in the estimated states without neglecting kinematic and dynamic model constraints. Also due to the sampling based nature of this approach no assumptions on discontinuities in measurement uncertainty with respect to vehicle position are considered. Results show that this

approach improves the precision of state estimation and that a naive planner will not be able to generate a successful path in an environment with bounded uncertainty in most cases. Furthermore, it was noted that uncertainty is close to null at the start, increasing during straight sections and when leaving feature-rich areas and decreasing at the goal provided a perfectly working re-localisation system. Results show that the scaling factor of the monocular vision-based system, that requires excitation, was fundamental to maintain the uncertainty within bounds whilst the UAV flew away from the origin. In this regard, authors remark that uncertainty bounding boxes need to be significantly enlarged when crossing unknown areas with uncertainty scale reducing in non direct paths while remaining unchanged for direct paths.

A* ALGORITHM

The use of the A* path planner to plan in the presence of uncertainty is not frequently used when compared to RRT and its variants. However this does not imply that this graph-based method is unsuitable to operate in an uncertain environment. In this regard, Liao *et al.* [19] developed a 3D, A* based, closed-loop path planning algorithm for Vertical Take-Off and Landing (VTOL) UAVs in GPS-denied, obstacle-rich environments. This algorithm provides collision-free shortest path from the current UAV position to any final goal point. The authors remark that this method is implicitly robust to uncertainty mainly due to the closed-loop approach.

MARKOV DECISION PROCESSES (MDP) AND PARTIALLY OBSERVABLE MARKOV DECISION PROCESS (POMDP)

The MDP algorithms generate policies that allow the agent to formulate a series of actions to maximise return or minimise cost functions without neglecting uncertainties [52]. MDP assume that states are completely observable while POMDPs incorporate uncertainties in sensing and partial observability of the agent [53, 54]. Vanegas *et al.* [20] developed an on-line POMDP algorithm that relies only on on-board localisation sensors to generate UAV control actions in the presence of uncertainty. To improve the quality of the control actions the POMDP algorithm is allowed to calculate for longer times although this will increase the uncertainty in motion due to longer prediction time [6].

SLIDING MODE CONTROL

Sliding mode control has inherent benefits that makes it a potential candidate for mobile robotics. Such benefits include robustness against system uncertainties, good dynamic response and stability under large disturbances [55]. Robustness against different uncertainties such as matched uncertainty in nonlinear systems is achieved by the sliding mode algorithm's systematic approach [56].

LINEAR METHODS

Linear algorithms can handle disturbances system control and model uncertainties and therefore can be used for path planning under uncertainty. These algorithms can describe the environment completely while modelling kinematic and dynamic constraints [57]. In this regard, Mixed Integer Linear Programming (MILP) methods merge binary and integer logical constraints to model the system and environment [57–59]. Optimal control can be used to find a path based on a set of differential equations [60]. This

method can be regarded as a variant of linear methods. In optimal control methods there exist a infinite number of variables' states as opposed to linear approaches. Using linear chance constraints to model uncertainties in optimal methods makes the method more computational efficient [57].

Reactive Path Planning Methods

Reactive path planning strategies are also potential candidates that can be employed to mitigate uncertainty in real-time. This approach uses only local knowledge of obstacles and threats to plan without generating a global plan either offline or at the initiation stage [5]. Reactive path planning systems make use of either vision-based or depth sensor-based systems to constantly update the path planning algorithm with real-time environmental information. For better reactiveness to uncertainties or changing environmental situation, feedback controllers can be employed. Reducing the complexity of feedback controllers can enhance robustness [4].

Potential Fields and Probabilistic Maps

Potential fields assign a potential function to free space with the agent reacting to forces due to repelling potential from obstacles with the goal node having the lowest potential [4]. A Laplacian potential field method was developed by Connolly [61] with the scope of minimising collisions with obstacle fields during random movement. Similarly, Lazanas *et al.* [62] considered potential fields to navigate through regions but with no uncertainty. Oppositely, Lihua *et al.* [25] developed the Modified Artificial Potential Field (MAPF) to navigate through an uncertain environment with randomly moving obstacles and pop-up threats. Obstacles are modelled by non-isotropic spatial volume creating repulsion potential fields. Results for test carried out at 5km altitude and maximum speed of 60m/s show an improvement over the standard Artificial Potential Field approach [25].

In the same category as potential fields are probabilistic maps that define the exposure to threats as a function of location and time. The undeterminisitic nature of dynamic environments has motivated Zengin *et al.* [63] to develop such a probabilistic map any apply it to the problem of dynamic path planning. The authors use state-space search to approach the goal location while avoiding threats.

Other methods

Path planning in dynamic environments with uncertainty are too complex for classical approaches to guarantee a solution. These algorithms tend to be inefficient and may lock in local minima without producing non-optimal paths [4].

Simultaneous Localisation and Mapping (SLAM) is important in situations of path planning under uncertainty, although it does not directly address the planning phase [4]. This method is able to provide localisation information and hence attenuate the level of uncertainty. SLAM has been used for real-time laser mapping in a remotely controlled helicopter flight [64] and to provide situational awareness under uncertainty in real-time in the DARPA Grand Challenge [65].

Receding Horizon Control (RHC) can also be employed for path planning under uncertainty. Goerzen *et al.* [4] remark that an online RHC framework can modify an offline generated path to handle in flight uncertainties that are either unknown or partially

known at the initiation stage. In this regard, Kuwata *et al.* [66] used the RHC method to generate trajectories for an agent under the influence of atmospheric turbulence.

Q-learning is an optimisation method used to deal with uncertainties. Cui *et al.* [16] modelled uncertainties to control the acceleration and bank angle of a UAV using tracking error cost function optimised through Q-learning in a 2D environment without obstacles.

Shell Space Decomposition (SSD) scheme is a method that decomposes the environment into concentric shells radiating out from the start to the goal location with single or multiple control points in each shell region [41]. Zeng *et al.* [41] applied SSD with a quantum-behaved particle swarm optimisation path planner for an Autonomous Underwater Vehicle (AUV) with static, quasi-static and moving obstacles in a cluttered and uncertain environment. Paths are generated offline and results show a better performance with concentric circles and sphere algorithms [41].

Finally, in another approach, Schouwenaars *et al.* [67] developed a robust system that accounts for uncertainty in manoeuvring of vehicles. This system is based on a control architecture developed by Frazzoli [68]. This control system is based on the quantization of system dynamics to reduce computational complexity of motion planning in nonlinear, high dimension environment.

5.2.7. CONCLUSION

This section highlighted the need to consider uncertainty during path planning especially in dynamic environments. The different uncertainty sources considered in various work were divided into 4 main categories. The two main uncertainty modelling methods namely bounded shapes and probabilistic distributions were explained based on their applications in different environments with different obstacles and threats. Another subsection presented a resume of parametric quantification of uncertainty in different working environments. Also this part highlighted how different authors tried to reduce uncertainty or its effects. Finally, the different path planning methods that were used in different work not only for UAVs were presented and the resulting outcomes analysed and compared. This review helps in defining a path planning strategy for an agent operating within an environment with the respective kinematic and dynamic constraints and uncertainties.

5.3. A*, RRT, SMOOTHING AND REAL-TIME ALGORITHMS

5.3.1. INTRODUCTION

This section will initiate with a brief definition of the two most utilised algorithms, namely the graph-based A* and sampling-based RRT algorithms. The smoothing algorithm developed to mitigate the non-optimality of the RRT algorithm will then be explained. Finally, this section will conclude with a summary of the A* and RRT real-time path planning implementations.

5.3.2. THE A* ALGORITHM

Graph-based methods define the state space into an occupancy grid defining obstacles residing in grid points as unavailable points. Graph-based algorithms check the possi-

bility of a path between start and goal position using only obstacle-free grid points [69]. Graph-based algorithms only offer a guarantee of solution if an appropriate resolution is selected [70].

The standard A* algorithm uses a heuristic evaluation function ($f(n)$) to determine the cost of neighbouring grid points [71]. This evaluation function sums the cost from the current position to a prospective future position and the cost from the latter to its goal node [71, 72]. For a detailed explanation of this algorithm refer to our previous work [11–13].

5.3.3. THE RAPIDLY-EXPLORING RANDOM TREE (RRT) ALGORITHM

Sampling-based algorithms construct a path between start and goal positions by connecting randomly selected obstacle-free points in the configuration space [70, 73]. Opposite to graph-based methods, these algorithms offer a guarantee of solution within an infinite time as opposed to graph-based methods provided that a path exists [70].

The standard Rapidly-Exploring Random Tree (RRT) constructs a unidirectional tree by randomly planting seeds in obstacle-free points. A new tree branch is constructed by selecting a new point a predefined distance from the nearest tree node on the line interconnecting the latter node with the randomly selected point, provided that the line from the nearest tree node and the new node is obstacle-free. A path from start to goal points is formed when one tree branch reaches the goal provided the first tree node is the start position [74–76]. The RRT algorithm is efficient in complex high-dimensional environments but lacks optimality and requires smoothing [76–78]. Just as for the A* algorithm, a more detailed explanation of the RRT algorithm and its variants is provided in our previous work [11–13].

5.3.4. THE SMOOTHING ALGORITHM

The path generated by both A* and RRT algorithms is constructed by interconnecting path points. The developed smoothing algorithm randomly selects two of these path points and then randomly defines two points on the path segments connecting the initially selected path points with their respective next path points. If the interconnection of the latter two points is obstacle-free then all path points in between are neglected, creating a shorter path with lesser turns and lesser path points. This process is repeated until the percentage path length reduction over the last 20 iterations is less than 1% and for a minimum of 20 iterations. A more detailed explanation of this algorithm is provided in our previous work [12, 13].

This post-path construction smoothing algorithm was developed to mitigate the non-optimality characteristics of the RRT algorithm. Since in our previous work [11, 12], the A* algorithm produced the shortest possible path with the considered resolution, improvement with the smoothing algorithm was only marginal. Moreover, in a real-time A* implementation some parts of a smoothed, obstacle-free path which are very near to obstacles, can produce collisions as the graph points can shift due to discretisation in the next intermediate path construction as the UAV current position moves. Therefore, this smoothing algorithm was only applied for the RRT algorithms.

5.3.5. The Real-time Algorithm

Literature identifies the need for real-time path planning if the UAV is expected to travel in dynamic environments especially if the environment is partially unknown or totally unknown except for a limited Field of View (FOV) at the initiation stage [79–81]. Owing that the scope of this work is to design a path planning algorithm for such scenarios, an algorithm to emulate real-time behaviour was developed in a previous paper [13].

The developed real-time path planning algorithm selects an obstacle-free intermediate goal node in the direction of the final goal position a look-ahead distance from the current UAV position. This look-ahead distance is governed by the sensory system's range and FOV. Consequently, a path from the current UAV position to the intermediate goal node is constructed (if possible) using either the A* or RRT algorithms. The UAV current position travels along the intermediate path a predetermined distance determined from the UAV speed and maximum allocated intermediate time. It is assumed that actuator systems' response is modelled with high fidelity with no uncertainty and the UAV is not affected by external factors. Furthermore it is assumed that in between intermediate path planning iterations the environment remains static. This assumption is valid since a low intermediate time is considered. No solution exists when either no intermediate obstacle-free path is constructed in the allocated time, the total time to reach the goal from the initial position has been exceeded, or when an obstacle collides with the UAV. The various UAV parameter definitions are briefly explained in Section 5.4.3. For a more detailed explanation of the described real-time algorithm and performance results for both path planning algorithms, refer to our previous work [13].

5.3.6. Conclusion

This section briefly described the most utilised graph-based and sampling-based method, i.e. the A* and RRT algorithms. To mitigate the non-optimality of the RRT algorithm a smoothing algorithm was also developed. The real-time algorithm made up of either the A* or RRT with the smoothing algorithm provided a test platform to assess the performance of both configurations in view of real life UAV applications.

5.4. Environmental Scenarios, UAV Model, Path Planner Parameter Definition and Uncertainty Modelling and Quantification

5.4.1. Introduction

This section will describe how uncertainty factors are integrated within the environmental and UAV model frameworks. This section will initiate with a description of the modelled environmental scenarios. Then the UAV model and path planner's parameter constraints will be described. Bounded uncertainty is then defined for size and obstacle positions while randomised but bounded uncertainty is considered for UAV positions. In the final subsection, the percentage uncertainty bound values applied to both obstacles and UAV position are described.

5.4.2. ENVIRONMENTAL SCENARIOS

Four different scenarios with different complexities developed in our previous work [10] are considered as a test platform to assess the performance of the A* and RRT algorithms. These time variant environments are generated offline with an associated time stamp before the initiation of path planning. Cubes, V shaped obstacles and 2D planes with window openings are used in the development of these obstacle scenarios. The cubes and V obstacles are randomly placed in the environment with each point having the same probability of occupancy. The following list describes the construction of each of the four scenarios, namely:

- Scenario 1. 10 cubes of 0.1x0.1x0.1 with no rotation;
- Scenario 2. 10 cubes the same size as Scenario 1 but with random rotation at definition stage and with changing independent rotation with time iterates;
- Scenario 3. 10 V obstacles constructed by adjoining one side of each of the two planes with an angle of 53° between the planes. Each plane has a size of 0.1x0.112 and is randomly rotated as in Scenario 2. This plane size was considered so that it fits exactly into the considered cube; and
- Scenario 4. Two planes on the Y-Z axis separated by 0.4 each with a window of 0.2x0.2 as well as the obstacles in Scenarios 2 and 3 combined.

Figure 5.1 illustrates pictorially each scenario for a random time iterate without considering any uncertainty bounds. Since moving obstacles are considered, the positions of each obstacle for each scenario will change in every time iterate.

Table 5.1: Obstacle shapes

Shape	Size	Number of Planes	Closed/Open
Cube	0.1x0.1x0.1	6	Closed
V obstacle	0.1x0.112	2; (53^o with each other)	Open
Plane with window	1x1	1; window (0.2x0.2)	Open

5.4.3. UAV MODEL AND PATH PLANNER PARAMETER DEFINITIONS AND CONSTRAINTS

To assess the performance of the A* and RRT algorithms, nominal UAV and path planner parameter values tabulated in Table 5.2 and adapted from our previous work [13] are derived after considering real UAV model characteristics, features and constraints.

A cube of 1x1x1 was considered as the environmental space, with fixed start and goal points at (0,-0.5,0) and (0,0.5,0), respectively for all considered tests. For A* algorithm the environment and start and goal points were shifted by a random distance between 0 and half the distance between grid points to eliminate path length ripple as explained in [12].

The resolution, considered for graph-based methods such as the A* is defined as the number of grid points per dimension. Oppositely in sampling-based methods all

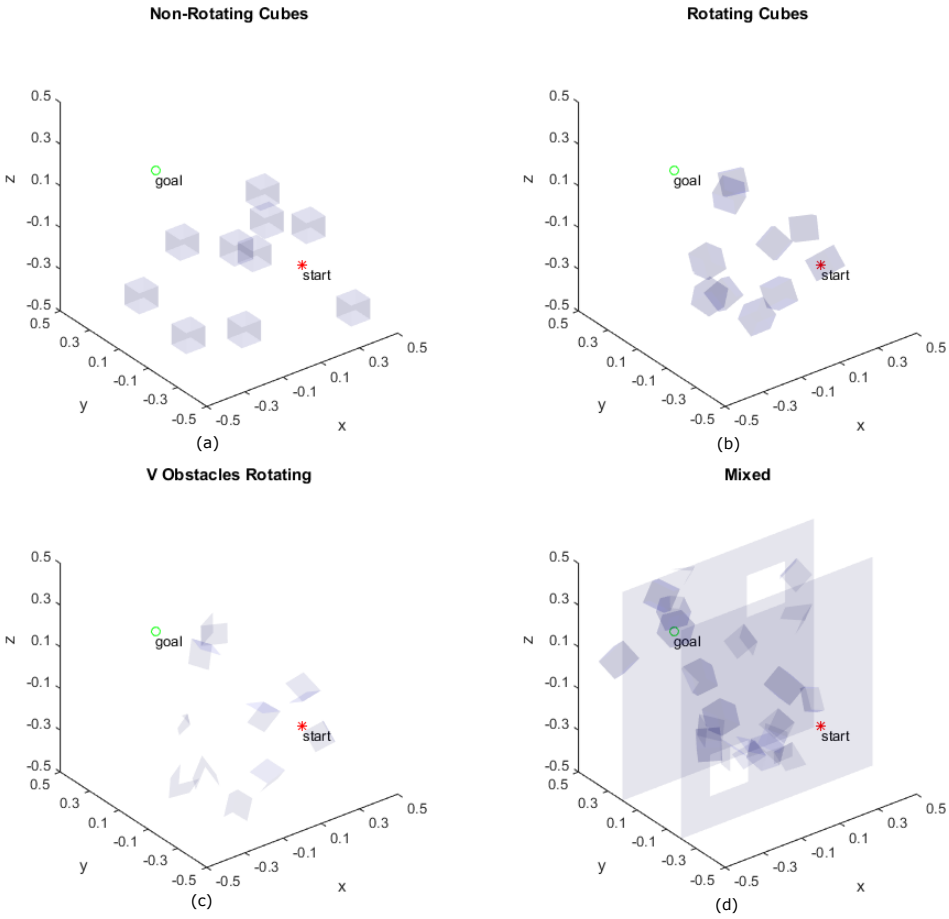


Figure 5.1: Environmental scenarios: (a) Non-rotating cubes scenario, (b) Rotating cube scenarios, (c) V obstacle rotating scenario and (d) Mixed scenario. These scenarios incorporate cubes, V obstacles and obstacle planes in the Y-Z with windows as openings.

Table 5.2: Real-time algorithm parameter definition

Parameter	Nominal Value	Units
Resolution (res)	21	[-]
Step size RRT (d_{step_RRT})	$\frac{1}{21-1} = 0.05$	[-]
Distance to travel per iterate (d_{s_step})	$\frac{2}{res-1} = 0.1$	[-]
Distance between current UAV position and prospective new intermediate goal point (d_{int_goal})	0.4 and 0.6 for Mixed case scenario	[-]
Maximum time to generate path segment ($t_{iterate\_max}$)	$\frac{d_{s\_step} \times 60 \times 60}{100 \times v_{UAV}}$	s
Maximum time to generate path ($t_{path\_gen\_max}$)	$10 \times t_{iterate\_max}$	s
Distance reduction factor (d_{factor})	0.8	[-]

obstacle-free space is available for path construction. Therefore, the tree branch length set as the distance between grid positions in A*, was considered for RRT to offer a fair comparison between the two methods. The distance to travel per iterate (d_{s_step}) is set at double the distance between grid positions and tree branch length while the distance between the current UAV position and an intermediate goal point (d_{int_goal}) is set to double d_{s_step}. For the mixed scenario, d_{int_goal} is increased from 0.4 to 0.6 since the planner needs to be able to visualise the second plane window to construct a path to the intermediate goal point that may result on final goal side of the second plane. If this distance is not increased the window will not reside within the range of the planner and consequently a no solution situation will result. All these parameters were set based on performance analysis in our previous work [13]. The maximum time to generate a path segment ($t_{iterate\_max}$) was defined as the time needed for the UAV to travel d_{s_step} while the maximum time to generate the whole path ($t_{path\_gen\_max}$) was set at 10 times $t_{iterate\_max}$. Finally, the distance reduction factor is used to reduce the distance in cases where prospective intermediate goal points or prospective UAV positions reside on an obstacle. Again this parameter was empirically defined after performance analysis in [13].

For each test situation the UAV speed remained constant. This parameter was varied between 0.01[-]/s and 0.1[-]/s in steps of 0.01[-]/s, where [-] represent modular distance units. These values were determined based on the nominal speeds for exploration situations in a nominal environment. The environmental time stamped space created $apriori$, assumed that obstacles move in random directions with random speeds smaller than the UAV speed.

5.4.4. BOUNDED UNCERTAINTY DEFINITIONS

In the literature review presented in Section 5.2, uncertainty was divided into four main categories, namely uncertainty in sensing systems, UAV model, environment and communication. The first three categories of uncertainties aforementioned will effect position and orientation of the UAV and obstacles within the range of the UAV sensing systems. Therefore, by modelling probabilistic and/or bounded uncertainties in position

and orientation of the UAV agent and obstacles, uncertainties in sensing systems, UAV model and environment will be catered for, as illustrated in Figure 5.2. Owing that our UAV is assumed to be stand-alone, no communication is exchanged with other third parties and therefore communication uncertainties are out of scope for the purpose of this work, as shown in Figure 5.2. Furthermore, for the scope of path planning it is assumed that the UAV is a point model moving at a specific speed in the intermediate goal point direction with no orientation. The actual UAV orientation will be taken into account by the path following algorithm which is no considered within the scope of this work. Therefore orientation uncertainty is also neglected.

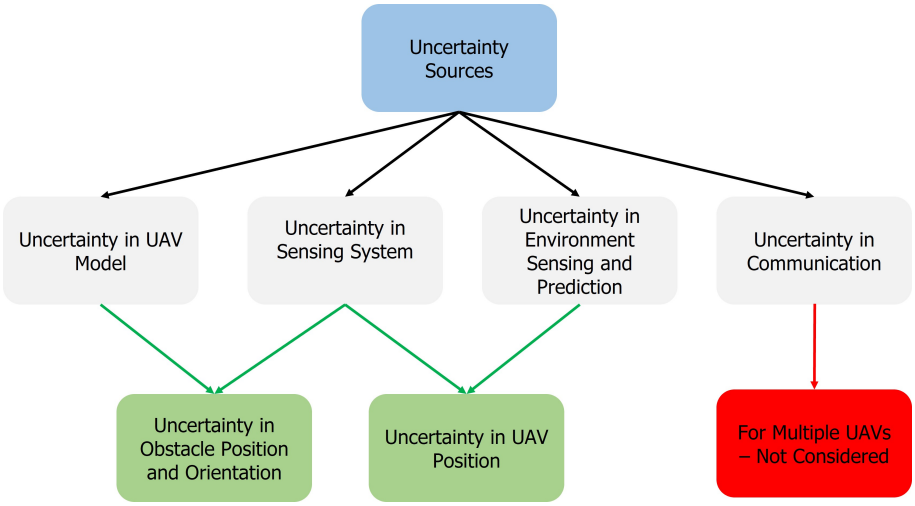


Figure 5.2: Illustration of uncertainty sources categorisation (refer Section 5.2.3), showing how these uncertainty sources are integrated within the developed uncertainty modelling environment

After considering both bounded and probabilistic uncertainty modelling methods, bounded uncertainty was considered to model uncertainty in both obstacles and UAV position. As described in literature, both methods were successfully applied to model different kinds of uncertainty in all four categories described earlier. The real-time path planning algorithms are designed to consider any point in the environmental space as either unoccupied or occupied by an obstacle, implying that the latter point is either free or unavailable. With probabilistic methods, each point has a probability based in the distribution function of either being free or unavailable and therefore the planner must define $apriori$ or in real-time a threshold to assign such point as free or unavailable. This will increase computational demand with respect to the bounded shape method which only defines a bound around the obstacle or UAV position and assumes that within the bound, parameters are known with certainty and outside the bound everything is totally unknown. Therefore through this rationale the latter method was selected.

Figure 5.3 illustrates the bounded boxes around the different obstacle shapes and the spherical bound around UAV position. 3D bounded boxes were considered since obstacle shapes can be regarded as 2D square planes. Therefore, 3D bounded boxes are shapes that offer equidistant bounds from the actual obstacle limits. A spherical

bound was considered for UAV position estimation as for fair modelling the uncertainty error must be equidistant in all directions for the estimated UAV position point. In this work, we will assume that uncertainty is time invariant in both obstacle position and orientation and UAV maximum deviation in position since we will assume that within the look-ahead distance, the sensing accuracy and precision of the UAV sensing systems and the UAV positional accuracy system are assumed to remain constant. Having said this, the effect on performance by varying uncertainty will be analysed in Section 5.5.

The actual UAV position in real-time path planning is assumed to reside with equal probability anywhere within the spherical bounds defined by the time invariant maximum deviation in position. Therefore for the scope of analysis, the current UAV position is randomly selected within the spherical bounds of the estimated future UAV position on the constructed path in the previous iterate. This randomisation process is repeated in the construction of each path segment while each scenario with a specific speed is repeated for 100 times.

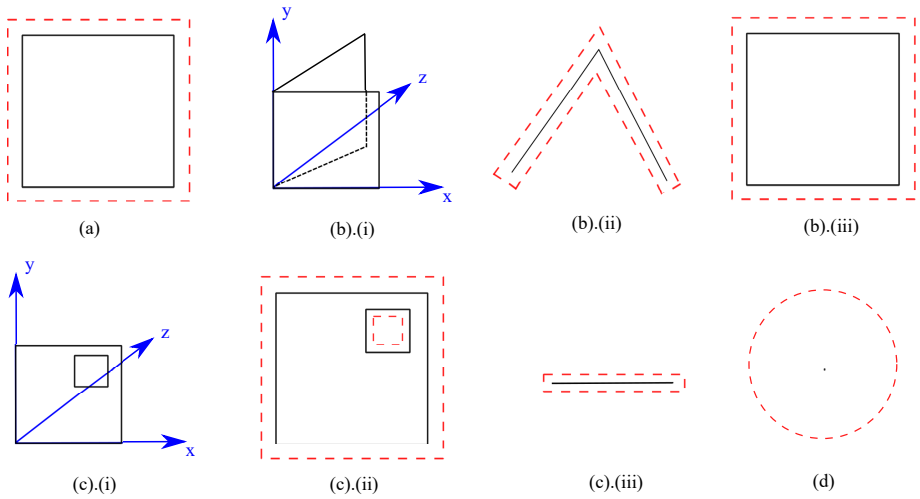


Figure 5.3: 2D illustrations of uncertainty bounds for all three categories of 3D obstacle shapes and the UAV position: (a) Cube (bounds are applied the same for all cube sides), (b) V-obstacle, (c) Planes in the X-Z with windows as openings and (d) UAV position

5.4.5. Uncertainty Quantification

Uncertainty in obstacle position and orientation is governed by either the precision and accuracy of the on-board UAV sensing systems only or by on-board sensing systems fused with other ground or associated off-ground sensing systems. In both cases, environmental situational awareness is limited and provided to the path planner with a degree of accuracy and precision depending upon the sensing capabilities of the aforementioned system or systems.

The uncertainty bounds of 3D obstacles namely cubes incorporate uncertainty by increasing the distance from the centre of the obstacle to each vertex by the predefined uncertainty. The connection of these new vertices will form the cubic uncertainty bounds.

The distance from the centre to the new vertex v ($d_{obst\_vertex}(v)$) is computed by Equation 5.1, where $\%_u$ denotes the percentage uncertainty, d_{obst_centre} denotes the distance from the centre of the obstacle to vertex v:

$$d_{obst\_vertex}(v) = \left(1 + \frac{\%_u}{100}\right) \times d_{obst\_centre}(v) \tag{5.1}$$

The new obstacle vertex is $d_{obst\_vertex}(v)$ distance from the centre of the obstacle on the line connecting the centre of the obstacle to the original obstacle vertex. The resultant uncertainty bounds will increase the percentage cube volume by multiple times with respect to the percentage uncertainty.

For the V-obstacle case, which is made of two 2D planes perpendicular to each other, uncertainty bounds are quantified by 10 planes enclosing the V-obstacle from all sides as illustrated in Figure 5.3 (b). Figure 5.3 (b).(i) illustrates the 3D view of the V-obstacle while (ii) and (iii) shows the X-Y and X-Z views incorporating uncertainty bounds, respectively. The percentage uncertainty factor in obstacle position is multiplied by the distance between the centre of the shape and each node of the obstacle and then added to the actual node position to identify the nodes of the 10 enclosing planes such that the distance between any node on the V-obstacle planes and the respective bound limit is equidistant. Therefore the uncertainty bounds of the enclosing planes around the V-obstacles are calculated as the cube obstacles using Equation 5.1.

For the Mixed case, incorporating all shapes including 2D planes with windows, the uncertainty bound for planes is constructed by using two other parallel planes with smaller windows with the edges of the parallel planes connected with each other to form a closed shape as illustrated in Figure 5.3 (c). Figure 5.3 (c).(i) shows the 3D view of the plane with a window in the X-Z plane while (ii) and (iii) show the X-Z and X-Y views with uncertainty bounds, respectively. The Mixed case scenario consists of two planes with one window each. Each plane with a window consists of 4 planes each described by 4 vertices. Since the developed planes occupy 96% of the area in the particular plane, the bounding planes cannot occupy more than 99% in the respective plane. Otherwise, the bounding planes will completely block the whole space, leaving no space for the planner to find a path to goal. In this regard, to ensure a bounding plane occupation of not more than 99%, the maximum bound increase ($bound_{max}$) is set at 0.05. The distance from an original plane vertex v to the new vertex $d_{plane\_vertex}(v)$ is calculated using Equation 5.2. The positions of the new vertex for bounding planes are defined at $d_{plane\_vertex}(v)$ distance from the original vertex.

$$d_{plane\_vertex}(v) = \left(bound_{max} \times \frac{\%_u}{100}\right) \tag{5.2}$$

The nominal uncertainty percentage values of obstacle shapes are decided based on the uncertainty quantification considered in different studies and presented in Section 5.2.5. As regards, obstacle uncertainty three studies were presented one that considered probabilistic and the others considered bounded uncertainty. Yang *et al.* [38], considered a constant circular and spherical uncertainty bound of an additional 0.5m radii for 5m radii obstacles for 2D and 3D environments, respectively. This amounts to 10% uncertainty bounds. Rathbun *et al.* [27], considered a 0.1km positional uncertainty

for obstacle aircraft which amounts to 2x the size of a medium sized passenger aircraft. This environment is different from the environment under review since in this case the size of obstacles and obstacle density is much larger with respect to the environmental space than Rathbun *et al.* study and thus only Yang *et al.* [38] is considered for the definition of uncertainty bounds for obstacles.

The maximum uncertainty in UAV position is modelled as a sphere as illustrated in Figure 5.3 (d). The radius of the uncertainty sphere r_{u_sphere} is equal to the multiplication of the respective percentage uncertainty and the distance the UAV moves per iterate d_{s_step}, as defined in Equation 5.3.

$$r_{u\_sphere} = \frac{\%_u}{100} \times d_{s\_step} \tag{5.3}$$

Cui *et al.* [16], considered a 10% distance and 0.2% angle uncertainty in system error distribution in an obstacle free scenario. The selected spherical uncertainty bound will be larger with respect to Cui *et al.* [16] for the considered speed since 0.2% angle uncertainty will translate to approximately 1.3% lateral deviation and in our case we are assuming a maximum deviation of 20%. This allows a safer navigation using a less accurate and precise positioning systems.

For all considered scenarios the uncertainty in either obstacle position and/or UAV position was varied between 2% and 20% in steps of 2% when the effect of this uncertainty on performance is under review. Otherwise, it is assumed that uncertainty is neglected.

5.4.6. CONCLUSION

This section presents the four different environmental scenarios with different difficulties that are used to assess the effect on path planning performance in real-time in the presence of time invariant uncertainty in both obstacle position and orientation as well as UAV position using bounded shapes. The considered uncertainty factors and their quantification incorporate a set of major uncertainty sources although other uncertainty factors neglected in this study may negatively impact performance of the considered path planning algorithms in real-time.

5.5. RESULTS

5.5.1. INTRODUCTION

The A* and RRT 3D real-time path planning algorithms, environmental scenarios and uncertainty in UAV and obstacle position and obstacle orientation described in the previous sections were implemented in MATLAB and tested using an Intel Xeon ES-1650, 3.2GHz. The path length, computational time and success rate are the performance measures considered. Unless uncertainty is under review, UAV speed is varied in steps of 0.01[-]/s, starting from 0.01[-]/s to 0.1[-]/s as described in Section 5.4, otherwise the UAV speed is set at 0.05[-]/s, while the parameters tabulated in Table 5.2 were assigned to the nominal values.

5.5.2. A* AND RRT WITH NO UNCERTAINTY

Before the inclusion of uncertainty in obstacle and UAV positions, the performance of both the A* and RRT algorithms in real-time in the presence of moving obstacles is assessed. This is important so as to compare the effect on performance of uncertainty additions to both A* and RRT. Figure 5.4 adapted from [10] shows the path length, computational time and success rate for A* and RRT as speed is varied between 0.01[-]/s to 0.1[-]/s for the constant parameters illustrated in Table 5.2.

The salient points that can be concluded from Figure 5.4 is that the path length for the first three scenarios namely the cube with no rotation, the cube with rotation and the V-obstacle produce a path very close to 1 for all scenarios for all speeds for both algorithms. The shortest path length is recorded for the cube with no rotation followed by the cube with rotation and the V-obstacle also for all speeds for both algorithms. The straight line distance from start to goal is 1 implying that the planner only deviated by a small percentage from the straight line owing that the obstacles are rarely in the line of sight from start to goal. A major increase in path length is recorded for both A* and RRT for all speeds as the planner needs to traverse through two planes with windows on opposite sides as illustrated in Figure 5.1. Overall the path length for RRT was shorter than A* for all scenarios and all speeds considered. Also speed has no effect on path length since speed only determines the maximum intermediate and total time allocation.

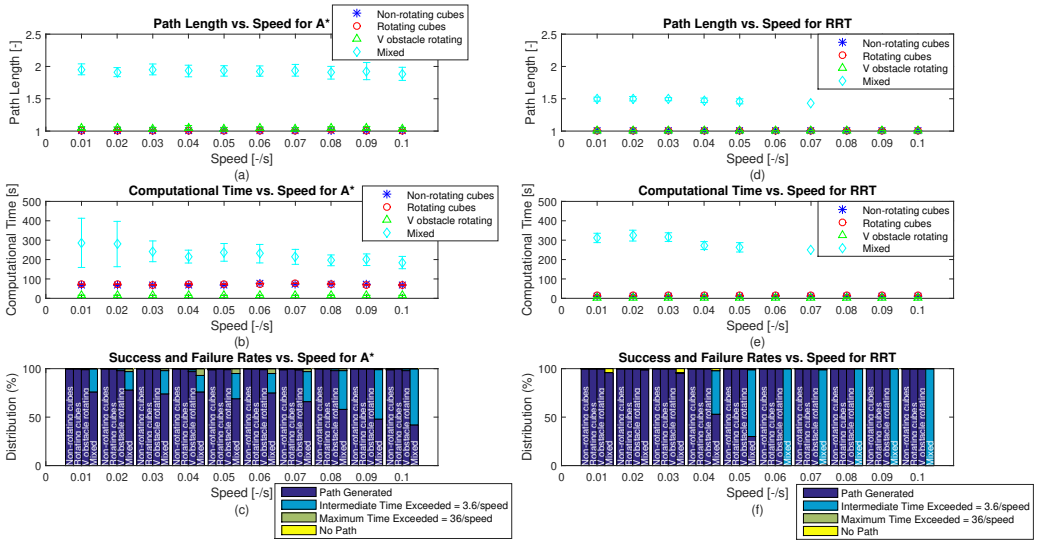


Figure 5.4: Performance parameters vs. speed: (a) Path Length for A*, (b) Computational Time for A*, (c) Success and Failure rates for A*, (d) Path Length for RRT, (e) Computational Time for RRT and (f) Success and Failure rates for RRT for 100 iterates for each considered situation (speed and scenario) with 95% confidence interval. ($res = 21, d_{s\_step} = 0.05, d_{int\_goal} = 0.4$ (for Scenarios 1,2 and 3) and $d_{int\_goal} = 0.6$ (for Scenario 4), $d_{factor} = 0.8$ and $t_{iterate\_max}, t_{path\_gen\_max}$ are a function of the UAV speed) Adapted from [10]

The computational time for A* is longer than that for RRT for all scenarios for all speeds. As for path length speed has no effect on computational time for both algorithms

for the reason described previously. The V-obstacle resulted in the lowest computational time followed by the cube without and with rotation for both algorithms with a major increase for the Mixed case. This order is attributed to the computational difficulty associated with each scenario with the V-obstacle consisting of only two 2D planes while cubes consist of 6 planes and the internally unavailable space. For both path length and computational time A* results in higher variance with respect to RRT for all scenarios for all speeds. This is attributed to the graph-based nature of the A* algorithm and the ripple reduction algorithm that re-plans the environment each iteration. Refer to [10, 12] for further details.

In terms of success rate both algorithms exhibit a near 100% success rate for the first three scenarios with RRT exhibiting a 100% success rate for the mentioned scenarios for all speeds while A* achieved a minimum of 95% success rate for the same set of scenarios. For the Mixed case, the RRT achieved a success rate of a minimum of 95% for speeds from 0.01[-]/s to 0.03[-]/s deteriorating to 0% as speed increases. The A* on the other hand never achieved this high success rate of the RRT algorithm but maintained success rate above 40% at the highest speed which is the worst case scenario. For further analysis of these results refer to Chapter 4 and [10].

Overall the performance of both the A* and RRT algorithms show that they can be applied to real-time path planning in the presence of moving obstacles as long as the allocated intermediate and total time is appropriate. The next step is to include uncertainty in UAV position as explained in Section 5.4.

5.5.3. A* AND RRT WITH UNCERTAINTY IN UAV POSITION

The inclusion of uncertainty in UAV position is factored as a percentage of the distance that the UAV is expected to move per iterate which is predetermined. During planning the path is expected not to reside less than this distance away from each obstacle. Figure 5.5 illustrates the performance response of the A* and RRT algorithms with uncertainty in UAV position. In this test two inter-related parameters are changing concurrently. One is the maximum uncertainty in UAV position that dictates the safe distance that the planner must keep to ensure a non-collision provided the obstacle remains fixed. Secondly, when the UAV is simulated to traverse the path in real-time, a random shift by a random distance in a random angle varying from 0 to the considered maximum UAV positional uncertainty is added to the calculation of the next intermediate UAV position on the previously constructed path to the intermediate goal point. This random distance is bounded between 0 and the maximum uncertainty defined in the horizontal axis of Figure 5.5.

The mean shortest path length for A* results is recorded for the non-rotating cube case increasing when rotation is introduced and increasing further in the V-obstacle and Mixed cases. For RRT, the V-obstacle is the shortest followed by the cube without and with rotation and the mixed scenario, although the difference in the first three scenarios is small. This order is equivalent for the no uncertainty cases for A* but changes with RRT mainly because the V-obstacle case experienced the largest deterioration. The V-obstacle rotating case is made up of two planes with each one having an area greater than each side of the cube. Due to the buffer distance the available volume between planes is reduced and therefore the planner needs to pass from outside in a larger num-

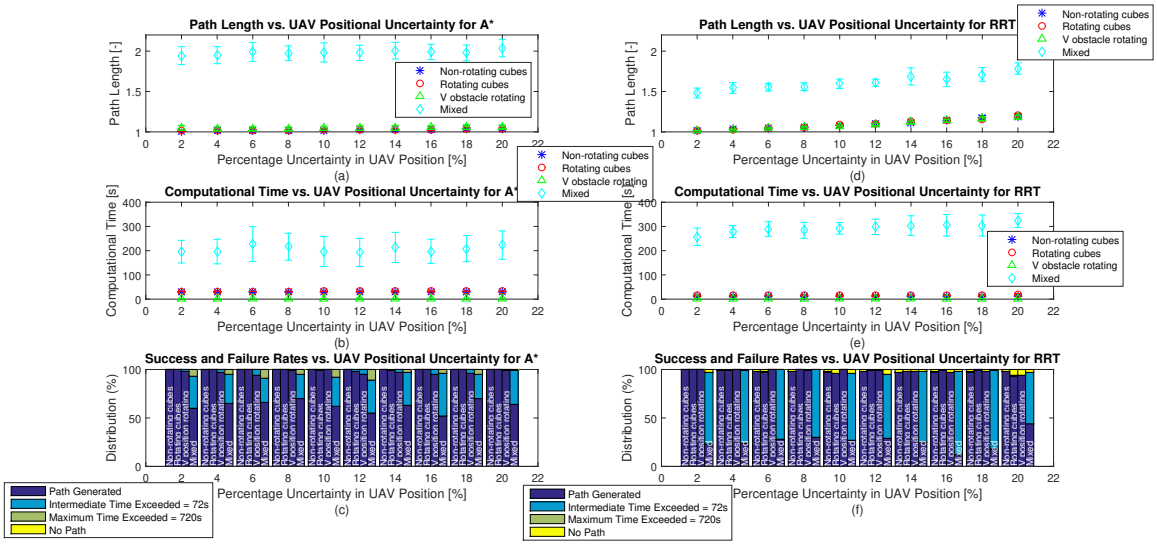


Figure 5.5: Performance parameters vs. Uncertainty in UAV position: (a) Path Length for A*, (b) Computational Time for A*, (c) Success and Failure rates for A*, (d) Path Length for RRT, (e) Computational Time for RRT and (f) Success and Failure rates for RRT for 100 iterates for each considered situation (obstacle uncertainty and scenario) with 95% confidence interval. ($res = 21, d_{s\_step} = 0.05, d_{int\_goal} = 0.4$ (for Scenarios 1,2 and 3) and $d_{int\_goal} = 0.6$ (for Scenario 4), $d_{factor} = 0.8$, $t_{iterate\_max} = 72s$ and $t_{path\_gen\_max} = 720s$.

ber of iterations, resulting in a longer path. As for the non uncertainty cases, it can be concluded that rotation increases path length since the effective movement of the cube is higher effectively requiring the UAV to travel further away to avoid collisions.

The path length for the first three scenarios is 2 times and 1.5 times for the Mixed case with respect to the other cases for all speeds for the A* and RRT algorithms, respectively. As for the no uncertainty cases this increase results since the planner needs to travel through obstacle plane windows on opposite sides of two parallel planes spaced by 0.4[-] from each other. Due to the graph-based nature of A* obstacles are defined with a buffer of half the distance between grid positions while for RRT every point not on an obstacle plane or within the object is accessible.

The mean path length for RRT is 7.4%, 6.9%, 4.0% longer and 18.4% shorter with respect to A* for Scenarios 1 to 4, respectively. For Scenarios 1 to 3 the success rate is close to 100% for both algorithms therefore a fair comparison can be made. But for the Mixed case, A* exhibited a higher success rate that could effect the mean path length owing that the easiest iterates (which are ultimately successful) are considered for RRT with more difficult iterates besides the easiest ones are considered for A*. This is also applicable to computational time analysis. Therefore, without uncertainty the path length recorded for RRT was always smaller than that for A*. So with the inclusion of UAV positional uncertainty both algorithms experienced an increase in path length with the major deterioration exhibited by the RRT algorithm. In fact, with the inclusion of uncertainty, A* exhibited a mean increase of 1.8%, 1.7% and 1.2% and a decrease of 2.7% for Scenarios

1 to 4, respectively for the same speed. A larger increase of 9.2%, 9.1%, 8.9% and 10.7% is recorded for RRT for Scenarios 1 to 4, respectively for the same speed. The decrease in path length for A* for the Mixed case is attributed to the lower success rate for A* for this case and therefore the easiest iterates are considered as explained earlier.

Further analysis of Figure 5.5 (a) and (d) shows that for both algorithms for all scenarios the path length increases with percentage uncertainty in UAV position. Refer to Section 5.4.5 for percentage uncertainty definitions. For A* the path length increases by 2.7%, 2.8%, 1.2% and 4.8% for Scenarios 1 to 4, respectively for uncertainty increases from 2% to 20%. A similar analysis for RRT shows that the path length increases by 16.7%, 18.2%, 16.7% and 20.4% for Scenarios 1 to 4, respectively. From this analysis, the comparison between the mean path length of A* and RRT and the comparison between the A* and RRT with and without uncertainties, it can be concluded that both algorithms experience an increase in path length with uncertainty and the RRT is more susceptible to increase in uncertainty in UAV position with respect to A*. For A* a buffer distance in obstacle definition is already considered even without uncertainty at half the distance between grid positions which is equal to 0.05[-] for the considered resolution. Therefore, for A* two buffer distances are added when uncertainty in position is under investigation while for RRT only positional uncertainty is included leading to a larger risk of collision for RRT with respect to A*. The major percentage path length increase is recorded for the Mixed case since it incorporates more than double the obstacles, restricting the construction of intermediate paths.

Computational time is lowest for the V-obstacle case for both algorithms for all ranges of uncertainties considered since the planner needs to only check for collision with only two planes per shape not six planes and their interior per shape as in the cube case. As described for other situations rotation increases computational time as the effective movement of the obstacle increases. Since the mixed case is more complex than the other three cases the required computational time is higher as a consequence.

With the inclusion of uncertainty A* recorded a decrease in mean computational time by 2.29 times, 2.29 times, 5.60 times and 3.7% for Scenarios 1 to 4, respectively for the same speed. Oppositely, for RRT an increase of 16.3%, 8.9%, 13.1% and 11.6% is recorded for Scenarios 1 to 4, respectively for the same speed. The decrease exhibited for A* results since, with uncertainty, the planner keeps a larger distance from the obstacles and therefore the chance of re-planning due to obstacle movement decreases while the obstacle definition time and path planning remain the same. This reduces computational time at the expense of longer path length. For RRT a smaller increase in computational time results with increase in uncertainty. By increasing uncertainty in RRT, tree branches are more restricted and therefore the planner requires more time to construct the path although as for A* the re-planning time reduces. But since the path construction time is greater than the re-planning time, the combined effect is that the total computational time increases.

The mean computational time for A* is 2.43, 2.04, 1.13 and 0.704 times with respect to RRT for Scenarios 1 to 4, respectively. As for the non-uncertainty case, for both algorithms, the shortest computational time is recorded for the V-obstacle case followed by the cube without rotation and with rotation and the Mixed cases. Although, without and with uncertainty A* reduced and RRT increased in computational time, this difference is

not enough for the A* to perform better than RRT in the first three scenarios. Also, results confirm that for complex cases, the A* can construct a path in less time than RRT with and without positional uncertainty. This is because irrespective of complexity the obstacle definition in the graph environment is not affected as all points within the range need to be checked for occupancy. Oppositely for RRT, the larger the size and number of obstacles the higher the computational time required to check for collisions.

The success rate for the first three scenarios is over 90% for all percentage uncertainties for both algorithms considered with all three scenarios. A deterioration of approximately 5% with increase in uncertainty is visible only for RRT with A*'s success rate independent of uncertainty consideration. As remarked for path length and computational time with increase in buffer distance to obstacles the RRT is mainly affected as such buffer restricts prospective tree branches while for A* it does not effect obstacle definition within sensing range, the most computational intensive part of the algorithm. For the Mixed case, a success rate between 52% and 70% and between 12% and 44% is recorded for A* and RRT, respectively. This shows that for the range of uncertainties considered for the Mixed case, the A* has double the chance of finding a feasible path from start to goal.

In comparison with the no uncertainty scenarios, an approximately 5% success rate deterioration is recorded for both algorithms for the first three scenarios for all the range of uncertainties considered. For the Mixed case, a success rate similar to the no uncertainty case for the same speed is recorded for both algorithms for all the range of uncertainties. As for the no uncertainty cases, the maximum intermediate time restriction is the main violation, especially in the Mixed case scenario for both algorithms. For A* a total time violation is also illustrated in less than 5% of the cases in the Mixed scenario. Oppositely, for RRT a number of No Path solutions are recorded (also less than 5%) for all scenarios increasing with increase in uncertainty. For A*, No Path solutions are less probable as in addition to UAV positional uncertainty buffer distance another distance equivalent to half the distance between grid positions is added to avoid quantisation errors that could lead to erratic obstacle definitions.

In conclusion, both algorithms resulted in almost 100% success rate for the first three scenarios for all uncertainty values considered with RRT resulting in shorter path length and computational time with respect to A*. Both algorithms show an increase in path length and computational time with increase in uncertainty bounds with RRT showing the larger growth. For the Mixed case, A* is 2 times more successful than RRT for all uncertainty bounds with respect to RRT although A* constructed longer paths.

5.5.4. A* AND RRT WITH UNCERTAINTY IN OBSTACLE POSITION AND ORIENTATION

Figure 5.6 illustrates the performance response of the A* and RRT algorithms with increase in obstacle uncertainty bounds. The uncertainty bounds are modelled and calculated as described in Section 5.2.5.

Figure 5.6 (a) and (d) illustrate the path length for A* and RRT, respectively. The response is similar to the no uncertainty cases with the difference that no results are illustrated for the Mixed case scenario for the RRT algorithm since no successful runs were

recorded for this case. In fact, the mean path length for A* increased by only 0.6%, 0.3%, 1.7% and decreased by 1.5% for the cube without and with rotation, V-obstacles and Mixed cases for the uncertainty with obstacle uncertainty with respect to the no uncertainty cases. Similarly, the same comparison for RRT show that the mean path length increased by only 0.3%, 0.2% and <0.1% for the first three scenarios. The small increase in the first three scenarios is attributed to the relative small increase in obstacle volume occupation. For the Mixed case, the decrease in mean path length results due to a reduction of more than 50% in success rate in the obstacle uncertainty test case with respect to the no uncertainty test cases. With lower success rate the easier paths are successful.

In comparison with the UAV positional uncertainty cases a lower increase in path length results for all scenarios for both algorithms as the uncertainty increases. In fact, comparing the path length at 20% with respect to 2% uncertainty an increase of 0.2%, 0.2%, 0.2% and 2.2% results for the non-rotating and rotating cube, V-obstacle and Mixed cases, respectively for the A* algorithm. A similar analysis for RRT showed an increase of <0.1%, 0.4% and 0.1% for the first three scenarios, respectively. For both algorithms, the variance is invariant with increase in obstacle uncertainty. Therefore, it can be concluded that the inclusion of obstacle uncertainty bounds have minimal effect on path length in the first three scenarios for both algorithms mainly due to the low occupation and also for the Mixed case using the A* algorithm.

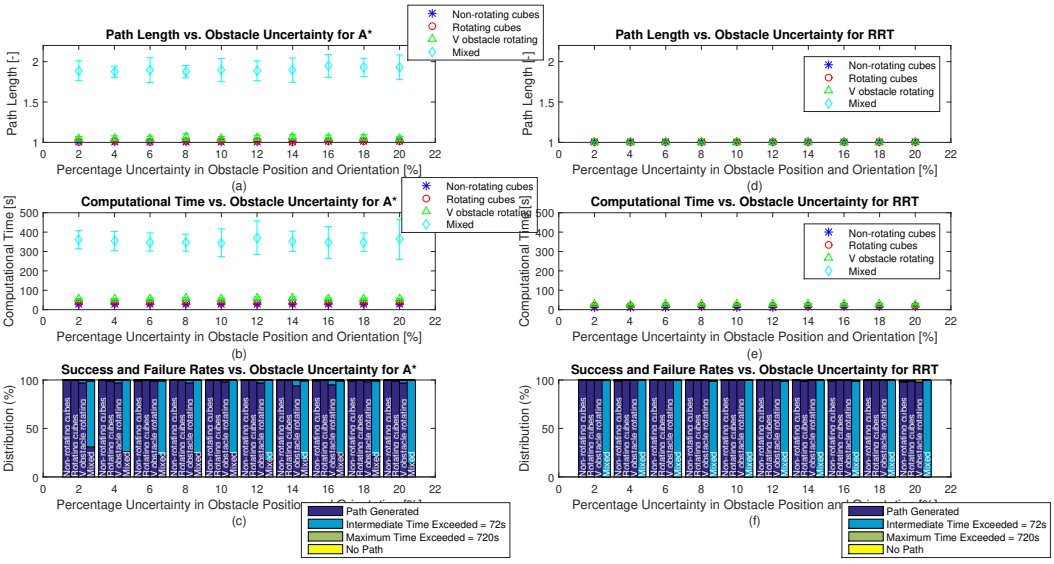
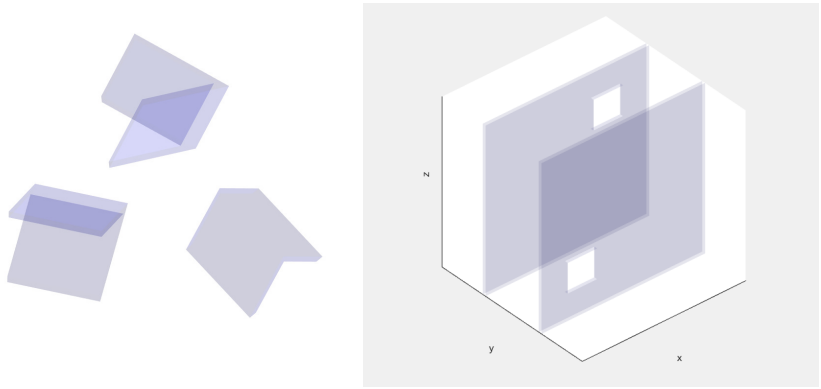


Figure 5.6: Performance parameters vs. Uncertainty in obstacle position and orientation: (a) Path Length for A*, (b) Computational Time for A*, (c) Success and Failure rates for A*, (d) Path Length for RRT, (e) Computational Time for RRT and (f) Success and Failure rates for RRT for 100 iterates for each considered situation (obstacle uncertainty and scenario) with 95% confidence interval. ($res = 21, d_{s\_step} = 0.05, d_{int\_goal} =$ 0.4 (for Scenarios 1,2 and 3) and $d_{int\_goal} = 0.6$ (for Scenario 4), $d_{factor} = 0.8$, $t_{iterate\_max} = 72s$ and $t_{path\_gen\_max} = 720s$.

The computational time results illustrated in Figure 5.6 (b) and (e) show that the cube with no rotation case exhibits the fastest time followed by the rotating cube, V-obstacle



(a) (b)

Figure 5.7: Two Dimensional objects modelled with uncertainty in obstacle position and orientation (20% test case): (a) V-obstacles (zoomed) and (b) Planes with windows constructed in Mixed Scenario.

and Mixed scenarios for both algorithms. This order is different for both algorithms to both the no uncertainty cases and the uncertainty in UAV position test cases. In fact, for A* the computational time is 0.427 and 0.429 times for the cube without and with rotation, respectively and 3.72 and 1.49 times for the V-obstacle and Mixed scenarios, respectively for the obstacle uncertainty case with respect to the no uncertainty cases. A similar comparison for RRT shows that the computational time increased by 23.6%, 26.3% and 12.9 times for the first three scenarios, respectively.

The analysis shows that with the increase in effective inaccessible volume of the cubes, A* required less computational time while RRT required more time with respect to the no uncertainty case. Adding uncertainty bounds around obstacles makes it more difficult to construct a path as the environment is more restrictive. This explains why RRT requires more time to construct tree branches to find a path from start to goal. For A*, the increase in the number of inaccessible grid positions will not increase the environment grid definition time as explained earlier and will make it easier to find a path, if possible, since fewer amount of free nodes are available.

For the V-obstacle case both algorithms experienced an increase with RRT showing the larger increase. The inclusion of obstacle uncertainty has changed the V-obstacle from a 2D shape to a 3D shape. This increases computational demand to define the grid environment for A* since the grid definition sub-algorithm needs to check and assign inaccessible grid points found internally within the obstacle. For RRT, the V-obstacle restricted the environment making it more difficult to construct tree branches. For the Mixed case scenario, the environment is more restrictive resulting in fewer options of tree branch construction, leading to a 0% success rate. For A*, the increase in computational time is attributed to the increase in grid definition time due to the change of 2D to 3D obstacles for the V-obstacles and planes.

Although the RRT experienced the higher increase in computational time, as for the non uncertainty situation, the RRT algorithm constructed the path in less time than the A* algorithm for the first three scenarios, implying that A*'s time consumption in environmental quantisation is larger and path construction than RRT's approach even with

the inclusion of uncertainty. This shows that the RRT algorithm remains more computationally efficient with respect to A* for simple scenarios with the inclusion of obstacle uncertainty in real-time.

Comparing the effect on computational time for obstacle uncertainty with respect to UAV positional uncertainty for A* shows a minor decrease of 2.1%, 1.6% and a relatively larger increase of 20.9 times and 71.0% increase for the cube without and with rotation, the V-obstacle and Mixed case scenarios, respectively. This results since the increase of the cube size will increase the number of inaccessible grid positions for the obstacle uncertainty case remaining the same size for the positional UAv uncertainty. For the V-obstacle and Mixed cases the 2D to 3D shift in obstacle modelling for the obstacle uncertainty case is the reason for the increase with respect to the UAV positional uncertainty case in which the V-obstacles and planes remains 2D. A similar analysis for RRT shows a 6.3%, 15.9% and 11.4 times increase for the first three scenarios, respectively when comparing the mean computational time with obstacle uncertainty with respect to UAV positional uncertainty. Obstacle uncertainty makes the environment more restrictive due to decrease in available volume resulting in larger computational demand requirement especially for the V-obstacle case.

The success rates for both algorithms illustrated in Figure 5.6 (c) and (f) show that a close to 100% success rate is recorded for both algorithms for the first three scenarios with a major drop for the Mixed case scenario for A* and 0% success rate for RRT. This shows that for the first 3 cases the success rate is indifferent for both algorithms in the range of uncertainty considered. For complex cases A* outperforms the RRT algorithm. The decrease in success rate as obstacle uncertainty increases is illustrated in the Mixed case for A* with a drop from 30% to 14% at 2% and 20% obstacle uncertainty, respectively. This is not shown for the other tests since the allocated time is either too high (First 3 scenarios) or too low (Mixed case, RRT). In fact, the maximum intermediate time violation is the major cause for unsuccessful runs.

In comparison with the no uncertainty cases, results for A* show no difference in success rate for the cube cases with a minor drop from 99% to an average of 96.9% in the V-obstacle due to the increased volume occupation. On the other hand a major drop from 69% to 22.2% is recorded for the Mixed case. The latter case includes both planes and V-obstacles that changed from 2D to 3D obstacles effectively increasing complexity. A similar analysis for RRT shows that for the first three scenarios the mean success rate with obstacle uncertainty is larger than 99.5% as opposed to 100% with no uncertainty. This implies that the time restrictions are too large for these cases. For the Mixed case a drop from 30% to 0% is recorded.

In comparison with UAV positional uncertainty, overall results show that obstacle uncertainty deteriorates success rate more than UAV positional uncertainty for the whole range of uncertainties considered for the Mixed cases with a less than 2% difference for the first three cases for both algorithms. For A*, a difference of less than 1% is recorded for the first 3 scenarios and a drop from 62.7% to 22.2% for the Mixed case for UAV positional and obstacle uncertainty, respectively. Similarly for RRT, a difference of less than 2% is recorded for the first 3 scenarios with a drop from 26% to 0% in mean success rate for the Mixed case.

In conclusion, the result show that a larger deterioration than the UAV positional

uncertainty results for the obstacle uncertainty especially for RRT mainly for the V-obstacles and Mixed cases in terms of path length, computational time and success rate.

5.5.5. A* AND RRT WITH UNCERTAINTY IN OBSTACLE POSITION AND ORIENTATION AND UAV POSITION

Figure 5.8 illustrates the performance response of the A* and RRT algorithms with the combined increase of obstacle position and orientation and UAV position uncertainties. The modelling of these uncertainties is as described in the previous two sub-sections and the scope of this analysis is to analyse the combined effect of these two unrelated uncertainties on the path planning performance of both algorithms at a predetermined speed.

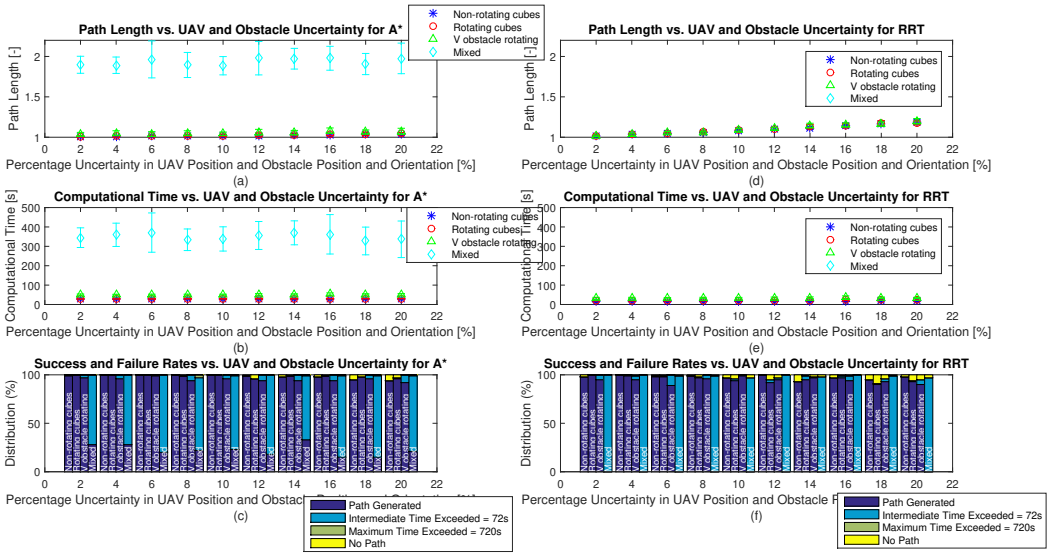


Figure 5.8: Performance parameters vs. Uncertainty in obstacle position and orientation and UAV position: (a) Path Length for A*, (b) Computational Time for A*, (c) Success and Failure rates for A*, (d) Path Length for RRT, (e) Computational Time for RRT and (f) Success and Failure rates for RRT for 100 iterates for each considered situation (obstacle and UAV uncertainty and scenario) with 95% confidence interval. ($res = 21, d_{s\_step} = 0.05, d_{int\_goal} = 0.4$ (for Scenarios 1,2 and 3) and $d_{int\_goal} = 0.6$ (for Scenario 4), $d_{factor} = 0.8$, $t_{iterate\_max} = 72s$ and $t_{path\_gen\_max} = 720s$.

Figure 5.8 (a) and (d) illustrate the path length for A* and RRT, respectively. The response in path length is a combination of the results of Figure 5.5 (a) and (d) and Figure 5.6 (a) and (d). For both A* and RRT algorithms an increase in path length results with increase in uncertainty with the major increase shown for RRT. For both algorithms the cube without rotation case produced the shortest path followed by the cube with rotation, V-obstacle and Mixed scenarios. Path length increases by 3.3%, 3.4%, 3.3% and 4.2% for the cube without and with rotation, V-obstacle and Mixed case scenarios, respectively for the A* algorithm as uncertainty increases from 2% to 20%. Similarly for

RRT, an increase in path length of 16.7%, 15.4% and 17.8% is obtained for the first three scenarios, respectively as uncertainty increases from 2% to 20%. For the Mixed case for the RRT algorithm a 0% success rate is recorded as in the obstacle uncertainty test. This mainly results due to the larger increase in path length for RRT with increase in uncertainty. The RRT algorithm constructed shorter average paths of 7.2%, 7.2% and 4.4% with respect to A* for the first three scenarios, respectively.

In comparison with the no uncertainty case, A* results show an average increase in path length of 1.8%, 1.6%, 1.9% and 0.2% for the cube without and with rotation, V-obstacle and Mixed case scenarios, respectively. The minor increase for the Mixed case is attributed to the lower success rate (>50%) and therefore the best performing paths are successful lowering the average path length. The UAV positional uncertainty tests result in an increase of 1.8%, 1.7%, 1.2% and 2.7% while the obstacle uncertainty result in 0.6%, 0.3%, 1.7% increase and 1.5% decrease with respect to the no uncertainty cases for the scenarios defined earlier, respectively. Analysis of these results show that both uncertainty types contribute to path length increase with UAV positional uncertainty having the predominant effect. Although the increase in obstacle size and buffer distance from UAV are equal, UAV positional uncertainty is added at each intermediate UAV position irrespective of whether an obstacle is in the vicinity or not. For the first three cases where the environment is mainly empty the deviations due to obstacles occur in few situations while deviations in UAV positional uncertainty occur at each intermediate UAV position. This explains the predominant effect of the UAV positional uncertainty results. For the Mixed case, which introduces a deviation from obstacle uncertainty at each intermediate UAV position due to the high obstacle density, results cannot be directly compared due to different success rates for the uncertainty cases considered.

Similarly for RRT, a comparison with the no uncertainty cases shows an average increase in path length of 9.1%, 9.4% and 10.2% for the first three scenarios, respectively. The UAV positional uncertainty tests result in an increase of 9.2%, 9.1% and 46.6% while for obstacle uncertainty an increase of 0.3%, 0.2% and <0.1% is recorded for the first three scenarios, respectively with respect to the no uncertainty test case. As for A*, the major increase is recorded due to UAV positional uncertainty with obstacle uncertainty adding a low share to path length increase for the same reason described for A*. The difference for the V-obstacle scenario is attributed to the lower success rate for the combined uncertainty cases.

Figure 5.8 (b) and (e) illustrate the computational response time for the A* and RRT algorithms, respectively. The cube without rotation is the fastest scenario followed by the cube with rotation, V-obstacle and Mixed case scenarios (if success rate is not 0%) for both algorithms for all the uncertainty range considered. This order is different from the no uncertainty and UAV positional uncertainty tests and in line with the obstacle uncertainty test cases. The computational time comparison for A* at 2% and 20% uncertainty in obstacle and UAV position uncertainty results in a 4.4%, 6.2%, 2.2% increase and 2.4% decrease for the non-rotating cube, rotating cube, V-obstacle and Mixed case scenarios, respectively for 20% uncertainty with respect to 2% case. The decrease in Mixed case is attributed to the decreasing success rate. A similar analysis for RRT shows an increase of 35.1%, 27.9% and 7.6% for the first three scenarios implying that RRT's computational demand is more affected by uncertainty in comparison with A*. But RRT remains faster

than A* for the first three scenarios by 74.8%, 33.9% and 53.6%, respectively mainly because of the time consuming grid definition of the environment.

A* results for combined obstacle and UAV positional uncertainties with respect to the no uncertainty case show an average decrease in computational time by 2.51 and 2.50 times and an increase of 3.38 and 1.48 times for the cube without and with rotation, V-obstacle and Mixed case scenarios, respectively. The decrease in the cube cases is mainly attributed to the reduced number of accessible grid positions making it faster to find a path, if possible. The shift from 2D to 3D obstacles in the V-obstacle and planes (Mixed case) increases the computational time to define inaccessible grid positions residing within obstacle bounds. Furthermore, the UAV positional uncertainty tests result in a decrease of 2.29, 2.29, 5.60 and 1.15 times while the obstacle uncertainty results in 2.34 and 2.33 times decrease and a 3.72 and 1.49 increase with respect to the no uncertainty cases for the scenarios defined earlier, respectively. This analysis shows that in terms of computational time obstacle uncertainty is predominant for the V-obstacle and Mixed cases since the environment grid definition is predominately time consuming in 3D obstacles.

Similarly for RRT, a comparison with the no uncertainty cases shows an average increase in computational time of 47.7%, 51.8% and 15.8 times for the first three scenarios, respectively. The UAV positional uncertainty tests result in an increase of 16.3%, 8.9% and 13.1% while obstacle uncertainty test result show an increase of 23.6%, 26.3% and 12.9 times for the first three scenarios, respectively with respect to no uncertainty cases. As both positional and obstacle uncertainties increase computational time their effect in restricting path construction is summed up in the combined test case.

The success rates for both algorithms illustrated in Figure 5.8 (c) and (f) show a success rate near 100% for the first three scenarios for both algorithms for the range of uncertainties considered with a major drop for A* and a drop to 0% for RRT for the Mixed case scenario in line with the obstacle uncertainty test cases. As concluded for the obstacle uncertainty case, the success rate for the first three scenarios is the same for both algorithms with A* performing better than RRT in the Mixed case scenario. For A* the success rate drops by 5%, 4%, 5% and 7% for the cube without rotation, with rotation, V-obstacle and Mixed case, respectively from 2% to 20% uncertainties. For the same analysis, the RRT results in a success rate drop of 1%, 6%, 5% and 0% (0% success rate). Therefore it can be concluded that success rate drops with increase in uncertainty for both algorithms.

As previously, the maximum intermediate time violation is the major cause of unsuccessful runs with 78 tests out of 4000 resulting in No Path solution only for RRT with the number increasing with increase in percentage uncertainty. This violation only occurs for RRT since the planner does not incorporate half the distance between grid position as in A* and therefore the movement of an obstacle towards the UAV can result in a collision, as explained earlier. This risk will improve path length but is dangerous and can lead to a collision especially in the presence of uncertainty.

Comparing the success rates for the uncertainty in obstacle and position with respect to the no uncertainty test cases for A* show a decrease of 0.7%, 1.3%, 3.8% and 46.7% for the cube without rotation, with rotation, V-obstacle and Mixed case scenarios, respectively. Also, the UAV positional uncertainty tests show a drop <7% for all scenarios and for

obstacle uncertainty a drop <3% for the first three scenarios with a major drop of 46.8% for the Mixed case scenario all with respect to no uncertainty cases. This analysis show that success rate is hindered mainly by obstacle uncertainty especially in the Mixed case since in this situation the environment is already obstacle dense without increase in the obstacle bounds. In general, bounds will increase the inaccessible grid points making it more difficult to find a solution while UAV positional uncertainty will only limit the minimum distance to the obstacle which is already buffered by half the distance between grid positions.

A similar analysis for RRT shows a drop in success rate of 2.5%, 4.5%, 5.7% and 30% for the combined obstacle and UAV positional uncertainty with respect to the no uncertainty test case for the same speed for the cube without rotation, with rotation, V-obstacles and Mixed case scenarios, respectively. The UAV positional uncertainty test case results in a drop of 2.1%, 2.0%, 1.5% and 4% for the same scenarios as previously while for the obstacle uncertainty test case results in a drop <1% (First three scenarios) and 30% for the Mixed case with respect to the no uncertainty test cases. The analysis shows that, as opposed to A*, both uncertainties contribute to the drop in success rate owing to the nature of the RRT algorithm that both uncertainties will restrict tree branch construction increasing the path planning time with the consequence increasing unsuccessful runs.

In conclusion, with the inclusion of both obstacle and UAV positional uncertainties path planning performance further deteriorates, especially for RRT, with respect to the consideration of individual uncertainties.

5.5.6. CONCLUSION

This section presented, compared and analysed the real-time path planning performance of both A* and RRT algorithms in 4 different scenarios with and without uncertainty in obstacles and UAV position. Results show that:

1. Uncertainties deteriorate path length, computational time and success rate with the larger deterioration exhibited for the RRT algorithm;
2. RRT performed better in the cube and V-obstacle scenarios in terms of path length and computational time with A* performing better in complex scenarios;
3. The deteriorating factor of different uncertainty sources on path planning performance is different;
4. The inclusion of both uncertainties at the same time further hinders performance, especially for RRT.

In conclusion, this analysis confirms that uncertainties shall be considered in path planning, as their effect especially in complex scenarios, will determine the safety and success of the mission.

Table 5.3 summarises the results presented in this section by showing the effect of each uncertainty type and their combined effect on path planning performance. A +- sign shows that the effect of the uncertainty type on path planning performance measure is minimal and a + or - notation imply that the uncertainty type and path planning

performance measure are directly or inversely proportional, respectively. The additional sign show that the uncertainty type highly effects the path planning performance measure.

Table 5.3: Relational table between uncertainty type and path planning performance

Parameter	Path Length		Planning Time		Success rate	
	A*	RRT	A*	RRT	A*	RRT
UAV positional uncertainty	+-	+	–	+	+-	+-
Obstacle uncertainty	+-	+-	+-	++	–	–
UAV positional and obstacle uncertainty	+-	+	+-	++	–	–

5.6. CONCLUSION AND FUTURE WORK

This paper analysed the path planning performance of the A* and RRT algorithms in 3D real-time UAV path planning in different complexity scenarios with moving obstacles in the presence of uncertainties in UAV position and obstacle position and orientation. Literature identified the need to consider uncertainty sources in UAV path planning owing to their effect on constraints and disturbances within which a UAV must safely operate. In this regard, two different uncertainties, the UAV position and obstacle position and orientation, that incorporated all identified uncertainty sources, for the considered UAV, were quantified and modelled using bounded shapes. The effect on path planning of these uncertainties was assessed at different UAV speeds. The UAV model, path planner parameters and four different complexity scenarios were defined in view of real UAV models and the environment into which these are expected to operate. The path length, computational time and success rate were the performance measures considered, as uncertainty was varied between 2% and 20%.

Results show that both types of uncertainty deteriorate path planning performance of both A* and RRT algorithms for all scenarios considered with RRT exhibiting the larger effect on performance due to both types of uncertainties. RRT results in the fastest and shortest paths with approximately the same success rate as A* (>95%) for the cube and V-obstacle scenarios deteriorating significantly to even 0% success rate for the Mixed scenario. In latter case A* performs better. The combination of both types of uncertainty into the same test further deteriorates performance especially for RRT. Also owing to RRT's ability to shorter path nearer to obstacles with respect to A* results show that RRT has a higher risk of collision than A*. The results show that both algorithms can be applied in low obstacle density environments with moving obstacles in real-time path planning in the presence of uncertainty. In complex scenarios more time is required, especially for RRT, for comparable success rates. Finally, this work shows that 3D real-time path planning in different obstacle density, moving obstacle environments in the presence of uncertainty is possible.

Future enhancement shall include the analysis of the effects of other parameters

including look-ahead distance, distance to travel per iterate and allocated time on the performance of both path planning algorithms. A future work is the implementation of the developed 3D real-time path planning algorithms to configure a real UAV for autonomous 3D UAV navigation in an indoor obstacle-rich environment.

REFERENCES

[1] International Civil Aviation Organisation, "Global Air Traffic Management Operational Concept", *Doc. 9854*, AN/458, Ed. 1, pp. 1-82, 2005.

[2] International Civil Aviation Organisation, "Unmanned Aircraft Systems (UAS)", *Cir. 328*, AN/190, pp. 1-54, 2011.

[3] Boskovic, J. D., Knoebel, N., Moshtagh, N. and Larson, G.L., "Collaborative Mission Planning & Autonomous Control Technology (CoMPACT) System Employing Swarms of UAVs", *AIAA Guidance, Navigation and Control Conference*, Chicago, IL, 10-13 Aug., 2009, pp. 1-24.

[4] Goerzen, C., Kong, Z. and Mettler, B. "A Survey of Motion Planning Algorithms from the Perspective of Autonomous UAV Guidance", *Journal of Intelligent & Robotic Systems*, Vol. 57, pp. 65—100, 2010.

[5] Dadkhah, N. and Mettler, B., 'Survey of Motion Planning Literature in the Presence of Uncertainty: Considerations for UAV Guidance", *Journal of Intelligent & Robotic Systems*, Vol. 65, pp. 233—246, 2012.

[6] Vanegas, F. and Gonzalez, L. F., "Uncertainty based online planning for UAV target finding in cluttered and GPS-denied environments", *IEEE Aerospace Conference*, Big Sky, MN, 5-12 Mar., 2016, pp. 1-9.

[7] Chakrabarty, A. and Langelaan, J. W., "Energy maps for long-range path planning for small-and micro-uavs", *AIAA Guidance, Navigation and Control Conference*, Chicago, IL, 10-13 Aug., 2009, pp. 1-13.

[8] Benenson, R. Petti, S., Fraichard, T. and Parent, M., "Integrating Perception and Planning for Autonomous Navigation of Urban Vehicles," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China, 9-15 Oct. 2006, pp. 98-104.

[9] Yao, P., Wang, H. and Su, Z., "Real-time path planning of unmanned aerial vehicle for target tracking and obstacle avoidance in complex dynamic environment," *Aerospace Science and Technology*, Vol. 47, pp. 269-279, 2015.

[10] Zammit, C. and van Kampen, E. J., "3D real-time path planning of UAVs in dynamic environments", *AIAA Guidance, Navigation and Control Conference*, AIAA SciTech Forum, Kissimmee, FL, 8-12 Jan., 2021.

[11] Zammit, C. and van Kampen, E. J., "Comparison between A* and RRT Algorithms for UAV Path Planning", *AIAA Guidance, Navigation and Control Conference*, AIAA SciTech Forum, Kissimmee, FL, 8-12 Jan., 2018, pp. 1-23.

[12] Zammit, C. and van Kampen, E. J., "Advancements for A* and RRT in 3D path planning of UAVs", *AIAA Guidance, Navigation and Control Conference*, AIAA SciTech Forum, San Diego, CA, 7-11 Jan., 2019, pp. 1-17.

[13] Zammit, C. and van Kampen, E. J., "Comparison of A* and RRT in real-time 3D path planning of UAVs", *AIAA Guidance, Navigation and Control Conference*, AIAA SciTech Forum, Kissimmee, FL, 6-10 Jan., 2020.

[14] Kim, Y., Gu, D.-W., Postlethwaite, I., "Real-time path planning with limited information for autonomous unmanned air vehicles", *Automatica*, Vol. 44, No. 3, pp. 696—712, 2008.

[15] Hsu, D., Kindel R., Latombe J. C., Rock, S., "Randomized kinodynamic motion planning with moving obstacles", *International Journal Robotics Research*, Vol. 21, No. 3, pp. 233—255, 2002.

[16] Cui, J.-H., Wei, R.-X., Liu, Z.-C. and Zhou, K., "UAV Motion Strategies in Uncertain Dynamic Environments: A Path Planning Method Based on Q-Learning Strategy", *Applied Sciences*, Vol. 8, No. 11, pp. 2169, 2018.

[17] LaValle, S. M. and Sharma, R. "A Framework for Motion Planning in Stochastic Environments: Modeling and Analysis", *IEEE lnternational Conference on Robotics and Automation*, Nagoya, Japan, 21-27, May 1995, pp. 3057-3062, 1995.

[18] Hoy, M., Matveev, A. S. and Savkin, A. V., "Algorithms for collision-free navigation of mobile robots in complex cluttered environments: a survey", *Robotica*, Vol. 33, No. 4, pp. 463-497, 2014.

[19] Liao, F., Lai, S., Hu, Y., Cui, J., Wang, J. L., Teo, R. and Lin, F. "3D Motion Planning for UAVs in GPS-Denied Unknown Forest Environment", *IEEE Intelligent Vehicles Symposium*, Gothenburg, Sweden, 19-22, Jun. 2016, pp. 246-251.

[20] Vanegas, F., Campbell, D., Roy, N., Gaston, K. J. and Gonzalez, L. F., "UAV tracking and following a ground target under motion and localisation uncertainty", *IEEE Aerospace Conference*, Big Sky, MN, 4-11 Mar., 2017, pp. 1-10.

[21] Achtelik, M. W., Lynen, S., Weiss, S., Chli, M. and Siegwart, R., "Motion- and Uncertainty-aware Path Planning for Micro Aerial Vehicles", *Journal of Field Robotics*, Vol. 31, No. 4, pp. 676-698, 2014.

[22] Prentice, S. and Roy, N. "The Belief Roadmap: Efficient Planning in Belief Space by Factoring the Covariance" *The International Journal of Robotics Research*, Vol. 28, No. 11-12, pp. 1448-1465, 2009.

[23] He, R., Bachrach, A. and Roy, N., "Efficient planning under uncertainty for a target-tracking micro-aerial vehicle", *IEEE International Conference on Robotics and Automation*, Anchorage, AK, 3-7, May 2010, pp. 1-8.

[24] Florence, P. R."Integrated Perception and Control at High Speed", Master Dissertation, Massachusetts Institute of Technology, Feb. 2017.

[25] Lihua, Z., Xianghong, C. and Fuh-Gwo, Y. "A 3D collision avoidance strategy for UAV with physical constraints" *Measurement*, Vol. 77, pp. 40-49, 2016.

[26] Kothari, M. and Postlethwaite, I., "A Probabilistically Robust Path Planning Algorithm for UAVs Using Rapidly-Exploring Random Trees", *Journal of Intelligent & Robotic Systems*, Vol. 71, pp. 231-253, 2013.

[27] Rathbun, D., Kragelund, S., Pongpunwattana, A. and Capozzi, B., "An evolution based path planning algorithm for autonomous motion of a UAV through uncertain environments", *Digital Avionics Systems Conference*, Irvine, CA, 27-31, Oct. 2002, pp. 8.D.2-1-8.D.2-12.

[28] LaValle, S. M. and Sharma, R. "A Framework for Motion Planning in Stochastic Environments: Applications and Computational Issues ", *IEEE lnternational Conference on Robotics and Automation*, Nagoya, Japan, 21-27, May 1995, pp. 3063-3068.

[29] LaValle, S. M., *Planning algorithms*, Cambridge: Cambridge university press, 2006.

[30] Majumdar, A. and Tedrake, R. "Funnel libraries for real-time robust feedback motion planning", *The International Journal of Robotics Research*, Vol. 36, No. 8, pp. 947-982 2017.

[31] Moore, J., Cory, R. and Tedrake, R. "Robust post-stall perching with a simple fixed-wing glider using LQR-Trees", *Bioinspiration & Biomimetics Journal*, Vol. 9, No. 2, pp. 1-24 2014.

[32] Shah, S. K, Pahlajani, C. D., Lacock, N. A. and Tanner, H. G. "Stochastic receding horizon control for robots with probabilistic state constraints", *International Conference on Robotics and Automation (ICRA)*, St. Paul, MN, 14-18, May, 2012, pp. 2893-2898.

[33] Blackmore, L., Ono, M. and Williams, B. C. "Chance-constrained optimal path planning with obstacles", *IEEE Transactions on Robotics*, Vol. 27, No. 6, pp. 1094-1080, 2011.

[34] Luders, B., Kothari, M. and How, J.P. "Chance constrained RRT for probabilistic robustness to environmental uncertainty", *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Toronto, Ontario, Canada, 02-05 Aug. 2010, pp. 1-21.

[35] Page, L. A. and Sanderson, A. C. "A path-space search algorithm for motion planning with uncertainties", *IEEE International Symposium on Assembly and Task Planning*, Pittsburgh, PA, 2-5, Aug. 1995, pp. 334-340.

[36] Lazanas, A. and Latombe, J.-C. "Landmark-based robot navigation", *Algorithmica*, Vol. 13, No. 5, pp. 472-501, 1995.

[37] Larson, J., Bruch, M. and Ebken, J. "Autonomous navigation and obstacle avoidance for unmanned surface vehicles", *Defense and Security Symposium, SPIE conference*, Kissimmee, Orlando, FL, 17—21, Apr. 2006, pp. 1-12.

[38] Yang, L., QI, J., Jiang, Z., Song, D., Han, J. and Xiao, J. "Guiding Attraction based Random Tree Path Planning under Uncertainty: Dedicate for UAV", *International Conference on Mechatronics and Automation*, Tianjin, China, 3—6, Aug. 2014, pp. 1182-1187.

[39] Pepy, R. and Lambert, A. "Safe Path Planning in an Uncertain-Configuration Space using RRT", *IEEE International Conference on Intelligent Robots and Systems*, Beijing, China, 9-15 Oct. 2006, pp. 5376-5381.

[40] van den Berg, J., Wilkie, D., Guy, S. J., Niethammer, M. and Manocha, D. "Lqg-obstacles: Feedback Control with Collision Avoidance for Mobile Robots with Motion and Sensing Uncertainty", *IEEE International Conference on Robotics and Automation*, St. Paul, MN, 4-18 May 2012, pp. 346-353.

[41] Zeng, Z., Lammas, A., Sammut, K., He, F. and Tang, Y. "Shell space decomposition based path planning for AUVs operating in a variable environment", *Ocean Engineering*, Vol. 91, pp. 181-195, 2014.

[42] Gonzalez, J. P. and Stentz, A., "Planning with uncertainty in position an optimal and efficient planner", *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Edmonton, Alta., Canada, 2-6 Aug. 2005, pp. 1-8.

[43] Wen, N., Xiaohong, S., Ma, P., Zhao, L. and Zhang, Y. "Online UAV path planning in uncertain and hostile environments", *International Journal of Machine Learning and Cybernetics*, Vol. 8, pp. 469-487, 2017.

[44] Daftry, S., Zeng, S., Khan, A., Dey, D., Melik-Barkhudarov, N., Bagnell, J. A. and Hebert, M. "Robust Monocular Flight in Cluttered Outdoor Environments", *ArXiv*, Vol. 1604.04779, pp. 1-10, 2016.

[45] Matthies, L., Brockers, R., Kuwata, Y. and Weiss, S. "Stereo vision-based obstacle avoidance for micro air vehicles using disparity space", *IEEE International Conference on Robotics and Automation (ICRA)*, Hong Kong, China, 31 May-7 Jun. 2014, pp. 3242-3249.

[46] Liu, S., Watterson, M., Tang, S. and Kumar, V. "High speed navigation for quadrotors with limited onboard sensing.", *IEEE International Conference on Robotics and Automation (ICRA)*, Stockholm, Sweden, 16-21 May 2016, pp. 1484-3249.

[47] Kuwata, Y., Karaman, S., Teo, J., Frazzoli, E., How, J. and Fiore, G. "Real-Time Motion Planning With Applications to Autonomous Urban Driving", *IEEE Transactions on Control Systems Technology*, Vol. 17, No. 5, pp. 1105-1118, 2009.

[48] Fulgenzi, C., Tay, C., Spalanzani, A. and Laugier, C. "Probabilistic navigation in dynamic environment using rapidly-exploring random trees and Gaussian processes", *IEEE International Conference on Intelligent Robots and Systems*, Nice, France, 22-26 Sep. 2008, pp. 1056—1062.

[49] Kewlani, G., Ishigami, G. and Iagnemma, K. "Stochastic mobility-based path planning in uncertain environments", *IEEE International Conference on Intelligent Robots and Systems*, St. Louis, MO, 11-15 Oct. 2009, pp. 1183-1189.

[50] Melchior, N.A. and Simmons, R. "Particle RRT for path planning with uncertainty", *IEEE International Conference on Robotics and Automation*, Rome, Italy, 10-14 Apr. 2007, pp. 1617-1624.

[51] Aoude, G. S., Joseph, J., Roy, N. and How, J.P. "Mobile Agent Trajectory Prediction using Bayesian Nonparametric Reachability Trees", *Infotech@ Aerospace*, St. Louis, MO, 29-31 Mar. 2011, pp. 1587-1593.

[52] Papadimitriou, C. H. and Tsitsiklis, J. N. "The Complexity of Markov Decision Processes", *Mathematics of Operations Research*, Vol. 12, No. 3, pp. 441-450, 1987.

[53] Thrun, S., Burgard, W. and Fox, D. 'Probabilistic Robotics', Massachusetts: MIT Press, 2005.

[54] Pineau, J., Gordon, G. and Thrun, S. "Anytime Point-Based Approximations for Large POMDPs", *Jounal of Artificial Intelligence Research*, Vol. 27, No. 1, pp. 335-380, 2006.

[55] Utkin, V. I. 'liding Modes in Control Optimization', Berlin: Springer–Verlag, 1992.

[56] Shah, M. Z., Samar, R. and Bhatti, A. I. "Guidance of Air Vehicles: A Sliding Mode Approach", *IEEE Transactions on control system technology*, Vol. 23, No. 1, pp. 231-244, 2015.

[57] Yang, L., Qi, J., Song, D., Xiao, J., Han, J. and Xia, Y. "Survey of Robot 3D Path Planning Algorithms", *Journal of Control Science and Engineering*, Vol. 2016, pp. 1-22, 2016.

[58] Culligan, K., Valenti, M. Kuwata, Y. and How, J. P. "Three dimensional flight experiments using on-line mixed-integer linear programming trajectory optimization", *American Control Conference*, New York, NY, 9-13 Jul. 2007, pp. 5322-5327.

[59] Masehian, E. and Habibi, G. "Robot path planning in 3D space using binary integer programming", *International Journal of Mechanical System Science and Engineering*, Vol. 23, pp. 26-31, 2007.

[60] Anderson, S., J., Peters, S. C., Pilutti, T. E. and Iagnemma, K. "An optimal-control-based framework for trajectory planning, threat assessment, and semi-autonomous control of passenger vehicles in hazard avoidance Scenarios", *International Journal of Vehicle Autonomous Systems*, Vol. 8, No. 2-4, pp. 190-216, 2010.

[61] Connolly, C. I. "Harmonic functions and collision probabilities", *International Journal of Robotics Research*, Vol. 16, No. 4, pp. 497-507, 1997.

[62] Lazanas, A. and Latombe, J. C. "Motion planning with uncertainty: a landmark approach", *Artificial Intelligence*, Vol. 76, No. 1-2, pp. 287-317, 1995.

[63] Zengin, U. and Dogan, A., J. C. "Probabilistic trajectory planning for UAVs in dynamic environments", *AIAA "Unmanned Unlimited" Technical Conference, Workshop and Exhibit*, 20-23 Sep., 2004, Chicago, IL, pp. 1-12.

[64] Thrun, S., Diel, M. and Hahnel, D. "Scan alignment and 3-D surface modelling with a helicopter platform", *International Conference on Field and Service Robotics*, Yamanashi, Japan, 14-16 Jul. 2003, pp. 1-6.

[65] Thrun, S. and Montemerlo, M. "The graph slam algorithm with applications to large-scale mapping of urban structures", *International Journal of Robotics Research*, Vol. 25, No. 5-6, pp. 403-429, 2006.

[66] Kuwata, Y., Schouwenaars, T., Richards, A. and How, J. "Robust constrained receding horizon control for trajectory planning", *AIAA Guidance, Navigation, and Control Conference*, San Francisco, CA, 15-18 Aug. 2005, pp. 1-12.

[67] Schouwenaars, T., Mettler, B., Feron, E. and How, J. P. "Robust Motion Planning Using a Maneuver Automaton with Built-in Uncertainties", *American Control Conference*, Denver, Co, 4-6 Jun. 2003, pp. 2211-2216.

[68] Frazzoli, E. 'Robust Hybrid Control for Autonomous Vehicle Motion Planning', Ph. D. thesis, MIT, June 2001.

[69] Gonz*à*lez, D., P*è*rez, J., Milan*è*s, V., and Nashashibi, F., "A Review of Motion Planning Techniques for Automated Vehicles", *IEEE Transactions on Intelligent Transportation Systems*, Vol. 17, No. 4, pp. 1135-1145, 2016.

[70] Ghandi, S. and Masehian, E., "Review and taxonomies of assembly and disassembly path planning problems and approaches", *CAD Computer Aided Design*, Vol. 67-68, No. October, pp. 58-86, 2015.

[71] Tseng, F. H., Liang, T. T. and Lee, C. H. Chou, L. D. and Chao, H., "A Star Search Algorithm for Civil UAV Path Planning with 3G Communication", *10th International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP)*, Kitakyushu, Japan, 27-29 Aug. 2014, pp. 942-945.

[72] Hart, P. E., Nilsson, N. J. and Raphael, B., "A Formal Basis for the Heuristic Determination of Minimum Cost Paths", *IEEE Transactions on Systems Science and Cybernetics*, Vol. 4, No. 3, pp. 100-107, 1968.

[73] Short, A., Pan, Z., Larkin, Z. and van Duin, S. "Recent Progress on Sampling Based Dynamic Motion Planning Algorithms", *IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, Alberta, Canada, Jul. 2016, pp. 1305-1311.

[74] LaValle S. M. "Probabilistic roadmaps for path planning in high-dimensional configuration spaces", *IEEE Transactions on Robotics and Automation*, Vol. 12, No. 14, pp. 566-580, 1996.

[75] Lavalle S. M. and Kuffner J. J., "Randomized kinodynamic planning", *International Journal of Robotics Research*, Vol. 20, No. 3, pp. 378-400, 2001.

[76] LaValle, S. M. and Kuffner, J. J. "Randomized kinodynamic planning", *Proceedings of the IEEE International Conference on Robotics and Automation*, Detroit, MI, 10-15 May 1999, pp. 473-479.

[77] Devaurs, D., Siméon, T. and Cortés, J. “Optimal Path Planning in Complex Cost Spaces With Sampling-Based Algorithms”, *IEEE Transactions on Automation Science and Engineering*, Institute of Electrical and Electronics Engineers, 2015.

[78] Geraerts, R. and Overmars, M. “Creating high-quality paths for motion planning”, *International Journal of Robotics Research*, Vol. 26, No. 8, pp. 845-863, 2007.

[79] Sujit, P. B., and Ghose, D., “Search by UAVs with Flight Time Constraints using Game Theoretical Models,” *AIAA Guidance, Navigation and Control Conference*, San Francisco, CA, 15-19 Aug. 2005, pp. 1-11.

[80] Bethke, B., Bertuccelli, L. How, J. P., “Experimental Demonstration of MDP-Based Planning with Model Uncertainty,” *AIAA Guidance, Navigation and Control Conference*, Honolulu, HI, 18-21 Aug. 2008, pp. 1-22.

[81] Bollino, K., Lewis, L. R., Sekhavat, P. and Ross, I. M., “Pseudospectral Optimal Control: A Clear Road for Autonomous Intelligent Path Planning,” *AIAA Infotech Aerospace Conference and Exhibit*, Rohnert Park, CA, 7-10 May 2007, pp. 1-14.

6

Conclusions and Recommendations

Drones overall will be more impactful than I think people recognize, in positive ways to help society.

Bill Gates

The modelling and integration of uncertainties into the developed real-time 3D UAV path planning algorithms concludes the planned research journey of this dissertation. The contents of this dissertation started in Chapter 1 with the identification of challenges hindering the use of UAVs in indoor environments. These challenges target: real-time 3D UAV path planning (Challenge 1) in dynamic environments (Challenge 2) and including uncertainty considerations (Challenge 3). The main research goal was formulated as:

Research Goal

Assess the performance of state-of-the-art path planning rationales in the context of UAVs operating in *3D real–time*, *dynamic* indoor environments in the presence of *uncertainty* and identify a customised configuration based on the application.

To reach this research goal, five research questions were posed in Chapter 1. These research questions were addressed in Chapter 2 to Chapter 5 and the main findings and the implications on UAV path planning in indoor environments will be summarised in Section 6.1. Finally, Section 6.2 concludes this dissertation with an explanation of the limitations of this work and future recommendations with the ultimate aim of reducing the challenges that are hindering the use of UAVs in indoor environments.

6.1. CONCLUSIONS

6.1.1. MAIN FINDINGS

Research Question 1, stated below, targets the review and comparison of path planning algorithms for UAVs, and was tackled in Chapter 2. It required a literature review of the state-of-the-art 2D and 3D path planning algorithms that have a potential to be applied to guide a UAV operating in typical indoor environments.

Research Question 1

What is the state-of-the-art in the field of path planning for UAVs in 3D and how these algorithms compare?

The literature review presented in Chapter 2 shows that graph–based and sampling–based methods are potential candidates for 3D UAV path planning. These two path planning rationales have developed into a number of variants with the A* and RRT being the most utilised graph-based and sampling-based path planning algorithms, respectively. These two algorithms, together with a static obstacle-rich 3D environment, were implemented in MATLAB to allow the two algorithms to be tested in the same environment.

Tests showed that RRT algorithm's tree propagation is limited by tree branch length. Therefore, to assess the effect of this limitation this restriction is eliminated in the RRT without step size constraint variant, developed originally in this work. Furthermore, the MRRT algorithm, a variant of the RRT algorithm, which grows multiple trees, instead of just one is also implemented.

For A*, tests show an inherent ripple in path length with changes in resolution for all scenarios. Analysis shows this ripple is the result of A*'s graph–based nature which creates situations in which an increase in resolution, which theoretically shall slightly decrease the path length, effectively generates longer or shorter paths. This ripple can be mitigated by randomly shifting the environment in all three dimensions by a distance varying between zero and half the distance between adjacent graph points.

Literature and tests show that the A* generates almost optimal paths while the RRT generates non–optimal paths. Therefore, a smoothing algorithm is developed and applied to A* with the ripple reduction algorithm, RRT, RRT without step size constraint and MRRT as a post path planning stage.

From the comparison of the graph-based and sampling-based rationales, addressed in Chapter 2, it was concluded that:

1. All path planning algorithms construct a non-colliding path in all scenarios considered;
2. A* generates almost optimal paths while the RRT generates non–optimal paths;
3. A* only explores areas necessary for path construction while the RRT evenly explores the environment;
4. A* generates shorter paths in less time with respect to RRT, for all scenarios considered, even if the post path planning smoothing algorithm is considered;
5. The A* ripple reduction (A^*_R) algorithm reduces ripple by 46% to 48% in terms of standard deviation for all conditions considered, without an increase in path length and computational time;
6. The RRT without step size constraint results in less path construction time with respect to the standard RRT, but also leads to longer and oscillating paths which require more smoothing, for all scenarios;
7. The evenly–distributed MRRT generates longer unsmoothed paths in shorter planning times but requires more smoothing with respect to RRT for all considered scenarios; and
8. The developed smoothing algorithm takes less than 1% of the path generation time and is mainly effective for sampling-based algorithms.

The algorithms developed to answer Research Question 1 are integral to address Research Question 2, stated below and tackled in Chapter 3, through developing and assessing a real–time 3D path planning platform in static environments.

Research Question 2

Can the selected path planing algorithms be applied in real-time environments using the computational demand onboard small UAVs?

The development of this platform is influenced by the conclusions drawn in Chapter 2. A literature survey highlighted the need of real-time path planning of 2D autonomous systems [1–3] emphasising the greater need for real–time path planning for UAVs which need to manoeuvre in complex 3D environments. In Chapter 3, a real-time path planning platform is constructed based on a set of user-defined parameters and system constraints. This system assumes that, first, the environment within a look-ahead distance from the UAV position is known with certainty while the remaining is unknown and second, the path planning algorithm must construct a path in less than or equal to the time required by the UAV to move one iteration.

In Chapter 2 it was concluded that the A* with ripple reduction and the standard RRT are best performing path planning algorithms for the graph-based and sampling-based rationales, respectively. In Chapter 3, these two algorithms with smoothing are integrated within the real-time platform and tested in the same scenarios as in Chapter 2. Results show that:

1. The A* algorithm outperforms the RRT algorithm in both path length and computational time, for all scenarios considered, with the difference increasing with scenario complexity;

2. A* is successful by more than 90% in all tests for all scenarios considered, provided the look-ahead distance is at least double the distance moved per iterate;

3. The RRT algorithm results in a lower success rate than A* owing to the longer computational time required to construct intermediate paths with respect to A*; and

4. For both algorithms, performance depends on the definition of empirical values for each parameter.

This analysis triggered Research Question 3, stated below, that replaces static obstacles with moving and rotating obstacles at a speed less than or equal to the UAV speed. In Chapter 4 a dynamic environment is constructed to assess the path planning performance of the 3D real–time path planning platform developed in Chapter 3.

Research Question 3

What is the effect, if any, on path planning performance if static obstacles are replaced with dynamic obstacles?

The need to consider dynamic environments is underlined in literature owing that UAVs must operate in time-varying environments [4]. Four different scenarios with different complexity made up of rotating and non-rotating cubes, rotating V-shaped obstacles and static 2D planes with windows are considered.

A real-time environment with a limited sensing range creates situations where an intermediate goal point is not available. Waiting for the intermediate goal point to be clear in dynamic environments may result in the clearance or not of the intermediate goal point due to obstacle movement. Two rationales were developed to mitigate this

situation. In the *waiting* rationale, the UAV waits in its current position until the defined intermediate goal position becomes available. In the *moving* rationale, the intermediate goal position is neared to the current UAV position, consequently increasing the chances of moving closer to the final goal position. These two rationales were integrated within the two path planning rationales and tested in all scenarios with dynamic obstacles.

Results show that:

1. The *moving* option yields better overall results in terms of path length, computational time and success rate for A* and RRT with respect to the *waiting* option;

2. Both A* and RRT produce similar results in relatively simple scenarios with RRT showing better results in path length, computational time and success rate;

3. As speed increases in complex scenarios the success rate drops due to lack of path planning time in both A* and RRT; and

4. For complex scenarios the RRT is better in terms of path length provided time is not limited, while the A* algorithm is less susceptible to time constraints.

Chapter 5 identifies the need of uncertainty consideration in real-time 3D UAV path planning, owing that uncertainty may negatively impact path planning performance if neglected [5]. This chapter tackles Research Question 4, stated below, by identifying and describing the uncertainty sources present in the sensing systems, UAV model, environmental sensing and prediction and communication.

Research Question 4

Do uncertainties effect 3D path planning of UAVs? If yes, how can these uncertainties be modelled?

Literature suggested the bounding shapes and probabilistic distributions methods as key candidates for uncertainty modelling in UAV applications. After considering their characteristics, uncertainty is modelled using bounded shapes around the current UAV position and obstacle volume.

Chapter 5 integrates the uncertainty into the real-time 3D UAV path planning algorithm with dynamic obstacles developed in Chapter 4. Uncertainty bounds are quantified based on literature and varied between 2% and 20% for both the UAV position and the position of obstacles. Tests are performed using both path planning rationales with the *moving* method on the real-time path planning platform using the same scenarios described in Chapter 4. The analysis with different uncertainty bounding percentages will help understand the effect of uncertainties for a UAV operating within indoor environments, suggesting how these uncertainties can be mitigated hence tackling Research Question 5.

Research Question 5

Can uncertainties be mitigated to ensure collision–free 3D path planning of UAVs in real–time in the presence of dynamic obstacles?

The main findings are that:

1. Both sources of uncertainty (position of UAV and obstacles) deteriorate path planning performance of both A* and RRT algorithms for all scenarios considered with RRT exhibiting the largest performance degradation;
2. The inclusion of both sources of uncertainty at the same time further deteriorates performance, especially for RRT;
3. RRT results in the fastest and shortest paths with approximately the same success rate as A* for relatively simpler scenarios;
4. A* performs better in terms of success rate in the relatively complex case; and
5. RRT has a higher risk of collision than A* as in RRT the UAV moves closer to obstacles than with A*.

In the next subsection, the findings from each chapter will be further discussed in the context of real UAV path planning in indoor environments.

6.1.2. IMPLICATIONS TO REAL UAV PATH PLANNING

In all the chapters, three performance measures where considered: path length, path generation time and success rate. An optimal path length is desired, especially when frequent re-charging is not possible or when the UAV is expected to travel long distances. In such situations, UAV's onboard energy resources become the bottleneck. A low path generation time is also desired especially in time-varying, obstacle-rich environments, since the planner needs to continuously react to ensure a non-colliding path. Since in the considered application the UAV can only use onboard computational resources which must be shared with the sensing and actuator systems, this performance parameter becomes more important. Finally, the success rate will affect the safety and robustness of the path planning system and its consideration for real implementation.

From Chapter 2, results show that A* performs better than RRT both in terms of path length and path generation time in offline situations with static obstacles with 100% success rate for both in all scenarios considered. This implies that A* shall be used in situations where the UAV is expected to operate in a static environment and the path is defined prior flight. Increasing resolution in A* and decreasing tree branch length in RRT reduces path length and increases computational demand. Therefore, these parameters shall be defined in view of mission requirements and environment sensing systems.

A* allows the environment to be discretised differently, making optimal use of resources. Oppositely, RRT and its variants are suited to generate paths efficiently in evenly-distributed and focused 3D area exploration applications. RRT and its variants can perform better in terms of path construction time and length than A*, if the randomly– selected nodes are intelligently selected in view of obstacle shape, position and dynamics.

Chapter 3 shows that 3D real-time path planning can be realised using standard UAV onboard systems. This chapter develops a 3D real-time path planning platform and outlines the best empirical values for the different external and internal parameters, namely UAV speed, sensor range, computational power, distance to travel per iterate, maximum time to generate a path and distance factor. If not restricted by mission demands and/or hardware limitations, the setting of these parameters will configure the developed 3D real-time path planning platform, optimising its performance to a specific user-defined indoor application.

Chapter 4 shows that the 3D real-time path planning platform with both A* and RRT algorithms has potential to be used in low-density dynamic obstacle scenarios. Computational demand increases with scenario complexity, especially for RRT. The *waiting* variant is better suited where it is safer to remain in the current position than risking a collision. In home environments this is usually the case as the UAV are not allowed collide with obstacles, especially if these are persons. While the *moving* variant is ideal in situations where goal achievement is paramount even at the expense of collision or where other hostile objects are operating in the same space and therefore stopping will make the UAV more vulnerable. Such situations include search and rescue and spying inside buildings.

Finally, Chapter 5 deals with uncertainty modelling and integration. It shows that uncertainty must be considered as it has a significant effect on path planning performance. Performance depends on how accurate uncertainty can be modelled. With bounding shapes, this depends on the maximum variants of each respective uncertainty. If the UAV is to operate within the same environment throughout its lifetime, a learning algorithm can fine-tune uncertainty bounds with experience. If the UAV is expected to operate in different scenarios, uncertainty modelling can become a challenge.

This resume analysizes the main findings of this dissertation from an applications' point of view. The intent is to guide future UAV designers to select the appropriate configuration based on the application. This work has some limitations and future recommendations that shall enhance its findings. These will be discussed in Section 6.2.

6.1.3. SUMMARISED CONCLUSIONS

The main finding of this thesis can be summarised as follows:

1. The A* algorithm performs better in terms of path length and computational time than RRT in offline situations with static obstacles. The A* algorithm's graph-based nature allows the environment to be discretised according to mission requirements and environmental situations, while the RRT algorithm evenly explores the environment, wasting time in exploring areas not necessary for path construction.

2. The A* algorithm generates shorter paths in less time than RRT in real-time path planning in the presence of static obstacles. The difference in performance increases with environmental complexity. The path planning performance of both algorithms depends on the definition of different parametric values, such as look-ahead distance and speed.

3. In the presence of dynamic obstacles, for both algorithms, the *moving* variant, that allows the UAV to move closer to obstacles, increasing the chance of collision,

produces better real-time path planning results than the *waiting* variant. Both variants can be used, however, depending on the mission requirements and constraints.

4. Both sources of uncertainty (UAV position and obstacles) independently deteriorate path planning performance of both algorithms, with RRT exhibiting the largest performance degradation. Using the RRT algorithm leads to shorter, faster but also riskier paths for relatively simple scenarios with respect to the A* algorithm, with the situation reversing in complex scenarios.

6.2. LIMITATIONS, FUTURE WORK AND MAIN RECOMMENDATIONS

This section will discuss the limitations of the developed real-time 3D UAV path planning and recommend future advancements that are expected to increase the potential use of autonomous UAVs in indoor environments. It will conclude with the main recommendations for 3D real-time path planning of UAVs in dynamic environments in the presence of uncertainty.

6.2.1. LIMITATIONS AND FUTURE WORK

In Chapter 2, different path planning algorithms presented in literature were discussed. Although an extensive literature review is presented, a further in-depth and broader review likely finds more algorithms within the graph-based and sampling-based classes. Examples include the basic Theta* [6], Anytime D* [7], RRT* [8] and Transition-based RRT [9] that can potentially attain an equivalent or even better performance.

The standard RRT, the RRT without step-size constraint and MRRT algorithms were all implemented without any tree-branch biasing. Biasing on the line directly connecting the start and goal points will potentially reduce the path length and planning time in a low obstacles density environment [9].

The developed smoothing algorithm is relatively basic, and further work or the development of a new smoothing algorithm could improve the final path. Care must be taken, however, not to consider a computationally too intensive algorithm, as onboard UAV computational resources are limited. Also, the smoothing algorithm shall be selected upon the characteristics of the path following algorithm and the UAV actuator systems.

In this dissertation, the constructed path from start to goal is made up of points, that when connected, create a non-colliding path. This creates abrupt point turns that requires the UAV to stop. This is not always possible, nor desired, and a path following algorithm needs to be developed to construct smoother trajectories between path points such that the UAV can follow at the speed requested by the application. Therefore, a path following algorithm needs to be integrated with the developed path planning system in view of UAV model constraints and mission specifications.

The mean UAV speed is assumed constant in this work although the effect of speed on path planning performance is tested and analysed. This does not imply that the UAV

will traverse the path points at constant speed but that the mean speed over the particular path segment is constant. This means UAV speed is used to determine the maximum time to construct an intermediate path. The addition of a speed controller in combination with the path following algorithm will allow the UAV to follow the path better, as the controller can adjust the speed based on the complexity of the path being followed.

The scenarios constructed are a representative set of the difficult to worst case scenarios which the UAV is expected to encounter in an indoor environment. Modelling of real static environments, path planning using different start and goal points and the real UAV traversing will further increase the value of this work.

In Chapter 3 the selected path planning algorithms were tested in a simulated real-time environment assuming that the standard PC computational power is available on-board the UAV. Although care is taken in limiting the use of the processor to only MATLAB's 3D path planning files, background processes could not be eliminated. The implementation of the 3D path planning platform into a real UAV will be of additional value.

It is further assumed in this chapter that the sensing system can accurately sense all obstacles within its sensing range in all 3D and provide these data to the path planning algorithm in real-time. A generic sensing range with a 360^o FOV is considered in this work and the sensing range effect on path planning performance analysed. But further work on determination of this distance on a range of sensing technologies is required for the implementation of the 3D real-time path planning algorithm in real life.

For the assessment of the real-time 3D path planning algorithm, a set of generic shapes with different difficulty of evasion were considered in Chapter 4. More shapes with different sizes and different obstacle-density scenarios can improve the robustness of the results.

Further, it was assumed that obstacles remain static as the UAV moves from the current to a future position. In parallel, the path planning algorithm is constructing a new intermediate path. Real path planning tests in a typical environment using a standard UAV can confirm the selected maximum intermediate and total times and assess whether such time is enough to ensure non-colliding paths in the specific application. The maximum intermediate time can be reduced by increasing computational power.

Further work could analyse the effects of other parameters on the performance of path planners with both waiting and moving rationale. These parameters shall include resolution, look–ahead distance, distance to travel per iterate, intermediate and total time and distance factor. This analysis will flag the best configuration for both algorithms in the considered test scenarios, offer a range of path construction difficulties.

In Chapter 5, a set of uncertainties was modelled and tested. Future work should involve assessing uncertainties in environment sensing and UAV position systems in a typical environment using a standard UAV. This will help the UAV designer to predict the path planning performance of the proposed system and adapt accordingly based on the UAV and application.

Uncertainty is modelled using the bounding shapes method for all uncertainty sources. Further work into investigating and comparing both the bounding shape and the probabilistic distribution methods, or possibly other techniques for each specific uncertainty sources, will provide an overview of the best method to utilise for each uncertainty source based on the environment and UAV model capabilities.

The uncertainty sources are all grouped and integrated into obstacle and UAV position uncertainties. Communication uncertainty is not studied in this thesis as the operation of a single UAV is assumed. Also, UAV orientation uncertainty is neglected as a UAV point model, moving at constant speed with no orientation is assumed. The actual UAV orientation will be taken into account by the path following algorithm which is no considered within the scope of this work. These two uncertainty sources, together with the others listed, can be included to assess their effects in a multi UAV environment and using UAVs of different shapes and characteristics.

Further analysis of the effects of other parameters, including look-ahead distance, distance to travel per iterate and allocated time on the performance of both path planning algorithms will lead to more awareness and knowledge of the effects of uncertainty on path planning performance.

6.2.2. MAIN RECOMMENDATIONS

Based on the limitations and future work discussed in the previous subsection, the following five main recommendations are drawn:

1. Develop a tree-branch seed algorithm to enhance the convergence rate of the RRT, and also a better smoothing algorithm. Assess both performances independently and concurrently with the standard RRT and the A* algorithms.

2. Integrate a path following algorithm with the path planning algorithm and assess their combined performance in all situations and configurations considered.

3. Assess the performance of real onboard UAV sensing technologies, model the response of the state-of-the-art, integrate within the developed real-time path planning platform and compare their performance with the generic 360^o range sensor considered for this work.

4. Analyse the effect of UAV parameters, such as look-ahead distance, on performance using a larger set of regular and irregular shapes, without assuming that obstacles remain static in between iterations.

5. Compare the effects on path planning performance of bounding shapes and probabilistic distribution methods for uncertainty modelling, with a wider range of individual uncertainty sources, including communication uncertainty.

6.2.3. CLOSING STATEMENTS

The implementation of the developed 3D real-time path planning algorithms to configure a real UAV for autonomous 3D UAV navigation in an indoor obstacle-rich environment is the ultimate future aim that can lead into the commercialisation of this system for use in domestic applications such as assisting in video production, in-house item delivery and potentially dangerous window and chimney cleaning.

Moreover, this real-time 3D UAV path planning system can also be proposed for the integration in outdoor UAV applications. In outdoor environments the situation

changes both in terms of obstacles, external factors such as wind and rain and also regulation, as the UAV must share public space with other time-variant autonomous systems, manually-controlled systems, static obstacles and people.

This dissertation's ultimate goal derived from the research goal was to contribute in reducing the gap that still exists to allow UAVs to be used in domestic environments, to assist people in their everyday activities. Ultimately, this dissertation has made the integration of UAVs a step closer to reality.

REFERENCES

[1] Sujit, P. B., and Ghose, D., "Search by UAVs with Flight Time Constraints using Game Theoretical Models," *AIAA Guidance, Navigation and Control Conference*, San Francisco, CA, 15–19 Aug. 2005, pp. 1–11.

[2] Bethke, B., Bertuccelli, L. How, J. P., "Experimental Demonstration of MDP–Based Planning with Model Uncertainty," *AIAA Guidance, Navigation and Control Conference*, Honolulu, HI, 18–21 Aug. 2008, pp. 1–22.

[3] Bollino, K., Lewis, L. R., Sekhavat, P. and Ross, I. M., "Pseudospectral Optimal Control: A Clear Road for Autonomous Intelligent Path Planning," *AIAA Infotech Aerospace Conference and Exhibit*, Rohnert Park, CA, 7–10 May 2007, pp. 1–14.

[4] Mac, T. T., Copot, C., Tran, D. T. and Keyser, R. D. "Heuristic approaches in robot path planning: A survey", *Journal of Robotics and Autonomous Systems*, Vol. 86, Aug. 2016, pp. 13–28.

[5] Cui, J.–H., Wei, R.–X., Liu, Z.–C. and Zhou, K., "UAV Motion Strategies in Uncertain Dynamic Environments: A Path Planning Method Based on Q–Learning Strategy", *Applied Sciences*, Vol. 8, No. 11, pp. 2169, 2018.

[6] K. Daniel, A. Nash, S. Koenig and A. Felner, Theta^*: Any-angle path planning on grids, *J. Artificial Intelligence*, **39** (2010) 533–579.

[7] M. Likhachev, D. Ferguson, G. Gordon, A. Stentz and S. Thrun, Anytime search in dynamic graph, *Artificial Intelligence*, **172**(14) (2008) 1613—1643.

[8] S. Karaman and E. Frazzoli, Optimal kinodynamic motion planning using incremental sampling-based methods, in *Proc. 49th IEEE Conf. Decision and Control*, (Atlanta, GA, 2010), pp. 7681–7687.

[9] D. Devaurs, T. Siméon, and J. Cortés, Optimal Path Planning in Complex Cost Spaces With Sampling-Based Algorithms, *IEEE Trans. on Automation Science and Engineering*, Institute of Electrical and Electronics Engineers, **13**(2) (2015) 415–424.

ACKNOWLEDGEMENTS

This dissertation is the result of seven wonderful years at one of the best universities in the World. It has been an honour for me to even be considered to study within TU Delft. Although only one name is present on the cover of this dissertation, this work would not have been possible without the direct and indirect contribution of a number of people in Delft and outside.

My first thank you goes to my promotor Prof. ir. Max Mulder for believing in me from day one and allowing me to pursue this Ph. D. remotely and part time while lecturing in Malta. I vividly remember how happy I was to receive a reply for my email stating my interest in pursuing a Ph. D. in TU Delft on the beginning of January 2015. I also remember how tense I was in my first Skype interview with you, Dr. ir. Erik-Jan van Kampen and Dr. Qi Ping Chu. But I really appreciate that all of you kept me calm and made me feel welcome. Since then you have always been available, always congratulating me in my small achievements, motivating me in continuing to enhance my work and introducing me to Dutch food. Your enthusiasm and sharp analytical skills deeply inspires me. Also, I am very grateful for always encouraging me to attend conferences and exchanges with other Universities that have improved my network and improved my discipline, research and transferable skills. You always made sure that I had all the support possible even though I was residing in Malta for the majority of the time.

My second thank you goes to my co-promotor and daily supervisor Dr. ir. Erik-Jan van Kampen. Thank you for believing in me from day one. I remember the first time that we met face-to-face at the old Delft station. At that time, I really appreciated the care and support you treat your students with by cycling all the way from the furthest faculty block to the train station and back just to welcome me in TU Delft. But this small gesture was just the beginning. You have been always available 24/7, accompanying me through this journey. Although residing remotely I knew that you were always with me, I just had to drop you an email or a message. Thank you for listening to my technical and not so technical issues and requests and to patiently try to understand and help me out to a solution. Your time management and work attic are an example to everyone. Thank you for always making me feel welcome at TU Delft, offering to share your office with me while I am in Delft and introduce me and promoting me and my work with your colleagues both in Delft and in the World.

My third thank you goes to Dr. Qi Ping Chu who I hope is enjoying his well deserved retirement. Although we only met a couple of times, you also made feel welcome and was always ready to exchange a discussion being technical or not. I think your enthusiastic approach is an inspiration to all students. I vividly remember when we together with Dr. ir. de Visser and Dr. Thomas Lombaerts enthusiastically drove to the 11pm Space-X rocket launch and landing in Cape Canaveral on 7th January 2018.

Another thank you goes to all other academic staff within the Control & Simulation department that I had the honour to meet whilst I was in Delft, namely Dr. ir. Daan M.

Pool and Ir. Olaf Stroosma. I would like to thank Dr. ir. Pool for organising and including me in the road trip along the Pacific Coast Highway, the visit to NASA Ames and the NBA match. In regards, to the NASA Ames visit I would like to thank Dr. Thomas Lombaerts and Dr. ir. Peter Zaal TU Delft Alumni for making this visit possible and for being our guides throughout the whole visit. I would also like to thank all support staff, especially Ms. Bertine Markus for always being available and assist me in the best possible way in all the was required.

I would like to thank all my fellow Ph. D. candidates and M. Sc. students that I had the opportunity to meet and exchange discussions with. A special thanks goes to Dirk van Baelen for always being a point of reference with other students, for including me in all extracurricular activities both in Delft and during conferences. Thank you for everything that you did for us as a group and for always making me feel welcome. Moreover, I would like to thank Wei, Tijmen, Marc, Martin, Twan, Xuerui, Tigran, Stephan, Imrul, Rowenna, Luc and Daniel for making conference trips memorable.

Also, I would also like to thank the Graduate School administration both at the central office and at our faculty. Thank you for being supportive although I was not residing in TU Delft. Also I would like to congratulate all the lecturer in all the GS courses I followed for enhancing my research and transferable skills. Another thanks goes to Mr. Yetim from the International office that thoroughly assisted me in all that was required in all my international exchanges.

During my Ph. D. journey I was mainly staying in Gozo, Malta working as a lecturer at the Malta College of Arts, Science and Technology (MCAST). I would like to use this opportunity to thank all my fellow colleagues at this campus for their encouragement and for making it possible for me to travel to Delft and conferences by being flexible in exchanging lecture slots. Also, I would like to thank Mr. Godwin Grech the campus director for always being supportive in all my research ventures while keeping our students as top priority.

Special thanks goes to my fiancée Ms. Liliana Borg for always being present, for supporting me in all ups and downs, for continuously motivating me and for always listening to my thoughts. Your brilliant mind helps me think outside the box. I am also grateful to my fiance's family especially Mr. Angelo Borg, Mrs. Maria Borg and Mrs. Carmela Micallef for welcoming me in your houses and hearts.

I am deeply grateful to my sisters Mrs. Maria Cauchi and her husband Mr. Mario Reno Cauchi and Ms. Pauline Zammit and my brother Mr. Joseph Zammit for supporting me unconditionally. I would also like to thank my grandmother Mrs. Francesca Pace for her encouragement in my studies. I am also grateful to my nieces Ms. Martina Cauchi and Ms. Julia Cauchi for their company whilst writing this dissertation. Finally, but most importantly, I am forever grateful to my parents, Mr. Vincent Zammit and Mrs. Josephine Zammit for your unconditional love. Without you I would not be writing this today.

Christian Zammit
Victoria, Gozo, Malta, September 2021

CURRICULUM VITÆ

Christian ZAMMIT

27-09-1989 Born in Victoria, Gozo, Malta.

EDUCATION

2015–2021 Ph. D. in Aerospace Engineering
Delft University of Technology, The Netherlands
Thesis: 3D Path Planning for UAVs in Dynamic environments in the presence of Uncertainties
Promotor: Prof. dr. M. Mulder

2015–2016 Graduate Teaching Certificate in Vocational Education and Training (GTC in VET) with Distinction
Malta College of Arts, Science and Technology (MCAST)

2011–2014 Master of Science in Engineering (Cum Laude)
University of Malta, Malta
Thesis: An Algorithm for the Automatic Taxi of Fixed Wing Aircraft

2007–2011 Bachelor of Science in Engineering
University of Malta, Malta
Thesis: Idle State Detection in Motor Imagery Based BCI

AWARDS

2015 Best Paper of Session, 34^{th} DASC
2007 Best Science Student, Sir M. A. Refalo, Institute for Further Education

LIST OF PUBLICATIONS

.1. JOURNAL PAPERS

3. **C. Zammit** and E. van Kampen, *Real-time 3D UAV path planning in dynamic environments in the presence of uncertainty*, Journal of Guidance, Control and Dynamics, Submitted for publication.

2. **C. Zammit** and E. van Kampen, *A Comparative Analysis of the A* and RRT Algorithms in Real-time 3D UAV path planning*, Journal of Aerospace Science and Technology, Submitted for publication.

1. **C. Zammit** and E. van Kampen, *Comparison between A* and RRT Algorithms for 3D Path Planning*, Journal of Unmanned Systems, Accepted for publication.

.2. PEER-REVIEWED CONFERENCE PAPERS

9. **C. Zammit** and E. van Kampen, *3D real–time path planning of UAVs in dynamic environments in the presence of uncertainty*, Proceedings of AIAA Guidance, Navigation and Control Conference, Nashville, TN, 11-15 Jan., 2021, pp. 1-25, AIAA-2020-0861.

8. **C. Zammit** and E. van Kampen, *3D real–time path planning of UAVs in dynamic environments*, Proceedings of AIAA Guidance, Navigation and Control Conference, Nashville, TN, 11-15 Jan., 2021, pp. 1-22, AIAA-2020-0861.

7. **C. Zammit** and E. van Kampen, *Comparison of A* and RRT in real-time 3D path planning of UAVs*, Proceedings of AIAA Guidance, Navigation and Control Conference, Orlando, FL, 6-10 Jan., 2020, pp. 1-25, AIAA-2020-0861, doi:10.2514/6.2020-0861.

6. **C. Zammit** and E. van Kampen, *Advancements for A* and RRT in 3D path planning of UAVs*, Proceedings of AIAA Guidance, Navigation and Control Conference, San Diego, CA, 7-11 Jan., 2019, pp. 1-17, AIAA-2019-0920, doi:10.2514/6.2019-0920.

5. **C. Zammit** and E. van Kampen, *Comparison between A* and RRT Algorithms for UAV Path Planning*, Proceedings of AIAA Guidance, Navigation and Control Conference, Kissimmee, FL, 8-12 Jan., 2018, pp. 1-23, AIAA-2018-1846, doi:10.2514/6.2018-1846.

4. **N. Cauchi**, K. Theuma, C. Zammit, J. Gauci and D. Zammit-Mangion, *A Decision Support tool for Weather and Terrain Avoidance during Departure*, Proceedings of IEEE/AIAA 34th Digital Avionics Systems Conference (DASC), Prague, Czech Republic, 13-17 Sep., 2015, pp. 1-15, doi:10.1109/DASC.2015.7311384. (**Best Paper of Session**).

3. **C. Zammit** and D. Zammit-Mangion, *A control technique for automatic taxi for a fixed wing aircraft*, Proceedings of AIAA Intelligent Systems Conference, National Harbour, MD, 13-17 Jan., 2014, pp. 1-15, AIAA-201-1163, doi:10.2514/6.2014-1163.

2. **C. Zammit** and D. Zammit-Mangion, *An enhanced automatic taxi control system for a fixed wing aircraft*, Proceedings of AIAA Guidance, Navigation and Control Conference, National Harbour, MD, 13-17 Jan., 2014, pp. 1-16, AIAA-2014-1300, doi:10.2514/6.2014-1300.

1. **A. Alapetite**, R. Fogh, D. Zammit-Mangion, C. Zammit, I. Agius, M. Fabbri, M. Pregnolato, L. Becouam *Direct tactile manipulation of the flight plan in a modern aircraft cockpit*, International Conference on Human-Computer Interaction in Aerospace (HCI-Aero 2012), Brussels, Belgium, 12-14 Sep., 2012, pp. 1-4.