Autonomous navigation of a rotary wing flying vehicles for precision agriculture

Adel Mokrane

THESE DE DOCTORAT

NNT : 2023UPAST200

Autonomous Navigation of Rotary Wing Flying Vehicles for Precision Agriculture

Navigation Autonome des Engins Volants à Voilures Tournantes pour l'Agriculture de Précision

Thèse de doctorat de l'université Paris-Saclay et de l'université Abou Bekr Belkaid -Tlemcen-

École doctorale n° 580, sciences et technologies de l'information et de la communication (STIC)
Spécialité de doctorat : Robotique
Graduate School : Sciences de l'ingénierie et des systèmes
Référent : Université de Versailles Saint-Quentin en Yvelines (UVSQ)

Thèse préparée dans les unités de recherche **LISV (Laboratoire d'Ingénierie de Systèmes de Versailles)** et **LAT (Laboratoire d'Automatique de Tlemcen)**, sous la direction de **Abdelaziz BENALLEGUE**, Professeur, et **Amal CHOUKCHOU-BRAHAM**, Professeur, le co-encadrement de **Abdelhafid EL-HADRI**, Maitre de Conférences et **Brahim CHERKI**, Professeur

Thèse soutenue à Tlemcen, le 21 décembre 2023, par

Adel MOKRANE

Composition du Jury

Membres du jury avec voix délibérative

Amine HADJ ABDELKADER Professeur, Univ. Tlemcen	Président
Rochdi MERZOUKI Professeur, Univ. Lille	Rapporteur & Examinateur
Abdellah MOKHTARI Professeur, Univ. Oran 2	Rapporteur & Examinateur

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre : Navigation Autonome des Engins Volants à Voilures Tournantes pour l'Agriculture de Précision

Mots clés : Navigation autonome, Quadrirotor, Agriculture de Précision.

Résumé : Alors que les applications d'agriculture de précision (AP) sont déjà bien établies pour les quadrirotors, la navigation autonome, en particulier dans les champs agricoles complexes, non structurés et capricieux, reste un défi permanent pour ces véhicules. Les stratégies de navigation aérienne autonome ne doivent pas se limiter à garantir que la cible est atteinte sans entrer en collision avec les obstacles. Elles doivent également viser à identifier le chemin et la trajectoire optimaux (ou sous-optimaux) que le quadrirotor doit utiliser pour déplacer de son point de départ à sa destination, en tenant compte de toutes les contraintes pratiques qui peuvent s'appliquer. En général, la navigation aérienne autonome ne peut pas être résolue directement. Cependant, elle peut être divisée en plusieurs sous-problèmes : planification de la trajectoire, génération et optimisation de la trajectoire, replanification de la trajectoire et contrôle et suivi de la trajectoire. Cette thèse propose une solution complète et efficace du problème de navigation autonome pour un quadrirotor afin d'accomplir des missions de vol sures et stables pour la télédétection dans les champs agricoles. La solution est multi-phase et basée sur une combinaison d'algorithmes utilisés pour la première fois dans un scénario de AP. Certains de ces algorithmes ont été choisis avec précision parmi ceux actuellement disponibles dans la littérature afin d'identifier la meilleure combinaison d'algorithmes de navigation autonome. Dans la première phase, une définition hors ligne de la trajectoire optimale a été utilisée. Cette phase s'est généralement d´déroulée en deux étapes consécutives. La première étape utilisait des représentations de l'environnement, principalement des cartes artificielles d'occupation (OGM) et des cartes numériques d'élévation (DEM), pour générer des trajectoires géométriques optimales et localiser des points de référence de position. Des contraintes contenant les points de passage extraits et la vitesse/accélération à ces points de passage ont été formées. Dans un deuxième temps, un algorithme de régulation quadratique linéaire (LQR) a été adopté pour générer des trajectoires minimales optimales. Le générateur de trajectoires LQR traite les contraintes des points de passage comme étant souples. Cela garantit à la fois la relaxation des contraintes des points de passage et la génération de trajectoires de position stables. Dans la phase de replanification de la trajectoire, un algorithme de champ potentiel artificiel (APF) amélioré a été utilisé pour replanifier localement la trajectoire du quadrirotor en temps réel. L'algorithme APF amélioré utilise des forces artificielles pour éloigner le véhicule de tout obstacle inattendu. Dans la phase finale, un contrôleur géométrique a été conçu pour suivre les trajectoires générées tout en pointant vers une direction de pointage spécifiée. Dans ce cas, le contrôleur devait utiliser les mesures vectorielles bruitées fournies par l'unité de mesure inertielle (IMU) pour construire l'attitude du quadrirotor en temps réel le long de la trajectoire de position générée. Le contrôleur géométrique a été mis en œuvre sur le groupe euclidien spécial SE(3) afin d'éviter les singularités associées aux angles d'Euler ou les ambiguïtés accompagnant la représentation en quaternions. Les performances de la stratégie de navigation autonome proposée ont été d´démontrées à l'aide de simulations illustratives dans différents scénarios et les résultats ont confirmé l'efficacité de la stratégie proposée. Les résultats ont confirmé l'efficacité de la stratégie proposée. En particulier, des trajectoires de guidage géométrique sures ont été obtenues. Des trajectoires de position optimales satisfaisant aux contraintes des points de passage ont été générées avec succès en minimisant l'instantanéité du quadrirotor.

Title : Autonomous Navigation of Rotary Wing Flying Vehicles for Precision Agriculture

Keywords : Autonomous navigation, Quadrotor, Precision Agriculture.

Abstract : While Precision Agriculture (PA) applications are already well-established for quadrotors, achieving autonomous navigation especially in intricate, unstructured, and fickle agricultural fields still remains an ongoing challenge for such vehicles. Autonomous aerial navigation strategies should not be limited to ensuring that the target is reached without colliding with the obstacles. It should also aim to identify the optimal (or sub-optimal) path and trajectory for the quadrotor to travel from its starting location to the destination, taking into consideration any practical constraints that may apply. In general, the autonomous aerial navigation cannot be solved directly. However, it can be divided into multi-phase sub-problems: path planning, trajectory generation and optimization, trajectory re-planning and trajectory control and tracking. As its core, this thesis proposes a complete and efficient solution of the autonomous navigation problem for a quadrotor to accomplish safe and stable flight missions for remote sensing purposes in agricultural fields. The solution is multi-phase and based on a combination of algorithms used for the first time in a PA scenario. Some of these algorithms were accurately among those currently available in the literature aiming to identify the best combination of autonomous navigation algorithms. In the first phase, an offline definition of optimal trajectory was used. This phase was typically performed in two consecutive stages. The first stage made use of environment representations, mainly artificial Occupancy Grid Maps (OGMs) and Digital Elevation Maps (DEM), to generate optimal geometric paths and locate reference position way-points. Constraints containing the extracted position way-points and velocity/acceleration at these way-points were formed. In the second stage, a Linear Quadratic Regulator (LQR) algorithm was adopted to generate optimal minimum snap trajectories. The LQR trajectory generator treats the constraints at the way-points to be soft. This would guarantee both the relaxation on fulfilling the way-points constraints and a generation of stable position trajectories. In the trajectory re-planning phase, an Improved Artificial Potential Field (APF) algorithm was used to locally re-plan the quadrotor trajectory in real-time. The Improved APF uses artificial forces to move the vehicle away from any unexpected obstacle. In the final phase, a geometric controller was designed to track the generated trajectories while pointing towards a specified pointing direction. The controller in this case was required to use the noisy vector measurements provided by the Inertial Measurement Unit (IMU) to construct the quadrotor's attitude in real-time along the generated position trajectory. The geometric controller was implemented on the Special Euclidean SE(3) group aiming to avoid singularities associated with Euler angles or ambiguities accompanying quaternion representation. The performance of the proposed autonomous navigation strategy was demonstrated using illustrative computer simulations in different scenarios and the results have confirmed the effectiveness of the proposed strategy. In particular, safe geometric guiding paths were achieved. Optimal position trajectories that satisfy the way-points constraints were successfully generated with minimized quadrotor's snap.

Résumé

Bien que les applications d'agriculture de précision (AP) pour les quadcoptères soient déjà bien établies, la navigation autonome, en particulier dans les champs agricoles complexes, non structurés et imprévisibles, reste un défi constant pour ces véhicules. Les stratégies de navigation aérienne autonome ne doivent pas se limiter à atteindre la destination sans entrer en collision avec des obstacles. Elles doivent également viser à déterminer le chemin et la trajectoire optimaux que le quadcoptère doit utiliser pour se déplacer de son point de départ à sa destination, en tenant compte de toutes les contraintes pratiques qui peuvent s'appliquer. En général, la navigation aérienne autonome ne peut pas être résolue directement. Cependant, elle peut être divisée en plusieurs sous-problèmes : la planification du mouvement, la génération et l'optimisation de la trajectoire, la replanification de la trajectoire, et le contrôle et le suivi de la trajectoire. Dans cet article, nous proposons une solution complète et efficace au problème de la navigation autonome pour un quadrotor afin d'effectuer des missions de vol sûres et stables pour la télédétection dans les champs agricoles. La solution est à plusieurs niveaux et repose sur une combinaison d'algorithmes utilisés pour la première fois dans un scénario de navigation autonome. Certains de ces algorithmes ont été soigneusement sélectionnés parmi ceux actuellement disponibles dans la littérature afin de déterminer la meilleure combinaison d'algorithmes de navigation autonome. Tout d'abord, une définition hors ligne de la trajectoire optimale a été effectuée. En général, cette phase s'est déroulée en deux étapes consécutives. Dans la première phase, des représentations de l'environnement, principalement des cartes d'occupation artificielle et des cartes d'élévation numériques, ont été utilisées pour générer des trajectoires géométriques optimales et trouver des points de référence de position. Les contraintes ont été formées à partir des points de repère extraits et de la vitesse/accélération au niveau de ces points de repère. Ensuite, un algorithme de régulation linéaire quadratique (LQR) a été utilisé pour générer des trajectoires optimales minimales. Le générateur de trajectoires LQR traite les contraintes d'intersection comme étant lisses. Cela garantit à la fois le relâchement des contraintes liées aux points de passage et la génération de trajectoires de position stables. Dans la phase de replanification de la trajectoire, un algorithme APF (Artificial Potential Field) amélioré a été utilisé pour la replanification locale de la trajectoire du quadcoptère en temps réel. L'algorithme APF amélioré utilise des forces artificielles pour éloigner le véhicule d'obstacles inattendus. Dans la phase finale, un contrôleur géométrique a été développé pour suivre les trajectoires générées pointant dans une direction spécifique. Dans ce cas, le contrôleur devait utiliser les mesures vectorielles bruitées de l'unité de mesure inertielle pour construire la position en temps réel du quadrotor le long de la trajectoire de position générée. Le contrôleur géométrique a été mis en œuvre dans le groupe euclidien spécial SE(3) pour éviter les singularités liées aux angles d'Euler ou les ambiguïtés dans la représentation en quaternions. La performance de la stratégie de navigation autonome proposée a été démontrée par des simulations illustratives dans différents scénarios, et les résultats ont

confirmé l'efficacité de la stratégie proposée. Les résultats ont confirmé l'efficacité de la stratégie proposée. En particulier, des trajectoires de guidage géométrique sûres ont été obtenues. Des trajectoires de position optimales ont été générées, satisfaisant aux contraintes des points de passage et minimisant l'élan du quadrotor. Il a été démontré que le contrôleur géométrique suivait les trajectoires générées et atteignait une stabilité asymptotique quasi globale pour les rotations.

Dedication

To my family.

Acknowledgments

Firstly, I would like to thank my thesis advisors Prof. Amal CHOUKCHOU-BRAHAM, Prof. Abdelaziz BENALLEGUE, Prof. Brahim CHERKI and Dr. Abdelfid El-HADRI for their support throughout the duration of this thesis. I could not have imagined having better advisors for my Ph.D study. Pursuing this degree has been a tremendous growth experience for me and I am immensely grateful for having had the opportunity to work in their research groups. Our meetings together and their feedback throughout this project were instrumental in guiding me from the early start through to the end of this thesis.

Secondly, I would like to thank my colleagues in both laboratories: Laboratoire d'Automatique de Tlemcen LAT and Laboratoire D'Ingénierie des Systèmes de Versailles LISV for all of their support over the past few years. I was very fortunate to be able to share a lab with so many talented researchers who were always willing to lend a hand. A special thanks also to my friend Ismat MESLOULI for his support. It was a great pleasure to share a work place with him.

Thirdly, I would also like to extend a special thanks to my parents for all of their love and support throughout my graduate studies. You have always encouraged me to pursue my goals and I would not have been able to complete this degree without you by my side.

Besides, I would like to thank the external members of the jury ... for accepting to review my thesis manuscript.

Finally, this research is supported by the Partenariats Hubert Curien (PHC) Tassili and the Direction Générale de la Recherche Scientifque et du Développement Technologique (DGRSDT), Algeria.

Contents

Résumé ii

Dedication iv

Acknowledgments v

Contents vi

List of Figures ix

1 Introduction 1
- 1.1 Overview . . . 1
- 1.2 Thesis Scientific Context . . . 2
- 1.3 Research Question . . . 3
- 1.4 Thesis Contribution . . . 3
- 1.5 Thesis Organization . . . 4

2 Quadrotors: State of The Art 6
- 2.1 Introduction . . . 6
- 2.2 UAV Systems Classifications . . . 6
  - 2.2.1 Based on Size and Weight . . . 6
  - 2.2.2 Based on Control Configuration . . . 9
  - 2.2.3 Based on Autonomy Level . . . 13
- 2.3 UAV Applications . . . 13
  - 2.3.1 Military Applications . . . 14
  - 2.3.2 Civilian Applications . . . 14
  - 2.3.3 Transport Applications . . . 15
- 2.4 Quadrotor Autonomous Vehicle Systems . . . 16
  - 2.4.1 Hardware Modular Approach . . . 16
  - 2.4.2 Software Modular Approach . . . 22
- 2.5 Quadrotor Autonomous Navigation: Definition . . . 22
- 2.6 Quadrotor Autonomous Navigation: Applications . . . 23
- 2.7 Conclusion . . . 25

3 Quadrotors Path Planning **28**
3.1 Introduction . . . 28
3.2 Robot Path Planning . . . 28
3.2.1 Path Planning: Definition . . . 28
3.2.2 Configuration Space . . . 29
3.2.3 Types of Path Planning . . . 30
3.2.4 Path Planning Paradigms . . . 31
3.2.5 Map-Based vs. Mapless Path Planning Methods . . . 31
3.3 Map Representation for Path Planning . . . 32
3.4 Robotic Mapping . . . 35
3.4.1 Mapping without Localization . . . 35
3.4.2 Simultaneous Localization and Mapping . . . 36
3.4.3 Types of Robotic Maps . . . 37
3.5 UAV Path Planning Taxonomy . . . 38
3.5.1 UAV Point-to-Point Path Planning . . . 38
3.5.2 UAV Coverage Path Planning . . . 46
3.6 Conclusion . . . 48

4 Quadrotors Trajectory Generation & Control Strategies **50**
4.1 Introduction . . . 50
4.2 Quadrotor Dynamics Model . . . 50
4.3 Quadrotors Trajectory Control Strategies . . . 52
4.3.1 Linear Trajectory Control Systems . . . 53
4.3.2 Nonlinear Trajectory Control Systems . . . 56
4.3.3 Intelligent Trajectory Control Systems . . . 58
4.3.4 Geometric Control Systems . . . 59
4.4 Trajectory Generation for Quadrotors . . . 61
4.4.1 Continuous Geometrical Curved-Based Methods . . . 62
4.4.2 Optimal Control-Based Methods . . . 63
4.4.3 Differential Flatness Based Methods . . . 66
4.5 Conclusion . . . 67

5 Quadrotor Autonomous Navigation: Methodology **69**
5.1 Introduction . . . 69
5.2 Global Autonomous Navigation Structure . . . 69
5.3 Aerial Terrain Mapping . . . 70
5.4 Optimal Path Finding . . . 71
5.5 Trajectory Generation and Optimization . . . 72
5.5.1 Trajectory Generation Constraints . . . 72
5.5.2 LQR Optimal Control Trajectory Generation . . . 73
5.6 Online Trajectory Re-planning . . . 75
5.7 Trajectory Control and Tracking . . . 77
5.8 Conclusion . . . 83

6 Simulation Experiments **85**
6.1 Introduction . . . 85
6.2 Experiment Setup . . . 85
6.2.1 Simulation Setup . . . 85
6.2.2 Environmental Settings . . . 86
6.3 Results & Discussions . . . 87
6.3.1 Path Planning . . . 87
6.3.2 Trajectory generation . . . 88
6.3.3 Trajectory Control . . . 90
6.3.4 Trajectory Re-planning . . . 93
6.4 Discussion . . . 95
6.5 Conclusion . . . 100

7 General Conclusion & Perspectives **102**

A List of Publications **104**
A.1 International Conference Papers . . . 104
A.2 International Journal Articles . . . 104

List of Figures

2.1 UAV classification based on size and weight. 7
2.2 (*a*) Micro and (*b*) nano UAVs. 7
2.3 CPX4 mini UAVs. 8
2.4 Tactical UAVs: (*a*) Watchkeeper WK450 and (*b*) Sagem Spewer. 8
2.5 A British MQ-9A Reaper UCAV. 9
2.6 (*a*) MQ-1 Predator MALE UAV and (*b*) Global Hawk HALE UAV 10
2.7 Different types of UAVs (*a*) rotary-wing, (*b*) fixed-wing (*c*) flapping-wing (Ornithopters) (*d*) flapping-wing (Entomopters). 10
2.8 Rotary-wings UAVs: (*a*) tricopter, (*b*) quadcopter, (*c*) hexacopter and (*d*) octocopter. 11
2.9 Fixed-wings UAVs: (*a*) M^2AV Carolo, (*b*) eBee. 12
2.10 Flapping-wings UAVs: (*a*) Nano Humming, (*b*) The Harvard RoboBee and (*c*) The BionicOpter. 12
2.11 Hybrid-wing UAV design. 13
2.12 Communication means of UAVs through GCS. 14
2.13 Some military applications of UAVs (Udeanu, Dobrescu, and Oltean, 2016). 15
2.14 Civilian applications of UAVs (Sivakumar and TYJ, 2021). 16
2.15 Transport applications of UAVs: (*a*) ambulance drone, (*b*) delivery drone and (*c*) air taxi. 17
2.16 Hardware components that built up a quadrotor. 18
2.17 Quadrotor configurations: (*a*) X-quad, (*b*) +-quad and (*c*) H-quad. 18
2.18 Scheme of propellers for maneuvering in "+" configuration (Lukmana and Nurhadi, 2015). 19
2.19 FCSs examples: (*a*) Pixhawk PX4 FMUv5, (*b*) Chemira Autopilot for Paparazzi, (*c*) Ardupilot Mega (APM) and (*d*) N3 DJI. 20
2.20 Quadrotor software module for autonomous navigation. 22
2.21 Quadrotor autonomous navigation elements. 23
2.22 Quadrotor autonomous navigation structures. 26

3.1 Robot path planning. 29
3.2 (*a*)The workspace and (*b*) the configuration space of a robot. 30
3.3 Different path planning paradigms. 32
3.4 Map-based vs. mapless path planning. 33
3.5 Continuous map vs. hybrid map. 34
3.6 Exact cell decomposition vs. fixed cell decomposition. 34

3.7 Occupancy-based grid map (Nuss et al., 2018). . . . 36
3.8 (*a*) Physical environment (*b*) topological representation. . . . 38
3.9 Point-to-point path planning taxonomy. . . . 39
3.10 Illustration of the APF algorithm. . . . 39
3.11 Probabilistic Roadmap algorithms: (*a*) VG, (*b*) VD (Siegwart, Nourbakhsh, and Scaramuzza, 2011). . . . 40
3.12 RRT evolution illustration (Siegwart, Nourbakhsh, and Scaramuzza, 2011). . . . 41
3.13 Dijkstra algorithm flowchart. . . . 43
3.14 A* algorithm flowchart. . . . 44
3.15 RL schematic diagram. . . . 45
3.16 (*a*) Trapezoidal decomposition vs. (*b*) boustrophedon decomposition (Galceran and Carreras, 2013). . . . 47
3.17 (*a*) Wavefront distance transform vs. (*b*) Coverage path (Galceran and Carreras, 2013). . . . 47
3.18 (*a*) Approximate cell decomposition vs. (*b*) Coverage spiral path (Galceran and Carreras, 2013). . . . 48

4.1 Quadrotor inertial and body-fixed frames. . . . 51
4.2 SE(3) trajectory of a quadrotor (Dhullipalla et al., 2019). . . . 52
4.3 Trajectory generation and control structure. . . . 53
4.4 PID controller structure for a quadrotor (Jiao et al., 2018). . . . 54
4.5 LQR/LQG control structure. . . . 55
4.6 *(a)* Conventional proportional controller vs. *(b)* gain scheduled controller (Sawyer, 2015). . . . 55
4.7 Control structure of $\mathcal{H}_{\text{inf}}$ controller (Doyle et al., 1988). . . . 56
4.8 Control structure of an adaptive controller. . . . 58
4.9 Fuzzy logic controller applied to a quadrotor. . . . 59
4.10 Sliding mode ANN control structure (Raiesdana, 2020). . . . 60
4.11 Schematic visualization of the different mappings. . . . 61
4.12 Geometrical curves: (*a*) Dubin (R: Right, L: Left, S: Straight (*b*) polynomial (*c*) splines (*d*) Bézier. . . . 63
4.13 Differential flatness property and its mapping. . . . 67

5.1 An overview of the navigation global structure. . . . 70
5.2 The map with both main lines (black) and auxiliary lines (red). . . . 71
5.3 The 3D local goals extraction flowchart. . . . 72
5.4 The improved APF method: (*a*) the improved resultant force model for solving the GNRON problem, (*b*) the RHG for solving the local minima problem. . . . 76
5.5 The flowchart of the proposed trajectory re-planning method based on the improved APF. . . . 78
5.6 The structure of the geometric controller. . . . 79

6.1 Environment representation of Scenario ① in (*a*) 2D and (*b*) 3D. . . . 86
6.2 Environment representation of Scenario ② Terrain 1 in (*a*) 2D and (*b*) 3D. . . . 87
6.3 Environment representation of Scenario ② Terrain 2 in (*a*) 2D and (*b*) 3D. . . . 87
6.4 The generated local goals. . . . 88
6.5 The generated coverage paths and way-points: (*a*) Terrain 1, (*b*) Terrain 2 88

6.6 Trajectory on $SE(3)$ obtained by (a) our algorithm, (b) algorithm in (Dhullipalla et al., 2019). 89
6.7 The generated optimal trajectory for Terrain 1: (a) 2D, (b) 3D. 90
6.8 The generated optimal trajectory for Terrain 2: (a) 2D, (b) 3D. 90
6.9 The vector measurements of the flight in Scenario ①. 91
6.10 The vector measurements of the flight in Scenario ②: (a) Terrain 1, (b) Terrain 2. . 91
6.11 The position tracking results of the flight in Scenario ①. 92
6.12 The position tracking results of the flight in Scenario ② in Terrain 1. 92
6.13 The position tracking results of the flight in Scenario ② in Terrain 2. 93
6.14 The attitude tracking results of the flight in Scenario ①. 93
6.15 The attitude tracking results of the flight in Scenario ② in: (a) Terrain 1, (b) Terrain 2. 94
6.16 (a) The pointing, (b) thrust directions and (c) their representations in 3D for Scenario ①. 94
6.17 (a) The pointing, (b) thrust directions and (c) their representations in 3D for Scenario ② Terrain 1. 95
6.18 (a) The pointing, (b) thrust directions and (c) their representations in 3D for Scenario ② Terrain 2. 96
6.19 The global position and the re-planned trajectories: (a) single obstacle, (b) multiple obstacles. 96
6.20 The thrust direction obtain by (a) our algorithm, (b) algorithm in (Dhullipalla et al., 2019). 98
6.21 The pointing direction obtain by (a) our algorithm, (b) algorithm in (Dhullipalla et al., 2019). 98
6.22 The pointing direction (top-view) obtain by (a) our algorithm, (b) algorithm in (Dhullipalla et al., 2019). 99
6.23 The position error profile for Scenario ①. 99
6.24 The position error profile for Scenario ② for: (a) Terrain 1, (b) Terrain 2. 100

Chapter 1

Introduction

1.1 Overview

Unmanned Aerial Vehicle (UAV) or drone is the term that refers to the aircraft that can fly without a human pilot on-board0. UAVs are gaining more interest in applications that can reduce human effort or where human piloting is impractical. These applications involve structural inspection, Precision Agriculture (PA), security, emergency response, surveillance and photography. Among these vehicles, Unmanned Aerial Vehicles or UAVs have seen substantial progression in the past decades. They have become very popular due to their relative compact size, high maneuverability and low manufacturing cost. UAVs can be either tele-operated remotely by a pilot or autonomously fly by relying on the information provided by the different sensors mounted on them. Although UAVs were primarily dedicated for military purposes, however with the the advent of the Internet of Things (IoT) and the development in the computing efficiency, mini and micro UAVs have received a significant attention from the robotics community.

Several schemes have been adopted for to classifying the UAVs, however, these classifications are not unique. In order to do so, a large number of different characteristics are used. Mass, size, altitude of operation, autonomy, propulsion system, flying principle ... etc. are some of these characteristics. It is worth noting that each scheme has its advantages and drawbacks. The most common classification is based on the type of wings. Hence, UAVs can be rotory-wing, fixed-wing or flapping-wing. A quadrotor, which is one of the most well-known under-actuated multi-rotor UAV systems, consists of two pairs of rotors in cross configuration capable of spinning at different angular velocities for achieving a certain motion (Sun et al., 2020). Thanks to their significant mechanical simplicity, several commercial quadrotors models have been developed for autonomous applications by the use of advanced mechanical and electrical technologies in association with fast processors and accurate sensor measurements.

As the demands of autonomous quadrotors are increasing in the recent years, navigation problems have become an extensive research subject in the field of robotics. Hence, navigation systems are critical elements of autonomous quadrotors where higher level of autonomy and more stable flight are sought for a robust and efficient flight missions especially when dealing with complex tasks. Autonomous navigation in known environments involves mainly four processes: path planning, trajectory generation, trajectory control and trajectory re-planning (Kanellakis and Nikolakopoulos, 2017). The first and the last processes aim to determine the shortest path between two configura-

tions. Trajectory generation uses an optimization algorithm where the input is the geometric information produced by the first process in order to generate a smooth position/velocity/acceleration profiles for the quadrotor. Trajectory control adopts control laws in order to track the generated trajectory profiles. Autonomous navigation can be categorized into inertial, satellite and vision-based navigation (Arafat, Alam, and Moh, 2023). Inertial navigation processes the information provided by an Inertial Measurement Unit internally to drive the quadrotor from its starting location to a predefined location. Satellite navigation is based on the data coming from a positioning sensor like GPS. Visual sensors can provide online information about the environment and perform visual-based navigation.

Autonomous navigation can be classified also based on the structure designed. On one hand, coupled structure autonomous navigation combines the planning and the trajectory control processes. The advantage of such structure is its effectiveness in real-time applications. However, it tends to obtain results which are heuristic only (not optimal). Besides, coupled structure has a heavy reliance on the on-board sensors which are not immune to errors. As a result, catastrophic control action may occur. On the other hand, decoupled structure uses a simple decouple design of the mentioned processes to solve the problem of autonomous navigation. Such structure is advantageous since the trajectories may be obtained from a maximization/minimization of a certain objective function (time, effort, energy, a position derivation, etc). However, it is mainly dedicated for offline applications. Owing to both structures capabilities to provide solutions for the autonomous navigation problem, combining them in one single structure improves the accuracy and the effectiveness of the navigation system.

From these aspects, this thesis presents an autonomous navigation approach for a single quadrotor where the path planning and part of the trajectory generation (position generation) are performed using the offline decoupled structure, however, the other part of the trajectory generation (attitude generation), the tracking controller and the trajectory re-planning are implemented using the online coupled structure. This autonomous navigation approach provides effective and accurate solution for quadrotors for both indoor and outdoor scenarios.

1.2 Thesis Scientific Context

This thesis is part of an Algerian-French joint supervision between the University of Paris Saclay and the University of Abou Bekr Belkaid, Tlemcen. It is entitled "Navigation Autonome d'un Engin Volant à Voilures Tournantes pour l'Agricuture de Précision". The work presented in this report is the fruit of a thesis co-financed by the French Ministry of Higher Education, Research and Innovation (MESRI) and the Algerian Ministry of Higher Education and Scientific Research (MESRS). The thesis is supervised by Professor BENALLEGUE Abdelaziz (Laboratoire d'Ingénierie des Systèmes de Versailles, Université de Versailles Saint Quentin en Yvelines, France) and Professor CHOUKCHOU-BRAHAM Amal (Laboratoire d'Automatique de Tlemcen LAT, Université de Tlemcen, Algeria) and co-supervised by Dr. El HADRI Abdelhafid (Laboratoire d'Ingénierie des Systèmes de Versailles, Université de Versailles Saint Quentin en Yvelines, France) and Professor CHERKI Brahim (Laboratoire d'Automatique de Tlemcen LAT, Université de Tlemcen, Algeria). The two laboratories join thanks to this thesis subsidized by the programme of Partenariat Hubert Curien PHC Tassili. This thesis is a continuation of work started at the laboratory level on the synthesis of control, observation, data fusion and the design of a UAV. Its purpose is to generate an optimal trajectory for an already built UAV platform in the interest of navigating, localization and mapping in the context of PA.

1.3 Research Question

The main topic of this thesis is to investigate the different navigation strategies (path planning, trajectory generation and control methods) developed for autonomous vehicles especially UAVs.

Nowadays, agriculture is facing many challenges. These challenges can be economic such as productivity (quality and quantity), cost effectiveness and labour shortage in rural areas. They can be also global such as population increase, urbanization, environment degradation, and of course climate change (Gnip, Charvat, Krocan, et al., 2008). In addition, reliable monitoring of crops and accurate detection and identification for analyzing the plants condition are critical to be performed regularly to lessen economic expenditures, trade disruptions and even human health risks. Under these conditions, more advanced technologies and tools derived from scientific advances, research and development activities should absolutely be a priority. Although there exist satellites and ground-based monitoring and detection solutions, the upper hand of UAVs as they offer better detailed resolution, low cost implementation and high maneuverability and flexibility which can not be defeated. Our first research question is how to deploy UAVs to perform remote sensing in agricultural fields.

Navigation is one of the most prominent and essential abilities of autonomous vehicles as it has been and still the focus of research in the robotics field. The necessity of the autonomous aerial navigation algorithm should not be satisfied with the ability to achieve the objective without colliding with the obstacles only, but also should attempt to solve for a possible optimal or heuristic motion from an initial position configuration to the objective while satisfying the UAV kinematics and dynamics constraints. Hence, our second research question is how to improve the accuracy and the effectiveness of the UAV in known and partially known environment with obstacles.

At present, many strategies have been developed by various researchers for path planning, trajectory generation and tracking control. Each strategy has its own strengths and flaws in terms of computational intensiveness, ability in handling maximum uncertainty, the requirement of an accurate real-time navigation, ability in handling vehicle's kinematics and dynamics constraints, etc. However to exploit such strengths, these strategies can be combined together in a decouple structure. Thus, the last research question is how to build such decouple structure of navigational techniques in order to offer an efficient and complete autonomous navigation solution for UAVs.

1.4 Thesis Contribution

In this thesis, an extensive research from various perspectives have been conducted to address the above research questions. For the first question, we use a quadrotor vehicle system with a camera mounted on it. We envision to fly the quadrotor at low altitudes within irregular and unstructured agricultural scenarios for sake of performing remote sensing. This is can be achieved by pointing the camera toward the plants. For the second question, we develop a collision-free navigation technique which helps the quadrotor execute three-dimensional flight missions with high degree of autonomy and stability. Mainly two methods are presented: the point-to-point and the coverage aerial navigation. For the last question, we conduct a feasibility screening among the possible already known navigation technologies. We attempt to design a complete and efficient solution by accurately selecting navigation algorithm among those currently available in the literature aiming to identify the best combination of them. This thesis contributes in the area of autonomous navigation for quadrotors for PA scenarios. The main contribution can be described as follows:

- A collision-free point-to-point and coverage path planning algorithms for a quadrotor is proposed. The objective is to enable the quadrotor to plan geometric paths from a starting location to a target one without hitting any obstacle in a partially known environment. The input of the algorithms is a map representation and the output is a minimum distance geometric path while keeping safe distance margin from the obstacles. The presented path planning algorithms can navigate the quadrotor via optimal path successfully.

- An offline trajectory optimization based on Linear Quadratic Regulator (LQR) is designed. The input to the LQR algorithm is waypoints which are extracted from the planned geometric path. The output is a smooth minimum snap trajectory. The position trajectory must satisfy the pointing direction constraints along the whole flight. Depending on the scenarios, a corridor constraint is added in order to keep the quadrotor fly within the plants rows.

- A nonlinear real-time geometric control law that helps the quadrotor generate its attitude trajectory is implemented. The control law uses raw vector measurements provided by the on-board sensors such as Inertial Measurement Unit (IMU). These vectors allow us to avoid using the desired attitude directly in the control law as presented in most existing algorithms. The nonlinear geometric control is used also for tracking both the generated position and attitude trajectories.

- Since the environment where the quadrotor navigate is partially known, an online reactive trajectory algorithm is proposed. This algorithm is enabled when an unknown obstacle is detected. It aids the quadrotor re-plan its trajectory locally and commends it to move away from the unknown obstacles. The vehicle takes the current position (when obstacle detection occurs) as the starting configuration and the next turning point in the global path as the goal configuration.

- The performance and the effectiveness of the presented navigation strategy have been confirmed by illustrative computer simulations in different scenarios. The simulation results demonstrated the effectiveness of the presented strategy by comparing it with other strategies.

1.5 Thesis Organization

The thesis is organized into seven chapters:

Chapter 2 presents a comprehensive state of the art that is related to UAVs, specifically their different classifications, autonomous navigation and its applications. Chapter 3 provides the different path planning algorithms and techniques developed in literature. The detailed review focuses on the algorithms applied on quadrotors and discusses their advantages and drawbacks. The first part of Chapter 4 develops the quadrotor dynamic model that is used in this thesis. However, the second part is devoted to the various strategies designed for both trajectory generation and control for quadrotors. The methodology of the proposed autonomous navigation strategy is described in Chapter 5. A detail mathematical development of each algorithm is presented. Chapter 6 shows the different simulation experimental setups, results and discussions of the proposed strategy. Chapter 7 summarized the final conclusion and provides some possible directions for future work.

Chapter 2

Quadrotors: State of The Art

2.1 Introduction

Research on Unmanned Aerial Vehicles or UAVs has been increasingly developed over recent decades. UAVs havebeen used across the world for several applications ranging from military applications (e.g., enemy surveillance), civil applications (e.g., search and rescue, remote sensing) to commercial applications (e.g., package delivery). However, the development of these vehicle systems still faces many challenges because of the application complexity that increases with such development particularly with the aim of switching to fully autonomous operations. In addition, many UAVs applications are evolved to autonomously operate in complex environments where they require reliance on onboard sensors to perceive the environment they navigate in and to perform the application tasks effectively. The aim of this chapter is to introduce the different UAVs systems and give insights about autonomous navigation of such vehicle systems.

2.2 UAV Systems Classifications

An Unmanned Aerial Vehicle, which is better known with its generic term drone, can refer to intelligent autonomous aircraft that flies without human pilot onboard. Several factors, such as size, control configuration, autonomy and mean takeoff, are used in order to classify UAVs.

2.2.1 Based on Size and Weight

Based on the size and weight of the vehicle, UAVs can be categorized into four different groups: micro ($< 1kg$), small ($1 - 25kg$), medium ($25 - 150kg$) and large ($> 150kg$) (Zakaria, Abdallah, and Elshafie, 2012) (See Figure 2.1).

Micro and Nano UAVs

Micro and nano UAVs are small drones with less than fifteen centimeters in size and few tens to hundreds grams. They can fly at lower altitudes (under 300 meters). Micro and nano UAVs are equipped with propellers driven by electric motors. The designs for such class of UAVs have focused on making aerial vehicles that can operate in both inside and outside buildings, flying along

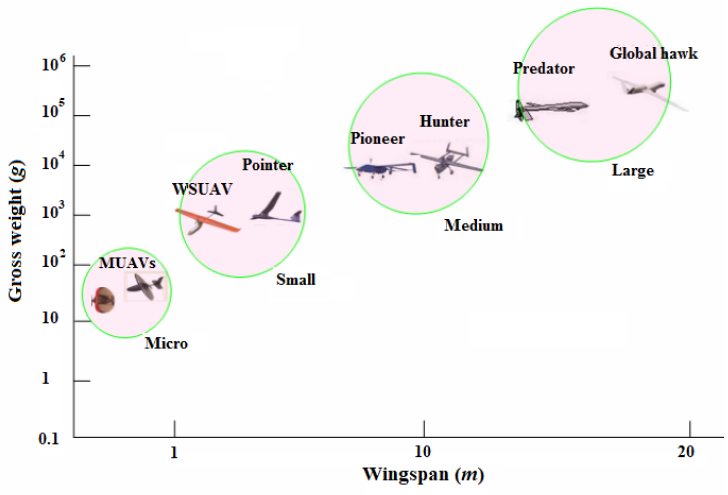

Figure 2.1: UAV classification based on size and weight.

hallways, carrying sensor devices such as transmitters and miniature TV cameras. The autonomy of such type of UAVs is about twenty minutes for a radius of action of about ten kilometers. Figure 2.2 depicts a micro and a nano UAV.

Figure 2.2: (a) Micro and (b) nano UAVs.

Mini UAVs

Mini UAVs or Mini Air Vehicles (MAVs) are types of aerial vehicles that have an endurance of few hours and dimensions of the order of a meter. MAVs can fly up to an altitude of 300 meters and operate at distances up to around 30 kilometers while carrying light payloads. Like micro UAVs, MAVs can be electrically powered. This what makes them relatively slow (some tens of kilometers per hour) and have limited obstacle avoidance abilities. The technological development required for this class of UAVs involves sensors such as cameras and lidars. MAVs are characterized by their

robustness, low noise signature, stable observation sensors and man portable (folded in a rucksack). CPX4 is an example of such UAV as depicted in Figure 2.3.

Figure 2.3: CPX4 mini UAVs.

Tactical UAVs

Tactical UAVs or TUAVs have an operational range from 30 to 500 kilometers and an altitude ranging from 200 to 5,000 meters. They have also a radius of action up to more than one hundred kilometers with autonomy of tens of hours. With the aid of visible and infrared optical sensors, most of the TUAVs are used for surveillance and reconnaissance mission purposes. Watchkeeper WK450 and Sagem Spewer manufactured by Thales and Safran, respectively, are example of such class of UAVs.

Figure 2.4: Tactical UAVs: (a) Watchkeeper WK450 and (b) Sagem Spewer.

Combat UAVs

Combat UAVs or Unmanned Combat Air Vehicles (UCAV) are fighter planes which are equipped with weapons systems or intelligence information gathering. They can perform missions of recon-

naissance, attack and shooting thanks to their technical advances in the areas of automation, data transmission, satellite link, precision guidance and stealth technology. Figure 2.5 shows an example of an UCAV.

Figure 2.5: A British MQ-9A Reaper UCAV.

MALE and HALE

MALEs or Medium Altitude Long Endurance UAVs have a wing span of 10 to 20 meters and an operational range of 500 to 1,000 kilometers. They have autonomy of thirty hours and can fly between 5,000 and 15,000 meters above sea level and they can carry a payload ranging from 1,500 to 3,500 kilograms. The search for extending the range, endurance, maximum altitude and payload capacity of MALE UAVs missions has led to the HALE or High Altitude Long Endurance UAVs whose dimensions are comparable with those of a civilian plane. HALE UAVs are able to fly 7 to 8 kilometers in one single day at very high altitude and carry a payload of one to two tons. Both MALE and HALE UAVs can carry missions such as strategic reconnaissance and surveillance, tactical reconnaissance, early detection of missile launching, target identification and designation and jamming. MQ-1 Predator and Gloabl Hawk in Figure 2.6 are examples of MALE and HALE UAVs, respectively.

2.2.2 Based on Control Configuration

A typical classification of UAVs is based on control configurations. They can be classified into rotary-wing drones, fixed-wing drones, hybrid-wing drones and flapping-wingdrones (See Figure 2.7).

Rotary-wings UAVs

Rotary-wings UAVs use rotor blades to produce a forceful thrust which can be used for lifting and propelling. This type of aerial vehicles is characterized by vertical takeoff and landing (VTOL) which can make the vehicle to hover at a place (Schauwecker et al., 2012). Rotary-wing UAVs can be either single-rotor or multi-rotor. Single-rotor UAVs (e.g., helicopters) have been not exploited much as multi-rotor UAVs. These latter are designed by number and location of propellers on the frame.

Figure 2.6: (a) MQ-1 Predator MALE UAV and (b) Global Hawk HALE UAV

Figure 2.7: Different types of UAVs (a) rotary-wing, (b) fixed-wing (c) flapping-wing (Ornithopters) (d) flapping-wing (Entomopters).

Multi-rotor drones are advantageous in terms of hovering capabilities and maintaining the speed abilities making them the ideal choice for civilian applications like monitoring and surveillance. The most popular multi-rotor UAVs are tricopters, quadrotors, hexacopters and octocopters as shown in Figure 2.8. All of these vehicles are underactuated systems and have similar dynamical models for control. Quadrotors come at cheaper prices and are faster and highly maneuverable. Hexacopters and octocopters are known by their flight stability and higher payload capacity. However, all multi-rotors require more power consumption which limits the vehicles endurance (Singhal, Bansod, and

Mathew, 2018).

Figure 2.8: Rotary-wings UAVs: (a) tricopter, (b) quadcopter, (c) hexacopter and (d) octocopter.

Fixed-wings UAVs

Fixed-wing UAVs have a simple design. The manufacturing of such type of drones is saturated nowadays thanks to the further developments and improvements of such type. Fixed-wing aerial vehicles, which are horizontal takeoff and landing (HTOL) vehicles, use a forward accelerating speed applied to fixed wings components to produce a lift. This lift is controlled by the air flowing velocity and steep angle (Singhal, Bansod, and Mathew, 2018). Unlike rotary-wing UAVs, fixed wings UAVs cannot hover at a certain place due to their typical non-holonomic constraints. Besides, they require a higher initial speed. However, fixed wings vehicles are characterized by long endurance, less power consumption and thrust loading less than 1. Figure 2.9 shows two examples of fixed-wing UAVs which are M^2AV Carolo and eBee.

Flapping-wing UAVs

A special UAV configuration that is inspired by birds (Ornithopters) and insects (Entomopters) is flapping-wing. This type of UAVs is known by lightweight and flexible wings. These configurations allow agile maneuvers while being more discreet than rotary wings, which represents another definite advantage for reconnaissance or surveillance missions. Like hybrid UAVs, flapping-wings UAVs are under development due to the complicated flapping mechanism which results in complex dynamics and power problems (Gerdes, Gupta, and Wilkerson, 2012). However unlike fixed-wing UAVs, flapping-wing drones have stable flights especially under windy conditions. Examples of flapping-wings UAVs are shown in Figure 2.10

Figure 2.9: Fixed-wings UAVs: (a) M^2AV Carolo, (b) eBee.

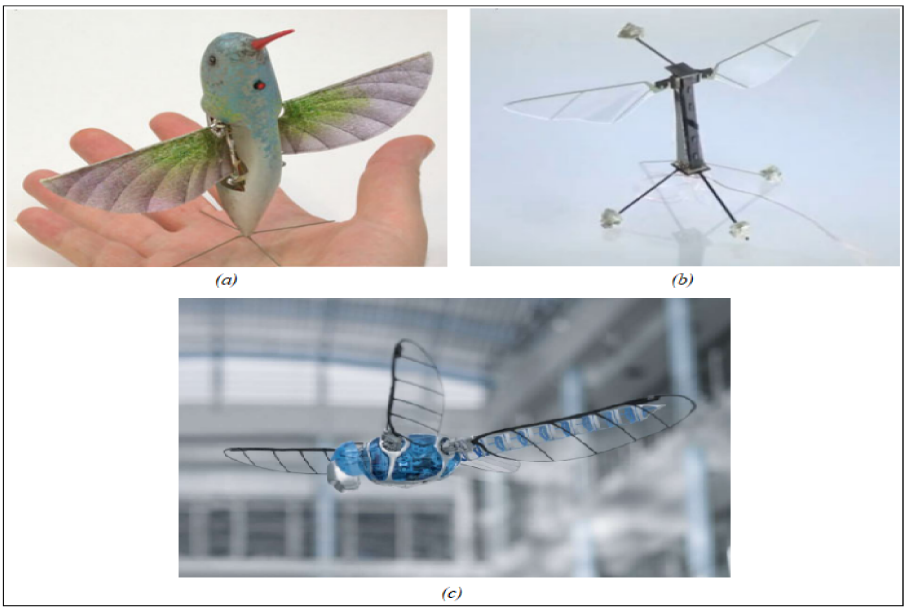

Figure 2.10: Flapping-wings UAVs: (a) Nano Humming, (b) The Harvard RoboBee and (c) The BionicOpter.

Hybrid-wing UAVs

Hybrid-wing UAVs are types of aerial vehicles which combine both configurations of rotary-wing and fixed-wing. By doing so, the drawback of a type of an UAV is compensated with the advantage of another type. This may lead to have VTOL, hovering and long endurance vehicle (See Figure 2.11). Hybrid drones require more reliable and sophisticated control algorithms especial to adapt with flight mode switching. This is why they are still under research and development.

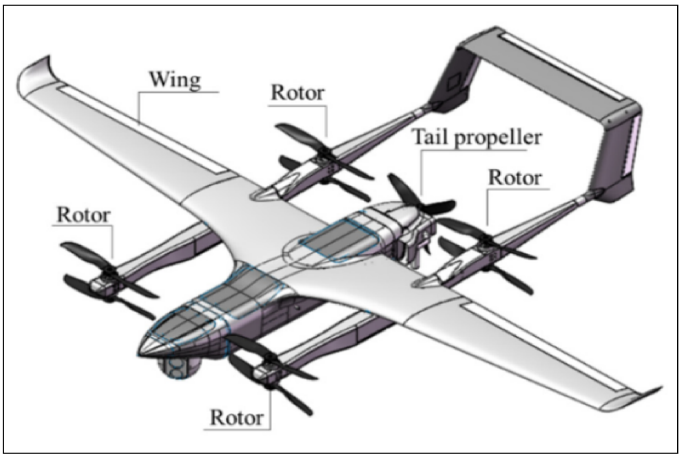

Figure 2.11: Hybrid-wing UAV design.

2.2.3 Based on Autonomy Level

Another UAV classification is based on autonomy level. Autonomy can be defined as the ability to perform a mission/task that is assigned to the UAV with minimum human intervention. Thus, the UAV autonomy level depends on the complexity of the mission/task. The classification is as below (Huang, 2004):

- **Fully autonomous controlled UAVs**: These are the UAVs that can carry out the assigned mission/task without any intervention for a human being. They are fully automated vehicles systems where all decisions are made onboard using sensory information that perceive and interact with the environment changes.
- **Semi-Autonomous UAVs:** In this type of UAVs, when the aerial vehicle is not able of making decisions, a human being can interfere to make those decisions. This can happen in cases where high level flight mission is delegated to the vehicle.
- **Tele-operated UAVs:** These UAVs are operated by human operator that uses the information gathered by sensors to directly send control signals in the aim of performing the mission/task successfully. This type of aerial vehicles is used mainly in Beyond-Line-of-Sight (BLOS)missions.
- **Remotely controlled UAVs:** The UAV is controlled manually by a human operator using a remote control from a Ground Control Station (GCS) (See Figure 2.12). Most of these UAVs are used in Line-of-Sight (LOS) missions.

2.3 UAV Applications

the role of UAVs has changed drastically in the last decades. Their applications have been extended from military to cover civilian and transport applications. To make them more suitable for these

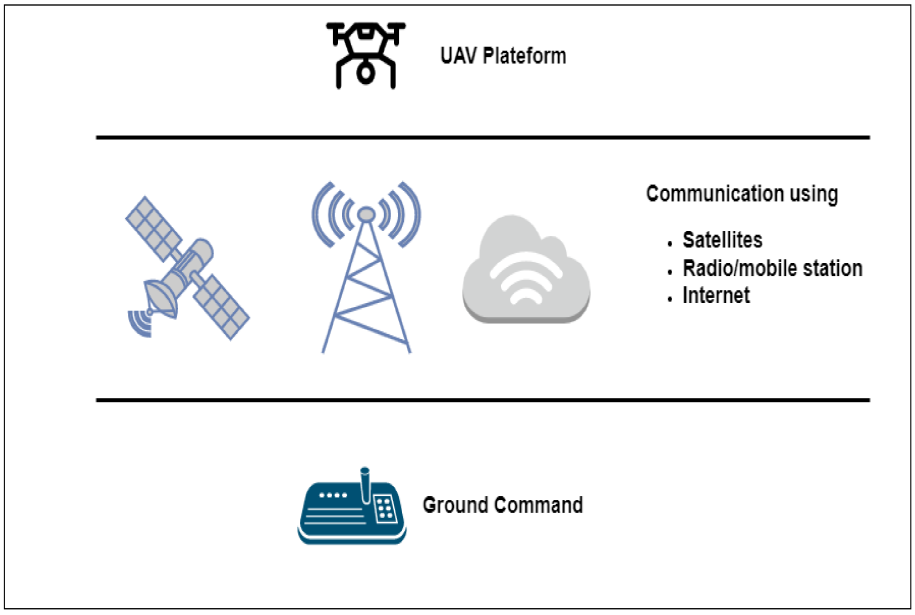

Figure 2.12: Communication means of UAVs through GCS.

applications, the UAVs' configurations have been changed in favor of smaller dimensions. Besides, there have been lot of research works to develop their autonomy and endurance. Some of these applications are discussed in this section.

2.3.1 Military Applications

UAVs, especially tactical, combat, HALE and MALE UAVs, have several applications within the military and defense area. Very advanced UAV technologies have emerged in this domain as they make it possible to perform military operations in a more effective and less risky way. These technologies are now leading to have numerous Intelligence, Surveillance and Reconnaissance (IRS) and Reconnaissance, Surveillance, and Target Acquisition (RSTA) capabilities. Other applications of military UAVs are shown in Figure 2.13.

2.3.2 Civilian Applications

UAVs can be deployed in various civilian uses thanks to their high maneuverability, low cost and high efficiency. In the last years, UAVs have supported public safety, search and rescue (SAR) operations and disaster management. UAVs play important responsibilities in rescue operations in case of natural or man-made disasters like floods, Tsunamis or terrorist attacks ... etc. They can be used to provide communications coverage in support of such operations. UAVs can be also a solution to provide medical supplies especially in the areas that are inaccessible (Tanzi et al., 2016). UAVs can be also used for construction and infrastructure inspection. To have a better visibility about the project progress, project managers fly UAVs to monitor constructions buildings, wind turbines dams ... etc. Furhermore, UAVs can be also deployed for inspecting high voltage

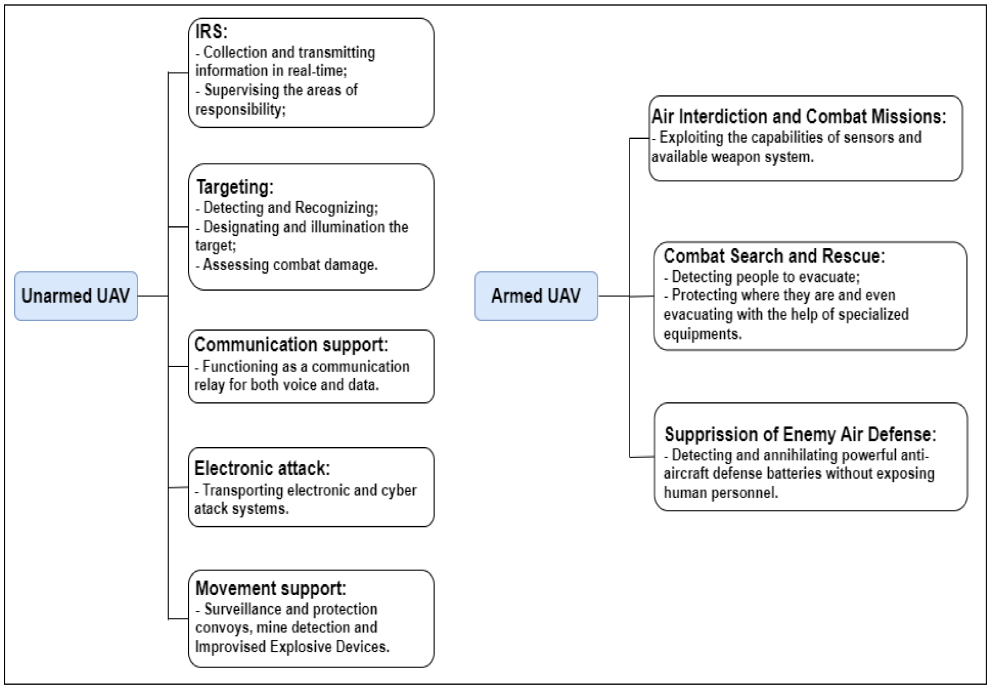

Figure 2.13: Some military applications of UAVs (Udeanu, Dobrescu, and Oltean, 2016).

power transmission lines. Another civilian application of UAVs is real-time monitoring of road traffic. They can monitor large continuous road segments more effectively compared to traditional road monitoring tools (loop detectors, video cameras ... etc.). In addition to these, other civilian applications of UAVs can be shown in Figure 2.14.

2.3.3 Transport Applications

Considering the technological development, UAVs are becoming a crucial part of the modern logistics industry. They can be used for transporting food, packages and goods. Because of traffic congestion issues, many industries have begun to employ UAVs for transportation. these aerial vehicles are starting to emerge by integrating into the current infrastructures of transportation. UAVs for transport can accelerate the delivery time significantly while reducing its cost. In healthcare service, UAVs, which are called ambulance drones (See Figure 2.15 (a)), can travel between a pick up location and a delivery location to deliver medicines, blood samples and immunizations. Considering also the increasing of e-Commerce on one hand and the demise of snail mail on the other hand, postal companies are investing in UAVs to make package delivery fast and more suitable (Bekhti et al., 2017) (See Figure 2.15 (b)). Companies and researchers are working on exploring new methods to overcome the challenges and the issues related to airspace safety, the privacy of citizens, theft, payload requirements, navigation systems ... etc. Another application of UAVs is passenger transport. Passenger UAVs or "air taxi" (See figure 2.15(c)) have shown their technical capabilities to transport passengers between or within cities (Kellermann, Biehle, and Fischer, 2020). Although

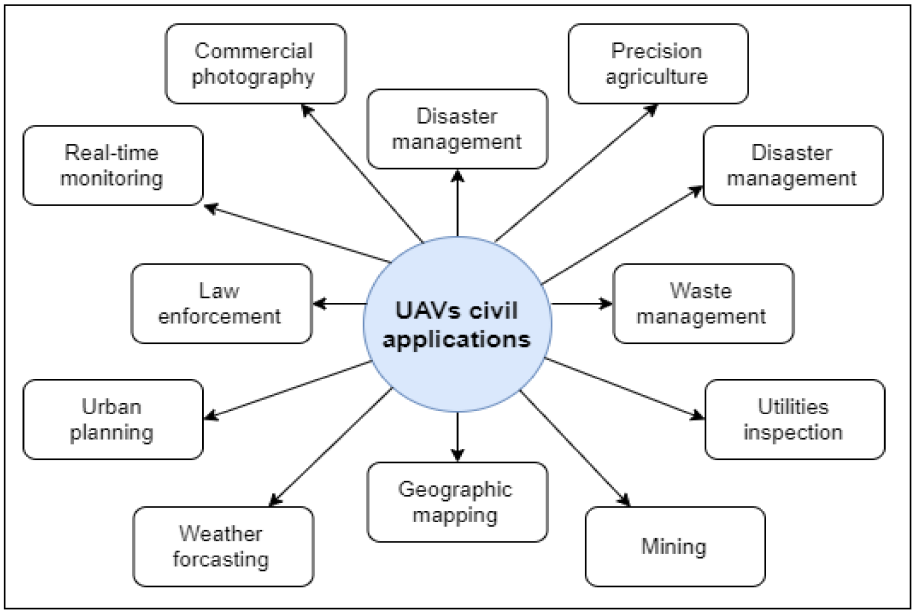

Figure 2.14: Civilian applications of UAVs (Sivakumar and TYJ, 2021).

this application is still in its infancy, this marks the start of a new era where low level airspace may be expanded to the third dimension of transportation.

2.4 Quadrotor Autonomous Vehicle Systems

A quadrotor vehicle is a rotary-wing UAV that consists of four (04) rotors located at the extremities of a cross structured frame. The flight motion of a quadrotors is controlled by the speed of its rotors; they can roll, pitch, yaw and accelerate along their common orientation. As mentioned before, quadrotors are multi-rotor vehicles having certain key characteristics like VTOL, hovering capability, slow precise movements ability, higher payload capacity and higher degree of maneuverability. Quadrotors are mechanically simple but they are inherently stable. Thus, they require feedback control algorithms to make them fly with stability. In order to design a quadrotor, two modular approaches are adopted: hardware and software.

2.4.1 Hardware Modular Approach

A simpler hardware approach can be used to build a quadrotor with a frame, a propulsion system and a Flight Control System (FCS) (See Figure 2.16.

Quadrotor Frame

The quadrotor frame can have a common configuration, either +, X, or H configuration (See Figure 2.17). In the “plus” configuration, a single rotor leads the quadrotor; however, in the “cross” and

Figure 2.15: Transport applications of UAVs: (a) ambulance drone, (b) delivery drone and (c) air taxi.

"H" configuration, two rotors lead the vehicle. The cross and the plus configurations are considered the most popular. Even though the first configuration is more stable than the second one however this latter is easier to control (Agrawal and Shrivastav, 2015). The X-copter is considered the most symmetrical while its weight is concentrated at the vehicle center of gravity. This may lead to improve its stability, whereas the vulnerability to aerodynamics disturbances is increased. Despite the simplicity of the X-frame configuration, the shortage of space for integrating other hardware components remains a big problem. The "plus" frame configuration has the same footprint as the X-frame flipped with 45°. Such configuration has an advantage of each motor being responsible for rotation movements (roll, pitch, and yaw) in one axis only. As a result, this allows applying finer control system strategies. Nevertheless, it is more likely to break because most impacts involve only solid contact with the front arm. The H-frame configuration is an antique design where the vehicle's arms are located in front of a long bus shaped carriage. Due to its hefty design and odd configurations, the research in H-copter has been recently ignored.

Propulsion System

The propulsion system comprises four motors, four propellers, four Electronic Speed Controllers (ESCs) and a source of power (batteries).

At the beginning, quadrotors were equipped with DC motors with gearboxes. Recently, they have been replaced by Brushless DC motors because of the life limit and the friction caused by the brushes. Brushless DC motors are synchronous motors having longer life, more reliable, better

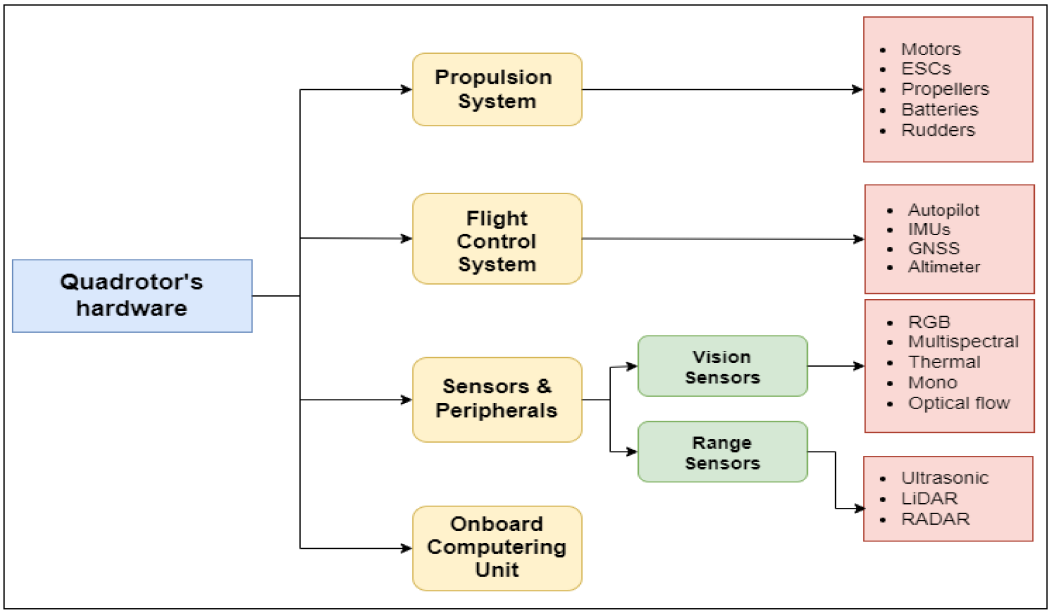

Figure 2.16: Hardware components that built up a quadrotor.

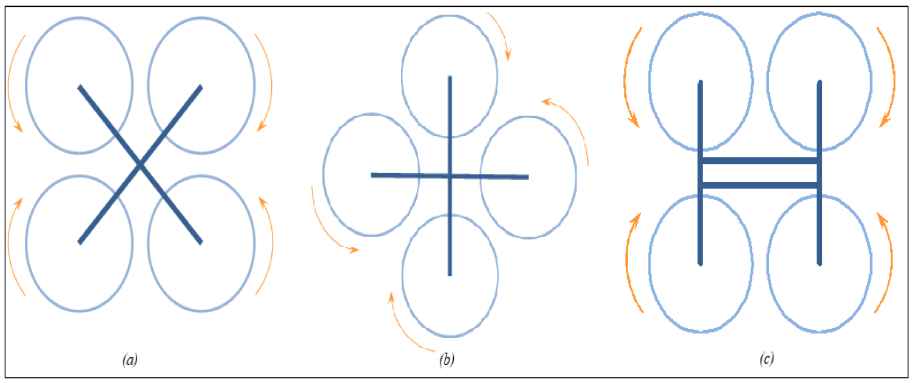

Figure 2.17: Quadrotor configurations: (a) X-quad, (b) +-quad and (c) H-quad.

efficiency and lighter than DC motors. Besides, they offer better thrust-to-weight ratios. Kv and current rating are properties that characterize motors. The Kv ratings shows how a motor transforms the power it receives into speed. The current ratings, however, show the maximum current that can be drawn by the motor.

The propellers are spun by the Brushless DC motors to create lift thrust to the quadrotor. They are characterized by their diameter, the larger the propeller, the more lift it provides, and pitch, the smaller the pitch, the more traction the propeller offers at low speeds. The propellers can be made from several materials like plastic, carbon, fiber reinforced polymer. However, the most common ones are nylon and carbon. To ensure flight, the propellers must not all turn in the same direction.

Hence, they can turn either in CW (Clockwise) direction or in CCW (Counter Clockwise) direction (See Figure 2.18).

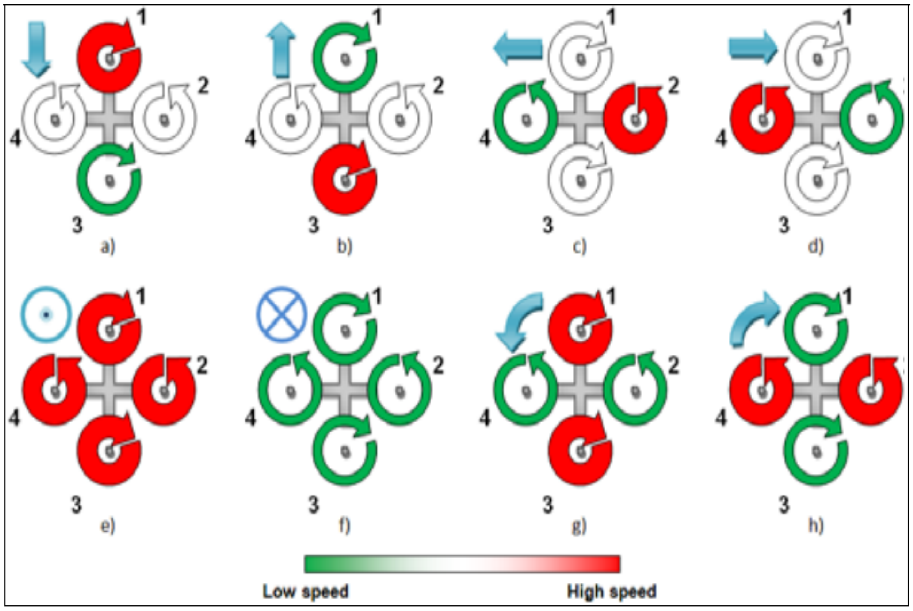

Figure 2.18: Scheme of propellers for maneuvering in "+" configuration (Lukmana and Nurhadi, 2015).

ESCs are what allow the flight controller to control the speed and direction of motors at a specific time. They must be able to handle the maximum current the motor can draw without overheating according to their current rating, and be able to supply it at the correct voltage. In design, the choice of the ESCs must be done carefully in order to guarantee that the motors draw enough current.

The batteries provide electrical power to the quadrotor's motors and other electronic components. The most used batteries are made of Lithium Polimer or LiPo due to their small weight, high energy density, longer run periods, and ability to be recharged. However, they are not completely safe because they contain pressurized hydrogen gas and tend to burn and/or explode when there is a problem.

Flight Control System (FCS)

Flight Control System or FCS is the brain of the quadrotor. It is an Integrated Circuit (IC) component, composed of a microprocessor, sensors and input/output pins, responsible of ensuring stable flights and makes remote control and autonomous flight possible by setting the right power for each motor (Valavanis and Vachtsevanos, 2015). The principle work of the FCS is simple. It obtains the quadrotor's states (position, velocity, orientation, etc) from sensors such as Inertial Measurement Units (IMUs), barometers/altimeters, and Global Navigation Satellite Systems (GNSS), converts radio command signals into actuator pulses and maintains desired states to achieve proper flight mission performance. FCS contains also a computing module, like a microcontroller, coupled with

the autopilot to implement its logic for effective flight control mission. FCSs come with build-in software and can be categorized into two classes: closed-source and open-source software. Closed-source software can not be modified. However, open-source FCSs are developed such that they allow customers to modify the software based on their own needs. In this way, universities and industries can cooperate on solving issues related to aerial vehicles such as their reliability, functionality, endurance and fault tolerance. These issues are mainly associated with the FCSs hardware and software. Examples of FCSs with open-source software are depicted in Figure 2.19.

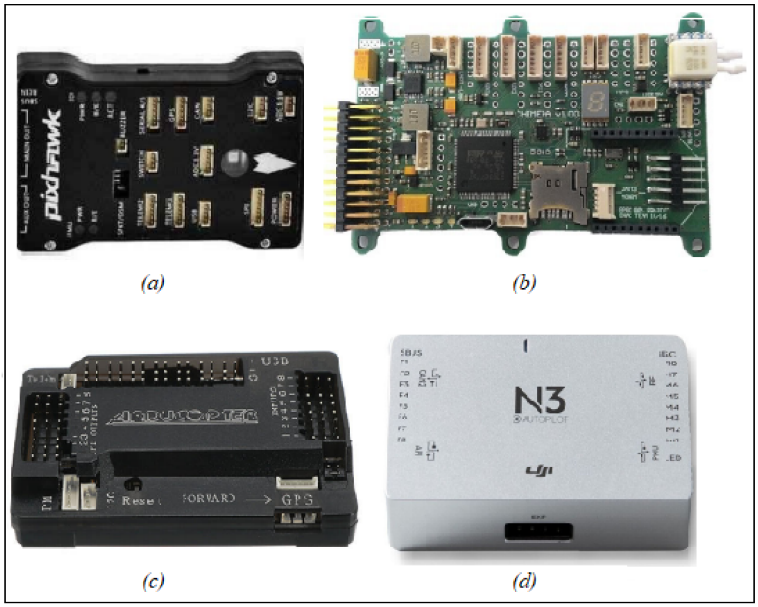

Figure 2.19: FCSs examples: (a) Pixhawk PX4 FMUv5, (b) Chemira Autopilot for Paparazzi, (c) Ardupilot Mega (APM) and (d) N3 DJI.

Sensors for Navigation and Localization

Sensors are crucial components for both the stability and the autonomy of the quadrotor. These components provide information about the location of the vehicle for the user (drone pilot) or the autopilot. The sensors of the quadrotor can be divided into three categories:

- Proprioceptive sensors: they provide information on the current state of the quadrotor, i.e., on its state at a given moment. These sensors measure the position, speed or acceleration of the machine relative to a reference state.
- Exteroceptive sensors: they provide information on the environment in which the drone is flying, such as mapping, temperature, etc.
- Exproprioceptive: sensors: combination of proprioceptive and exteroceptive.

The design of more efficient sensors ensures perfection and autonomy for quadrotor. These sensors are: IMUs, accelerometers, magnetometers, gyroscope, GNSS, barometer and imagers.

An IMU is an electronic navigation system that provides the angular orientation of a body with respect to an inertial reference. It typically consists of three accelerometers, three magnetometers and a gyroscope for obtaining an accurate attitude of the body. To execute any flight maneuver, the vehicle's orientation read by the IMU should be provided to the flight controller. Micro Electro-Mechanical Systems or MEMS technologies are the most used recently in quadrotors thanks to their cost, simplicity, and dimensions. MEMS have made it possible to have accelerometers, gyroscopes and magnetometers integrated into an electronic circuit weighing around ten grams.

Accelerometers provide the linear acceleration of the frame along the three body-frame orthogonal axes. The principle of accelerometers is based on the deformation or displacement of a body during acceleration. The advantage of an accelerometer is its great ease in revealing a several data such as acceleration, speed, displacement, force, etc. Nevertheless, obtaining the body displacement necessitates a double integration of the acceleration. Consequently, this leads to problems of precision. In this case, fusing with other sensors such as gyrometers makes it possible to adjust the measurements.

On one hand, magnetometers are essentially magnetic compasses that measure the direction and/or the intensity of a magnetic field, and in particular the direction of the earth's magnetic field. Sensitivity to external magnetic disturbances is considered as the major issue of such sensor. On the other hand, Gyroscope is a device that provides angular rates of a frame with respect to the body reference frame of the quadrotor. The attitude of this latter can be determined by integrating the angular rate provided by the gyroscope in a small period of time.

The GNSS is a navigation system that relies on satellites to provide the geo-spatial positioning of the aerial vehicle. It consists of a constellation of satellites that orbit the Earth. Signals, which are continuously transmitted from these satellites, are utilized by users to provide their three-dimensional position. The GNSS consists mainly of three types of satellites technologies: Global Positioning System or GPS, Glonass and Galileo. The GPS, which was developed by the United States Department of Defense (DoD) for military purposes, is a satellite positioning and navigation system containing 24 satellites spread over six orbits (four satellites per orbit) revolving around the globe (2 turns in 24 hours) and located at an altitude of 20,200 kilometers with an inclination of 55° relative to the equator or Medium Earth Orbit (MEO) (Subirana, Hernandez-Pajares, and Miguel Juan Zornoza, 2013). The constellation of the Nominal Glonass is made of 24 MEO satellites which are located in three different orbital planes which endow 8 satellites equally spaced. They are located at an altitude of 19,100 kilometers with an inclination of 64.8°. Last but not least, the constellation of Galileo contains 27 operational and 3 spare MEO satellites at an altitude of 23,222 kilometers with an inclination of 56°. Above all, due to atmospheric disturbances, the signal propagations of these constellations are altered. This may result in measurement errors. These latter can be minimized by fusing them with sensory data from imagers (lasers or cameras). Lasers and cameras represent great sources of information that can be exploited for quadrotor navigation and localization. Lasers can be ultrasonic, lidar, radar, etc. Cameras can be RGB, multispectral, thermal, etc.

Barometers are instruments that provide atmospheric pressure. They are used as sensors to determine the altitude of the quadrotor with respect to a reference level. The main disadvantage of such sensors is their sensitivity to atmospheric conditions alteration such as wind.

2.4.2 Software Modular Approach

The quadrotor software module uses the computing unit to run several processes in parallel. A middleware messaging system provides the ability to interchange messages between the different processes on the same computing module or with other computing modules on the same network. The software implemented on the computing module depends on the application to which the quadrotor is delegated (Elmokadem and Savkin, 2021). For instance n remote sensing, the quadrotor software is dedicated to both ensure safe mobility in the environment where it navigates and collect imagery data that can be processed after the mission is ended.

The module, which is related to the quadrotor mobility, is one of the fundamental modules. The aim of this module is to autonomously generate safe-collision navigation plans for the aerial vehicle. In most cases, the autonomous navigation module has four (04) submodules (See Figure 2.20): perception, localization and mapping, motion planning and obstacle avoidance and control (Laghmara et al., 2019). However in other cases, one or more submodules can be discarded. For example, the motion planning module can be coupled with the control module without the need to use the localization and mapping module.

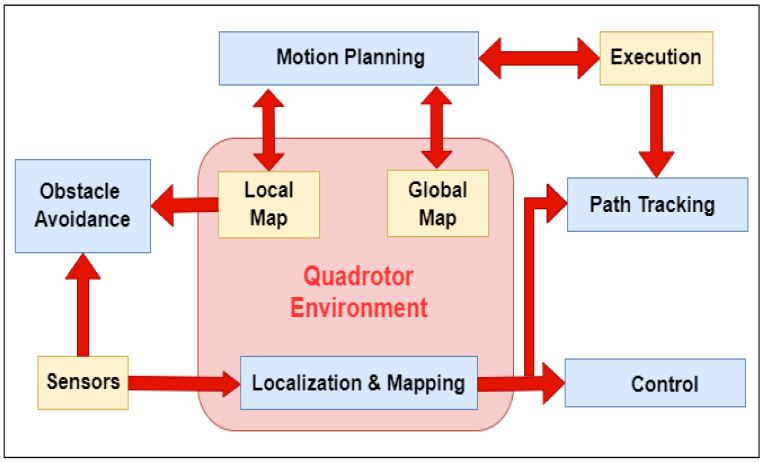

Figure 2.20: Quadrotor software module for autonomous navigation.

2.5 Quadrotor Autonomous Navigation: Definition

Most quadrotors require higher performance autonomous navigation techniques for the purpose of completing the tasks they are assigned effectively and efficiently. Quadrotor autonomous navigation can be defined as the process that the vehicles safely and quickly plans and executes its path in order to achieve the target location. Autonomous navigation is the core element for ensuring the vehicle collision-free motion. During this process, information like current position and velocity, heading direction, starting and target location are provided for the quadrotor system aiming to complete the scheduled mission successfully.

In general, quadrotor autonomous navigation techniques include three (03) key elements (See Figure 4.4.2): perception, planning and control (Alanezi et al., 2022). These elements are mainly

used to improve the vehicles autonomy until reaching the final destination. Perception uses the information gathered by the different sensors for determining the current state of the quadrotor (e.g. position and orientation) as well as the representation of the surrounding environment (e.g. obstacles). Several algorithms have been developed for perception. These algorithms comprise algorithms for object detection, localization, object tracking and Simultaneous Localization and Mapping (SLAM). Afterwards, the information gathered by perception are used in planning for making motion decisions. Here, two types of planning are differentiated: path planning (global/local) and trajectory planning/generation. The former concept refers to the process of generating a geometric collision-free between a starting and target positions with no timing law. However in the latter concept, the collision-free path associates with a timing law to provide a trajectory which has higher derivative information (e.g. velocity, acceleration, jerk, etc.) (Elmokadem and Savkin, 2021). The purpose of the control is to ensure that the vehicle follows the generated path/trajectory in the planning step. Control algorithms have been implemented for path/trajectory tracking, obstacle avoidance and stability (Alanezi et al., 2022).

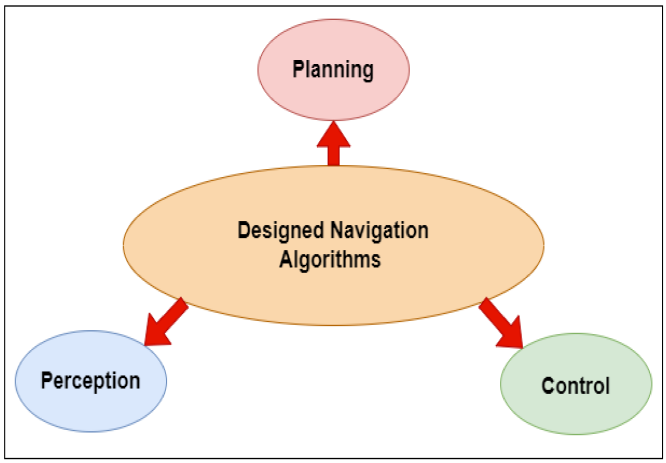

Figure 2.21: Quadrotor autonomous navigation elements.

Depending on the complexity of the autonomous navigation problem, different structures have been used with quadrotors. Structures can be designed by either coupling or decoupling the planning and the control elements (See Figure 4.4.3). Coupling structures combine planning and control resulting in complicated reactive control elements. However, decoupling structures are the most common thanks to their simple designs (Elmokadem and Savkin, 2021).

2.6 Quadrotor Autonomous Navigation: Applications

Quadrotor autonomous navigation has been applied in wide applications and is being recently envisioned for larger applications with the design technology advancement. Many of the current applications still do not use fully autonomous design systems because of two main reasons: (1) the different operational risks that these systems may experience, and (2) the immaturity of research associated with these applications. Precision Agriculture (PA), Search and Rescue (SAR), oil and gas exploration, wildlife monitoring and construction are the most relevant applications where

quadrotors are required to attract more interest in developing aerial vehicle system technologies with high level navigation autonomy.

Quadrotor Autonomous Navigation in PA

Agriculture is one of the most important sectors and contributes with a huge role in the world economic status. It is the most promising and challenging sector since it is dependent on weather conditions, soil conditions and water irrigation quantity and quality. On the other hand, the world population is increasing day by day and projected to reach 9.4 billion people in 2050 (Boretti and Rosa, 2019). Hence, food production in sustainable manner must increase to satisfy the need of the large population. This can be done by implementing modern technologies in agricultural production. These technologies will contribute in solving problems related to agriculture and establish proper farming processes by introducing Smart Farming or Precision Agriculture (PA).

PA, dubbed "the farming of the century" is gaining more attraction in today's modern technology-driven world. It refers to that type of computerized farming where advanced technologies are used to monitor and optimize agricultural activities for the aim of improving farm productivity in terms of quantity and quality. In order to do so, PA takes the advantage of the current technology and concepts and uses them for regulating temporal and spatial variations (soil, yield, crop, field and management) in all agricultural output elements (Tsouros, Bibi, and Sarigiannidis, 2019).

Quadrotor systems are one of the technologies that have taken PA one step further. Quadrotors offer great prospects for acquiring in-field information in an easy, fast and cost-effective manner compared to other methods like satellites and manned aircrafts (Tsouros, Bibi, and Sarigiannidis, 2019). Quadrotor vehicles are able to fly at low altitudes in agricultural fields for sake of collecting images of the crops with an ultra-high spatial resolution (few centimeters) such that the performance of the monitoring systems is improved. Besides, quadrotors are more efficient that Ground Vehicles (GVs) systems since they can cover larger field areas in short period of time and in non-destructive manner.

Autonomous navigation quadrotor systems are very commonly used in remote sensing (RS) applications under PA. Such systems are equipped with different types of sensors and can be used for identifying which zones of the fields need different management. In this way, the farmers are able to react on time and with the required quantity of products in any problem detected. Autonomous navigation quadrotor systems can be used in different applications under PA. Among these applications, weed mapping is the most popular. Weeds are undesirable plants that grow in agricultural crops causing many problems such as losses in crop yields and harvesting. In order to control these weeds, herbicides are sprayed over the entire agricultural field, even in areas with free weeds. Such activity leads to overuse quantities of herbicides resulting in evolution of herbicide-resistant weeds, cost as well as pollution problems. These problems are solved in PA using Site-Specific Weed management (SSWM) approach. Rather than spraying the whole field, SSWM applies the herbicides only on spots where undesirable weeds present with higher stress (Hunter III et al., 2020). In order to do so, an accurate weed map should be generated a priori. Autonomous navigation quadrotor systems can be employed to collect aerial images of the whole field. These images can be used to generate a precise weed cover map showing the spots where the herbicides are required the most. Under the same concept, crop spraying is also an application of autonomous quadrotor systems. These later are very useful thanks to their lower operator exposure and their enhanced ability to apply products in timely resolved way.

Autonomous navigation quadrotor systems represent a great solution for monitoring and assessing the health of the crops. They can monitor the crops constantly in time and in non-destructive way to detect diseases and avoid economic losses as a consequence of reduced yield quantity and quality. This can be achieved by data processing techniques that use crop imaging information collected by the quadrotors to spot changes in the biomass and health of the plant (Patel et al., 2013). Periodic monitoring flight missions can be scheduled to detect the diseases especially in their early stages allowing farmers to intervene by providing special treatment of infection using targeted spraying.

Under the same context of monitoring, autonomous navigation quadrotor systems are commonly used to monitor the growth of the vegetation and estimate the yield. These systems are means of collecting information and visualization of crops for mainly two reasons: (1) recording the variability observed on the field, and (2) monitoring the crop growth. At this stage, crop imaging is also used to analyze the biomass and nitrogen status information in order to determine the management actions required for the crop such as the use of fertilizers (e.g., use of nutrients) (Duggal et al., 2016). In addition, the crop imaging analysis can be performed using 3D digital map models of the crop by measuring various parameters like the crop height, the Leaf Area Index (LAI), and the vegetation greenness index or the Normalized Difference Vegetation Index (NDVI).

Another important application of the autonomous navigation quadrotor systems under PA is crop irrigation management. The extensive consumption of water for crop irrigation (about 70% of water consumed worldwide according to (Saccon, 2018)) urges to highlight the need for precision irrigation methods. The purpose of such methods is to improve the water use efficiency by applying the resource in the right time, places and quantities. This can be done using autonomous quadrotor systems that can incorporate appropriate sensors to divide the field into different irrigation zones and identify which zones of a crop that require more water. Precision irrigation methods help farmers to save time and water resources and increase the crop quality and productivity. In addition to the aforementioned PA applications, autonomous navigation quadrotor systems have been also used for soil analysis, mammal detection and phenotyping (Tsouros, Bibi, and Sarigiannidis, 2019).

2.7 Conclusion

In this chapter, we briefly presented the main UAV system architectures and their classifications based on size and weight, control configuration and the level of autonomy. Afterwards, we grouped the different applications of such vehicles into mainly three categories: military, civilian and transport. Then, we described the quadrotor autonomous navigation system focusing on its applications under the PA context. In the next chapter, we will divide quadrotor autonomous navigation system into multi-phase processes where each phase will be described in details.

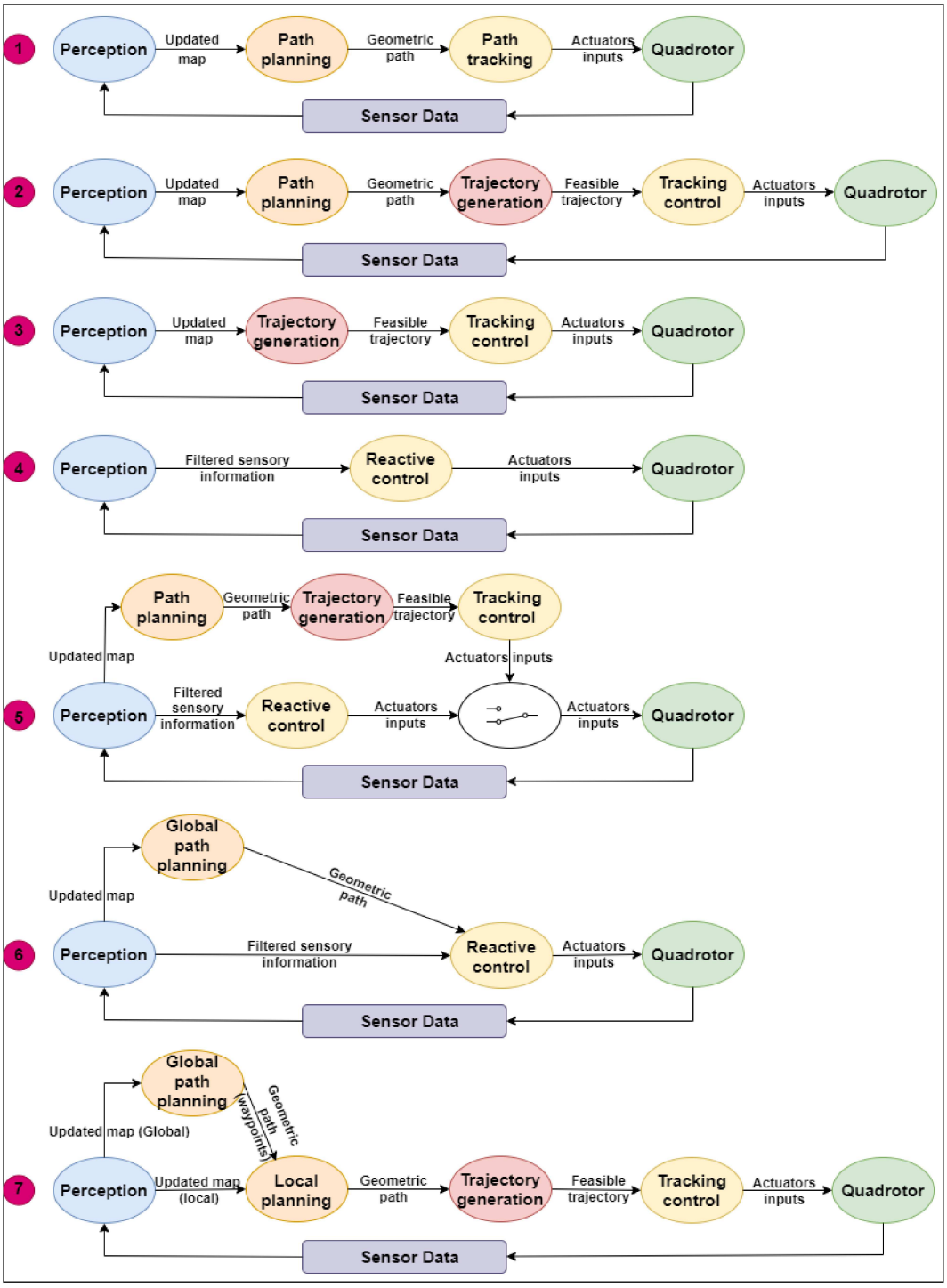

Figure 2.22: Quadrotor autonomous navigation structures.

Chapter 3

Quadrotors Path Planning

3.1 Introduction

A crucial part of autonomous robotic system is to ensure that the robot can move while avoiding collisions with the obstacles present in its environment. Generally, this problem can be addressed by path planning. Path planning is an essential task from the control engineering perspectives. It is considered as a hot spot in the field of robot autonomous navigation research. When dealing with UAVs, the path planning research is essential since it is correlated with the autonomy of such vehicles, the built-in components required, guidance, endurance, and functionality. Since UAVs suffer from the limited payload problems, the insertion of many batteries and power banks is not as option. Hence, path planning becomes the primary issue to solve UAVs time-limited problems for performing the required tasks. This chapter aims to introduce the concept of path planning, its types and paradigms used for quadrotor platforms.

3.2 Robot Path Planning

3.2.1 Path Planning: Definition

Robot path planning (RPP) related problems have been extensively studied, generally focusing on manipulators and ground mobile vehicles. From a technical point of view, path planning is a process of deliberatively producing a path or a set of way-points for a robot from a starting location to a goal location while taking into account the environmental and physical constraints of the robot in order to achieve a collision free path (Lin and Saripalli, 2017). In more technical terms, it is defined by (Dudek and Jenkin, 2010) as "determining a path in configuration space between the initial configuration of the robot and a final configuration such that the robot does not collide with obstacles and the planned motion is consistent with the kinematic constraints of the vehicle". A *configuration* in the definition refers to the position and the orientation of the robot while a *configuration space* is the set of all possible configurations (See Section 3.2.2 for more detail).

RPP is associated with a several terms that can be used interchangeably. Motion planning, which frequently associated with manipulators, is widely used to plan paths that are feasible and safe. Trajectory planning is which deals with the robot's dynamics, to plan the next move. Obstacle/collision avoidance is a part of motion planning; it uses the robot's current sensed information

for moving away from immediate obstacles while ensuring stability and safety (Giesbrecht, 2004).

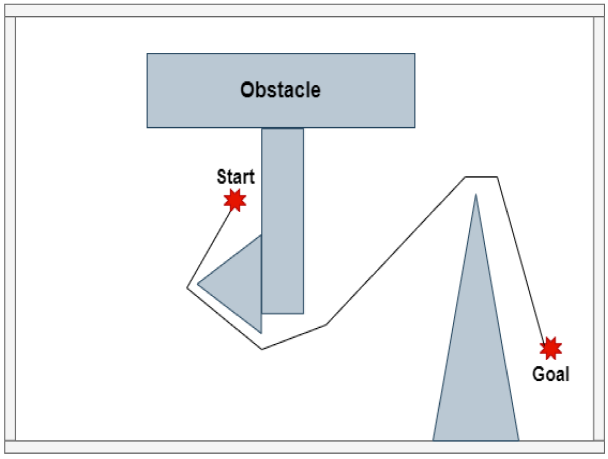

Figure 3.1: Robot path planning.

Path planning paradigms takes into consideration four main criteria (Teleweck and Chandrasekaran, 2019). Optimization is the criterion which guarantees that the selected path is the best among all of the possible solutions in terms of distance, time computation, cost, and so on. In case of UAVs where the autonomy problem is recurrent, optimization plays a primordial role in minimizing the flight distances and time, hence, increasing the vehicles' endurance and decreasing revelation to possible risks. Besides, optimizing the computational time is required especially in real-time applications. Completeness criterion ensures the path planning algorithm finding all possible solutions for the path at hand. Accuracy/precision is another criterion that can be taken into account for the purpose of driving all states of a robot from an initial configuration to a final one. Execution time is considered as an important path planning criterion since it ensures the best-case settings to handle the designated problem. It refers to the time taken by the robot to complete the entire path (Atiyah, Adzhar, and Jaini, 2021). There often a trade-off among these criteria. For instance, on one hand, path optimality has to be discarded in order to reduce the computational time. On the other hand, higher computational time is required for an optimal path to be generated. Thus, the consideration of these criteria has to be done before the process of path planning takes place.

3.2.2 Configuration Space

In path planning, the robot and the environment (referred to as workspace) through which it travels, are represented in some manner so that paths can be planned in a search space. This latter considers all possible states that can exist. However for planning motions in case of multiple Degrees of Freedom (DOFs), the notion of configuration space is used. The configuration space or C-space refers to the set of all possible transformations applied to the robot. C-space is an important technique that offers solutions for motion planning problems varying in geometry and kinematics. The configuration space consists of reducing the robot's size to a point and enlarging the size of the obstacles according to the robot's dimensions (Raja and Pugazhenthi, 2012).

The term "configuration space" is introduced in (LaValle, 2006) as follows:

For an n DOFs robot, the configuration space of a robot, denoted C, is an n-dimensional manifold that contains the set of all transformations. In particular, the C-space for a 2D rigid body is called $SE(2)$, for 3D rigid body $SE(3)$, and for multiple independent rigid bodies is the Cartesian product of the configuration spaces of each of them. The C-space contains two regions: an obstacle region and a free region as shown in Figure 3.2. The obstacle region, $C_{obs} \leq C$, is all configurations that the robot collides with obstacles or each other. All the leftover configurations are denoted by free region, $C_{free} = C/\ C_{obs}$. Using the definition above, path planning is defined as finding a

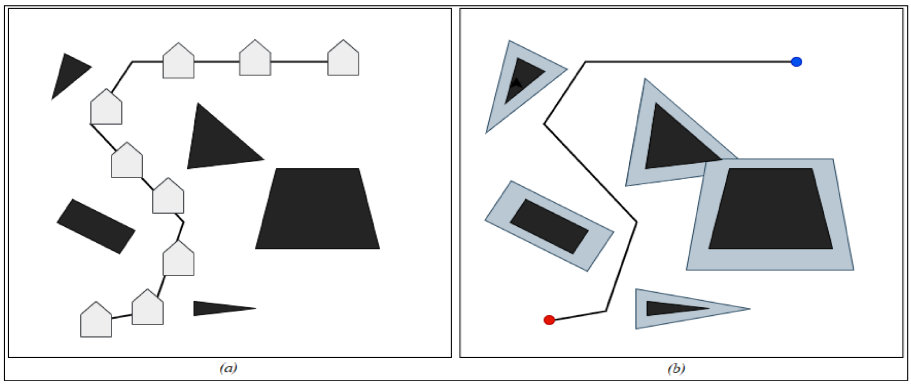

Figure 3.2: (a)The workspace and (b) the configuration space of a robot.

continuous path $\tau : [0, 1] \rightarrow C_{free}$, such that $\tau(0) = q_I$ and $\tau(1) = q_G$ where $q_I, q_G \in C_{free}$ are the initial and target configuration, respectively.

3.2.3 Types of Path Planning

The field of RPP can be categorized mainly in two major aspects: the point-to-point path planning and the coverage path planning. In recent years, researchers and experts have concentrated their research on the first aspect; however, the study of the second one has relatively reduced.

Point-to-Point Path Planning

The objective of robot point-to-point path planning approach consists of finding a collision-free path from a starting configuration to a destination configuration while optimizing a certain parameter like time, distance or energy. This approach is in a sense analogous to a single particle in a potential field with attraction and repulsion points while the particle being a robot, the attraction point being a destination configuration and the repulsion point being obstacles (Chakraborty et al., 2022).

Coverage Path Planning

The coverage path planning (CPP) or the complete CPP (CCPP) refers to as the task of traversing the robot's whole environment while taking the motion restrictions and avoiding the collision with the obstacles present in that environment (Cabreira, Brisolara, and Paulo R, 2019). The CPP often arise in robotics applications. It is considered as an integral task to many robotics systems like

vacuum cleaners, painter robots, lawn mowers, autonomous harvesters... etc. Coverage planning necessitates assumptions on the abilities of the robot to sense its environment, get to know its position in that environment (map knowledge and positioning abilities) and plan its route, efficiently. The study of the CPP started in the eighties', Cao et al. in (Cao, Huang, and Hall, 1988) delineated the conditions that a robot must take into account to perform a coverage operation:

- The robot must cover completely all points in its environment;
- The robot must fill the environment without overlapping routes;
- Continuous and sequential coverage process without repetition of the routes is entailed;
- The robot must avoid obstacles (if present);
- Motion routes, which are simple (straight lines or circles), must be used (for sake of simplicity);
- Optimal path is obtained under predefined conditions.

However, taking into considerations all the above conditions in a complex environment is unfeasible in most situations. Thus, priority consideration is required.

The CPP problem can be related to other problems known in literature. The lawn mower problem is one of them. The lawnmower problem, which states to find a path to cut all the grass of a pre-defined ROI is proven to be NP-hard (Non-deterministic Polynomial-time) (Dudek and Jenkin, 2010). Another problem is the "piano mover's problem" which is based on finding a collision free path from a starting configuration to a target configuration while avoiding obstacles. The problem is also known to be NP-hard. Consequently, even the CPP is considered as an NP-hard problem.

3.2.4 Path Planning Paradigms

The existing robot path planning techniques in general can be categorized into deliberative (offline global planning), sensor-based (online local planning) and hybrid as shown in Figure 3.3. In the deliberative methods, the information of the environment, presented as a map, is known to the robot prior the planning process. As the information includes global data, the process is slow; however it can generate optimal paths if one exists. Sensor-based methods rely on the acquired information from the current robot's surrounding environment (i.e., a local map) using sensory observations to plan collision-free paths in real time. In this way, the planned paths are locally optimal only. However, they can be trapped in local minima. Sensor-based methods provide great solutions for navigation in partially-known, unknown and dynamic environments thanks to their computational performance (Hoy, Matveev, and Savkin, 2015). In most practical cases, optimality is sacrificed for speed of computation when dealing with fast and limited computing power robots like UAVs. Hybrid methods combine the advantages of both deliberative and sensor-based methods for planning advanced behaviors (Elmokadem and Savkin, 2021). The robot uses the global planning for guidance and the local planning for handling unknown and dynamic obstacles, simultaneously.

3.2.5 Map-Based vs. Mapless Path Planning Methods

Path planning paradigms, by their turns, can be divided into map-based and maples methods depending on the environment information accessibility (See Figure 3.4) (Bonin-Font, Ortiz, and Oliver, 2008). The former methods depend on local (or global) environment map representation to

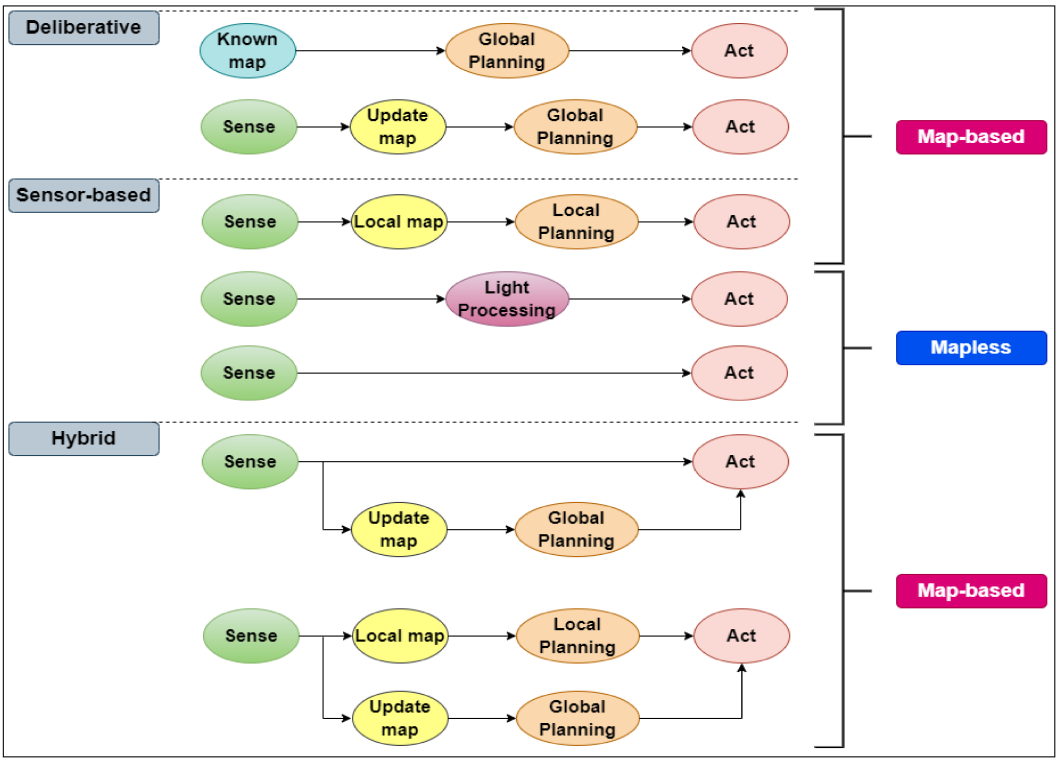

Figure 3.3: Different path planning paradigms.

plan safe and feasible paths using deliberative and/or sensor-based planning algorithms. Map-based approaches have a particular reliance on the size and the complexity of the environment map. Consequently, problems related to computational complexity, planning time and memory requirements are inevitable. On the other hand, the latter methods, known also as reactive methods, do not depend on local (or global) maps but on sensory observations for making motion decisions. Mapless methods when coupled with a control provide great solutions for obstacle avoidance problems in terms of computational complexity. However, non-optimal path, local minima and limited Field of View (FOV) are the most challenging issues that face such methods.

3.3 Map Representation for Path Planning

Robots must have a representation of their environment, a mechanism for choosing targets and a method for effective navigation to a target to operate autonomously. Combined with efficient localization abilities, an explicit environment representation guarantees the robot to navigate effectively. Maps are the most natural environment representations. They may contain other information, including reflectance properties of objects, unsafe regions or difficult to traverse, and information gained from prior experiences. Maps are required for at least three different tasks classes; they are used for (Dudek and Jenkin, 2010):

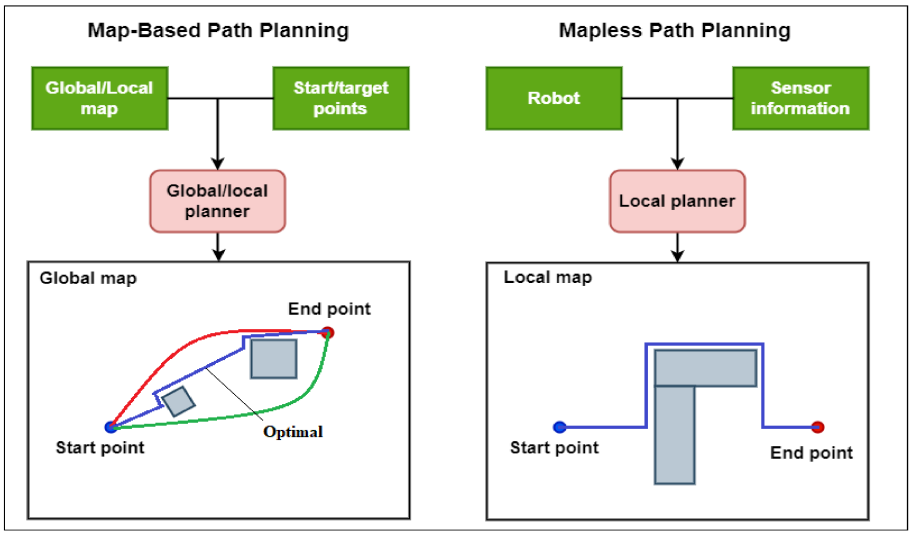

Figure 3.4: Map-based vs. mapless path planning.

1. Establishing what parts of the environment are free for traversing. This is a requirement to present and manipulate that part of the environment that is free of obstacles. This region is known as free space;

2. Recognizing regions or locations in the environment;

3. Recognizing specific objects within the environment.

The problem of map representation is a dual of the problem of representing the robot's possible position or positions. Decisions taken regarding the representation of the environment can affect the choices available for robot position representation. Hence, the position representation fidelity is correlated with the map fidelity. Choosing a particular map representation should be done by taking in consideration three fundamental relationships (Siegwart, Nourbakhsh, and Scaramuzza, 2011):

1. The map precision must match with the precision with which the robot requires to achieve its target;

2. The map precision and the features type represented must match with the precision and data types returned by the robot's sensors;

3. The map representation complexity affects directly the computational complexity of analysis about mapping, localization, and navigation.

There are mainly three ways to represent the robot's map: continuous, discrete and hybrid representations. A continuous map representation can be used as an exact decomposition of the robot environment. The main advantage of continuous maps is that the features positions can be reconstructed precisely in a continuous space (See Figure 3.5(a)). However, augmenting the dimensionality of such representation may lead to a computational explosion. Discrete representation can

be either exact decomposition or fixed decomposition. Both decompositions deal with environments which are occupied by polygonal obstacles. Exact decomposition tessellates the space obstacle-free region into areas of free space. In this way, the representation becomes compact since each area can be stored as a single node (See Figure 3.6(a)). Fixed decomposition is used to tessellate the continuous real environment and convert into a discrete map approximation. Such transformation is demonstrated in Figure 3.6(b) which shows what happen to obstacle-filled and obstacle-free areas during this conversion. Hybrid representation is considered as a combination of the both representations (continuous and discrete). The environment is taken discrete but the motion planning algorithm is considered as continuous environment (See Figure 3.5(b). This strategy is used to avoid the difficulties of software implementation of continuous environment (Siegwart, Nourbakhsh, and Scaramuzza, 2011). (Siegwart, Nourbakhsh, and Scaramuzza, 2011).

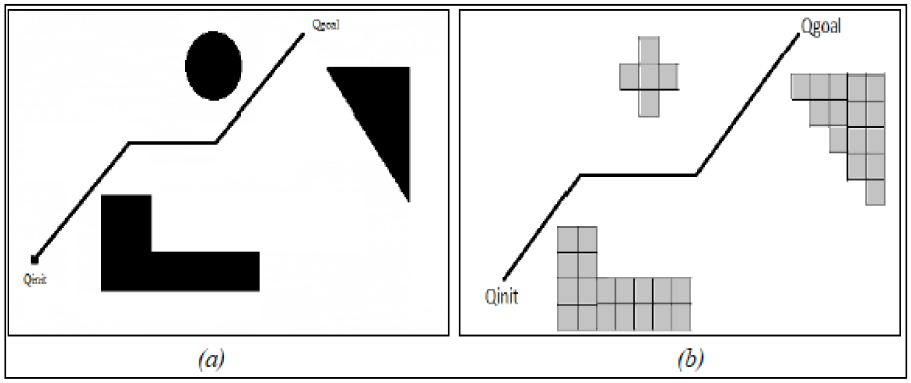

Figure 3.5: Continuous map vs. hybrid map.

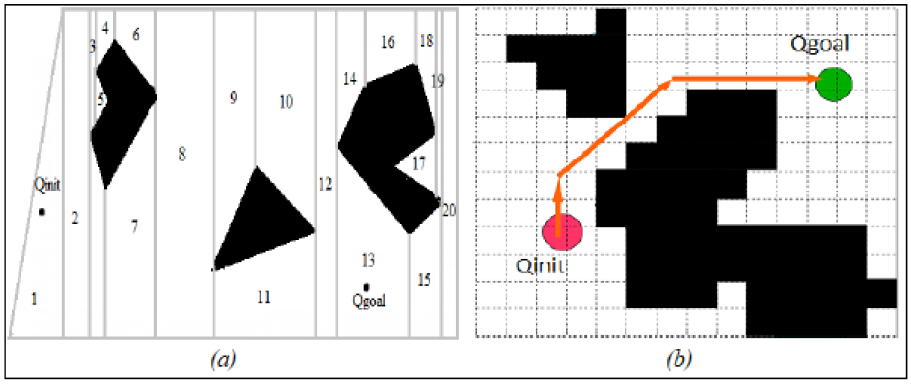

Figure 3.6: Exact cell decomposition vs. fixed cell decomposition.

3.4 Robotic Mapping

In last decades, robotic mapping has been extensively studied in robotics and Artificial Intelligence (AI). The problem of robotic mapping deals with constructing spatial models of physical world through robots. Such problem is generally is challenging, especially for complex environments, and regarded as one of the most major problem for implementing efficient path planning algorithms. Robotic mapping can be achieved using sensors that enable robots to perceive the outside world. These sensors include cameras, range finders using sonar, laser, and infrared technology, radar, tactile sensors, compasses, and GPS (Thrun et al., 2002).

All state-of-the-art robotic mapping algorithms are characterized by their probabilistic feature. They use probabilistic models of both the robot and its environment for building maps from sensor measurements. Probabilistic techniques are employed mainly because of two reasons: (1) robot mapping uncertainty and (2) sensory noise. They solve the robotic mapping problem by formulating distinctly different noise sources models and their effects on measurements. The basic principle of most of these techniques is Bayes rule (Thrun et al., 2002):

$$p(x|d) = \eta p(d|x)p(x) \tag{3.4.1}$$

Where x is the map, d is the measurement data (say range sensors, odometry). Bayes rules solve the mapping problem by multiplying $p(d|x)$, the probability of observing the measurement d given the map x, and $p(x)$, the prior. η is a normalizer that ensures $p(x|d)$ is a valid probability distribution.

In robotic mapping, Bayes filters, which extend Bayes rule to deal with temporal estimation problems, are the most dominating schemes. These filters recursively compute sequence posterior probability distributions over quantities known as states and denoted by x_t at time t. Let us define two different types of components: sensor observation at time t, denoted by z_t (e.g., camera image), and control signal asserted in the time interval $[t-1, t)$, denoted by u_t (e.g., motion commands). The generic Bayes filter calculates a posterior probability over the state x_t via the following recursive equation:

$$p(x_t|z_t, u_t) = \eta p(z_t|x_t) \int p(x_t|u_t, x_{t-1})p(x_{t-1}|z_{t-1}, u_{t-1})dx_{t-1} \tag{3.4.2}$$

3.4.1 Mapping without Localization

Mapping without localization algorithms, which were introduced by (Elfes, 1989) in late 80's, refer to a family of robotic algorithms that address the problem of constructing abstract probabilistic map representation of an environment with the assumption that the pose of the robot is already known. The basic idea of mapping without localization is to represent a map of the environment as an evenly spaced grid of binary random variables each referring the occupancy of an obstacle at that location. This may result in maps known as occupancy grid maps (OGMs). Occupancy maps use sensor observations for representing the environment as a block of cells, each one either occupied, so that the robot cannot pass through it, or free, so that the robot can traverse it. The sensor readings report the status of a set of grid cells that can be verified without reference to the rest of the map. Initially, all grid cells are assigned a value of 0.5 which means that they are unvisited yet (See Figure 3.7). The readings of the sensors are compared to the map changing the probability that observed cells are occupied (Li and Ruichek, 2014).

Occupancy grid mapping algorithms uses Bayes filtering to compute approximate posterior estimates for the binary random variables. The state x and the observation z are the only required

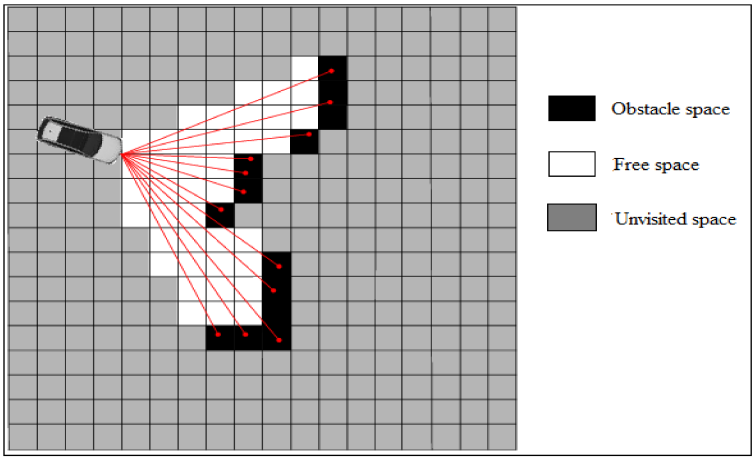

Figure 3.7: Occupancy-based grid map (Nuss et al., 2018).

components to solve this filtering problem. There is no need for the control signals since the position of the robot is initially known. Given the sensor data $z_{1:t}$, the states $x_{1:t}$ and assuming that the map does not change with time, a model of the map m can be estimated by filtering each map cell m_i independently assuming that they are in fact independent. Hence, the map posterior is the products of the maps marginal probabilities (Thrun, 2003):

$$p(m|z_{1:t}, x_{1:t}) = \Pi_i p(m_i|z_{1:t}, x_{1:t}) \tag{3.4.3}$$

Occupancy grid maps cannot be accurate, but by selecting a small enough cell size they can offer all the necessary information. Besides, they offer a uniform framework for fusing data from several sensors. However, more accurate probabilistic models of sensors are required. Occupancy maps approach takes into consideration the assumption that the position of the robot is known. This may lead to an accumulation of position errors over time while mapping is being performed. An efficient alternative is to perform both mapping and localization in parallel (Dudek and Jenkin, 2010).

3.4.2 Simultaneous Localization and Mapping

Navigation in unknown environments requires both mapping and localization to be performed simultaneously. This is known as Simultaneous Localization and Mapping (SLAM). In SLAM, both the position trajectory of the robot and its location of landmarks in the environment are estimated online without any prior information of the location (Taketomi, Uchiyama, and Ikeda, 2017). The SLAM process is also probabilistic. A probability distribution, which has to be computed for all times t, is given by

$$p(x_t, m|z_{0:t}, u_{0:t}, x_0) \tag{3.4.4}$$

Where $x_{0:t} = \{x_0, x_1, ..., x_t\} = \{x_{0:t-1}, x_t\}$: The history of robot locations, $u_{0:t} = \{u_1, u_2, ..., u_t\} = \{u_{0:t-1}, u_t\}$, $m = \{m_1, m_2, ..., m_t\}$. The set of all landmarks and $z_{0:t} = \{z_1, z_2, ..., z_t\} = \{z_{0:t-1}, z_t\}$: The set of all landmark observations. p relates both the landmark locations and the robot's state at time t given sensor observations and control input commands with the initial state of the robot.

Using the Bayesian Theorem, a solution for the SLAM problem can be derived recursively. In order to do so, an observation model and motion model are required (Durrant-Whyte and Bailey, 2006). The former model refers to the probability of obtaining an observation z_t knowing the locations of both the robot and the landmark. It can be expressed as:

$$p(z_t|x_t, m) \tag{3.4.5}$$

However, the latter model refers to the probability of obtaining a current location x_t knowing the current control command u_t and a previous robot location x_{t-1}:

$$p(x_t|x_{t-1}, u_t) \tag{3.4.6}$$

The probabilistic SLAM problem can be solved by constructing a suitable representations for the observation model in Equation 3.4.5 and the motion model in Equation 3.4.6. As stated before, the SLAM algorithms can be implemented recursively while this recursion is a function of the two models. SLAM algorithms are mainly two-step algorithms: time-update and correction and expressed respectively as follows (Durrant-Whyte and Bailey, 2006):

$$p(x_t, m|z_{0:t}, u_{0:t}, x_0) = \int p(x_t|x_{t-1}, u_t)p(x_{t-1}, m|z_{0:t-1}, u_{0:t-1}, x_0)dx_{t-1} \tag{3.4.7}$$

$$p(x_t, m|z_{0:t}, u_{0:t}, x_0) = \frac{p(z_t|x_t, m)p(x_t, m|z_{0:t-1}, u_{0:t}, x_0)}{p(z_t|z_{0:t-1}, u_{0:t})} \tag{3.4.8}$$

Both time-update and correction steps offer a recursive process for estimating the posterior in Equation 3.4.8. In literature, there are three common solutions for the probabilistic SLAM problem. The first solution is in the form of state-space model which endows a Gaussian noise. This solution uses the Extended Kalman Filter (EKF) to solve the problem (Castellanos, Neira, and Tardós, 2018). The second solution expresses the motion model in Equation 3.4.6 as a set of more general non-Gaussian probability distribution samples. This solution is known as Rao-Blackwellised particle filter or FastSLAM (Montemerlo et al., 2002). The last solution, which called GraphSLAM, addresses the SLAM problem by using a graph. The graph contains nodes which represents both the robot states and the landmarks on the map (Thrun and Montemerlo, 2006).

3.4.3 Types of Robotic Maps

Robotic mapping may result mainly in two specific representations of maps: metric or topological. In metric maps, raw allothetic data are converted into 2D space information related to the idiothetic data. The geometric features of the environment, such as the position of some objects, mainly the obstacles, are stored on an absolute reference frame (Filliat and Meyer, 2003). Metric maps are explicit occupancy maps that include connectivity information (See Figure 3.7). This is the main reason why they are the most used form of maps in robotics. Topological maps or relational maps, on the other hand, are type of maps that explicitly represent the connectivity information of the environment in the form of a graph (Dudek and Jenkin, 2010). In these maps, the allothetic data of places are stored, instead (See Figure 3.8). Hence, topological maps represent the environment sparsely in such a way they only represent key locations for robots to navigate using allothetic information. The main advantage of topological models is that they do not require metric sensors to construct 2D models of the environment. However, navigation using such models becomes difficult since their precision is low (Filliat and Meyer, 2003).

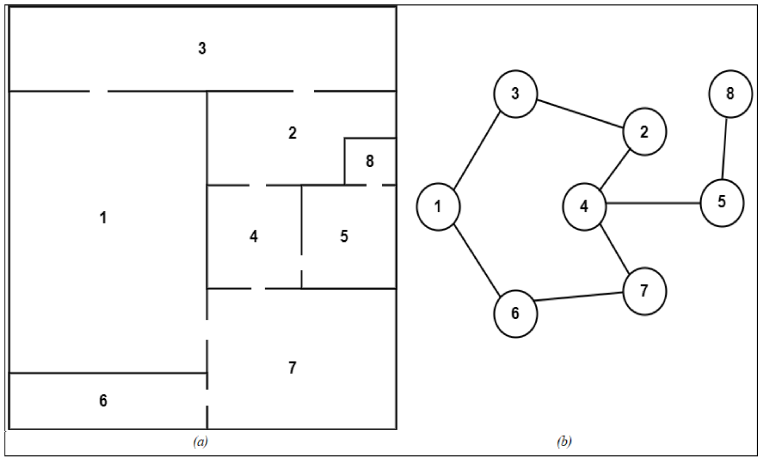

Figure 3.8: (a) Physical environment (b) topological representation.

3.5 UAV Path Planning Taxonomy

3.5.1 UAV Point-to-Point Path Planning

In the last decades, many point-to-point path planning algorithms for UAVs have been proposed in both 2D and 3D. These algorithms are constructed based on several theoretical hypotheses like soundness (no collision with the obstacles), completeness, path optimality, time complexity ... etc. The taxonomy of current methods of the developed path planning algorithm is shown in the following Figure 3.9.

Classical Paradigms

The first classical paradigm is Artificial Potential Field (APF). Path planning based on APF algorithms is based on a persuasive analogy proposed by Khatib in 1986 (Khatib, 1986): a robot, which behaves as a single particle, moves under the influence of a potential field U. The target location represents as an attraction point that draws the robot towards it while the obstacles represent repulsive potentials such that collisions are avoided (See Figure 3.10). At every location in the environment, the resultant force magnitude and direction, which the sum of the negative gradients of each potential, determines the motion the robot should take.

As stated before, an APF $U(q)$ is built from both components: the goal $U_{goal}(q)$ and the obstacles $U_{obs}(q)$. These components are used to generate attractive and repulsive forces, respectively. The robot is subjected to a total potential given by

$$U(q) = U_{goal}(q) + \sum U_{obs}(q) \tag{3.5.1}$$

The total potential field is then used to generate artificial forces given by

$$F = -\nabla U(q) \tag{3.5.2}$$

APF path planning algorithm offers several advantages: mathematical elegance and simplicity makes is suitable for UAV systems (Zhu, Cheng, and Yuan, 2016), smooth spatial paths can be

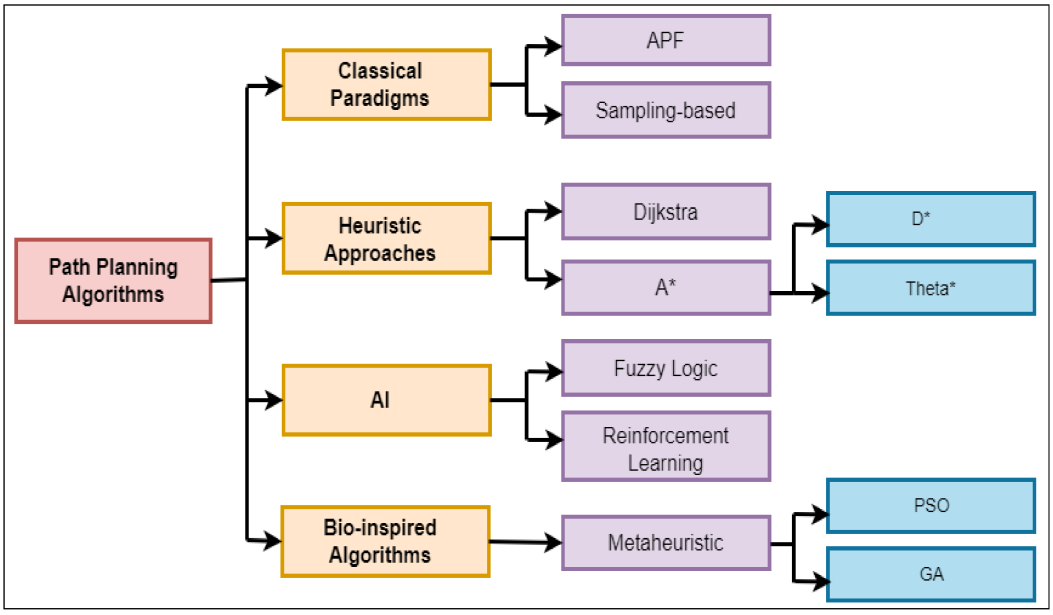

Figure 3.9: Point-to-point path planning taxonomy.

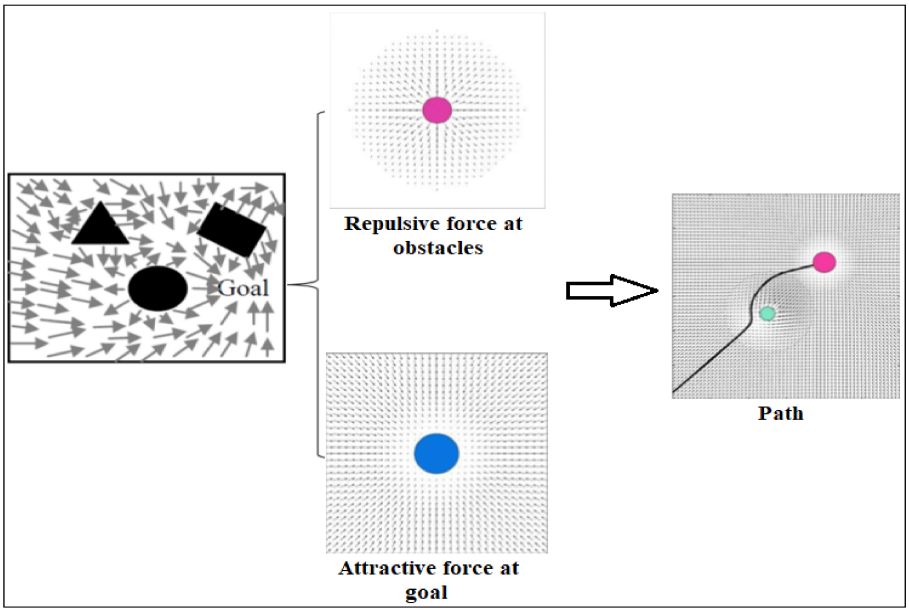

Figure 3.10: Illustration of the APF algorithm.

produced in real-time (Roussos, Dimarogonas, and Kyriakopoulos, 2010) and it can be coupled

directly to a tracking control algorithm (Santos et al., 2017). However, the main drawback of the APF algorithm is local minima issue. Several researches have been proposed to deal with the issue (Chen et al., 2016; Wang, Dai, and Ying, 2021).

The second classical paradigm is sampling-based method. Sampling-based path planning approaches require pre-defined information of the workspace where the robot is supposed to navigate. First, these approaches sample the robot's workspace into a set of nodes, or cells or other representations forms. Then, random search is performed to find a feasible path. Sampling-based approaches can be grouped into two algorithm types: Probabilistic Roadmap (PRM) and Rapidly exploring Random Trees (RRTs).

Path planning using PRM algorithms, which was proposed by Latombe in 1996 (Kavraki et al., 1996),consists of describing the connectivity of the continuous C-space in a network of one-dimensional curves for sake of having fewer states than the original C-space. PRM methods represent the environment by generating maps or graphs from sets of nodes and edges. Once the roadmaps are constructed, they are then utilized as standardized paths sets. In recent years, the PRM sampling-based algorithms have been greatly enhanced especially in case of UAVs (Yan, Liu, and Xiao, 2013; Tan et al., 2020). In literature, two types of PRM algorithms exist: the Voronoi Diagram (VD) and the Visibility Graphs (VG)(See Figure 3.11) (Giesbrecht, 2004). In the VD algorithms, the nodes are defined such that they are equidistant from all the points of the obstacles. Hence, the generated paths are relatively safe; however, they are not optimal. In addition, VD algorithms are not efficient, from practical point of view, especially when the dimensions are augmented (more than 3D) and complex data structures are required. The Visibility Graph, in contrast, makes use of the obstacle vertices including the starting and the target points to construct Visibility Line network that joints nodes pairs by a set of lines. Generalized VG is an extension of a simple VG where obstacles are in generalized polygonal shapes (Masehian and Amin-Naseri, 2004).

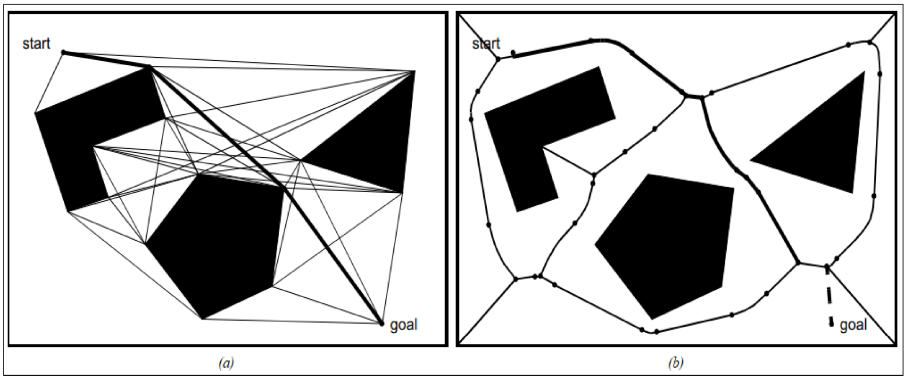

Figure 3.11: Probabilistic Roadmap algorithms: (a) VG, (b) VD (Siegwart, Nourbakhsh, and Scaramuzza, 2011).

RRT algorithms are further variations of the PRMs approaches. However instead of sampling the C-space, RRTs begin planning at the starting location and randomly expands a path, or tree, in the configuration space. The tree is constructed incrementally from samples built randomly from the search space and is inherently biased to grow towards large unsearched areas of the problem (See Figure 3.12). RRTs typically construct a graph during the search process and thus a priori only require

an obstacle map (without graph decomposition) (Siegwart, Nourbakhsh, and Scaramuzza, 2011). The Table 1 shows a pseudocode of the RRT algorithm where $RANDOM\_CONFIGURATION()$ selects randomly a coordinate q_{rand} in the domain D, $NEAREST\_VERTEX()$ finds the vertex in the tree closest to q_{rand} in D and $NEW\_CONFIGURATION$ produces a new configuration q_{new} in the tree by displacing a distance Δ from q_{near} to q_{rand}.

Algorithm 1 RRT algorithm

Input:
$q_{init} \leftarrow$ *Initial configuration*
$K \leftarrow$ *Number of vertices in RRT*
$\Delta \leftarrow$ *Incremental distance*
$D \leftarrow$ *the planning domain*
Output: $G \leftarrow$ *the RRT*
Processing:
Repeat K times
$q_{rand} \leftarrow RANDOM\_CONFIGURATION(D)$
$q_{near} \leftarrow NEAREST\_VERTEX(q_{rand}, G)$
$q_{new} \leftarrow NEW\_CONFIGURATION(q_{near}, q_{rand}, /(\Delta))$
Add vertex q_{new} *to* G
Add an edge between q_{near} *and* q_{new} G
End repeat
Return G

RRTs represent great sampling-oriented approaches for exploring pathways randomly. In addition to their simplicity, they have also a relatively high speed to plan paths in large and high dimensional search spaces. However, they are inefficient, and they create routes with sharp bends (Rodriguez et al., 2006). Since RRTs do not explicitly construct the entire configuration space, the

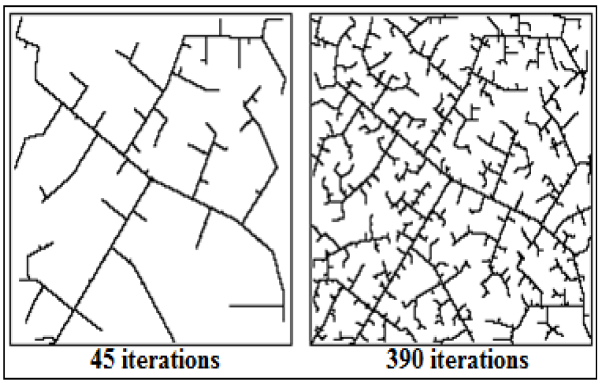

Figure 3.12: RRT evolution illustration (Siegwart, Nourbakhsh, and Scaramuzza, 2011).

problem of a growing search time with growing spatial dimensions is avoided. RRTs tend to have a lower algorithmic complexity in 3D space. Accordingly, they are the most suitable for solving path planning problems for single UAV (Killian and Backhaus, 2021; Nurimbetov et al., 2017) and

multi-UAVs (Liu et al., 2022). Recent researches have been developed many RRT algorithms. In terms of performance, RRT*, which hasan asymptotically optimal property, is an improved version of RRT algorithm. The RRT* always finds a solution even though the number of samples grow to infinity (Karaman and Frazzoli, 2011).

Heuristic Approaches

In many UAV applications, the search spaces are generally large and have many unexplored regions. Additional guidance, which gives the UAV information about the distance to the target location, could save considerable amount of time. Path planning, in this case, can be done by heuristic algorithms. These latter deal with nodes' and arcs' weight information and compute the cost by exploring among the nodes resulting in an optimal path. Network algorithms and Node Based Optimal Algorithms are other names of heuristic algorithms (Yang et al., 2016).

Dijkstra algorithm, proposed by E. W. Dijkstra in 1959, is a graph search algorithm which sorts solutions for the single-source shortest path problem for a graph with known non-negative edges'/arcs' (Dijkstra, 2022). In other words, the algorithm assumes any positive value and searches for a shortest path that depends on local path costs only. An improved of the Dijkstra Algorithm is the Bellman-Ford algorithm that generates an optimal path with positive and negative costs (De Filippis, Guglieri, and Quagliotti, 2012; De Filippis, Guglieri, and Quagliotti, 2011). The flowchart of the Dijkstra algorithm is shown in Figure 3.13 where **O** is an open list and c is the current configuration.

Developed in early 50s, A* star is considered one of the most used algorithm for motion in robotics. This algorithm merges the Dijkstra and Bellman-Ford characteristics to effectively analyze the domain for avoiding distributed obstacles. The A* algorithm is a graph search algorithm that finds a solution path from a given initial location to a given final location. It uses a heuristic estimate (Hart, Nilsson, and Raphael, 1968):

$$f(n) = g(n) + h(n) \tag{3.5.3}$$

Equation 3.5.3 provides an estimate of the best shortest route, where $g(n)$ is the cost from the starting node to the node n and $h(n)$ is heuristic estimate cost from the node n to the goal. A* algorithm is known also as the best first search. Usually, the Euclidean distance is used for calculating the heuristic cost. A* is computationally expensive since it has to pre-plan the entire path every time new information is added. In a grid map, where the starting location, target location and the obstacles are identified, A* algorithm starts by searching the 8 neighboring nodes of the starting location. The heuristics of those neighboring cells are calculated. Then, the process continues by choosing the nodes that are closer to the target location till the path is sorted. Figure 3.14 depicts the flowchart of the A* algorithm.

For 3D UAV flights, a number of A* algorithm variants have been implemented. One variant, the Dynamic A* or D* only updates new nodes while reducing computational demand (Zhang et al., 2020). The Lifelong Planning A* or LPA* is an incremental A* variant that improves the efficiency of path planning by information reuse. Such algorithm behaves more efficiently when it has specific start and goal locations nodes (Koenig and Likhachev, 2001). The variant D* Lite was developed to plan paths in case of a changing start node and a fixed goal node. D* Lite merges both the heuristic and incremental search for purpose of achieving path planning. The algorithm takes into consideration the information of the path between the current node and the goal one. Thus, D* Lite is now widely used in autonomous navigation systems(Hao et al., 2020). Another variant is

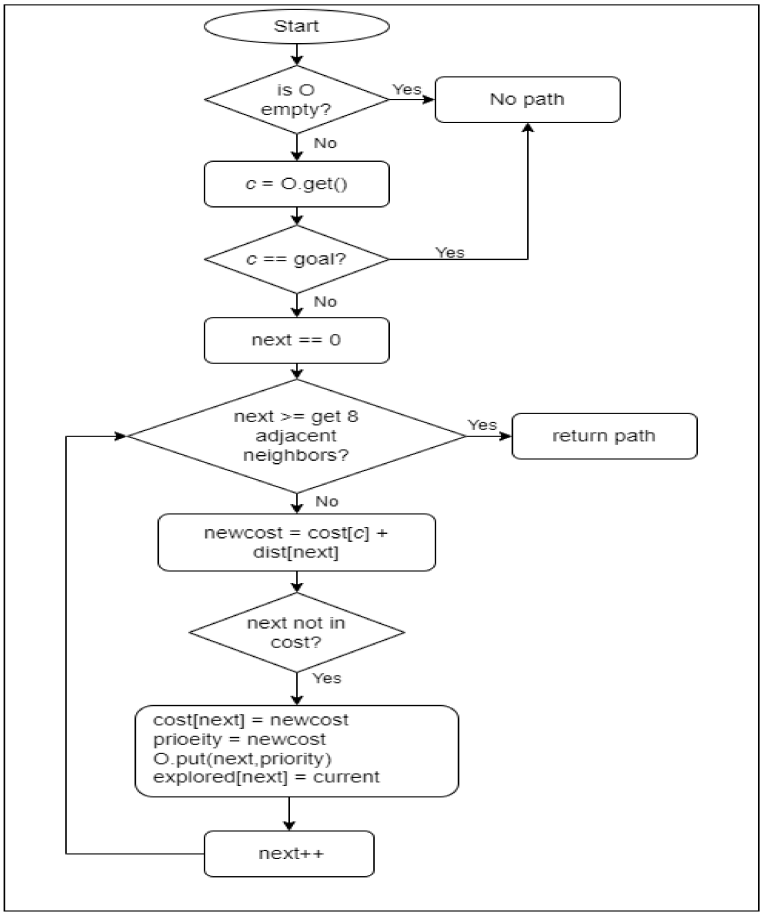

Figure 3.13: Dijkstra algorithm flowchart.

Lazy Theta* which can be used over octrees to find an optimal path between the start and target nodes for sake of exploration (Faria, Maza, and Viguria, 2019; Faria et al., 2019).

Artificial Intelligence Approaches

In artificial Intelligence or AI, the term planning is used generally to refer to the process of searching for a sequence of logical operators or actions in order to transform an initial state to a goal one. Hence, the aim here is to design systems that think intelligently and use decision-theoretic models to generate suitable operators. UAVs (or multi-UAVs) systems are relatively novel field of AI applicable for efficiently solving different complex navigation problems where conventional solutions require a considerable amount of hand-tuning. Therefore, AI techniques are increasingly being used to enhance autonomous UAV systems navigation (Rezwan and Choi, 2022).

Fuzzy logic (FL) imitates the human brain when making decisions in uncertain situations or dealing with incomplete knowledge. FL uses true real number values between 0 and 1 to form a

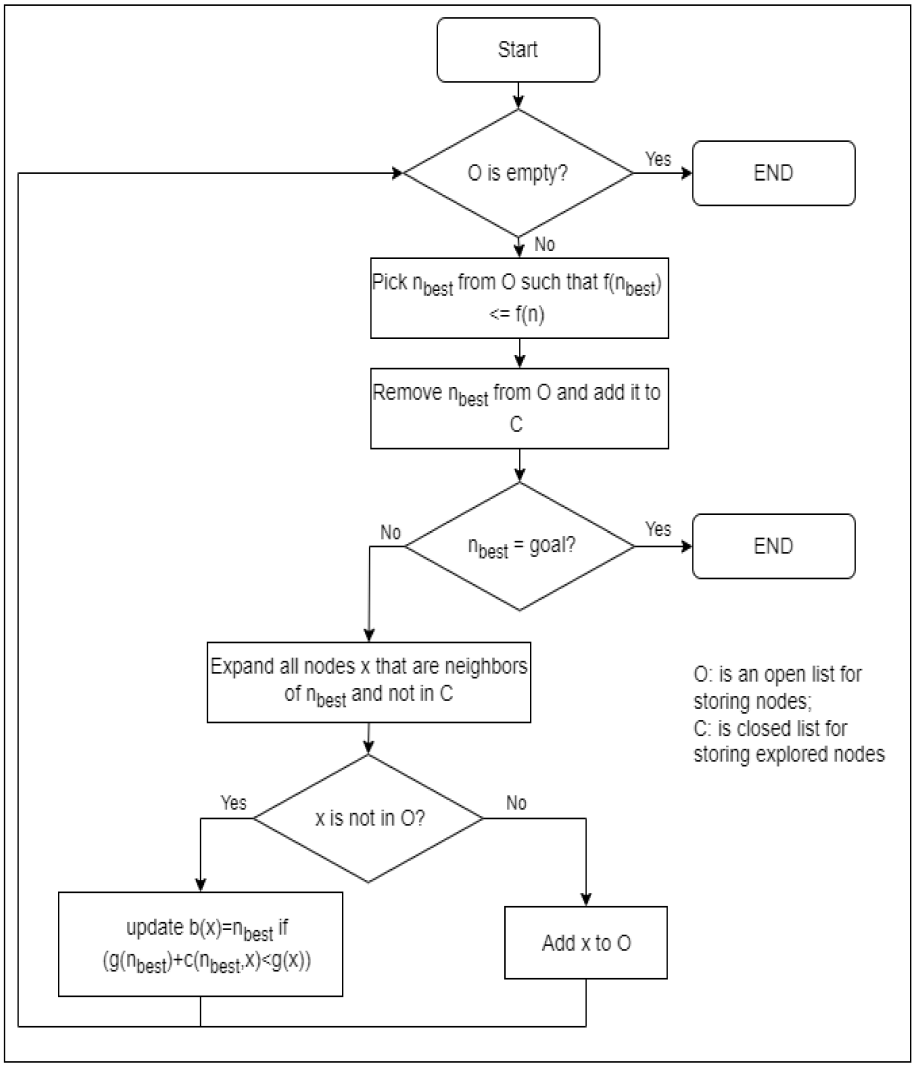

Figure 3.14: A* algorithm flowchart.

multi-valued logic. These values are used to deal with the concept of partial truth. In this latter, the truth value may range between completely true and completely false. FL has been the key for UAV path planning thanks to their high adaptability to objects that are unbounded and/or unclear (Masehian and Sedighizadeh, 2007). FL algorithms are used to solve path planning problems that are characterized with ambiguous optimization objectives. In literature, few works have been conducted on FL stand-alone algorithm for path planning due to difficulty in adjusting membership functions (between 0 and 1) such as in (Zhuoning et al., 2010; Sabo and Cohen, 2012). However, it is used in combination with other types of algorithm like Genetic Algorithm (Kermani and Afzalian, 2014), Ant Colony Algorithm (Taylor and Choi, 2014) and Multi-objective FL (Yanyang, Tietao,

and Xiangju, 2012).

Reinforcement Learning (RL) is a machine learning process where leaning is achieved by interacting with the environment. In RL, sequences of actions are applied to collect information from the environment. After each execution of an action, a feedback is obtained. The feedback is a trial-and-error process that can be used to evaluate the action made by the environment. The evaluation of the action by the environment is called an immediate reward which can be defined as an improved signal that stipulates the impact of the action on the result (Yu, Su, and Liao, 2020). The RL model is depicted by Figure 3.15

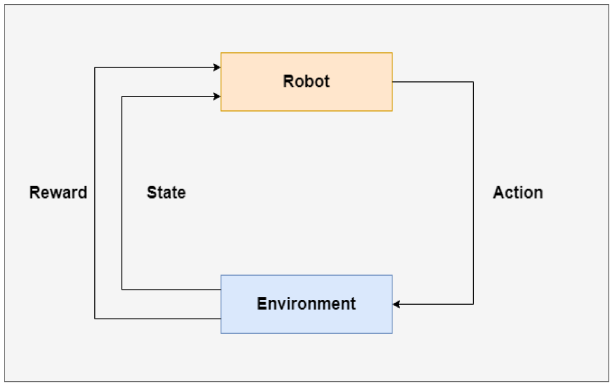

Figure 3.15: RL schematic diagram.

In UAV path planning, RL generates and updates the selection strategies of the vehicle actions. The environmental rewards are obtained based on the last state and last actions of the vehicle. Different variants of RL are used to plan paths for UAVs such as Q-Learning (Carvalho et al., 2022), Improved Q-Learning (Yan and Xiang, 2018) and Deep RL (DRL) (Theile et al., 2021).

Bio-inspired Approaches

Bio-inspired approaches are created from mimicking biological behavior to solve path planning problems. The algorithms exclude the process of constructing complex models of the robot's environment and suggest a strong method to achieve the goal location. This may avoid some problems where the aforementioned algorithms fail in solving NP-hard problems with large number of variables and nonlinear objective functions (one problem is the local minima) (Aghababa, 2012).

Genetic Algorithm (GA), which is based on a genetic reproduction mechanism and natural selection, is the most famous population-based numerical optimization method. GA is a reactive search algorithm that obtains a solution space for an optimal solution to a problem. GA is decomposed of chromosomes; each represents a solution, with different genes. The GA algorithm is implemented such that populations of possible solutions are created and evolve to reach a better solution (Galvez, Dadios, and Bandala, 2014).

Ant Colony Optimization (ACO) is another bio-inspired approach that has been used for solving UAVs path planning problems. The algorithm concept imitates the movement of a group of ant. The path of the whole group represents the solution space for the problem of optimization. As the time goes on, the shorter path is designated by the accumulation of pheromone. In this way, there will be an increase in the number of ants choosing this path. Finally, the corresponding path

represents the optimal solution for the path planning optimization problem (Cekmez, Ozsiginan, and Sahingoz, 2014).

Particle Swarm Optimization (PSO) is one of the most well-known metaheuristic approaches for path planning and formation problems, in general. PSO uses a stochastic optimization process based on the behavior representation of natural organisms such as bird flocking and fish schooling. The PSO principle is based on a population, named swarm, where each member, known as particle, can be a potential solution to the problem. After randomly initializing the position and the velocity of each particle in a continuous search space, the position and velocity values are updated after each iteration of the process (Alaliyat, Oucheikh, and Hameed, 2019). The position and the velocity of each particle are adjusted according not only to its own voyage experience but other particles' experiences as well. These values are referred to as personal best or p_{best}. Optimization is achieved when each particle reaches its p_{best}.

3.5.2 UAV Coverage Path Planning

In recent years, the adoption of UAVs for surface coverage applications has been emerged. The UAV coverage path planning (UAV-CPP) is being used in different fields. In UAV-CPP, the surface is tracked by the vehicle while taking into account several factors aiming to reduce the time of flight or save energy. Essentially, the planned path that covers the whole surface has to be optimal. Here, it is essential to point out that the aforementioned point-to-point path planning methods can be used in this purpose. However, most of these methods have been used in environments without obstacles. Thus, in environments where obstacles are present, adequate CPP algorithms are required.

Since CPP methods decompose the target environments into sub-regions called cells to achieve coverage, the classification of such methods is performed according to the type of decomposition used.

Exact Cellular Decomposition Coverage Methods

In the exact cellular methods, the free space of the environment is decomposed into simple, non-overlapping cells. The combination of all free-obstacle cells fills the free space. This latter can be covered easily and swept by the UAV using simple zig-zag motion patterns. Exact cellular decomposition CPP methods generate paths in two steps. First, the free space is decomposed into cells and stored the decomposition as an adjacency graph. Next, graph-based search is used to exhaustively walk through the adjacency graph (node visiting) to derive the optimal coverage path (Nasirian, Mehrandezh, and Janabi-Sharifi, 2021). Two of the most popular exact cellular decomposition methods are trapezoidal decomposition and boustrophedon decomposition. The first is an offline method that breaks down the environment into trapezoids. Simple back-and-forth motions can be used to cover each trapezoid. The boustrophedon method decomposes the free space similarly as the trapezoidal decomposition. However, vertices, where a vertical segment can be extended both above and below the vertex, are considered (See Figure 3.16). These vertices are called critical points (Galceran and Carreras, 2013).

Grid-Based Coverage Methods

Grid-based decomposition represents the environment as a grid of cells where each cell has the same size and shape. It is known also as approximate cellular decomposition since it approximates the shape of the free space and the obstacles. In this decomposition, each cell is associated with a

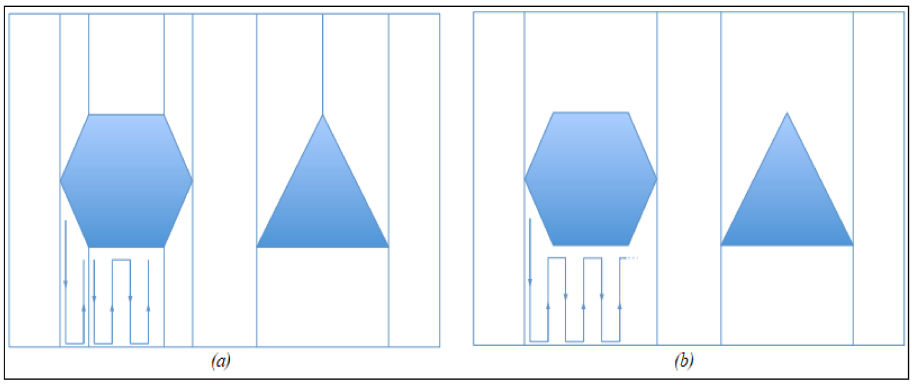

Figure 3.16: (a) Trapezoidal decomposition vs. (b) boustrophedon decomposition (Galceran and Carreras, 2013).

certain value to state whether an obstacle is present or not. In most cases, square shapes are used, however, other shapes may be used instead (e.g., triangular) (Oh et al., 2004).

After decomposing the environment into a grid, the search for an optimal coverage path can be done using several approaches. Grid-based coverage using wavefront uses a distance transform to propagate a wavefront from a starting location to a target location in order to assign a specific number to each grid cell. Initially the target location is assigned by a 0 and all of its surroundings are assigned a 1. After that, the algorithm assigns 2 to all of the unmarked cells neighboring the marked 1. The process continues incrementally until the starting location is achieved (See Figure 3.17).

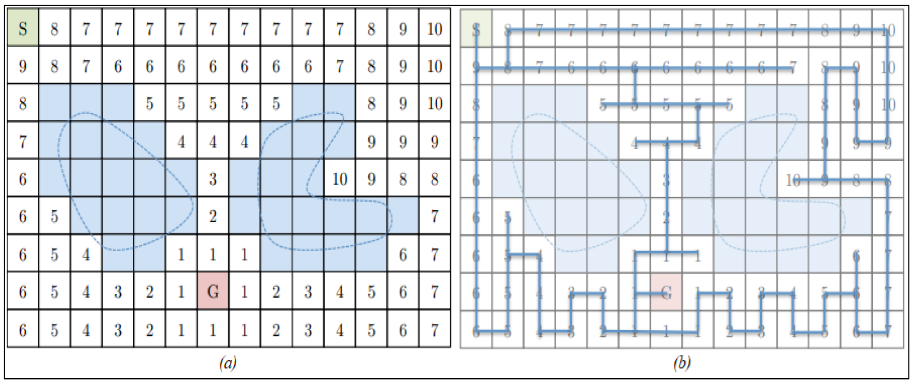

Figure 3.17: (a) Wavefront distance transform vs. (b) Coverage path (Galceran and Carreras, 2013).

Grid-based CPP using Spiral Spanning Trees is an online coverage technique where a path is generated following a systematic spiral pattern. Using onboard sensors, a spanning tree of the partial grid is constructed by the robot. This can be achieved by dividing bigger (or mega) cells into four cells of same size as the robot, first. The robot, which is located in its current cell,

moves to a new mega cell in the free space using an anti-clockwise direction. A new tree edge of the spanning tree is formed between the current cell and the new cell. The process proceeds in a recursive manner. When the current cell has no neighbors, the process stops. Finally, the robot uses the spanning tree to move from on cell to another until it reaches the tree end (Gabriely and Rimon, 2002).

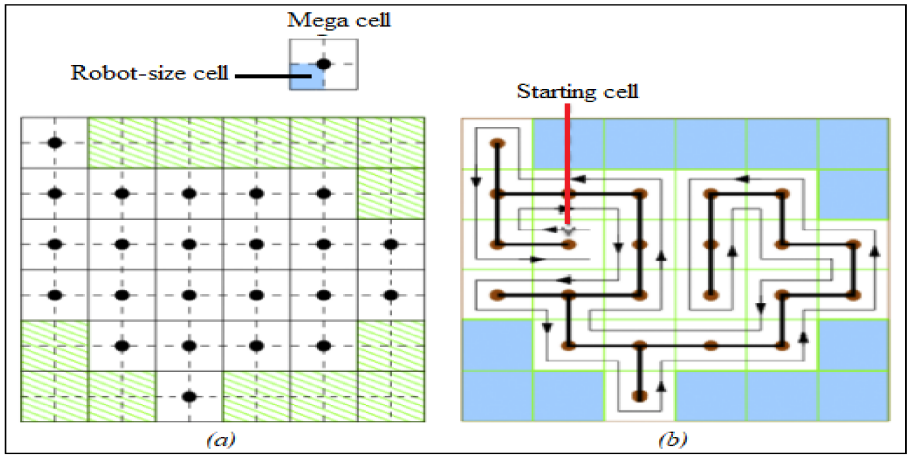

Figure 3.18: (a) Approximate cell decomposition vs. (b) Coverage spiral path (Galceran and Carreras, 2013).

3.6 Conclusion

The focus of this chapter was on path planning, which constitutes the initial stage of autonomous navigation. To commence the chapter, we established the definition of path planning and introduced various terms associated with it. Then, we divided the path planning process into two stages: map representation stage and query stage. Starting with the first stage, we described the different maps used for representing the environments where the quadrotor navigates. In addition, we presented the different methods that are used for building these robotic maps. At the end, categorized the approaches implemented for the query process into: point to point path planning and coverage path planning and reported the taxonomy of each.

Chapter 4

Quadrotors Trajectory Generation & Control Strategies

4.1 Introduction

Generation and execution of feasible trajectories are considered the core problems in quadrotor control. The trajectory generation step produces smooth trajectories as a function of time that the quadrotor must follow. The generated trajectories must take into account the vehicle's thrust and hardware limitations as well as satisfy the different constraints imposed by the control in order to perform complex and high-speed maneuvers. Despite quadrotors have relative mathematical simplicity and low mechanical complexity, designing feasible trajectory generation algorithms and tracking control laws is still challenging. The aim of this chapter is to investigate the different approach that have been developed for generating and tracking feasible and smooth time trajectories for the position and the orientation of quadrotors. However, before approaches for these are investigated it is necessary to give further details on quadrotor design.Hence, this chapter will cover the kinematic and dynamic models that will be used in the simulations.

4.2 Quadrotor Dynamics Model

In general, the flight motion of autonomous aerial vehicles with rotory wings, for applications like search-and-rescue, surveillance, inspection, etc, is characterized by a nonlinear dynamic behaviour. High stability, high precision hovering ability, high bandwidth, and high manoeuvrability are of great importance for successful execution of tasks within the application.

Quadrotor platforms have been extensively applied for research in flying robotics thanks to their ability to hover, high maneuverability, mechanical simplicity and robustness. Quadrotors are also known by their low rotational inertia which allow large translational accelerations and rapid rotational accelerations (Penin, 2018). However in practice, it is very difficult to design models of these vehicles that capture all the aerodynamic effects. Therefore, it is necessary to characterize the non-linearities for each flight configuration in order to provide them with autonomous flight and navigation capabilities. The work in (Bouabdallah, 2007) is one of the the earliest and most comprehensive works on quadrotor design and control. Other works like (Martinez Martinez, 2007)

and (Amir and Abbass, 2008) include the quadrotors aerodynamic effects. The model presented in this section is common to the majority quadrotor vehicles.

In order to derive the nonlinear model of the quadrotor, a set of coordinates systems for specifying the position, velocity, forces and moments acting on the vehicle must be defined, first. Let the inertial frame be the surface of the earth and the body frame be fixed on the quadrotor rigid body as shown in Figure 4.1. $\{\vec{b_1}, \vec{b_2}, \vec{b_3}\}$ are the unit vectors along the body-fixed frame axes $\{\mathcal{B}\}$ where by convention $\vec{b_1}$ is the longitudinal axis pointing to the front of the vehicle, $\vec{b_2}$ defines the lateral and right-facing axis of the vehicle and $\vec{b_3}$ defines the vertical axis of the vehicle and points down in our case. $\{\vec{a_1}, \vec{a_2}, \vec{a_3}\}$ are the unit vectors along the body-fixed frame axes $\{\mathcal{I}\}$ where $\vec{a_1}$ points to the North, $\vec{a_2}$ points to the East and $\vec{a_3}$ points upwards. This latter is supposed to be Galilean. It is assumed that the center of mass of the quadrotor coincides with the origin of the body-fixed frame. Further assumptions like the rigid structure and the axial symmetry of the quadrotor body and no induced drag, blade flapping and ground effects are taken into consideration.

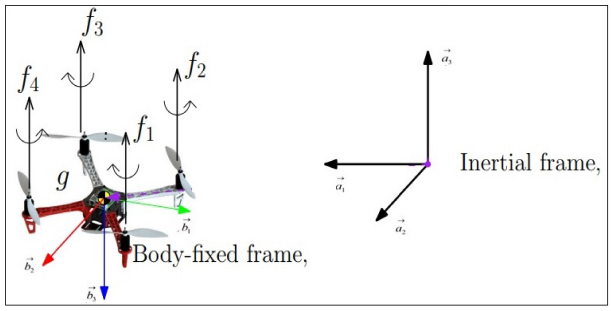

Figure 4.1: Quadrotor inertial and body-fixed frames.

The configuration of quadrotor modeled as a rigid body with a mass m and a moment of inertia $J \in \mathbb{R}^{3\times3}$ is represented by a position, denoted by $b \in \mathbb{R}^3$ and expressed in frame $\{\mathcal{I}\}$ and an orientation, denoted by $R \in SO(3)$ and defined as the rotation matrix from frame $\{\mathcal{B}\}$ to frame $\{\mathcal{I}\}$. Both the position and the orientation together is referred as its pose G and given by

$$G = \begin{bmatrix} R & b \\ 0 & 1 \end{bmatrix} \in SE(3) \tag{4.2.1}$$

Where the special Euclidean group in three dimensions $SE(3) \in \mathbb{R}^{3\times3}$ is the six-dimensional Lie group of rigid body motions (translational and rotational) that is obtained as the semi-direct product of $\mathbb{R}^3$ with $SO(3)$. Figure 4.2 shows a schematic diagram that depicts the trajectory on $SE(3)$ passing through several poses G.

The quadrotor satisfies the kinematics relation

$$\begin{cases} \dot{b} = v \\ \dot{R} = RS(\Omega) \end{cases} \tag{4.2.2}$$

Where v is the quadrotor's translational velocity expressed in $\{\mathcal{I}\}$; Ω is its rotational velocity expressed in $\{\mathcal{B}\}$ and $S(.)$ is the skew-symmetric cross product operator that gives the vector space isomorphism between $\mathbb{R}^3$ and $\mathfrak{so}(3)$, the Lie algebra of the Lie group $SO(3)$, which is a set of

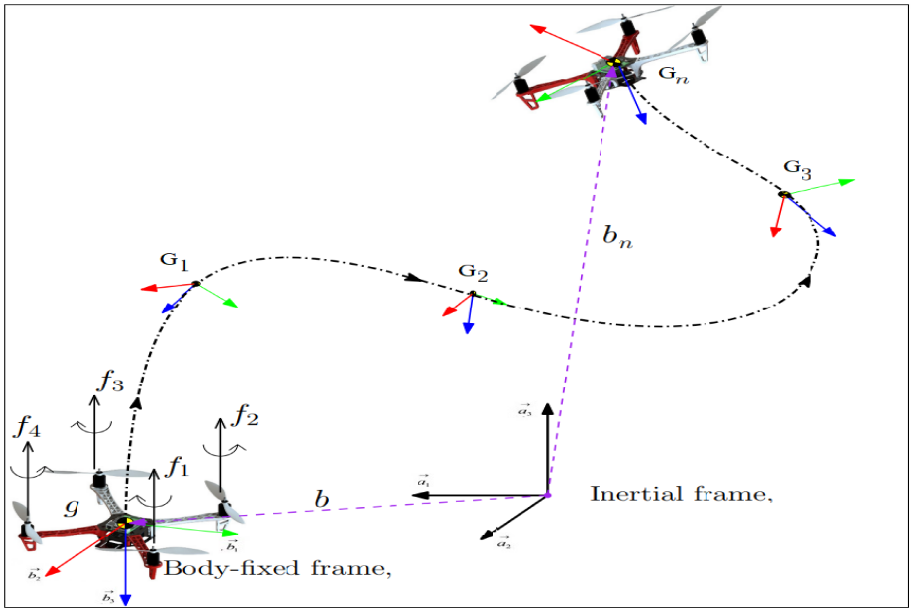

Figure 4.2: SE(3) trajectory of a quadrotor (Dhullipalla et al., 2019).

matrices $M \in \mathbb{R}^{3\times 3}$ such that

$$\mathfrak{so}(3) := \{M = \begin{bmatrix} 0 & -\alpha_3 & \alpha_2 \\ \alpha_3 & 0 & -\alpha_1 \\ -\alpha_2 & \alpha_1 & 0 \end{bmatrix} | M^T = -M\} \tag{4.2.3}$$

Where $\alpha_i \in \mathbb{R}$. The quadrotor dynamics are expressed by the following equations (Sanyal, Nordkvist, and Chyba, 2010)

$$\begin{cases} \dot{v} = \frac{f}{m} Re_3 - ge_3 \\ J\dot{\Omega} = J\Omega \times \Omega + \tau \end{cases} \tag{4.2.4}$$

where f is the thrust magnitude, $e_3 = [0,\ 0,\ 1]^T$, hence Re_3 is the thrust direction.

4.3 Quadrotors Trajectory Control Strategies

Quadrotors have been particularly selected in many research works thanks to their good maneuverability, stability and payload. Besides, they are agile vehicles that permit fast and accurate motion behavior. Originally, quadrotors have been the object of study on designing controllers that are capable of stabilizing them. Thus, different stabilization control techniques have been developed: Backstepping, Feedback Linearization, Sliding Mode Control, PID, optimal control, robust control, learning-based control, etc. (Özbek, Önkol, and Efe, 2016). Considering that these techniques have been extensively studied, today the challenge for quadrotors is to deal with path/trajectory control, fault tolerant control, path planning or obstacle avoidance problems.

The path/trajectory control problem can be defined as guiding the quadrotor to follow a pre-described path/trajectory in space. This can be mainly achieved using two different approaches: path following controller or trajectory tracking controller (Rubí, Pérez, and Morcego, 2020). The first approach solves the problem of following a reference path without assigning time information to it. However, in the second approach, the reference trajectory with assigned time information is tracked.

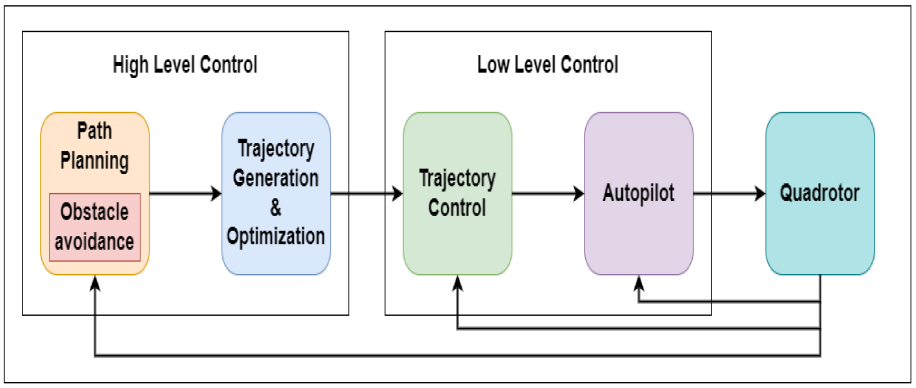

Figure 4.3: Trajectory generation and control structure.

In order to take advantage of the quadrotor capabilities, appropriate trajectory control systems must be implemented to allow reliable control over both the position and the attitude of the vehicle. A plethora of trajectory control systems, that allow rejecting disturbances in quadrotor's position and attitude, has been emerged (Li, Sun, and Jin, 2015). Most of these control systems use a nested structure, with an inner-loop and outer-loop, for controlling both the orientation and the position of the vehicle, respectively. Thus, simple combination of control systems may be allowed to form hybrid controllers aiming to obtain improved results. Besides, it is important to point out that most of the trajectory control problems have been solved by relying on a sort of linear dynamic assumptions for the controller design. Therefore, a trajectory control system can be either linear or nonlinear. This latter are characterized to be theoretically complicated and extensive studies are required to understand their functionality in terms of implementation. Yet, linear controllers are much easier. In addition, nonlinear control systems consider the quadrotor full dynamic system for operating in a wider operating region, hence, taking into account the vehicle's nonlinear aerodynamic effects. On the other hand, linear control systems have restricted operation regions (Partovi et al., 2012).

4.3.1 Linear Trajectory Control Systems

Quadrotors are inherently nonlinear, non-stable and complex systems, containing couplings between the different axes and which can be subject to significant disturbances like wind. Preliminary studies on designing controllers relied on the linearization of the vehicle dynamics around a hover point (Li, Sun, and Jin, 2015). In other words, the vehicle's speeds of translation and orientation are low, which makes it possible to neglect the couplings between the different axes. Consequently, a simplified dynamic model can be obtained and it is decoupled into four mono-input-mono-output subsystems of the dynamics of the vehicle around the equilibrium point: the altitude subsystem,

the longitudinal motion subsystem, the lateral motion subsystem, and the yaw subsystem. These subsystems are independent and controllable, su that linear control techniques can be used easily.

Several classical and modern linear control techniques have been successfully applied on quadrotors to track predefined trajectories. Many research works have applied PID controllers. In most cases, the control structure of the quadrotor contains an inner loop and an outer loop to stabilize the attitude and the position, respectively (Li and Li, 2011). PID controllers can be applied in both loops as shown in Figure 4.4. Otherwise, they can be applied in one single loop and combined with other type of controllers applied in the other loop. PID controller has several advantages such as it is easy to implement and its parameters are easy to optimize. Besides, PID can be used even in the situation of without the knowledge of quadrotor dynamic model. Nevertheless, using PID controller on quadrotors tend to limit its performance because of the under-actuation and nonlinearity characteristics of the vehicle model.

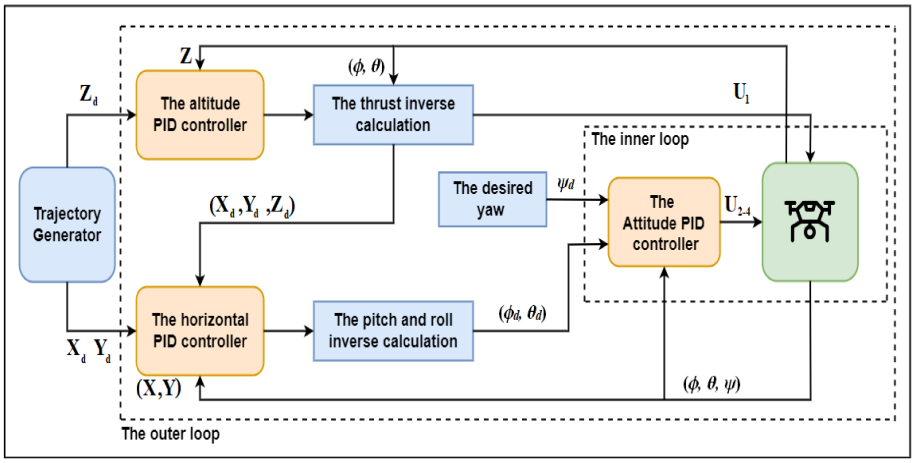

Figure 4.4: PID controller structure for a quadrotor (Jiao et al., 2018).

Another linear trajectory control technique is to use optimal control theory. The aim of such technique is to operate the vehicle dynamic system at a minimum cost. Two well-known optimal control techniques have been used to solve trajectory control problem: the Linear Quadratic Regulator (LQR) and the Linear Quadratic Gaussian (LQG). LQR controllers are still being designed thanks to their capabilities to handle complex dynamic systems like quadrotors. Although LQR controller is restricted to linear control laws, linearizing the quadrotor nonlinear dynamics shows that greater performances are achieved (Zulu and John, 2016). LQR can be combined with Kalman Filter (KF) and transformed into an LQG while preserving the control optimality (See Figure 4.5). This is done in order to have both an optimal controller and an estimator simultaneously. Besides, the LQR controller effectiveness is enhanced against the Gaussian white noise. As a result, clear vehicle performances can be improved. However, relying heavily on the linearization of quadrotor dynamics, detrimental effects can occur due to the parameter uncertainties (Montazeri, Can, and Imran, 2021).

Gain scheduling controllers have a wide application in industry. The use of such controllers has been extended to include autonomous vehicles like quadrotors where the scheduling of appropriate parameters can be autonomously achieved during the flight. In other words, knowing that the

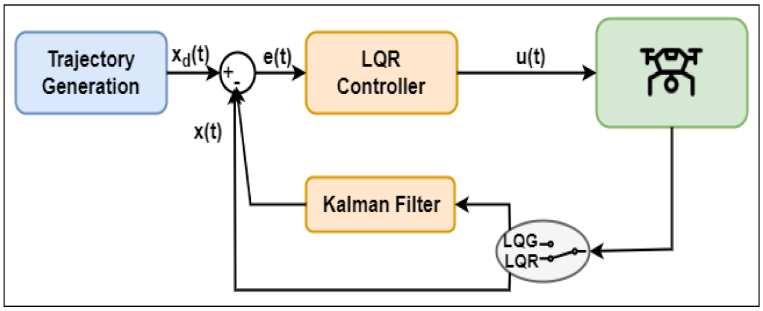

Figure 4.5: LQR/LQG control structure.

linearization about an operating point of the quadrotor nonlinear dynamics is only valid around that point where local asymptotic stability is ensured. Hence, by extending to a range of operating points, the gain scheduling can enhance the quadrotor linearization capability. This can be done by urging the controller gains to vary as the system dynamically evolves (Sawyer, 2015). Gain scheduling is an attractive approach since it uses linear control design techniques that are well-understood, theoretically mature and comprise computationally efficient synthesis processes. In addition, due to the plant linearization, the desired stability and the performance properties are achieved by the feedback system. In case of changing of operating conditions, the gain scheduling control systems tend to respond rapidly (Bett and Chen, 2005). However, the main drawback of the scheduling controllers is that many decisions to be made are system-dependent (the selection of appropriate scheduling gains and scheduling procedure).

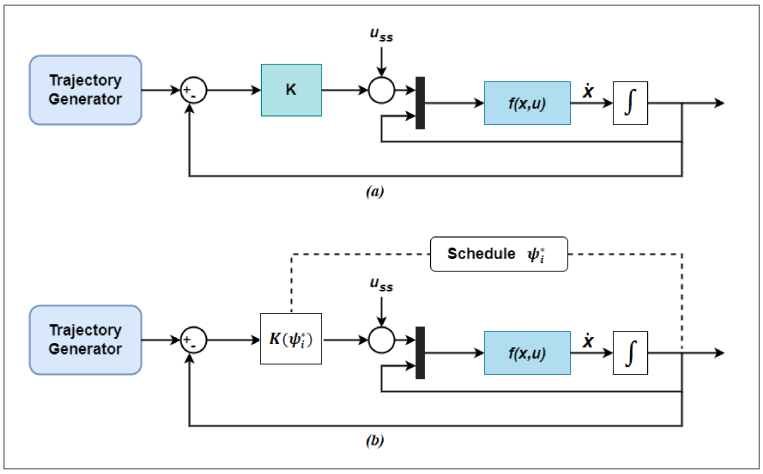

Figure 4.6: *(a)* Conventional proportional controller vs. *(b)* gain scheduled controller (Sawyer, 2015).

Unlike modern and classical approaches (PID and LQR), the $\mathcal{H}_{\text{inf}}$ control approach takes into account both performance and robustness explicitly; this is why it is also called robust control method. $\mathcal{H}_{\text{inf}}$ regards the control as a mathematical optimization problem where the aim here

is to obtain the correct gain values that stabilize the system. The $\mathcal{H}_{\text{inf}}$ control configuration was introduced by Doyle (Doyle et al., 1988) where the control structure shown in Figure 4.7 depicts how the $\mathcal{H}_{\text{inf}}$ works. The block P represents the generalized plant containing the quadrotor dynamics,

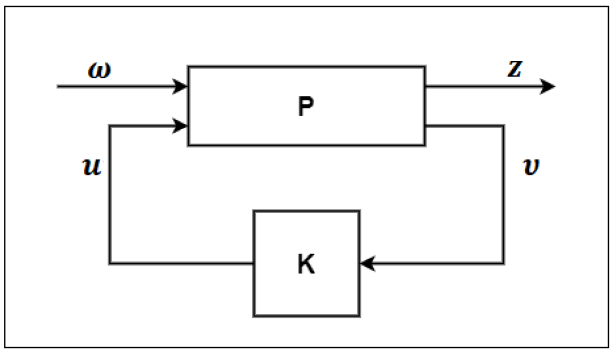

Figure 4.7: Control structure of $\mathcal{H}_{\text{inf}}$ controller (Doyle et al., 1988).

while the block K is the controller. ω is the input signal carrying external signals (e.g., sensor noise, uncertain disturbances) and the commands. Z is the output signal carrying the system states. v is the measured variable while u is the control input signal that enhances the plant P performances. The purpose of this structure is minimizing the output signal error Z using the measured variable v in K for manipulating the control input variable u. Designing such control system has demonstrated the rejection of external disturbances as well as dealing with parametric uncertainties.

4.3.2 Nonlinear Trajectory Control Systems

The linear control approaches mentioned above keeps stability only within a certain region of equilibrium points around which the quadrotor dynamics are linearized. The system becomes uncontrollable when uncertainties are introduced. However, two methods have been commonly used to handle the presence of uncertainties in nonlinear dynamical models: fixed gain control and adaptive control. The fixed gain control, which is a robust control approach, dominates the nonlinear disturbances in the system dynamics. Like so, the uncertain nonlinearities are handled within a certain bound that is required to be known priory. The adaptive method, however, is a type of control that is capable of handling the changing in the system parameters by adapting itself. The adaptive control does not require the uncertainties bound and proposes an adaption law for estimating the unknown parameters online in the closed-loop system.

Feedback Linearization

Feedback Linearization is one of the commonly used methods for controlling nonlinear dynamical systems. The purpose of this control method is to linearize a system in certain range of the state space using nonlinear inversion of the plant. In such a way, the plant nonlinearities are cancelled so that linear control theory can be applied (Byrnes and Isidori, 1991). Feedback Linearization has many advantages like simplicity in the control structure and implementation facility. However, it has some limitations such as it requires more exact model for avoiding loss of precision because of the process of linearization (Zulu and John, 2016). This kind of control method is frequently

applied in robot control like quadrotors. Several research works have been presented in literature like in (Bonna and Camino, 2015; Al-Hiddabi, 2009).

Backstepping

Backstepping is a famous technique extensively used for nonlinear control systems. This technique consists of breaking down the architecture of the control system into subsections. Backstepping control uses a recursion algorithm that stabilizes the system based on the Lyapunov stability where the objective is to force a set of given errors to be zeroed. First, a positive definite Lyapunov function is asserted for each error such that the time derivative of those functions is negative definite. In this way, errors converge to zero and the stability of the system is reached. In other words, the Backstepping approach links the choice of a control Lyapunov functions with the feedback controller design and assures global asymptotic stability of strict feedback systems (Vaidyanathan and Azar, 2016). This control approach is used in most of trajectory control designs since it is able to offer wider regions of attraction than any other controller type. Besides, the Backstepping control algorithm converges fast and ensures boundedness of tracking error globally. However, the approach, which suffers from the problem of "explosion of terms", is limited to systems in strict feedback form (Swaroop et al., 2000). Several works based on this control method have been successfully developed and tested on quadrotors (Madani and Benallegue, 2007; Rashad, Aboudonia, and El-Badawy, 2015).

Adaptive Control

Adaptive control provides many tools to tackle the autonomous quadrotors systems. The basic idea behind such control strategy is automatically adjusting the controller's parameters online based on the measured signals of the systems (Perez-Alcocer and Moreno-Valenzuela, 2019). In other words, when the parameters of the plant are unknown or change over time, the adaptive control requires to be considered for achieving and maintain the desired performance. It has the ability to adapt to compensate for the parameters variations of the quadrotor system especially in the presence of external signal disturbances. While compared with robust control strategies, adaptive control provides better performance for a large domain of uncertainty. However, this control technique has some limitations. The online estimation of parameters may not be an easy especially in case of cumbersome behaviors. In addition, the parameter estimation may have unwanted transient behavior in such a way guaranteeing system stability and achieving the required performance can be far from trivial (Giordano, Delamare, and Franchi, 2018). Figure 4.8 shows an example of the adaptive controller for a quadrotor.

Model Predictive Control

Model Predictive Control (MPC) is a famous control strategy that turns the control problem into an optimization problem. MPC is a crucial control method for handling multivariable control problems. It has been extensively applied on quadrotors (Seborg et al., 2016). The aim of MPC is to use the quadrotor dynamics model to estimate a sequence of future control values at any sampling time instant while minimizing the error. Only the first element of the estimated control sequence is utilized. The overall process is then repeated at the next sampling time. In such a way, the MPC approach predicts the future behavior of the closed-loop system (state and control input)

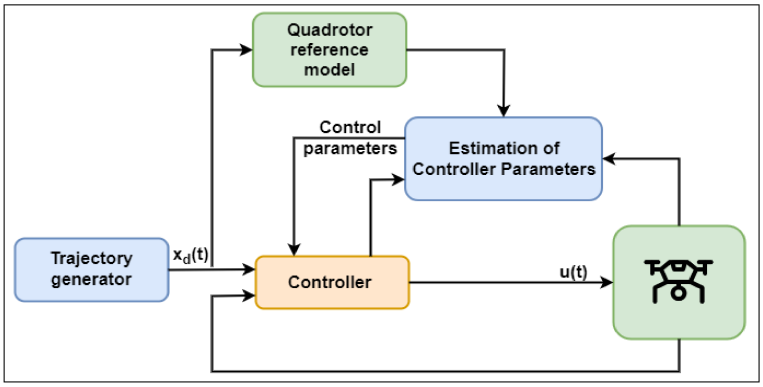

Figure 4.8: Control structure of an adaptive controller.

over a horizon of finite time. The strength of such approach is that it can handles multi-input multi-output (MIMO) systems having interactions between their inputs and outputs. In addition, MPC can handle systems at their optimized performance while satisfying some predefined constraints (Faulwasser and Findeisen, 2015). However just like adaptive control strategies, MPC performs optimization online. This may relatively require higher computational power.

4.3.3 Intelligent Trajectory Control Systems

The discipline of intelligent control describes control methods that endeavor to emulate important attributes of human intelligence. The attributes may include learning and adaptation, planning under the presence of large uncertainties and coping with tremendous data. Nowadays, this discipline attempts to cover everything that is not known as conventional control (Koutsoukos and Antsaklis, 1999). Intelligent control can be defined as a computationally efficient process to guide complex systems characterized with incomplete and inadequate representation and even under incomplete specifications in uncertain environment toward a certain objective. Intelligent control strategies have higher level of decision making scheme than conventional control ones. This leads to produce control signals based on qualitative (heuristic) understanding of the process (Gao, 1996). The main methods of intelligent control, which are used with trajectory control for quadrotors, include fuzzy logic and neural network. As suggested from the literature, the best performances of such intelligent control methods requires a combination with one or more of the aforementioned nonlinear control strategies to achieve higher robustness, adaptability, optimality, tracking ability, and disturbance rejection. Two well-known intelligent control systems are fuzzy logic and neural network.

Fuzzy logic control is a powerful problem-solving and well-known artificial intelligence control method that can be applied in several applications. Fuzzy logic control design are characterized by an easy implementation considering the fact that great performance can be obtained even if the system plant behavior is not realized. As a result, this type of control is applied with systems experiencing higher complexity and nonlinearity (Idrissi, Salami, and Annaz, 2022). Fuzzy logic controller contains mainly three parts: a fuzzifier, rule base and defuzzifier as shown in Figure 4.9. The dynamic behavior of the quadrotor can be the plant in the figure. The purpose of such control structure is to minimize the error as the systems evolves. The fuzzifier manages the numerical values of the plants and transforms them into class of fuzzy set members in such a way a decision

is made depending on the condition of each member. The defuzzifier, however, transforms the fuzzified output into numerical values that are fed back into the plant. The rule base is a control system that is responsible of making decisions in the form of if-then rules. These rules decide on the relationship between the input and the output variable based on the operator commands. Linguistic terms, which describe the set of member, are used to define such control rules such as NB (Negative Big), NS (Negative Small), Z (Zero), PS (Positive Small) and PB (Positive Big) (Zakariah et al., 2015).

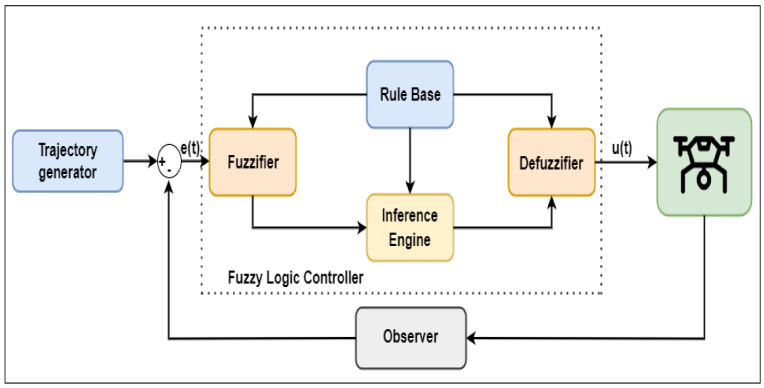

Figure 4.9: Fuzzy logic controller applied to a quadrotor.

Neural network control strategy has become a popular modern engineering control thanks to its ability to deal with intractable and complex nonlinear systems. It is inspired by the human central nervous system and uses an intelligent controller for adjusting itself online. Neural network control system tends to create nonlinear mappings between the inputs and the outputs of the system such that it allows for the quadrotor dynamic to be derived and a controller to be implemented based on a generated network of nodes and connections (Leondes, 1998). The purpose of developing such type of a controller is to urge the system states converge towards an equilibrium point by enforcing the feedback error converges to zero. Figure 4.10 shows an example of a sliding mode Artificial Neural Network (ANN) applied to a quadrotor. (Zakariah et al., 2015).

4.3.4 Geometric Control Systems

The linearization of both position and attitude models at the equilibrium point represent a great solution to preform quadrotor trajectory control. Although the attitude model linearization is effective in fixed-point hovering, in some cases it is not applicable especially to track time-varying reference trajectories. Consciously, it is crucial to study the nonlinear model of the quadrotor attitude. An appropriate visual representation of such model is to use the Euler angles. Hence, the nonlinear quadrotor model is generally expressed using position and Euler angles (Pizetta, Brandão, and Sarcinelli-Filho, 2019). Yet, it is commonly known that the Euler angle representation suffers from the singularities issues. That is, the quadrotor rotational matrix cannot be uniquely constructed at the singular point. Thus, it is necessary to bind the Euler angles with prior assumptions.

In order to overcome the above limitation, geometric control is introduced to deal with the quadrotor nonlinear model. Geometric control is the study of control systems evolving on non-Euclidean configuration manifolds. Besides, it can afford unique understanding of control theory

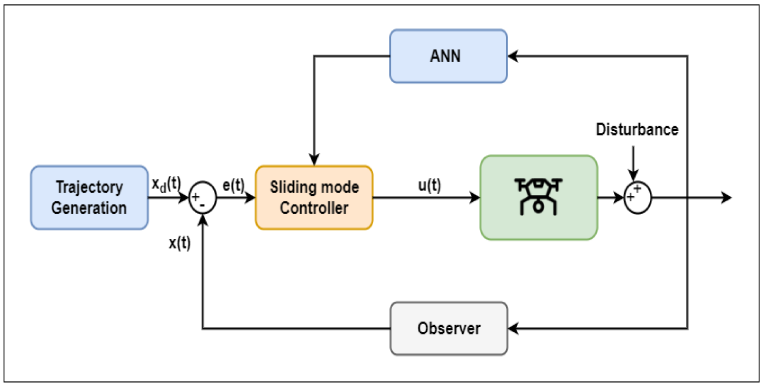

Figure 4.10: Sliding mode ANN control structure (Raiesdana, 2020).

that cannot be found from dynamic models constructed using local coordinates (Eslamiat et al., 2022). The geometric control strategy, which is capable of avoiding the singularity issue, uses the rotation matrices directly to design attitude controllers. The rotation matrices evolve on a smooth manifold which is the Lie group of 3×3 orthonormal matrices denoted $SO(3)$. The Special Orthogonal $SO(3)$ can well describe the rotation in the space and can be defined as

$$SO(3) = \{R \in \mathbb{R}^{3\times 3} | RR^T = I, det(R) = I\} \tag{4.3.1}$$

Where $RR^T = I$ is an orthogonality condition that imposes six constraints on the matrix with nine elements, thus making a matric with DOF of three. Every Lie group is associated with Lie algebra. Thus, The $SO(3)$ Lie group has an associated Lie algebra which is $\mathfrak{so}(3)$ (Eade, 2013). It represents the tangent space around the Lie group identity element and allows a complete characterization of the Lie group local properties. In other words, the Lie algebra is a vector space obtained by differentiating the group transformations along a given direction at the identity element. The Lie algebra $\mathfrak{so}(3)$ is defined as

$$\mathfrak{so}(3) = \{S(\boldsymbol{\phi}) = \boldsymbol{\Phi} = \begin{pmatrix} 0 & -\phi_3 & \phi_2 \\ \phi_3 & 0 & -\phi_1 \\ -\phi_2 & \phi_1 & 0 \end{pmatrix} \in \mathbb{R}^{3\times 3} \mid \boldsymbol{\phi} \in \mathbb{R}^3\} \tag{4.3.2}$$

Where $S(.)$ (equivalently $(.)^\wedge$) is the linear mapping that maps the vector $\phi in \mathbb{R}^3$ to $\mathfrak{so}(3)$. Φ is a skew-symmetric matrix. $SO(3)$ can be associated with $\mathfrak{so}(3)$ by the exponential mapping which is expressed by the following Rodrigues' formula.

$$R = exp(S(\boldsymbol{\phi})) = I_{3\times 3} + \frac{sin(||\boldsymbol{\phi}||)}{||\boldsymbol{\phi}||} S(\boldsymbol{\phi}) + \frac{1 - cos(||\boldsymbol{\phi}||)}{||\boldsymbol{\phi}||^2} S(\boldsymbol{\phi})^2 \tag{4.3.3}$$

Accordingly, the elements of $\mathfrak{so}(3)$ can be mapped to $SO(3)$ using the logarithmic mapping given by

$$S(\boldsymbol{\phi}) = log(R) = \begin{cases} 0, & \text{if } R = I_3 \\ \frac{\theta}{2sin\theta}(R - R^T), & \text{otherwise} \end{cases} \tag{4.3.4}$$

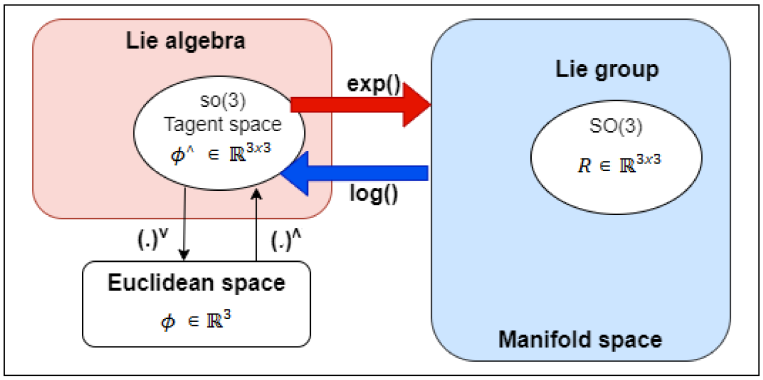

Figure 4.11: Schematic visualization of the different mappings.

Where θ is a rotation angle. Figure 4.11 visualizes the different mapping among $\mathbb{R}$, $\mathfrak{so}(3)$ and $SO(3)$.

In literature, several works attempted to design the aforementioned trajectory controllers inside the Lie group. One of the most cited works is presented in (Lee, Leok, and McClamroch, 2010). The authors introduced a geometric tracking control on the Special Euclidean group $SE(3) = \mathbb{R}^3 \times SO(3)$. The control completely avoids singularities that are commonly associated with Euler angle formulations. Other nonlinear geometric controllers have been also designed like PID (Goodarzi, Lee, and Lee, 2013), backstepping (Lee et al., 2013), and adaptive control (Goodarzi, Lee, and Lee, 2015).

4.4 Trajectory Generation for Quadrotors

Once the trajectory control algorithm is designed, the problem is reallocated to the higher level of the navigation task assimilated into trajectory generation or trajectory planning. Trajectory generation is an integral problem in motion control for vehicle systems like UAVs. It is the key solution for executing missions in cluttered environments. The trajectory planning algorithms generates a trajectory as a function of time that the UAV must follow for executing the assigned task. In order to do so, the planned trajectory must respect the thrust of the vehicle and its hardware limitations. Besides, it must also satisfy the constraints dictated by the vehicle control. In several applications, it is desirable to generate feasible trajectories such that the time required to execute the task or the energy consumption during the motion is minimized. Therefore, the trajectory generation problem is often regarded as an optimal control problem (LaValle, 2006).

An important criterion that has to be taken into consideration when designing trajectory generators is trajectory smoothness, in other words, trajectory having appropriate continuity features (continuous velocity, accelerations and jerks). The purpose behind generating smooth trajectories is to avoid the vehicle structural resonance and lessen the stresses to its actuators. In this way, smoothness will aid execute fast and aggressive motions for a quadrotor while assisting the actions of the controller. In addition, it is required to maintain the sensor measurements quality. For instance, motion blur in the images can be reduced using smooth vehicle motions. The smooth trajectories, which can be tracked by quadrotors quickly and accurately, must be C^3 *i.e.*, at least

continuous up to the third position derivative. The reason is that discontinuities in lateral acceleration necessitate instantaneous variations in the vehicle's attitude, discontinuities in lateral jerk necessitate variations in the vehicle's angular velocity as well.

In addition to the trajectory smoothness criterion, a trajectory generator should fulfill other requirements such as heading or pointing directions and corridor constraints. As stated in the previous chapter, path planners produce jagged paths and supply distinct way-points. The trajectory generator uses these way-points and generates feasible trajectories by taking into account the vehicle dynamic constraints (if any) and tackling optimal motion. The trajectory generation approaches can be divided into: continuous geometrical curve-based methods, optimal control-based methods and differential flatness based methods.

4.4.1 Continuous Geometrical Curved-Based Methods

The geometrical curved-based algorithms can be regarded as primarily geometric. The process of generating a trajectory comprises a geometric path generation, first. Thereafter, the path is time parameterized while satisfying the quadrotor dynamic constraints. The curves are characterized by special features to produce smooth trajectories such as continuous acceleration profile. As a pioneer, Dubins curves are often generated joining straight lines and arcs of circles in order to relate a set of way-points. By selecting the radius of the circle to be sufficiently large, these curves satisfy always upper bounds on curvature. Dubins showed in 1957 that for a forward (only) moving vehicle, the curve linking two assigned way-points with bounded curvature $\kappa \leq \bar{\kappa}$ is formed of segments and circles of radii $1/\bar{\kappa}$ (Dubins, 1957). Even though, Dubins guaranteed the shortest curve between two configurations; however, at the joints of segments and circular arcs, curvature continuity is not fulfilled. Another tool to generate smooth trajectory curves is polynomials. The trajectory generator approximates the desired curve way-points by a class of polynomial functions and produces a time-based control sequence from an initial configuration to a destination. Additional information such as position, velocity and acceleration constraints with duration of motion and are required. Cubic polynomials, which are expressed by Equation 4.4.1 are mostly used for quadrotors. However, higher order polynomials, which can permit to satisfy different states and inputs constraints, can be used.

$$x(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \tag{4.4.1}$$

Subjected to

$$\begin{cases} x(0) = x_0, \\ x(t_f) = x_f, \\ \dot{x}(0) = 0, \\ \dot{x}(t_f) = 0, \end{cases} \tag{4.4.2}$$

Splines are also means for planning curves for quadrotor vehicles. A spline can be defined as a piecewise polynomial curve whose segments are required to be polynomials of the same degree n (normally third-degree). Splines fulfill both the parametric (C^n continous) and the geometric continuities. The curvature of the spline is restrained using points called knots or control points. It is worth mentioning that the knots positions are decisive on the segments shape without any erratic behavior or breaks in continuities. However, the optimal computation of these positions is very time-consuming unless a set of cases are identified a priori (Judd and McLain, 2001). A Bézier curve is an alternative tool that uses control points for generating trajectories. Bézier curves generate continuous-curve trajectories by passing through the initial and the final way-points as the

whole trajectory remains within the convex hull defined by the knots. This states that the entire trajectory, except for the endpoints, will be inside a computable region. By definition, a Bézier curve $P(s)$ of degree n is a parametric curve, obtained from $n+1$ control points $(P_0, \ldots, P_n)$ is defined

$$P(s) = \sum_{i=0}^{n} B_i^n(s) P_i \quad s \in [0,1] \tag{4.4.3}$$

Where $B_i^n(s)$ is the i^{th} Bernstein polynomial of degree n, given by

$$B_i^n(s) = \binom{n}{i} s^i (1-s)^{n-i} \quad i \in \{0, 1, ..., n\} \tag{4.4.4}$$

Figure 2.21 shows examples of the different geometrical curve-based.

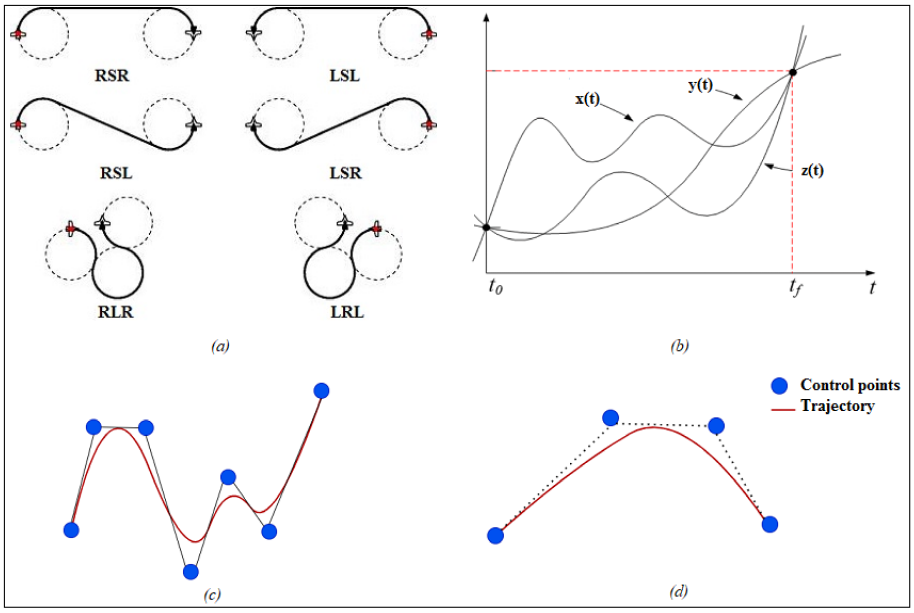

Figure 4.12: Geometrical curves: (a) Dubin (R: Right, L: Left, S: Straight (b) polynomial (c) splines (d) Bézier.

4.4.2 Optimal Control-Based Methods

Although the geometrical curve-based methods mentioned before are popular thanks to their polynomial natural choices for providing smooth, continuous motion and easy manipulation, differentiation and implementation, however, caution must be taken when higher order polynomials are utilized because of stability issues that can arise. Besides in case of aerial vehicles, problems related to dynamic feasibility and optimality can occur. A dynamically feasible trajectory can be defined as the one that the vehicle controls are capable of executing. For instance, sending acceleration commands that are beyond the aircraft's thrust capabilities is futile. As a result, the trajectory is

impossible to be tracked. On the other hand, optimality is also essential and to be desired even though it is rarely achieved in practice.

Quadrotors trajectory generation can be regarded as an optimization problem where some performance criteria (e.g., trajectory time, fuel expenditure, snap ... etc) have to be optimized (minimized) while fulfilling the physical and environment constraints. This means obtaining the best solution among all feasible solutions of the problem below

$$\begin{aligned} \min_{x} \quad & f(x) \\ \text{s.t.} \quad & g_i(x) \leq 0 \quad i = 1, ...n \\ & h_i(x) = 0 \quad i = 1, ...n \end{aligned} \tag{4.4.5}$$

Where $f(x)$ is the objective function to minimize over the variable x, the functions $g_i(x)$ are the inequality constraints and $h_i(x)$ are the equality constraints. Therefore, the aim here is to find the optimal value $x*$ such that $f(x*)$ has the smallest value among all the admissible solutions $f(x)$. Many algorithm methods have been implemented for solving the optimization problem above. Various classes of optimization problems have been developed. The problem is referred to as Linear Programming (LP) if the objective and the constraints functions satisfy the linearity property (Ignizio and Cavalier, 1994):

$$f(\alpha x + \beta y) = \alpha f(x) + \beta f(y) \tag{4.4.6}$$

For all $x, y \in \mathbb{R}^n$ and all $\alpha, \beta \in \mathbb{R}$. Thus, LP can be formulated as follow

$$\begin{aligned} \min_{x} \quad & c^T x \\ \text{s.t.} \quad & a_i^T x \leq b_i \quad i = 1, ...n \end{aligned} \tag{4.4.7}$$

Where $a \in \mathbb{R}^n$ and $b \in \mathbb{R}$. Sometimes the cost function and/or one of the constraints are not linear, this is referred to as nonlinear problem.

There exist two categories of optimization problems; convex and non-convex. Convex optimization refers to the type of problems where the cost and constraint function are convex and satisfies the convexity condition in (2.4). Solving the convex optimization problem leads to obtain an optimal solution which is global (Boyd, Boyd, and Vandenberghe, 2004).

$$f(\alpha x + \beta y) \leq \alpha f(x) + \beta f(y) \tag{4.4.8}$$

for all $x, u \in \mathbb{R}^n$ and all $\alpha, \beta \in \mathbb{R}$ with $\alpha + \beta = 1, \alpha \geq 0, \beta \geq 0$.

In the contrary, the non-convex optimization contains at least one non-convex function. Solving non-convex optimization problem is harder since it leads to an optimal solution that is local. Besides, several local minima solutions may be produced. It can be seen from equations 4.4.6 and 4.4.8 that linear programming is a convex optimization problem. Quadratic Programming (QP) is one of the recurrent convex optimization problems. QP is expressed as a quadratic function as the cost function with linear constraint functions as shown below

$$\begin{aligned} \min_{x} \quad & \frac{1}{2} x^T Q x + p^T x \\ \text{s.t.} \quad & Ax \leq b \\ & Cx = d \end{aligned} \tag{4.4.9}$$

where $Q \in \mathbb{R}^{n\times n}$, $Q \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$. The trajectory optimization problem in 4.4.5 can be a very difficult problem to solve. Algorithms used to solve the problem may take numerous iterations and function evaluations. The optimization problem has been solved using many methods which can be divided into: state-space, indirect and direct methods.

The state-space method uses the principle of optimality of sub-arcs. The generated trajectory is decomposed into sub-trajectories where each sub-trajectory of an optimal trajectory is considered as an optimal trajectory. In discrete time, this is referred as dynamic programming. However in continuous time, this is called the Hamilton-Jacobi-Bellman (HJB) equation. This latter solves the trajectory optimization problem by solving a significant number of sub-problems. Examples of such problem are grid-based path planning like A^*, Dijkstra ... etc (See Chapter 3). In the sate-space method, a global solution of the optimization problem is obtained even for case of non-convex optimization problem. However, the main drawback of such method is that the complexity of the global optimization increases exponentially with respect to the state-space dimension. This drawback is known as "curse of dimensionality" (Bellman and Dreyfus, 2015).

The indirect approaches employ the Pontryagin's Maximum Principle (PMP) which gives necessary conditions that the control and the state require to satisfy. PMP transforms the trajectory generation problem into an augmented Hamiltonian system of equations: either Ordinary Differential Equation (ODE) or Differential Algebraic Equation (DAE). The indirect approaches consider the trajectory generation as an Optimal Control Problem (OCP) of the form

$$\begin{aligned} \min_{x\in\mathcal{X},u\in\mathcal{U}} \quad & \int_0^T L(x(t),u(t))dt + E(T) \\ \text{s.t.} \quad & \dot{x}(t) = f(x(t),u(t)), \quad \forall t \in [0,T] \\ & h(x(t),u(t)) \geq 0, \quad h_0(x_0) \geq 0, \quad h_T(x_T) \geq 0, \\ & r(x(t),u(t)) = 0, \quad r_0(x_0) = 0, \quad r_T(x_T) = 0, \end{aligned} \tag{4.4.10}$$

Where the decision variable are the trajectory in state $\underline{x} : t \in [0,T] \rightarrow \mathrm{x(t)} \in \mathcal{X}$ and in control $\underline{u} : t \in [0,T] \rightarrow \mathrm{x(t)} \in \mathcal{U}$ (where $\mathcal{X}$ and $\mathcal{U}$ are the state and the control spaces, and the underlined symbol is used to differentiate the trajectory from the time value). L represents the integral (or running) cost, E is the terminal cost, f is the system dynamics and h and r functions represent arbitrary constraints. The Hamiltonian of the problem above is expressed as: beginequation

$$\mathcal{H}(x(t),u(t),(t),t) = L(x(t),u(t)) + \lambda^T f(x(t),u(t)) \tag{4.4.11}$$

While the necessary conditions of optimality are given by:

$$\begin{cases} 0 = \frac{\partial \mathcal{H}(x(t),u(t),(t),t)}{\partial u} \\ \dot{x}(t) = \frac{\partial \mathcal{H}(x(t),u(t),(t),t)}{\partial \lambda} \\ \dot{\lambda}(t) = -\frac{\partial \mathcal{H}(x(t),u(t),(t),t)}{\partial x} \\ \lambda(T) = \frac{\partial E(T)}{\partial x(T)} \end{cases} \tag{4.4.12}$$

The resulting Hamiltonian system of equations offers a complete and relatively cheap solution to the problem. However, it is unfortunately often too hard to be integrated as is due to non-linearity and the control structure complexity. Hence, the indirect approach is usually used on specific systems where the differential equations are simplified enough in order to be integrated (Böhme et al., 2017).

The direct approaches directly solve a discretized approximation of the original problem using numerical optimization techniques. The discretized approximation transforms the problem into a

Nonlinear Programming (NLP). Such approaches are advantageous since they work directly on the problem so there is no need of reformulation of the problem and solved by generic NLP solver packages (Geisert and Mansard, 2016). discretization of the original optimal problem leads to have a redundancy in the set of decision variables, the state $\underline{x}$ and control input $\underline{u}$ constrained by the system dynamics. Three methods are used to handle such redundancy: single shooting, collocation and multiple shooting methods. The direct single shooting, which is often called the control parameterization method, is the most widely used method for solving OCPs thanks to its simplicity. It integrates the whole trajectory from parameterized control variables using ODE/DAE solvers (Tassa, Erez, and Todorov, 2012). The cost function is calculated according to the integrator that solves the system dynamics differential equations. Although its simplicity, the co-state information cannot be provided using such method. Hence, the optimality of the solution is still questioned. The direct collocation method approximates both the states and the controls using a set of basic functions such as splines and exploits the properties of such functions to simplify the integration computations. In the direct collocation, the constraints on the dynamics on intermediate points named collocation points such that the solution of the OCP is required to satisfy the optimality conditions at these points. The direct multiple shooting method is similar to the direct single shooting, however comprises the state as decision variables in NLP at certain shooting points. Besides, it lifts the OCP to a higher dimension so that the optimal solution convergence is improved (Andersson, kesson, and Diehl, 2012). Nevertheless, this causes the NLP gets much larger.

4.4.3 Differential Flatness Based Methods

Solving the quadrotor trajectory generation problem in high dimensional space is challenging because of the under-actuation property of such aerial vehicle system. Both the trajectory control and generation can be simplified using the vehicle differential flatness property which makes the trajectory design easier. By definition, a differentially flat system is a system where all its states and inputs can be expressed as algebraic function of a set of outputs and derivatives (Ferrin et al., 2011). More precisely, a nonlinear system:

$$\dot{x} = f(x, u) \quad x \in \mathbb{R}^n, u \in \mathbb{R}^m \tag{4.4.13}$$

is termed flat if there is $\sigma \in \mathbb{R}^m$, of the form

$$\gamma = \xi(x, u, \dot{u}, ..., u^r) \quad \xi : \mathbb{R}^n \times (\mathbb{R}^m)^{r+1} \to \mathbb{R}^m \tag{4.4.14}$$

such that

$$x = \phi_x(\gamma, \dot{\gamma}, ..., \gamma^l) \quad \phi_x : (\mathbb{R}^m)^r \to \mathbb{R}^n \tag{4.4.15}$$

$$u = \phi_u(\gamma, \dot{\gamma}, ..., \gamma^l) \quad \phi_u : (\mathbb{R}^m)^{r+1} \to \mathbb{R}^m \tag{4.4.16}$$

where ξ, ϕ_x and ϕ_u are smooth functions and γ is the flat outputs.

The differential flatness property can reduce the algebra of the trajectory generation problem, in theory, and the algorithms computations in practice. In case of quadrotors, the 12 state space dimension can be decreased to a 4-dimentional space in which solving the dynamics 4.4.13 is not necessary. The importance of a system differential flatness is that all system dynamics can be expressed without integrating by the flat outputs and its derivatives. Figure 2.22 depicts that the problem of finding the curves that steer the system from an initial state/input $(x(0), u(0))$ to a final state/input $(x(T), u(T))$ is reduced to finding any sufficiently smooth curve that satisfies $\gamma^k(0)$ and

$\gamma^k(T)$ up to some finite number l. Hence, in case of differentially flat system, solving a Two-Point Boundary Value Problem (TBVP) is not required. After mapping all boundary conditions and trajectory constraints into the flat output space, the optimal trajectories can be generated in the flat output space and then elevated back to the state/input space. The differential flatness of the

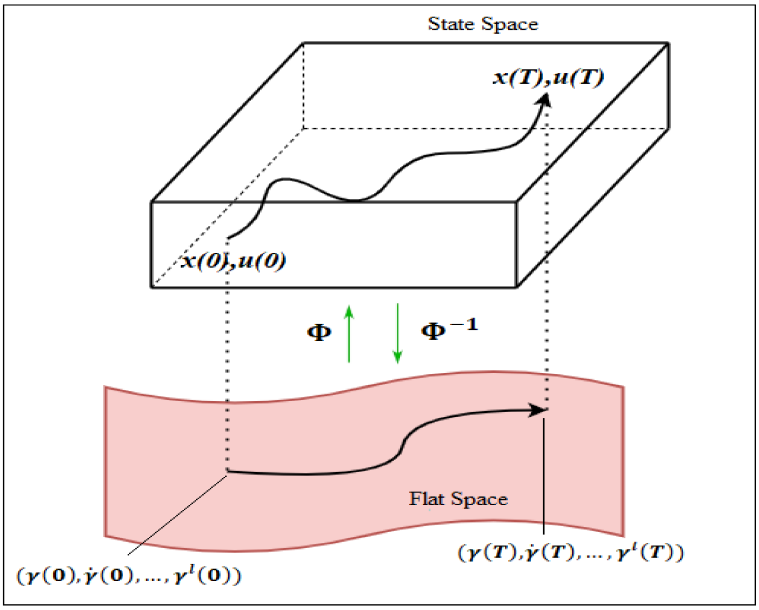

Figure 4.13: Differential flatness property and its mapping.

quadrotor system has been used in several applications. It is applied in (Mellinger and Kumar, 2011) to generate feasible polynomial trajectories that numerically minimize the vehicle snap. The generated trajectories aid the quadrotor to fly quickly through static and moving hoops. The concept of differential flatness is also applied in (Thomas et al., 2015) (Mellinger, Michael, and Kumar, 2012) to successfully perch the quadrotor on inclined surfaces where a geometric trajectory controller, which is developed in (Lee, Leok, and McClamroch, 2010), is used. The authors in (Yu, Cai, and Wang, 2016) use the concept to implement a trajectory generator, which generates minimum jerk trajectories, coupled with a nonlinear controller.

4.5 Conclusion

The aim of this chapter was on trajectory generation and control which are the second and the third stages of the quadrotor autonomous navigation. At the beginning, we described the dynamical model of the quadrotor used in this thesis. Then, we presented the different strategies used for solving the trajectory tracking problem and classified them into: linear, nonlinear, intelligent and geometric control systems. Most of these strategies are also used for trajectory generation. In addition to these, we reported other strategies mainly continuous geometrical, optimal control and differential flatness based methods.

Chapter 5

Quadrotor Autonomous Navigation: Methodology

5.1 Introduction

In the previous chapters, an extensive literature review about path planning, trajectory control and trajectory generation for case of quadrotors was presented. Besides, several approaches have been proposed to solve the quadrotor autonomous navigation problem using one or more of the aforementioned element(s). The approaches were categorized into mainly two structures: coupling structure and decoupling structure. The coupling structure combines essentially the planning and the control elements, however the decoupling one does not. To the best of our knowledge, none of these proposed approaches has presented a complete solution using all elements for the navigation problem.This chapter is devoted to the methodology that will be followed in order to create efficient and safe 3D navigation missions for a single quadrotor. In addition, mathematical modelling of each method of each element that builds up the navigation solution will be developed.

5.2 Global Autonomous Navigation Structure

The problem of 3D autonomous navigation for aerial vehicles is very challenging. In general, direct solutions do not exist; if they do, they are inefficient, complex and time consuming. Bearing this in mind, the problem can be decomposed into multi-phase sub-problems: path planning, trajectory generation, trajectory re-planning and trajectory control. A decoupling structure can be developed by provide a solution to each sub-problem. First, because of the numerous frequent challenges of each sub-problem, a feasibility screening among the developed solution technologies is conducted. After that, an accurate selection among those currently available is performed and exploited here for the first time in precision agriculture scenarios. The purpose of this selection process is to identify the best combination of sub-problem solutions which could result in suitable performances in terms of low computational requirements, reduced designed time, and optimal trajectory planning and tracking capabilities for fully 3D aerial vehicle navigation.

The proposed autonomous navigation approach in Figure 5.1 falls under the decoupling structure number 7 (See Figure 2.22 in Chapter 2). An offline definition of the optimal trajectory, which is

typically a two-consecutive step, is used. The first makes use of a representation of the environment to generate safe geometric paths. The purpose of this step is to locate reference way-points (or local goals). The second uses the extracted local goals to generate global trajectories that comply with position, velocity and acceleration constraints at these way-points. To do so, a Linear Quadratic Regulator (LQR) trajectory generator is developed. When unknown obstacles interfere the motion, a trajectory re-planning strategy directs the quadrotor to move away from them. An online Artificial Potential Field (APF) planner is used for keeping the vehicle safe from collisions. In order to track the generated trajectories while simultaneously pointing towards a predefined direction, a real-time geometric controller is constructed.

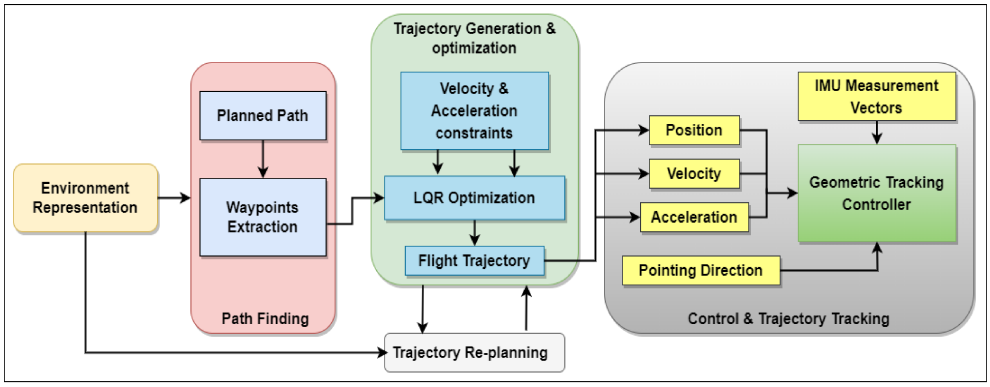

Figure 5.1: An overview of the navigation global structure.

5.3 Aerial Terrain Mapping

As stated in the previous chapters, Mapping is the process of modelling a robot's environment. Maps are consistent environment models that can be constructed using observations from the robot's onboard sensors. Robots use the constructed maps to locate themselves and make motion plans. Aerial mapping can be carried out using UAVs to construct maps of fields and terrains. In other words, the UAVs are capable of acquiring high accuracy information about the terrain geometry and the existence of static and dynamic obstacles. Accordingly, aerial mapping is an essential feature for autonomous vehicles to perform navigation in unknown environments.

This work adopts two main products of aerial terrain mapping: Occupancy Grid Maps (OGMs) and Digital Elevation Maps (DEMs). OGMs are well-known in robotics for storing the obstacle information for the purpose of planning safe and stable geometric paths. They discretize the 2D environment with fixed resolution into independent cells and each cell is associated with a binary variable estimating either it is free or occupied (eg., 1 for free space and 0 for obstacle space). OGMs are used in this work thanks to their simplicity, ease to maintain and their ability to guarantee planning stable and safe geometric paths. DEMs are another type of grid maps in 3D that exhibit the relief of the terrain in digital format at regularly spaced horizontal intervals. They are organized as square grid of columns and rows where each grid point refers to the height at that location. The methodology to follow in this work is to use OGMs to extract 2D coordinates along the x- and y-axes of the planned paths then use the DEMs to extract the paths last coordinate along the z-axis.

5.4 Optimal Path Finding

Optimal path searching is the first step of trajectory generation. It uses the OGMs to generate collision-free passable geometric guiding paths. The quality of the initial paths in terms of stability and safety affects the quality of the final flight path of the quadrotor. In this work, two scenarios are treated. Scenario ① considers a point-to-point flight mission in an environment that is free of obstacles. In this way, any point-to-point path planning algorithm can be chosen to effectively plan the geometric paths since the configuration space is obstacle-free. Scenario ②, however, considers a coverage flight mission in an environments filled with several obstacles. An Iterative Structured Orientation Algorithm (ISOA) is adopted as in (Horvath, Pozna, and Precup, 2018). The reason behind choosing such an algorithm is that it makes use of OGMs and generates optimal coverage paths by using the approach of main lines. These latter are a beam of parallel lines with an orientation in the OGM. Straight lines with maximum length bordered by the map and interrupted by the obstacles are guaranteed by the orientation. By connecting the beam of lines, optimal continuous coverage paths are achieved.

Given a modeled OGM $M = [M_{ij}]_{i=1..n, j=1..m} \in \mathbb{R}^{n \times m}$ as shown in Figure 5.2 where n is number of horizontal pixels and m is the number of vertical pixels with

$$M_{ij} = \begin{cases} 1, & if\, pixel(i,j)\, white \\ 0, & if\, pixel(i,j)\, black \end{cases} \tag{5.4.1}$$

The problem of finding an optimal path can be summarized as fellow

1. Find the appropriate beam of parallel lines and their orientation;
2. Get the main segments;
3. Connect these segments with the auxiliary segments to form a continuous optimal coverage path.

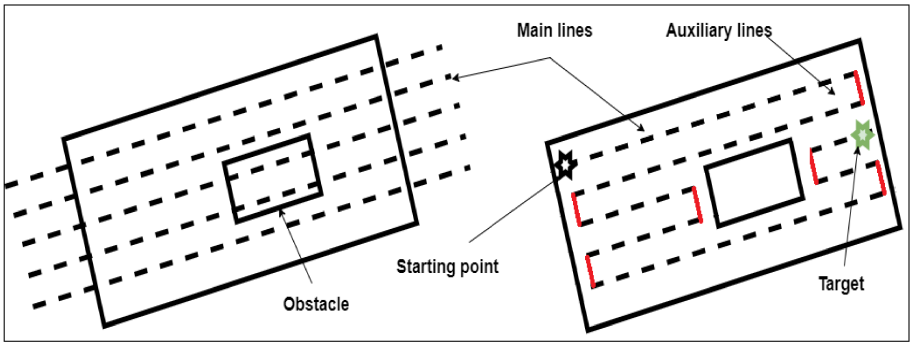

Figure 5.2: The map with both main lines (black) and auxiliary lines (red).

The geometric path is then used to extract the 2D local goals. Finally, the coordinate along the z-axis is added to these local goals by the use of the DEMs. The result of this step is a set of positions p_i in $\mathbb{R}^3$, be identified by an index set way-points $W = \{1, 2, ..., N\}$ required for trajectory generation. The flowchart of the process is shown in Figure 5.3.

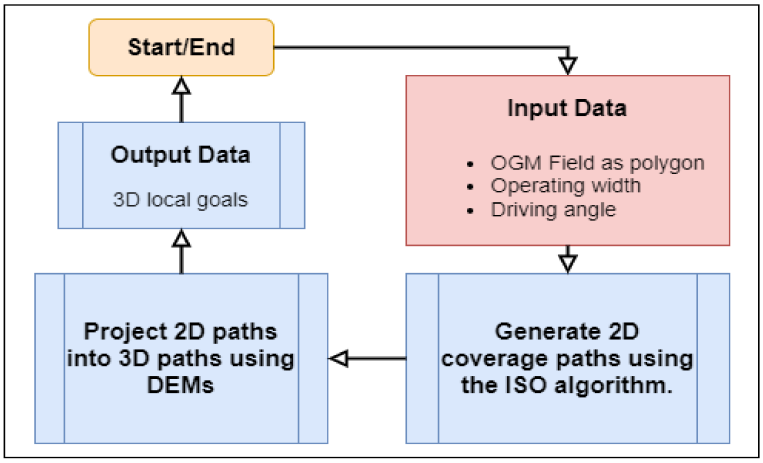

Figure 5.3: The 3D local goals extraction flowchart.

5.5 Trajectory Generation and Optimization

Although the geometric guiding path generated above is safe, it is inappropriate as the initial path since it does not consider the dynamic feasibility. Besides smoothness and temporal features should be considered. The aim of this section is compute feasible position trajectory for a quadrotor while following a collision-free path from a start point to a goal while minimizing a defined objective cost. To do so, the indirect approach, which employs the Pontryagin's Maximum Principle (PMP), is used in our case. Before computing the trajectories, certain constraints have to be taken into consideration depending on the flight mission requirements.

Trajectory generation should not only describe the desired trajectory accurately, but should also have smooth kinematics profiles in order to increase the precision and the durability of the system, maintain higher tracking accuracy while avoiding exciting natural modes of the mechanical structure or servo control system (Sencer and Ishizaki, 2015). To do so, the trajectory must satisfy certain constraints such as the vehicle's physical limits, safety regulations and sensor specifications.

5.5.1 Trajectory Generation Constraints

In this work, we adopt a method of decoupling these limits into constraints on the trajectory. The first constraint is clearly a position constraint to force the trajectory generated pass through all of the identified way-points. The second constraint depends on the flight mission scenario. In Scenario ①, a constraint on the vehicle's acceleration is used rather than on the velocity at the way-points as shown in Equation (5.5.1). The motivation behind this is to obtain trajectories that are smooth and with continuous curvatures. At each way-point, given a position vector p_i and a pointing direction vector s_i, expressed in $\mathcal{I}$ such that

$$(p_i, s_i) \in \mathbb{R}^3 \times \mathbb{S}^2 \quad \forall i \in W$$

Let t_i be the time required to reach the i^{th} way-point. The problem of trajectory generation is posed on $\mathbb{R}^3 \times \mathbb{S}^2$. The problem can be posed on $SE(3)$ by defining a vector which is orthogonal to

s_i. Let us define a unit vector q_i such that

$$q_i = \frac{e_3 - (e_3^T s_i)s_i}{||e_3 - (e_3^T s_i)s_i||} \tag{5.5.1}$$

such that $q_i \perp s_i$. Let us now define the quadrotor's acceleration constraints be given as follow

$$a_i = \frac{f}{m} q_i - g e_3 \tag{5.5.2}$$

assuming a nominal thrust magnitude $f \in \mathbb{R}$.

In Scenario ②, a flight corridor is satisfied using constraints on both the velocity and the acceleration. For the way-points at the coverage path corners, the velocities should be slowed down to zero while a maximum velocity is applied to other way-points. However, the acceleration should be always brought down to zero at every way-point.

5.5.2 LQR Optimal Control Trajectory Generation

An LQR approach is adopted in this work to generate position, velocity and acceleration trajectories that satisfy the mentioned constraints above. The trajectory generator treats these constraints at the way-points to be soft. This may lead to have relaxation on satisfying the way-point constraints exactly while providing a stable technique of trajectory generation. This latter can handle a great number of way-points to within desirable tolerance.

The system state $x(t)$ is described by concatenating the quadrotor's position, velocity, acceleration and jerk as follows

$$x(t) = [b(t) \quad \dot{b}(t) \quad \ddot{b}(t) \quad \dddot{b}(t)]^T \tag{5.5.3}$$

Where *b(t)* is the position of the quadrotor in the 3D environment at instant *t*. Using the position and three of its derivative in the state $x(t)$ guarantees that the control torque calculated using Equation 4.2.4 is at least C^3 continuous. The output *y(t)* is constructed such that the desired output y_i at the way-points is given by either (5.5.4) for Scenario ① or (5.5.5) for Scenario ②

$$y_i = [p_i \quad a_i]^T \tag{5.5.4}$$

Where a_i is expressed as in (5.5.1).

$$y_i = [p_i \quad v_i \quad a_i]^T \tag{5.5.5}$$

Where v_i and a_i are the velocity and the acceleration constraints of the vehicle at the way-points, respectively.

Problem: *Given: (1) a set of way-points* $p_i \in \mathbb{R}^3 \, \forall i \in W = \{1, 2, ..., N\}$*, and (2) a set of velocity* v_i*'s and acceleration* a_i*'s constraints at the way-points* p_i*'s, find the position* $b(t)$*, the velocity* $v(t)$ *and the acceleration* $a(t)$ *trajectories for* $\forall t \in [t_0, t_N]$ *such that*

$$min \quad J = \sum_{i=1}^{N} (y(t_i) - y_i)^T S (y(t_i) - y_i) + \int_{t_0}^{t_N} \frac{1}{2} (x^T Q x + u^T R u) dt \tag{5.5.6}$$

Subjected to the constraint equations:

$$\dot{x}(t) = Ax + Bu \tag{5.5.7}$$

$$y(t) = Cx(t) \tag{5.5.8}$$

Where $A = \begin{bmatrix} 0_{9\times3} & I_{9\times9} \\ 0_{3\times3} & 0_{3\times9} \end{bmatrix}$, $B = \begin{bmatrix} 0_{9\times3} \\ I_{3\times3} \end{bmatrix}$, $C = \begin{bmatrix} I_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & I_{3\times3} & 0_{3\times3} \end{bmatrix}$ *for Scenario ①, or* $C = \begin{bmatrix} I_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & I_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & I_{3\times3} & 0_{3\times3} \end{bmatrix}$ *for Scenario ②.* $I_{n\times n}$ *is the identity matrix of dimension* n, t_0 *is the starting time,* t_N *is the final time and the control input* u *is the snap.*

We are able to solve the problem using the PMP's principle discussed in Chapter 4. The augmented objective function J' is

$$J = \sum_{i=1}^{N} (y(t_i) - y_i)^T S(y(t_i) - y_i) + \int_{t_0}^{t_N} \frac{1}{2} x^T Q x + \frac{1}{2} u^T R u + \lambda^T (Ax + Bu - \dot{x}) dt \tag{5.5.9}$$

Where $\lambda \in \mathbb{R}^{12}$ is the Lagrangian multiplier or co-state that incorporates the constraint (5.5.7) to the cost function in (5.5.6). Using 4.4.11, the Hamiltonian $\mathcal{H}$ is expressed as

$$\mathcal{H} = \frac{1}{2} x^T Q x + \frac{1}{2} u^T R u + \lambda^T (Ax + Bu) \tag{5.5.10}$$

Then, the necessary conditions of optimality expressed in 4.4.12 are as follows

$$0 = \frac{\partial \mathcal{H}}{\partial u} = Ru + B^T \lambda \tag{5.5.11}$$

$$\dot{x}(t) = \frac{\partial \mathcal{H}}{\partial \lambda} = Ax + Bu \tag{5.5.12}$$

$$\dot{\lambda}(t) = -\frac{\partial \mathcal{H}}{\partial x} = -Qx - A^T \lambda \tag{5.5.13}$$

$$0 = \frac{1}{2} \frac{\partial (y(t_i) - y_i)^T S(y(t_i) - y_i)}{\partial x(t_i)} + \lambda(t_i^+) - \lambda(t_i^-) \tag{5.5.14}$$

Where t_i^+ is the time instant t_i when approached from times $t > t_i$ and t_i^- s the time instant t_i when approached from times $t < t_i$. The co-state can be expressed as $\lambda(t) = P(t)x(t) + \eta(t)$. The Optimal control $u^*(t)$ in Equation (5.5.11) is given by (the time t is dropped for simplicity):

$$u = -R^{-1} B^T (Px + \eta) \tag{5.5.15}$$

Differentiating the co-state equation $\lambda(t)$ yields

$$\dot{\lambda} = \dot{P}x + P(Ax + Bu) + \dot{\eta} \tag{5.5.16}$$

Replacing the control u in (5.5.16) and equating it with Equation (5.5.13) gives

$$-Qx - A^T Px - A^T \eta = \dot{P}x + PAx - PBR^{-1}B^T Px - PBR^{-1}B^T \eta + \dot{\eta} \tag{5.5.17}$$

By rearranging the above equation, we get

$$0 = (Q + A^T P + PA - PBR^{-1}B^T P + \dot{P})x + (-PBR^{-1}B^T + A^T)\eta + \dot{\eta} \tag{5.5.18}$$

The above equation is true if and only if

$$\begin{cases} 0 = (Q + A^T P + PA - PBR^{-1}B^T P + \dot{P})x \\ 0 = (-PBR^{-1}B^T + A^T)\eta + \dot{\eta} \end{cases} \tag{5.5.19}$$

In this way we can get

$$\begin{cases} \dot{P} = -A^T P - PA - Q + PBR^{-1}B^T P \\ \dot{\eta} = (-A^T + PBR^{-1}B^T)\eta \end{cases} \tag{5.5.20}$$

Replacing the co-state equation and $y_i = Cx_i$ in Equation (5.5.14), it yields

$$0 = \frac{1}{2}\frac{\partial (Cx(t_i) - Cx_i)^T S(Cx(t_i) - Cx_i)}{\partial x(t_i)} + P(t_i^+)x(t_i^+) + \eta(t_i^+) - P(t_i^-)x(t_i^-) + \eta(t_i^-) \tag{5.5.21}$$

Rearranging the above equation as follows

$$0 = C^T SCx(t_i) - C^T SCx_i + P(t_i^+)x(t_i^+) + \eta(t_i^+) - P(t_i^-)x(t_i^-) + \eta(t_i^-) \tag{5.5.22}$$

If we assume that $x(t_i) = x(t_i^+) = x(t_i^-)$ for small step, we get

$$0 = (C^T SC + P(t_i^+) - P(t_i^-))x(t_i) - C^T Sy_i + \eta(t_i^+) - \eta(t_i^-) \tag{5.5.23}$$

This is equivalent to say

$$\begin{cases} 0 = C^T SC + P(t_i^+) - P(t_i^-) \\ 0 = -C^T Sy_i + \eta(t_i^+) - \eta(t_i^-) \end{cases} \tag{5.5.24}$$

Or

$$\begin{cases} P(t_i^-) = P(t_i^+) + C^T SC \\ \eta(t_i^-) = \eta(t_i^+) - C^T Sy_i \end{cases} \tag{5.5.25}$$

However, at $i = N$

$$\begin{cases} P(t_N) = C^T SC \\ \eta(t_N) = -C^T Sy_i \end{cases} \tag{5.5.26}$$

Equations (5.5.20) are solved backward in time starting at $t = t_N$ and at every t_i where $i < N$; the boundary conditions are updated using Equation (5.5.25). In this way, the solutions of Differential Riccati Equations are obtained. These solutions are used in Equations (5.5.7) and (5.5.8) to solve the system dynamics. As a result, both the state vector $x(t)$ and the control input vector $u(t)$ are determined $\forall t \in [t_0, t_N]$.

5.6 Online Trajectory Re-planning

An online trajectory re-planner module is added in order to have an efficient complete solution for the navigation problem. The objective of this module is to cope with the unmodeled obstacles to maximize the safety of the quadrotor. The trajectory re-planner runs in several milliseconds to keep the vehicle close to the global path while simultaneously avoids unexpected obstacles. The polynomials, sampling-based and optimization methods discussed in Chapters 3 and 4 can be utilized to build the re-planning module.

In our work, an Improved APF method is used to perform the online trajectory re-planning (Fan et al., 2020). The quadrotor takes the current position as the starting point when an unknown obstacle is detected. It takes also the next turning point in the global path planning as the target point. An attractive and repulsive potential fields are constructed at the goal and around the obstacles, respectively. The joint action of the attractive and repulsive forces give assistance to the quadrotor to move toward the target. In contrast to the conventional APF, the improved APF solves both problems: the goal nonreachable with obstacles nearby (GNRON) and the local minima. The former problem is solved by adding a distance correction factor to the repulsive potential field function (See Figure 5.4(a)). The latter one is solved by the regular hexagon-guided (RHG) method that constructs a virtual regular hexagon helping the vehicle to escape from difficult positions as shown in Figure 5.4(b). The Improved APF is used because its model is simple and elegant. Besides, it is widely applicable for real-time implementation.

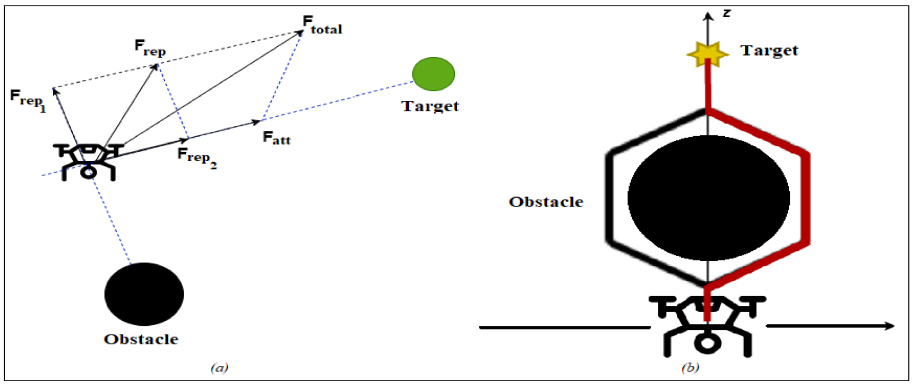

Figure 5.4: The improved APF method: (a) the improved resultant force model for solving the GNRON problem, (b) the RHG for solving the local minima problem.

Firstly, an attractive potential field is constructed at the target using the following expression:

$$U_{att}(P) = \frac{1}{2} k_{att}.d^2(P, P_g) \tag{5.6.1}$$

Where $d(P, P_g) = P_g - P$ is the Euclidean distance between the vehicle's position and the target. k_{att} is the attractive potential field constant. The attractive force of the quadrotor in the attractive potential field is the negative gradient of U_{att}:

$$F_{att}(P) = -\nabla U_{att}(P) = k_{att}.d(P, P_g) \tag{5.6.2}$$

Secondly, an improved repulsive potential field is constructed around the obstacles and can be defined as:

$$U_{rep}(P) = \begin{cases} \frac{1}{2}(\frac{1}{d(P,P_o)} - \frac{1}{d_o})^2 d^n(P, P_g) & d(P, P_o) \leq d_o \\ 0 & d(P, P_o) > d_o \end{cases} \tag{5.6.3}$$

Where P_o is the position of the obstacle, d_o is influence range of the repulsive potential field and n is an arbitrary real number which is greater than zero (for our case $n = 2$). $d^n(P, P_g)$ is the distance between the current quadrotor position and the target. The repulsive potential field is

called improved since the distance correction factor $d^n(P, P_g)$ is added to the conventional repulsive potential field known in literature. This creates a balance between the two kind of forces especially in case where a rapid increase in the repulsive force occurs. In this manner, the repulsive force can be decreased gradually when the quadrotor is adjacent to the target. Most important of all, this force ensures that the overall potential field at the target is the global minimum. Thus, the GNRON related problem is solved. Figure 5.4(a) depicts the resultant of the improved repulsive force F_{rep} in Equation (5.6.4). The direction of this force is not on the line between the quadrotor and the obstacle as in the conventional APF. The improved repulsive force has two components F_{rep1} and F_{rep2} expressed by Equations (5.6.5). The direction of F_{rep1} is on the line between the quadrotor and the obstacle, hence it makes the vehicle move away from the obstacle. However, the direction of F_{rep2} is on the line between the vehicle and the target. F_{rep2} guides the quadrotor to move toward the target.

$$F_{rep}(P) = -\nabla U_{rep}(P) = \begin{cases} F_{rep1}(P) + F_{rep2}(P) & d(P, P_o) \leq d_o \\ 0 & d(P, P_o) > d_o \end{cases} \tag{5.6.4}$$

Where F_{rep1} and F_{rep2} are expressed as

$$\begin{cases} F_{rep1}(P) = k_{rep}(\frac{1}{d(P,P_o)} - \frac{1}{d_o})\frac{d^n(P,P_g)}{d^2(P,P_o)} \\ F_{rep2}(P) = \frac{n}{2}k_{rep}(\frac{1}{d(P,P_o)} - \frac{1}{d_o})^2 d^{n-1}(P, P_g) \end{cases} \tag{5.6.5}$$

The total force $F(P)$ is given by

$$F_{total}(P) = F_{att}(P) + F_{rep1}(P) + F_{rep2}(P) \tag{5.6.6}$$

Figure 5.5 depicts a flowchart that explains the whole re-planning process. P_c refers to the current position of the quadrotor.

5.7 Trajectory Control and Tracking

Once the optimal trajectory is generated as previously described, it is necessary to design a control strategy to track the reference trajectory. In this work, a geometric controller strategy, that uses the quadrotor dynamics expressed globally on the Special Euclidean $SE(3)$ configuration manifold, is constructed to track predefined trajectories. Instead of using the heading and the thrust directions, which are mostly applied in literature to construct the vehicle attitude, inertial measurement vectors that can be provided by a sensor are used. In this way, new definitions of the errors between the real vectors and the desired ones are used (See Equations (5.7.6)-(5.7.7)). These errors are injected directly in the control law. Besides, the notion of a pointing direction is introduced instead of the heading direction. The controller structure is shown in Figure 5.6.

The quadrotor translational dynamics is controlled using the total thrust fRe_3. The magnitude of the total thrust f is directly controlled and its direction Re_3 is along the third body-fixed axis b_3. Hence, to obtain stabilized translational motion along a desired trajectory, the total thrust f and a desired direction of b_{3d} are selected. A direction is required to complete the degrees of freedom of the desired attitude $R_d \in SO(3)$. Thus, a pointing direction vector s^c, which has to be corrected online using the desired direction b_{3d}, is chosen. The desired attitude is then obtained as $R_d = [b_{2d} \times b_{3d}, b_{2d}, b_{3d}]$ where $b_{2d} = s^c$. This desired attitude is followed by the control moment τ.

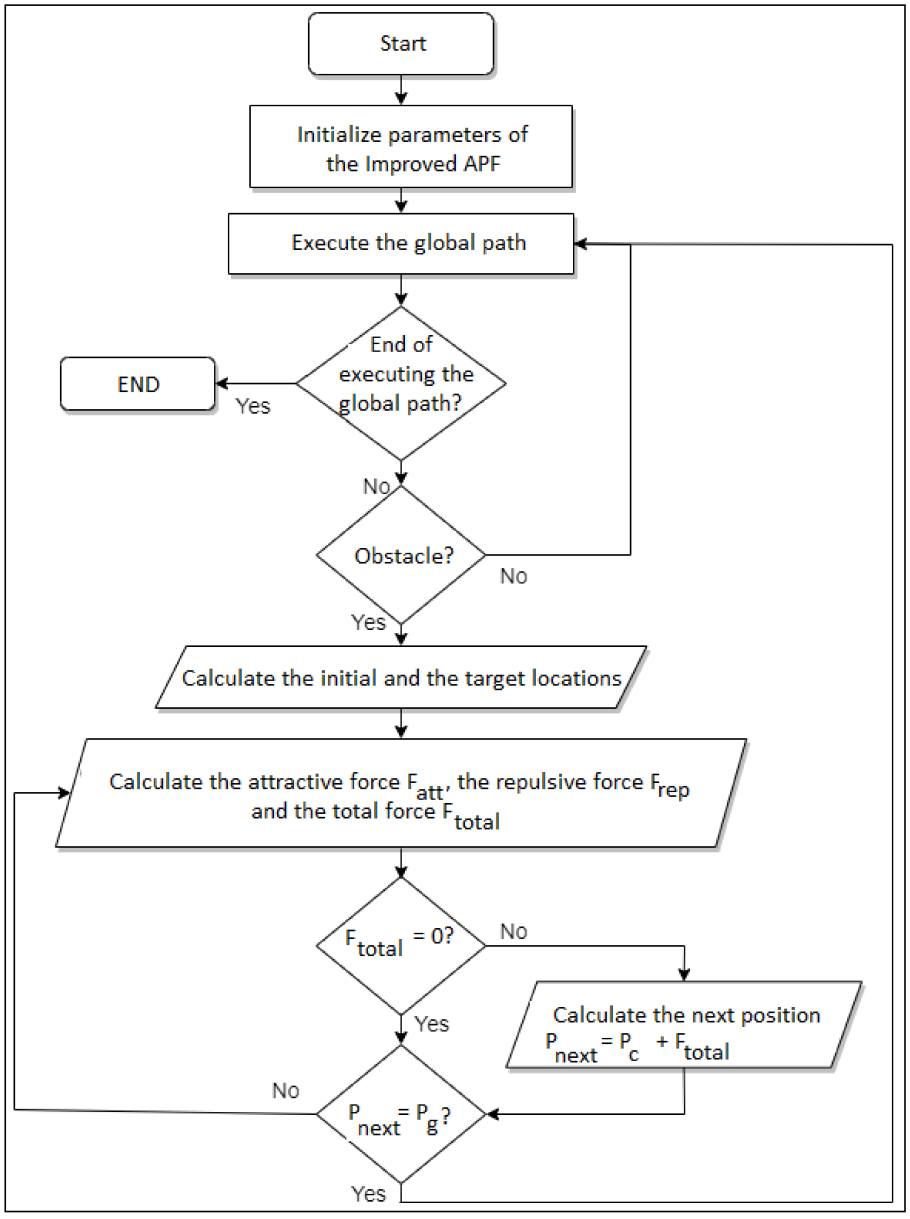

Figure 5.5: The flowchart of the proposed trajectory re-planning method based on the improved APF.

Section 5.5.2 results in an optimal desired position trajectories $x_d(t)$. The pointing directions

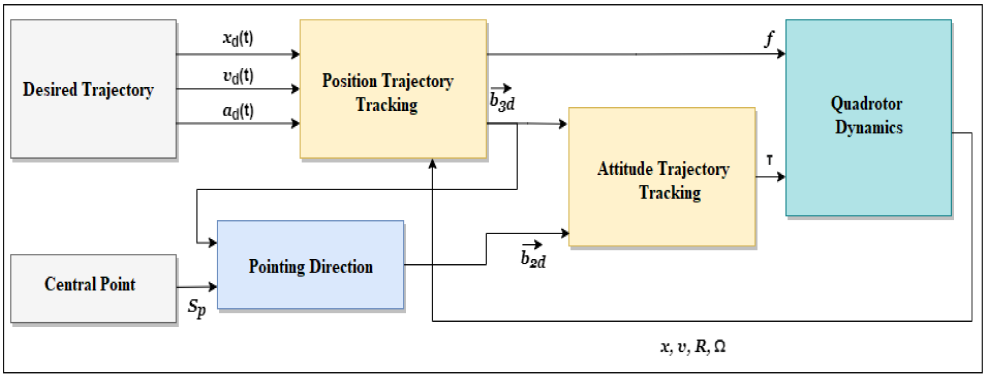

Figure 5.6: The structure of the geometric controller.

at every instant $t \in [t_0, t_f]$ are

$$s(t) = \frac{S_p - x_d(t)}{||S_p - x_d(t)||} \qquad \forall t \in [t_0, t_f] \tag{5.7.1}$$

Where S_p is a predefined centered point. The pointing directions can be corrected using the vector $b_{3d}(t)$ as shown in the expression below

$$s^c(t) = \frac{s(t) - (s(t)^T b_{3d}(t))b_{3d}(t)}{||s(t) - (s(t)^T b_{3d}(t))b_{3d}(t)||} \tag{5.7.2}$$

Where $s^c(t) = b_{2d}(t)$, and $s^c(t) \perp b_{3d}(t)$. The vector $b_{3d}(t)$ is expressed as

$$b_{3d}(t) = \frac{-k_x e_x - k_v e_v + m(ge_3 + a(t))}{||-k_x e_x - k_v e_v + m(ge_3 + a(t))||} \tag{5.7.3}$$

Where k_x, k_v are some positive constants, e_x and e_v are the tracking errors for the position $x(t)$ and the velocity $v(t)$, respectively. They are expressed as

$$\begin{cases} e_x = x - x_d \\ e_v = v - v_d \end{cases} \tag{5.7.4}$$

The vector $b_{1d}(t)$ can be produced as

$$b_{1d}(t) = b_{2d}(t) \times b_{3d}(t) \tag{5.7.5}$$

It is assumed that the quadrotor's attitude is unknown for measurements (unavailable for feedback). It is also assumed that it is equipped with sensors that provide unfiltered vector measurements (in the body-fixed frame). Hence, the only variables available are the vector measurements which are denoted by b_i.

Proposition: *Consider the rotational dynamics in (4.2.4). Define the following vector errors:*

$$\mathsf{z}_\rho = \sum_{i=1}^{k} \rho_i S(b_{id}) b_i \tag{5.7.6}$$

$$\begin{cases} e_{\Omega 1} = \Omega - \Omega_d \\ e_{\Omega 2} = \Omega_d \times \Omega \end{cases} \tag{5.7.7}$$

Where $\rho_i > 0$ and k is the number of measured vectors in the body frame ($k = 2$ in our case). The desired angular velocity is given by

$$\Omega_d = Vex([b_{1d}, b_{2d}, b_{3d}]^T [\dot{b}_{1d}, \dot{b}_{2d}, \dot{b}_{3d}]) \tag{5.7.8}$$

with $Vex(.) : \mathfrak{so}(3) \to \mathbb{R}^3$. The following control law

$$\tau = \Omega \times J\Omega + J\dot{\Omega}_d + J\mathsf{z}_\rho - Je_{\Omega 1} - Je_{\Omega 2} \tag{5.7.9}$$

guarantees Almost Global Asymptotic Stability (AGAS) of the body attitude and angular velocity to their desired values.

In order to prove the above proposition, the following lemma is used.

Lemma (Tayebi, Roberts, and Benallegue, 2013): *Assume that there are n vectors b_i, $i = 1, ..., n$ measured in the body attached frame, corresponding to n known inertial vectors r_i, $i = 1, ..., n$. Assume that the constant parameters γ_i are strictly positive and at least two vectors among the r_i vectors are non-collinear. Then, the following properties hold:*

- *The vector z_γ satisfies*

$$\mathsf{z}_\gamma \equiv \sum_{i=1}^{n} \gamma_i S(\hat{b_i}) b_i = -2\hat{R}^T(\tilde{q}_0 I - S(\tilde{q}))W_\gamma \tilde{q}, \tag{5.7.10}$$

  where the matrix $W_\gamma = -\sum_{i=1}^{n} \gamma_i S(r_i)^2$ is real symmetric and positive definite.

- *If the gains γ_i, $i = 1, ..., n$, are chosen such that W_γ has two by two distinct eigenvalues, the following holds true:*

  $\mathsf{z}_\gamma = 0$ is equivalent to ($\tilde{q}_0 = \pm 1, \tilde{q} = 0$) or ($\tilde{q}_0 = 0, \tilde{q} = \pm \mathsf{v}$), where v is a unit eigenvector of W_γ

Proof of Lemma: In order to prove the lemma above, we are going to use quaternions.

(1) Given the inertial vectors $r_i's$, they are related to the measured vectors $b_i's$ using the rotations R. Hence, $b_i = R^T r_i$. Equivalently, $\hat{b}_i = \hat{R}^T r_i$. Using the property $S(\hat{R}^T r_i) = \hat{R}^T S(r_i)\hat{R}$, we can write

$$\mathsf{z}_\gamma \equiv \sum_{i=1}^{n} \gamma_i S(\hat{b_i}) b_i = \hat{R}^T \sum_{i=1}^{n} \gamma_i S(r_i) \tilde{R}^T r_i \tag{5.7.11}$$

The rotation matrix error $\tilde{R} = R\hat{R}$ is re-written in terms of the unit quaternions and by using the properties $S(r_i)r_i = 0$ and $S(\tilde{q})r_i = -s(r_i)\tilde{q}$, z_γ can be written as

$$\mathsf{z}_\gamma = 2\hat{R}^T \sum_{i=1}^{n} \gamma_i S(r_i)(\tilde{q}\tilde{q}^T - \tilde{q}_0 S(\tilde{q}))r_i = -2\tilde{R}^T(S(\tilde{q})M_\gamma + \tilde{q}_0 W_\gamma)\tilde{q}, \tag{5.7.12}$$

Where $M_\gamma = -\sum_{i=1}^{n} \gamma_i r_i r_i^T$. Since $S^2(r_i) = r_i r_i^T - r_i^T r_i I$, it can be shown that

$$M_\gamma = \sum_{i=1}^{n} \gamma_i r_i^T r_i I - W_\gamma \tag{5.7.13}$$

Substituting (5.7.13) in (5.7.12), we get (5.7.10).

(2) For a given vector $\mathsf{y} \in \mathbb{R}$, we have

$$-\mathsf{y}^T S(r_i)^2 \mathsf{y} = \mathsf{y}^T (r_i^T r_i)\mathsf{y} - \mathsf{y}^T r_i r_i^T \mathsf{y} = ||r_i||^2 ||\mathsf{y}||^2 - (\mathsf{y}^T r_i)^2 \geq 0 \tag{5.7.14}$$

This shows that $-S(r_i)^2 \geq 0$. By assuming that r_1 and r_2 are non-collinear, it yields

$$\mathsf{y}^T W_\gamma \mathsf{y}^T = -\mathsf{y}^T (\sum_{i=1}^{n} \gamma_i S(r_i)^2)\mathsf{y} = -\gamma_1 \mathsf{y}^T S(r_1)^2 \mathsf{y} - \gamma_2 \mathsf{y}^T S(r_2)^2 \mathsf{y} + \mathsf{y}^T \mathfrak{E} \mathsf{y} \tag{5.7.15}$$

Where $\mathfrak{E} = -\sum_{i=3}^{n} \gamma_i S(r_i)^2 \geq 0$.

- $\mathsf{y}^T S(r_1)^2 \mathsf{y}$ is equivalent to $S(r_1)\mathsf{y} = r_1 \times \mathsf{y} = 0$, which is verified for all $\mathsf{y} \neq 0$, iff y and r_1 are collinear.
- Similarly for $\mathsf{y} \neq 0$, $\mathsf{y}^T S(r_2)\mathsf{y} = 0$ iff y and r_2 are collinear.
- r_1 and r_2 are non-collinear, they can not be collinear to the same y, hence $-\gamma_1 \mathsf{y}^T S(r_1)^2 \mathsf{y} - \gamma_2 \mathsf{y}^T S(r_2)^2 \mathsf{y} > 0$ for all $\mathsf{y} \neq 0$.

Therefore from Equation (5.7.15), $\mathsf{y}^T W_\gamma \mathsf{y}^T > 0$ for all $\mathsf{y} \neq 0$. This means that W_γ is a positive definite matrix.

(3) If $\mathsf{z}_\gamma = 0$, it is equivalent to say that

$$(\tilde{q}_0 I - S(\tilde{q}))W_\gamma \tilde{q} = 0 \tag{5.7.16}$$

From the above equation, it can be noticed that $(\tilde{q}_0 = \pm 1, \tilde{q} = 0)$ is the trivial solution. The other solution can be obtained by assuming $\tilde{q} \neq 0$ and multiply the Equation (5.7.16) by $\tilde{q}^T$, we get $\tilde{q}_0 \tilde{q}^T W_\gamma \tilde{q} = 0$. This shows that $\tilde{q}_0 = 0$ is a solution since W_γ is positive. Now, if $\tilde{q}_0 = 0$, we get

$$S(\tilde{q})W_\gamma \tilde{q} = 0 \tag{5.7.17}$$

Since W_γ is non-singular, the equality (5.7.17) is satisfied iff. $W_\gamma \tilde{q} = \lambda \tilde{q}$ for any scalar λ. This shows that $\tilde{q} = \pm \mathsf{v}$, where v is one of the unit eigenvectors of W_γ. □

Proof of Proposition: It is given that $\mathsf{z}_\rho = \sum_{i=1}^{k} \rho_i S(b_{id}) b_i$. Using the lemma, we can write

$$\mathsf{z}_\rho \equiv \sum_{i=1}^{k} \rho_i S(b_{id}) b_i = -2R_d^T (q_0^e I - S(q^e)) W_\rho q^e, \tag{5.7.18}$$

Where $W_\rho = -\sum_{i=1}^{k} \rho_i S(r_i)^2$ is a symmetric positive definite matrix according to the lemma. The tracking error $R^e = R R_d^T$ of the attitude that correspond to the unit quaternion errors

$$Q^e = Q \odot Q_d^{-1} \equiv (q_0^e, q^e) \tag{5.7.19}$$

Where $Q \in \mathbb{S}^3 = \{Q = (q_0, q) \in \mathbb{R} \times \mathbb{R}^3 | q_0^2 + q^T q = 1\}$. Given the attitude dynamics in (4.2.4) and the control law (5.7.9), if we consider $\bar{\omega} = R_d e_{\Omega 1}$, where $e_{\Omega 1}$ is given in Equation (5.7.7), the closed loop dynamics are obtained as

$$\dot{Q}^e = \begin{bmatrix} \dot{q}_0^e \\ \dot{q}^e \end{bmatrix} = \begin{bmatrix} -\frac{1}{2}(q^e)^T \bar{\omega} \\ \frac{1}{2}(q_0^e I + S(q^e))\bar{\omega} \end{bmatrix} \tag{5.7.20}$$

$$\dot{\bar{\omega}} = -\alpha\bar{\omega} - 2(q_0^e I - S(q^e))W_\rho q^e \tag{5.7.21}$$

Note that these dynamics are autonomous. Defining $\mathsf{x} = (Q^e, \bar{\omega})$ in the state space $\mathcal{X} := \mathbb{S}^3 \times \mathbb{R}^3$, the above dynamics can be written in the form

$$\dot{\mathsf{x}} = f(\mathsf{x}) \tag{5.7.22}$$

We can also notice that (5.7.21) can be written as

$$\dot{\bar{\omega}} = -\alpha\bar{\omega} + R_d z_\rho \tag{5.7.23}$$

Theorem: *Consider the rigid body dynamics (4.2.4) with the control law (5.7.9) resulting in the closed loop attitude dynamics given by (5.7.20)-(5.7.21). Then under assumptions of the lemma, the trajectories of (5.7.20)-(5.7.21) converges to the following subsets $\mathbb{S}^3 \times \mathbb{R}^3$, given by $\Theta_1 = (\pm 1, 0, 0)$ and $\Theta_2 = \{(0, \pm\mathsf{v}_1, 0), (0, \pm\mathsf{v}_2, 0), (0, \pm\mathsf{v}_3, 0)\}$, where v_i $(\mathsf{i} = 1, 2, 3)$ are unit eigenvectors of W_ρ.*

- *The equilibrium set Θ_1 is asymptotically stable with the domain of attraction containing $\Phi = \{\mathcal{X} := (Q^e, \tilde{\omega}) \in \mathbb{S}^3 \times \mathbb{R}^3 | \mathcal{X}^T P \mathcal{X} < c\}$ with $P = diag(0, 2W_\rho, \frac{1}{2})$ and $c < 2\lambda_{min}(W_\rho)$ and $\lambda_{min}(*)$ is the smallest eigenvalue of $(*)$*
- *The equilibria defined by the set Θ_2 are unstable and Θ_1 is almost globally asymptotically stable.*

Proof of Theorem: Let us propose the Lyapunov function candidate

$$V = 2(q^e)^T W_\rho q^e + \frac{1}{2}\bar{\omega}^T\bar{\omega} \tag{5.7.24}$$

With W_ρ symmetric positive definite. The time derivative of equation (5.7.24) in view of (5.7.20) and (5.7.21) is expressed as

$$\dot{V} = 2(q^e)^T W_\rho (q_0^e I + S(q^e))\bar{\omega} + \bar{\omega}^T(-\alpha\bar{\omega} - 2(q_0^e I - S(q^e))W_\rho q^e) \tag{5.7.25}$$

$$\dot{V} = 2\bar{\omega}^T(q_0^e I - S(q^e))W_\rho q^e + \bar{\omega}^T(-\alpha\bar{\omega} - 2(q_0^e I - S(q^e))W_\rho q^e) \tag{5.7.26}$$

After simplification, we obtain

$$\dot{V} = -\alpha\bar{\omega}^T\bar{\omega} \tag{5.7.27}$$

It can be noticed in (5.7.27) that $\dot{V} \leq 0$. Besides from the same equation, we have $\dot{V} = 0$ only $\bar{\omega} = 0$. Using the Equation (5.7.23) and $\bar{\omega} = 0$, we get $z_\rho = 0$. According to the lemma, this leads to the equilibrium sets $(q_0^e = \pm 1, q^e = 0, \tilde{\omega} = 0)$ or $(q_0^e = 0, q^e = \pm\mathsf{v}_\mathsf{i}, \tilde{\omega} = 0)$ which corresponds to Θ_1 and Θ_2, respectively. It remains to show that Θ_2 is unstable to complete the proof.

Let us define $\delta \equiv (q^e)^T\bar{\omega}$, and consider the dynamics of q_0^e and around $(q_0^e = 0, \bar{\omega} = 0)$ which corresponds to $(q_0^e = 0, \delta = 0)$

$$\dot{q}_0^e = -\frac{1}{2}\delta \tag{5.7.28}$$

$$\dot{\delta} = -\alpha\delta - 2\eta q_0^e \tag{5.7.29}$$

Where η is an eigenvalue of W_ρ. Hence, the system expressed by (5.7.28)-(5.7.29) can be written as follows

$$\dot{\mathsf{X}} = \begin{bmatrix} 0 & -\frac{1}{2} \\ -\alpha & -2\eta \end{bmatrix} \mathsf{X} = A\mathsf{X} \tag{5.7.30}$$

With $\mathsf{X} = \begin{bmatrix} q_0^e & \eta \end{bmatrix}^T$ and $A = \begin{bmatrix} 0 & -\frac{1}{2} \\ -\alpha & -2\eta \end{bmatrix}$. The two eigenvalues of A are real and of opposite sign. The characteristic equation of A is given by $P(\lambda) = \lambda^2 + 2\eta\lambda - \frac{\alpha}{2}$. Hence, the roots are $\lambda_{1,2} = -\eta \pm \sqrt{\eta^2 + \frac{\alpha}{2}}$. Since the equilibrium $(q_0^e, \eta) = (0, 0)$ is unstable, it can be concluded that Θ_2 is unstable. □

Given the proofs above and according to the Krasowsky-La Salle theorem, the equilibrium is almost globally asymptotically stable in this case. Hence, the proposed control law (5.7.9) guarantees AGAS. □

5.8 Conclusion

In this chapter, we presented the methodology that would be followed to design our 3D autonomous navigation strategy for the case of a quadrotor. To do so, we presented the different methods that were used in each phase of the strategy from planning to tracking. Besides, we developed the mathematical modelling that would be followed to implement these methods. In the next chapter, we will show the effectiveness of the proposed strategy by simulating the each method.

Chapter 6

Simulation Experiments

6.1 Introduction

Deployment of quadrotors for real world agricultural tasks such as plant spraying, pesticide application, pruning, etc., requires robotic vehicles to fly through complex 3D environments such as forests, orchards, greenhouses, vineyards, etc. Decoupled structures of navigation can offer motion cues for autonomous flights in partially or completely known 3D agricultural environments. However, coupled structures do not 'look ahead' and tend to bounce off obstacles rather than generating motions to avoid these obstacles. As a result, the flight mission fails. The aim of this chapter is to validate the effectiveness of the decoupled structure proposed in the previous chapter. Results are obtained for a quadrotor flying in agricultural fields. This chapter is based off our published article: Guidance, Navigation and Control for Autonomous Quadrotor Flight in Agricultural Field: Case of Vineyards (Mokrane et al., 2022).

6.2 Experiment Setup

This section is devoted to simulation experiments to test the theory presented in Chapter 5 and to validate the effectiveness of the complete autonomous navigation solution proposed in this thesis.

6.2.1 Simulation Setup

Simulations were conducted to test whether or not the decoupled structure of autonomous navigation in Figure 5.1 would work as expected. The presented simulations can be used as a proof of concept before applying it to a real quadrotor. All simulation experiments were performed by Matlab running on an HP laptop with a 2.90 GHz Intel® i7-10700 CPU and 32 Gb of RAM. All integration operations were simulated using Matlab fixed-step Solvers, updated at a sample time of 1 ms. The simulation have been carried out using a quadrotor with a mass of $m = 4.34$ kg, and a moment of inertia of $J = diag(0.820, 0.0845, 0.1377)$ kgm^2. The quadrotor started at $t_0 = 0$ s at rest from a given position $b_0 \in \mathbb{R}^3$ i.e., $x(t_0) = [b_0, 0_{3\times1}, 0_{3\times1}, 0_{3\times1}]^T$ with an initial attitude, without loss of generality, aligns with the inertial frame $\{\mathcal{I}\}$. Therefore, the quadrotor$'$s initial pose was given by

$$G_0 = \begin{bmatrix} I_3 & b_0 \\ 0 & 1 \end{bmatrix} \tag{6.2.1}$$

6.2.2 Environmental Settings

The main purpose of this work is navigating the quadrotor in a given agricultural environment. As stated in Chapter 5, two scenarios were taken into consideration. Scenario ① and Scenario ② were dedicated for point-to-point and coverage navigation flights, respectively, to perform Remote Sensing (RS). Scenario ① was used simulate the navigation flight of the quadrotor while pointing the imager (or sensor) toward a predefined object such as a tree or a bush. Scenario ② was used to simulate coverage navigation for the quadrotor flying between plants growing in lines. For our case, vineyards terrains were selected to be our experimental environments. Vineyards is an example of an intricate, unstructured, and fickle agricultural fields which can be challenging for robotic machines like quadrotors. Vineyards terrains are known by their steep slopes and limited room for maneuvering. We wanted the quadrotor to fly at relatively constant altitude between the vines. 2D and 3D environment representations were created with Matlab. Scenario ① was built of a free space configuration only as shown in Figure 6.1. However in Scenario ②, static obstacles were

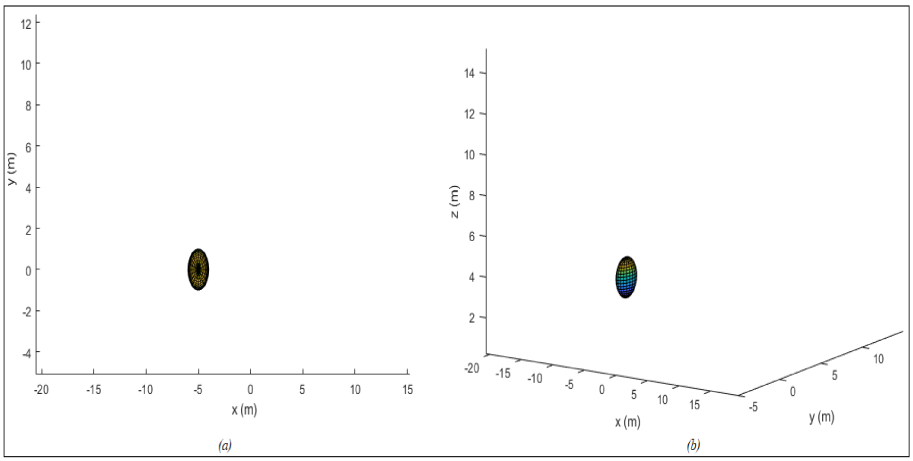

Figure 6.1: Environment representation of Scenario ① in (a) 2D and (b) 3D.

added. Since aerial imagery data was not available to build both real-life OGMs and DEMs, inflated artificial (generated) maps were used. The maps were hand-drawn using black and white colors then converted to binary occupancy maps. It was assumed that the environments presented as OGMs were ideal and known, i.e., there is only 0 and 1 in the map, representing free and occupied grids, respectively. Two vineyard terrain sub-scenarios were considered (See Figures 6.2 and 6.3). The first is a flat rectangle. The second is a tilted irregular hexagon.

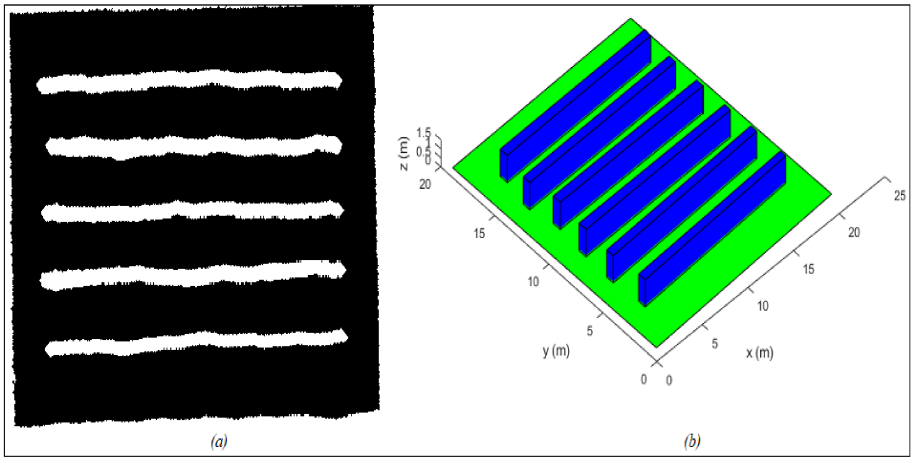

Figure 6.2: Environment representation of Scenario ② Terrain 1 in (a) 2D and (b) 3D.

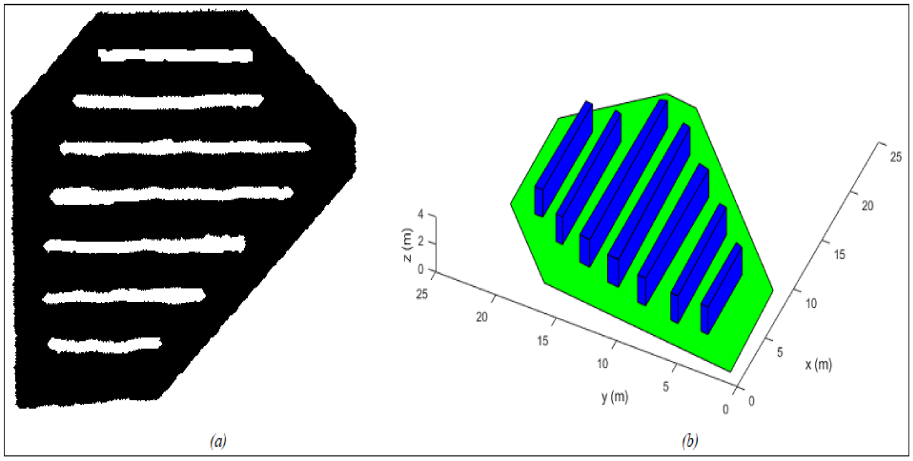

Figure 6.3: Environment representation of Scenario ② Terrain 2 in (a) 2D and (b) 3D.

6.3 Results & Discussions

6.3.1 Path Planning

In case of Scenario ①, the quadrotor was supposed to start from rest at the origin, i.e., $b_0 = 0_{3\times 1}$. Four way-points were specified. The way-points b_i were obtained from the expressions:

$$b_i = [4(\frac{i+2}{2})cos(\frac{\pi t_i}{20}) \quad 6(\frac{i+2}{2})sin(\frac{\pi t_i}{20}) \quad 0.6t_i] \tag{6.3.1}$$

where $t_i = 4i \;\; \forall i \in W = \{1, 2, 3, 4\}$. The results are shown in Figure 6.4.

In case of Scenario ②, The binary occupancy maps above were fed to Matlab where the ISOA coverage path planning algorithm was run. The distance between the main lines was initialized to

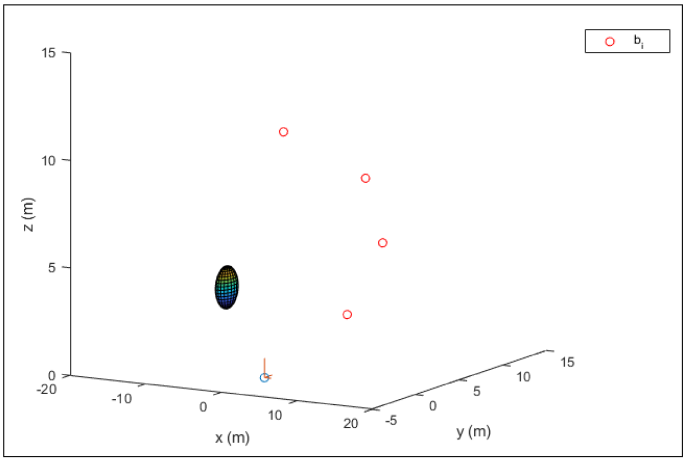

Figure 6.4: The generated local goals.

be $d = 2.5$ m. As a result, back and forth boustrophedon path was generated and way-points were extracted as explained in Section 5.4 as shown in Figure 6.5.

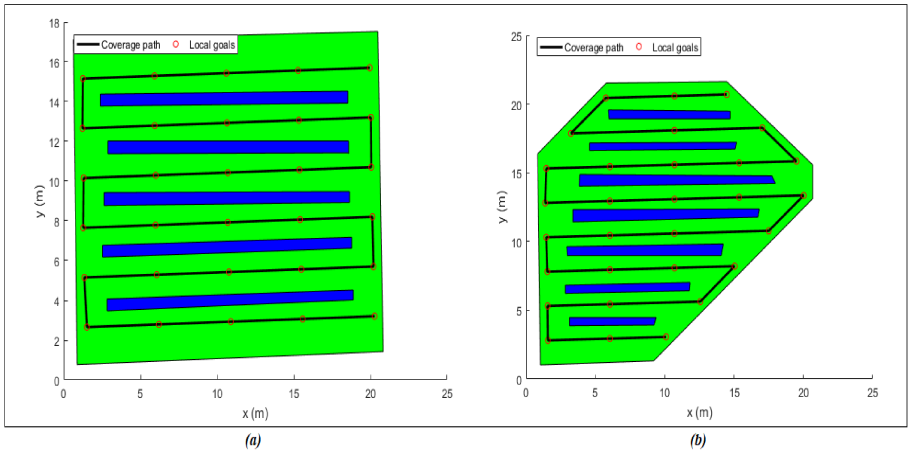

Figure 6.5: The generated coverage paths and way-points: (a) Terrain 1, (b) Terrain 2

6.3.2 Trajectory generation

The extracted way-points were not fed directly to the trajectory generator since they contained geometric information only, i.e., the position b_i. However, velocity/acceleration constraints were added to the way-points depending on the application. As stated in Section 5.5.1, only the acceleration constraint given by the Equation 5.5.2 was added in Scenario ①. Hence, s_i can be expressed

as

$$s_i = \frac{S_p - b_i}{||S_p - b_i||} \quad \forall i \in W \tag{6.3.2}$$

The pointing direction was the center of the object of the interest (a tree) and given by $S_p = [-5, 0, 4]^T$ m. The time required to reach the i^{th} way-point was fixed to be 4 seconds and the nominal force was given by

$$f = mg$$

The above information was used by the LQR trajectory generator with the weighting matrices below

$$Q = I_{12\times 12} \quad R = 10^2 * I_{3\times 3} \quad S = 10^4 * I_{6\times 6}$$

Figure 6.6(a) shows the resulting position trajectory maneuvering around the way-points and pointing towards the object of interest compared to the results in 6.6(b) obtained by the authors in (Dhullipalla et al., 2019). 6.6.

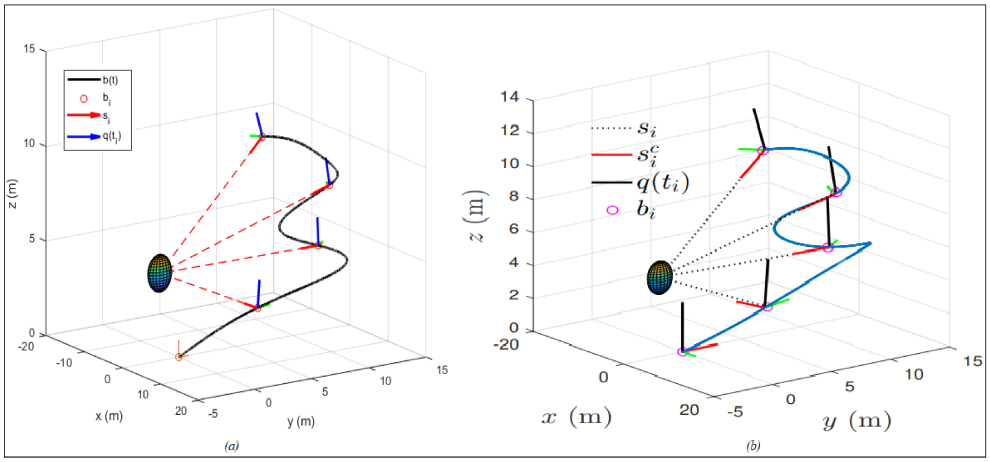

Figure 6.6: Trajectory on $SE(3)$ obtained by (a) our algorithm, (b) algorithm in (Dhullipalla et al., 2019).

In Scenario ②, a safe flight corridor (SFC) constrained the motion of the quadrotor between the vines. Constant velocity and acceleration constraints were imposed at the way-points as explained in Section 5.5.1. In our case, a velocity of $v = 1$m/s was chosen. The starting locations were $b_0 = [20.8, 1.4, 0]^T$ for Terrain 1 and $b_0 = [9.2, 1.325, 0]^T$ for Terrain 2. Again, this information was exploited by the LQR trajectory generator with the weighting matrices below

$$Q = I_{12\times 12} \quad R = I_{3\times 3} \quad S = 10^8 * I_{9\times 9}$$

The resulting position trajectories of both vineyard fields, maneuvering around the way-points, are shown in Figures 6.7 and 6.8 below.

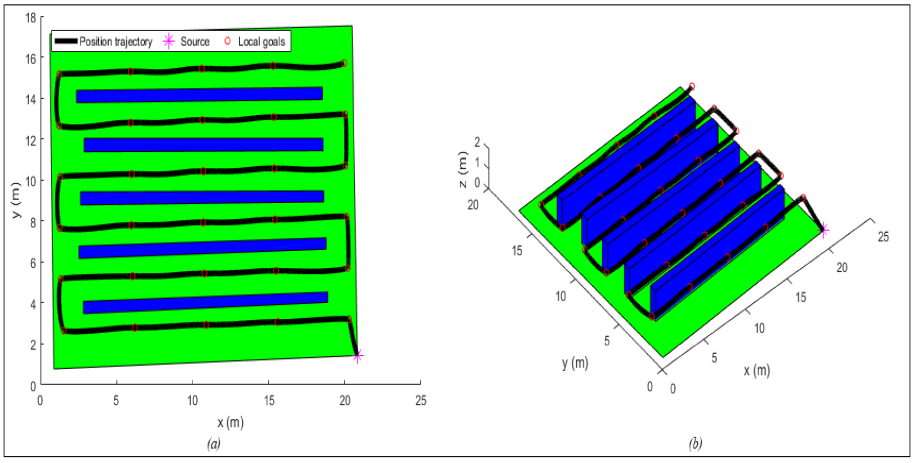

Figure 6.7: The generated optimal trajectory for Terrain 1: (a) 2D, (b) 3D.

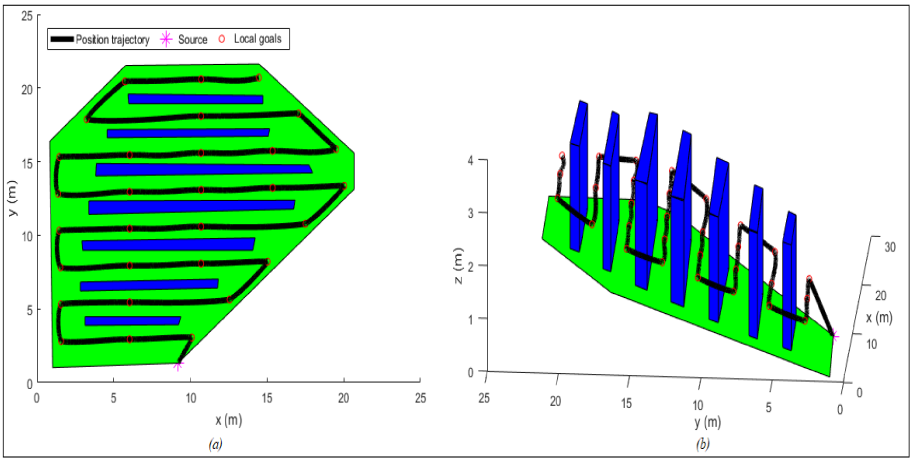

Figure 6.8: The generated optimal trajectory for Terrain 2: (a) 2D, (b) 3D.

6.3.3 Trajectory Control

The trajectory generator does not produce the position profile of the quadrotor only but even its velocity and acceleration profiles. These information were fed into the trajectory control in order to test its effectiveness. Without loss of generality, the object of interest was chosen to be the polygon centroid in case of Scenario ②, i.e., $S_p = [10.7, 9.2, 1]^T$ for Terrain 1 and $S_p = [9.3, 11.9, 2]^T$ for Terrain 2. Thus, the pointing direction was chosen to be expressed as in Equation 5.7.1 $\forall t \in [t_0, t_f]$. The control parameters were given by

$$k_x = 100 \quad k_v = 20 \quad \rho_i = 1$$

The control law in (5.7.9) contains the measurement vectors b_i and Ω. These vectors were considered to be noisy. A zero mean white noise with variance of 5.10^{-4} was added. The resulting noisy measurement vectors are shown in Figures 6.9 and 6.10. Figures 6.11, 6.12 and 6.13 show the

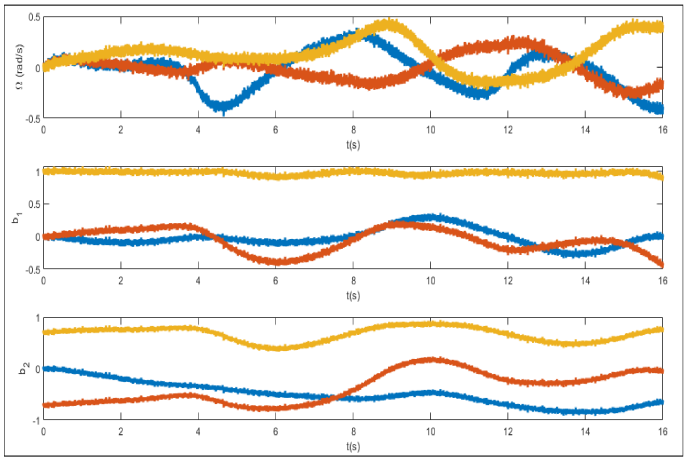

Figure 6.9: The vector measurements of the flight in Scenario ①.

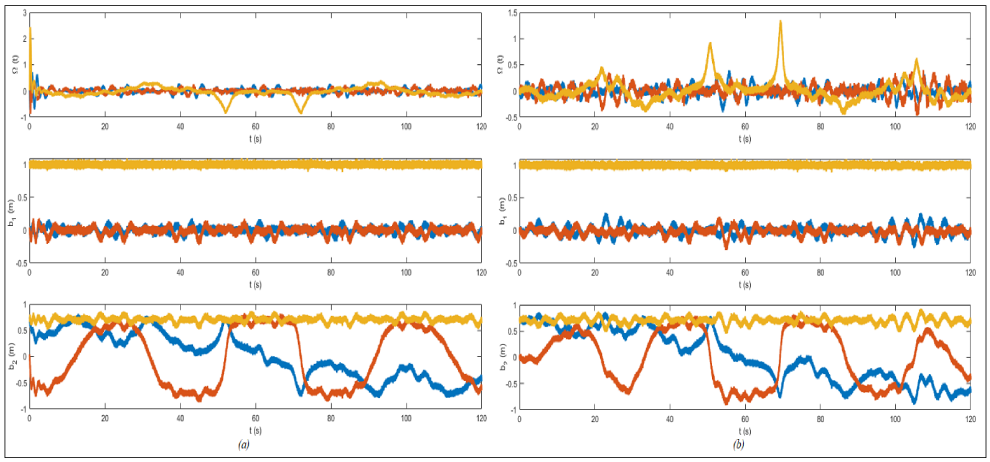

Figure 6.10: The vector measurements of the flight in Scenario ②: (a) Terrain 1, (b) Terrain 2.

position tracking results; however, Figures 6.14 and 6.15 show the attitude tracking results of the generated flight position trajectories. Figures 6.16,6.17,6.18(a) depict the pointing directions on $\mathbb{S}^2$. Figures 6.16, 6.17,6.18(b) show the thrust directions. However, Figures 6.16,6.17,6.18(c) show the aforementioned vector directions in 3D.

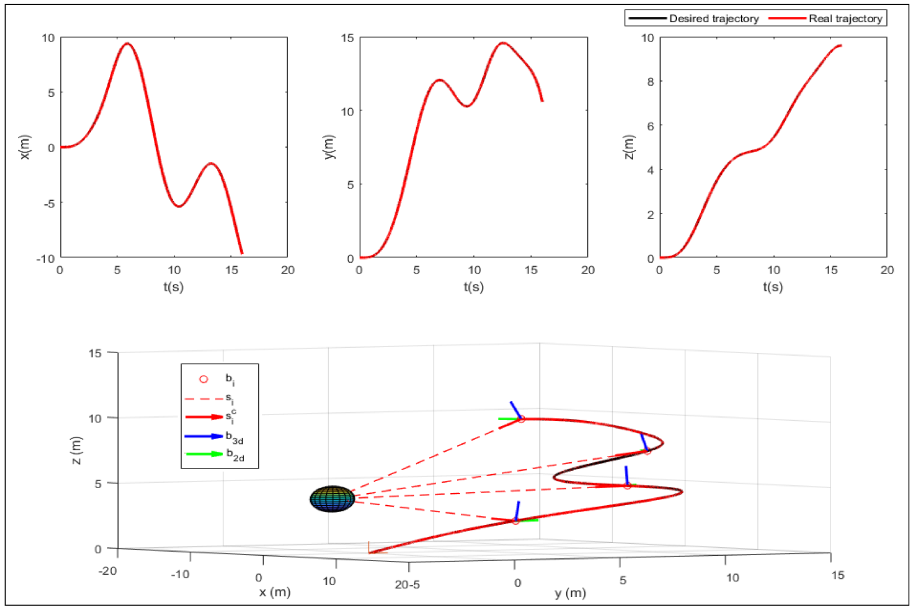

Figure 6.11: The position tracking results of the flight in Scenario ①.

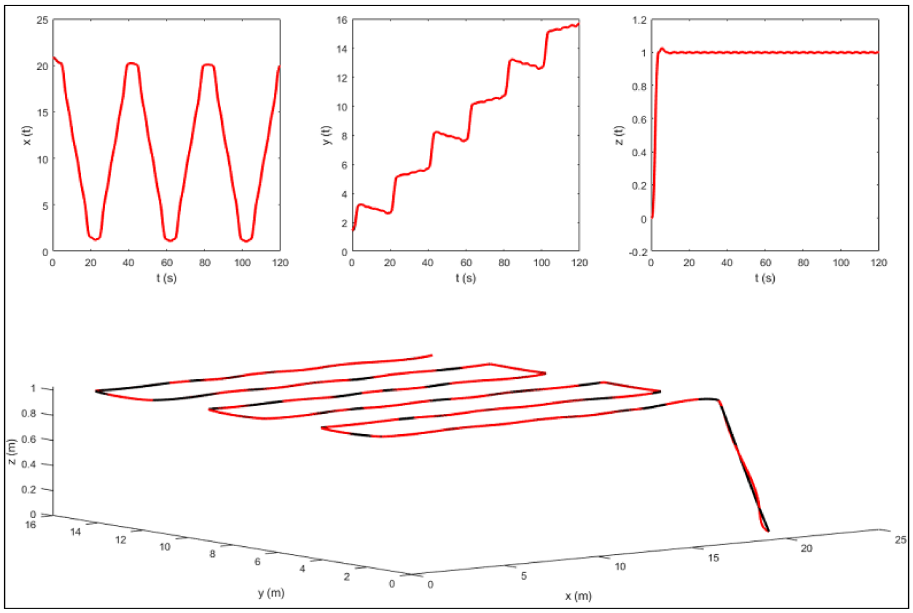

Figure 6.12: The position tracking results of the flight in Scenario ② in Terrain 1.

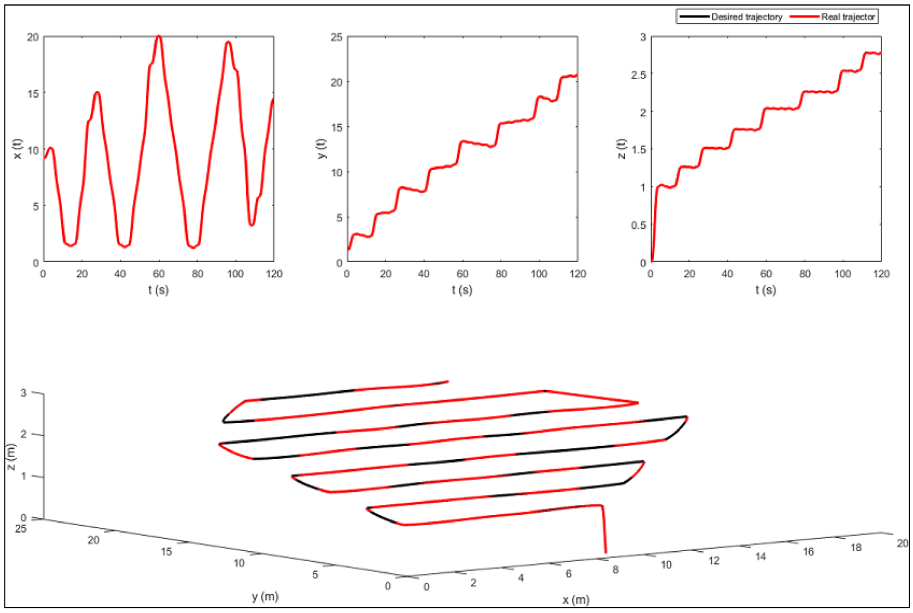

Figure 6.13: The position tracking results of the flight in Scenario ② in Terrain 2.

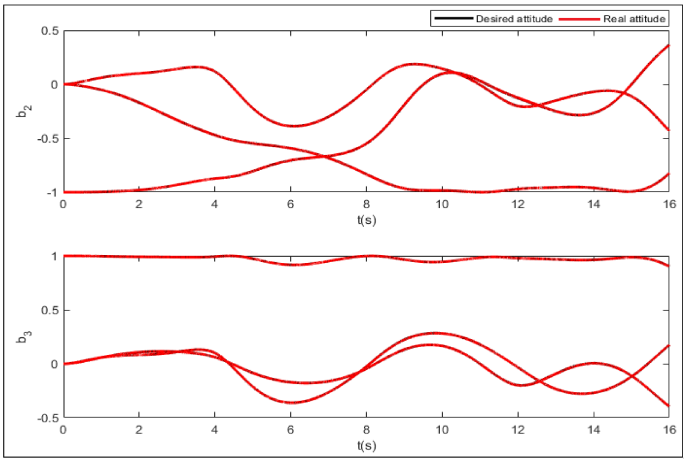

Figure 6.14: The attitude tracking results of the flight in Scenario ①.

6.3.4 Trajectory Re-planning

Terrain 1 of Scenario ② was selected to test the performance of the improved APF re-planning algorithm in 5.5. Several assumptions were taken into consideration:

- The quadrotor was modelled as a particle;
- The obstacles were static;

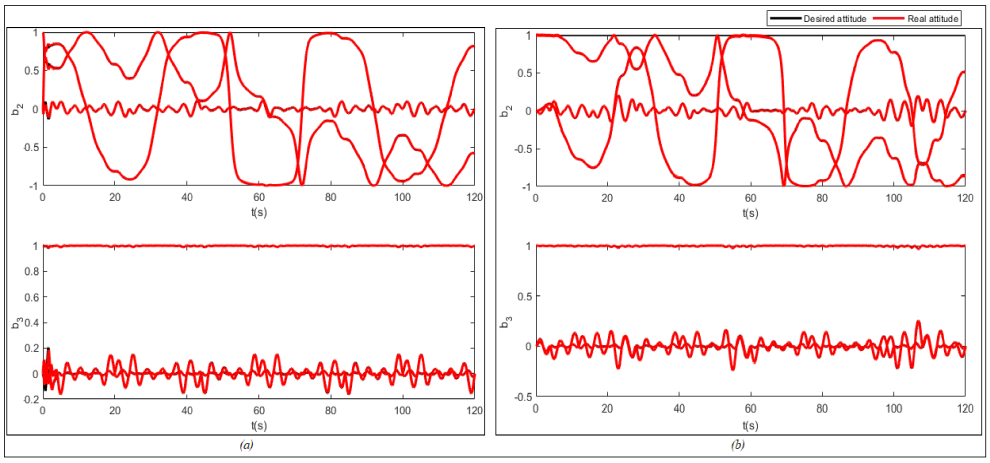

Figure 6.15: The attitude tracking results of the flight in Scenario ② in: (a) Terrain 1, (b) Terrain 2.

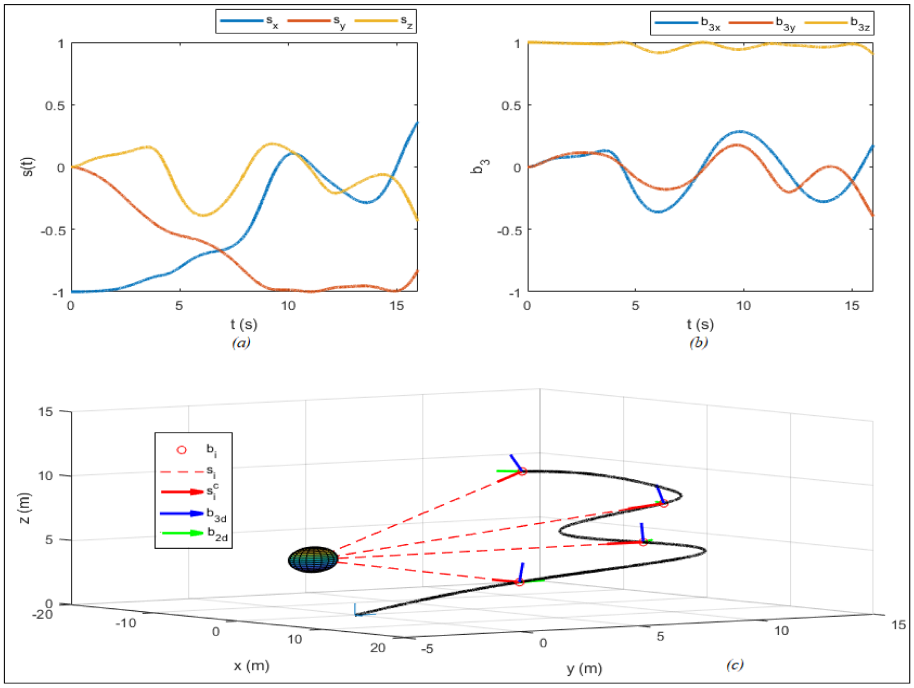

Figure 6.16: (a) The pointing, (b) thrust directions and (c) their representations in 3D for Scenario ①.

- For sake of simplicity, there were no uncertainties such as noise in the simulated measured ranges between the vehicle and the obstacles;

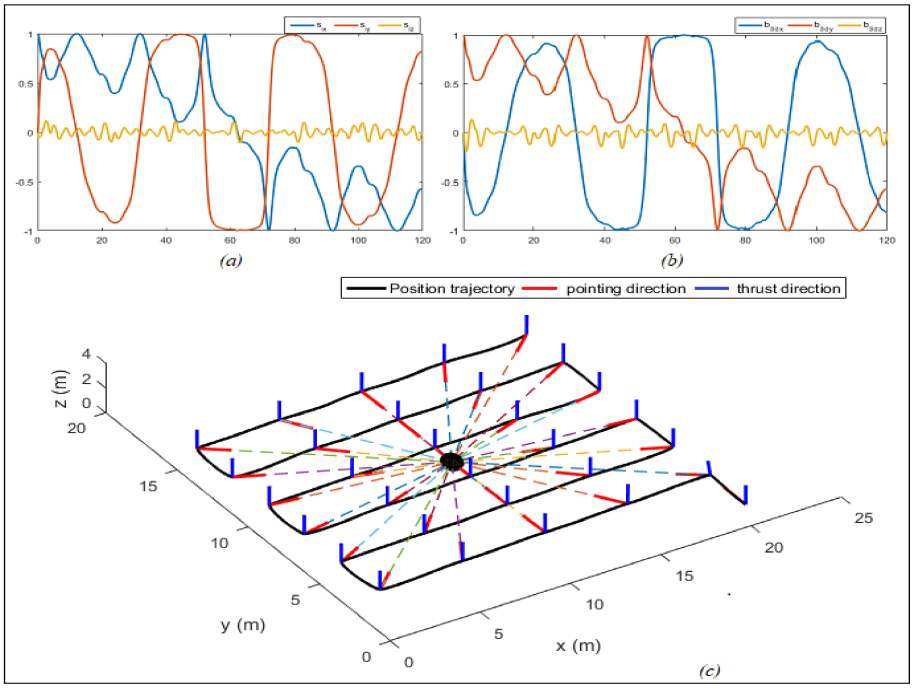

Figure 6.17: (a) The pointing, (b) thrust directions and (c) their representations in 3D for Scenario ② Terrain 1.

- The obstacle and the goal were symmetrically aligned. This is referred to as Symmetrically Aligned Robot-Obstacle-Goal (SAROG) problem;
- The local starting and the goal locations are calculated relative to the obstacle positions.

Besides, we supposed that the obstacles were in form of cylinders of equal sizes, located in the free environment between the vines. The influence range of a single obstacle was fixed at $d_{det} = 2.1$m. Figure 6.19 depicts the trajectory re-planned by the proposed method in case of single and multiple obstacles.

6.4 Discussion

In this chapter, a complete strategy for solving the autonomous navigation problem is presented. The strategy provides stable and efficient navigation method for autonomous quadrotor flight missions in agricultural fields. As mentioned before, the solution is based on a combination of algorithms. In path finding step, two scenarios are taken into consideration. Way-points are already defined in the point-to-point scenario. However, OGMs that maintain the environment information obtained from the terrain mapping process are adopted in the coverage scenario. Based on these maps, collision-free way-points are generated according to the ISOA. In trajectory planning, an LQR trajectory generator is adopted to to generate minimum snap trajectories while satisfying position/velocity/acceleration constraints at the extracted way-points. In trajectory tracking,

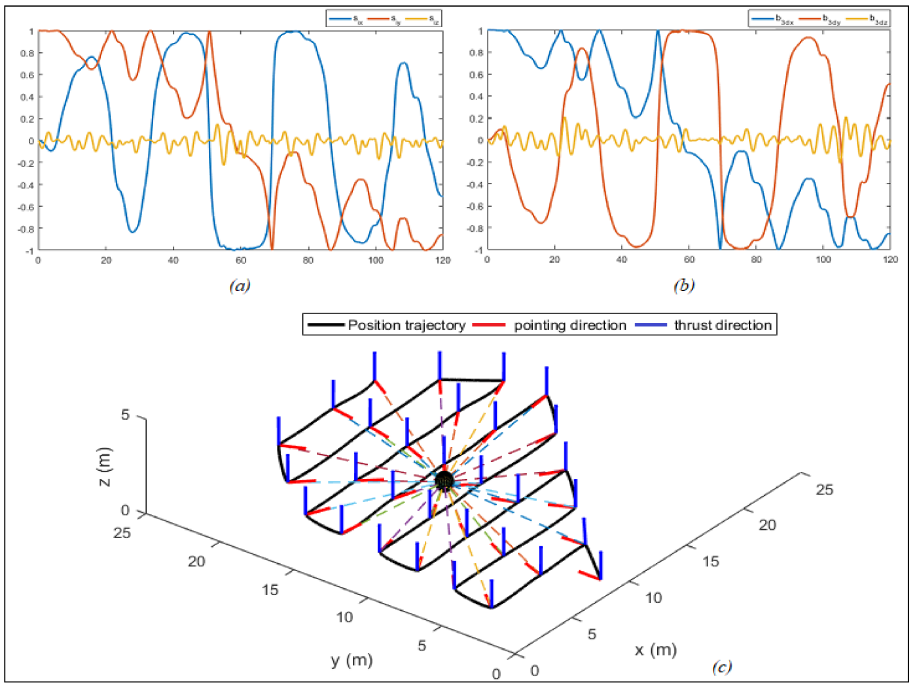

Figure 6.18: (a) The pointing, (b) thrust directions and (c) their representations in 3D for Scenario ② Terrain 2.

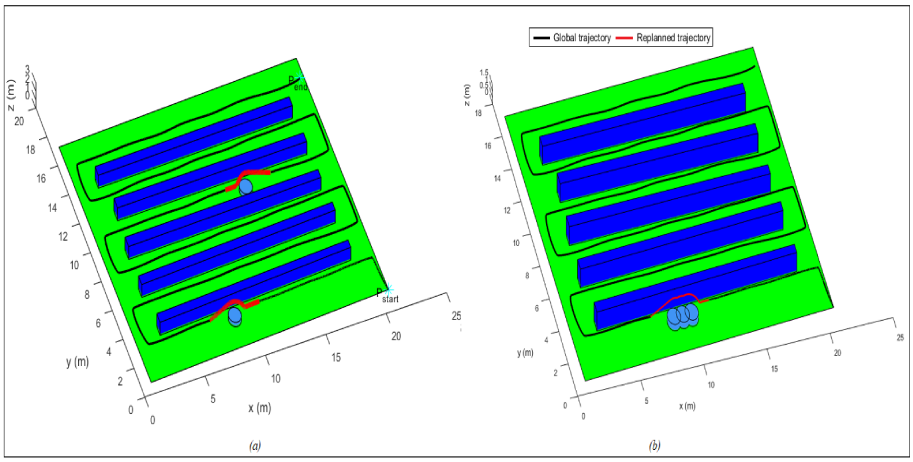

Figure 6.19: The global position and the re-planned trajectories: (a) single obstacle, (b) multiple obstacles.

a control law, which generates the attitude trajectories online using raw vector measurements, is implemented. The control law is used also to track the generated position and attitude trajecto-

ries while simultaneously pointing towards predefined direction along the whole flight mission. In trajectory re-planning, an Improved APF algorithm is used to enhance the safety of the quadrotor when unknown obstacles are detected.

To verify the effectiveness of the proposed strategy, a series of simulation experiments are conducted in this chapter. From the 2D view of 6.6, it can be seen that the ISOA efficiently generates complete optimal coverage paths. The back-and-fourth boustrophedon paths contain sharp turns which may contradict with the dynamics of the quadrotor. Hence, smoothing these paths represent a crucial step before sending any command to the vehicle. The ISAO is advantageous since it makes use of OGMs which offer a powerful technique for representing any environment and can be build from off-the-shelf sensors.

Figures 6.7, 6.8 and 6.9 depict the position trajectory produced by the LQR trajectory generator. It can be seen clearly that the trajectories are continuous and smooth. Besides, they maneuver around the extracted way-points successfully. This is achieved thanks to the LQR weighting diagonal terms S, Q and R. The choice of these terms is essential and plays a significant role in smoothing the trajectory features. However, choosing appropriate weighting terms is a challenging task. Figure 6.7 shows a comparison between the trajectories generated by our generator and the one discussed in (Dhullipalla et al., 2019). The quadrotor attitudes at the way-points are identical for both methods. However, a difference can be noticed in the trajectory curvatures. This difference is due to the minimized control input quantity; minimum snap trajectories are produced by our method, however minimum crackle (fifth derivative of the position) is used in (Dhullipalla et al., 2019).

The generated flight trajectories were tracked and analyzed. At the end of the flight mission, the quadrotor succeeded to track the desired trajectories despite the presence of white noise in the IMU. The comparison between the actual position/attitude trajectory (red) and the desired position/attitude trajectory (black) is depicted in Figures 6.11-6.15, showing that the two trajectories are identical, hence highlighting the effectiveness of the proposed control scheme to properly track the desired trajectories while fulfilling the pointing direction constraints at the local goals. The results obtained in Figure 6.14 are compared to the results of the algorithm used in (Dhullipalla et al., 2019) as shown in the following figures. By comparing the results, it can be seen that the obtained thrust directions by our method and the method in (Dhullipalla et al., 2019) are identical. In our case, b_{3d} is given by the Equation 5.7.3, however $q(t)$ in (Dhullipalla et al., 2019) is given by

$$q(t) = \frac{ge_3 + a(t)}{||ge_3 + a(t)||}$$

The two expression become the same if the parameters k_x and k_v are large enough to make the errors e_x and e_v vanish. In addition, it can be seen from Figure 6.21 that both algorithms guarantee the pointing direction at the way-points. The difference is that our algorithm fulfills the pointing direction constraint during the whole flight. However, the algorithm in (Dhullipalla et al., 2019) guarantees the pointing direction at the way-points only. This can be illustrated by Figure 6.22 which shows a top-view of Scenario ① (The same thing happens in Scenario ②).

A quantitative analysis of the position trajectories along the $x-$, $y-$ and $z-$ axes is shown in Figures 6.23 and 6.24 below. The absolute value of the maximum error after stability is reached, which is expressed as,:

$$e_{abs} = \sqrt{e_{x_{max}}^2 + e_{y_{max}}^2 + e_{z_{max}}^2}$$

can be calculated to be: 0.0014m in Scenario ①, 0.0028m for the first terrain and 0.0041m for the second one in Scenario ②. As it can be noticed, the errors are in the order of 10^{-3}m. Such

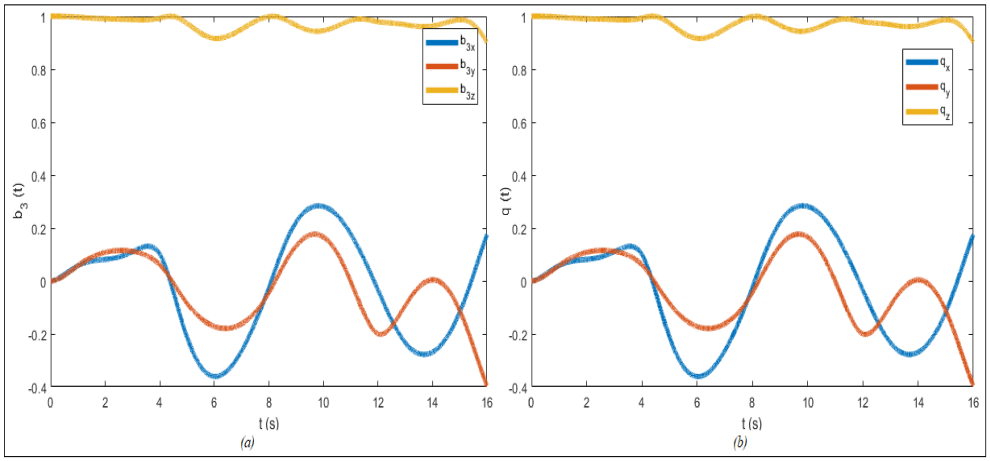

Figure 6.20: The thrust direction obtain by (a) our algorithm, (b) algorithm in (Dhullipalla et al., 2019).

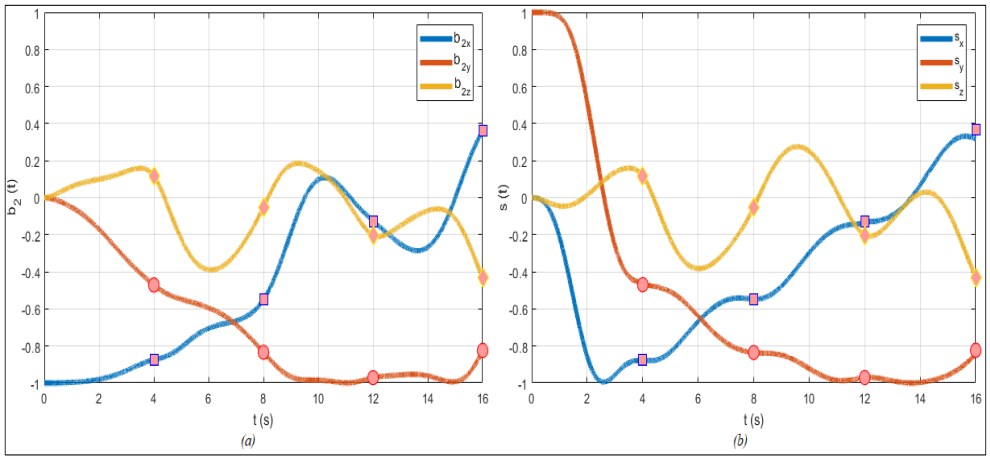

Figure 6.21: The pointing direction obtain by (a) our algorithm, (b) algorithm in (Dhullipalla et al., 2019).

errors are suitable relative to the size of the terrains. Hence, this data validates the efficiency of the proposed tracking algorithm.

Figure 6.19 shows the selected vineyard scenario where static obstacles were added between the vine rows to obstruct the motion of the quadrotor in its global trajectory. The figure depicts the quadrotor's avoidance trajectory near the static obstacles. Note that the black trajectory represents the desired re-planned trajectory obtained from the improved APF algorithm and the red trajectory represents the global trajectory obtained from the LQR trajectory generator. At every time step, the trajectory re-planner takes the position information of the global trajectory. Once the obstacle is sensed, it takes the position computed by the improved APF. It can be

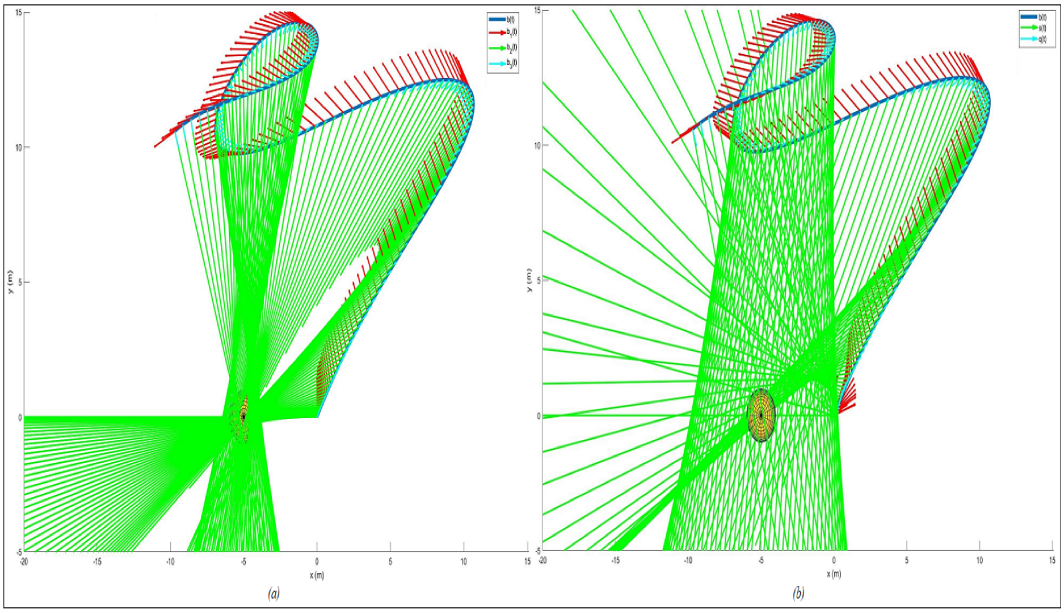

Figure 6.22: The pointing direction (top-view) obtain by (a) our algorithm, (b) algorithm in (Dhullipalla et al., 2019).

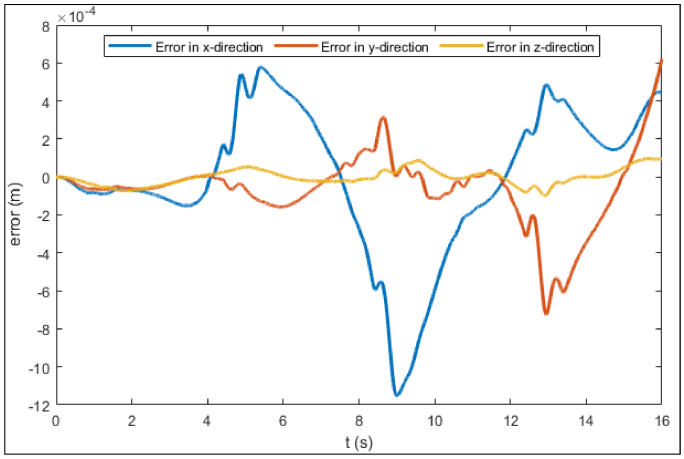

Figure 6.23: The position error profile for Scenario ①.

observed that the obstacle avoidance was performed vertically. This is advantageous since it would prevent collision with the vine rows if the avoidance was done horizontally. In addition, it can be noticed that the relative distance maneuver remains greater than collision threshold that represents a physical collision. Besides, it was assumed that the obstacle and the goal were symmetrically aligned, i.e. SAROG problem, and the local starting and the goal locations are calculated relative

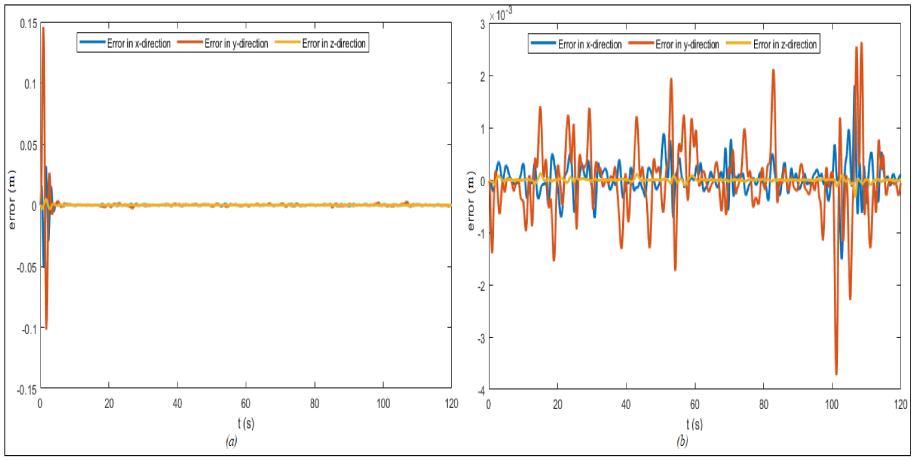

Figure 6.24: The position error profile for Scenario ② for: (a) Terrain 1, (b) Terrain 2.

to the obstacle positions. Hence, whenever multiple obstacles are available in same location, the improved APF algorithm calculates different local starting and target positions such that a smooth trajectory is replanned between the two positions while increasing the quadrotor safety as shown in Figure 6.19(b). Above all, one can say that the quadrotor completes the avoidance mission safely.

However, although the combination of the algorithms has offered a complete and efficient solution for navigation, there are some issues that need to be improved. Since the selected algorithms are used in phases sequentially, the output of a particular phase depends on the output of the previous one. Unreasonable outputs in a particular phase lead to intractable outputs in the next one. Consequently, the quality of the flight trajectories will be poor. Besides, the solution computational time depends on the dimensions of the environments. The more expansive the they are, the larger the computational time.

6.5 Conclusion

In this chapter, several simulation experiments have been conducted to show the effectiveness of the proposed approach developed in Chapter 5. First, a path planner that uses occupancy grids is used to generate obstacle-free geometric way-points. These latter are fed into the LQR trajectory generator to plan position trajectories while fulfilling the position/velocity/acceleration soft constraints at these way-points. A trajectory control is used to track the generated trajectories while pointing toward a predefined direction. The control, which is developed in the Special Euclidean $SE(3)$, uses the raw measurements vector provided by the IMU to follow the quadrotor trajectories with an absolute value of the maximum error in the order of 10^{-3}m. From the obtained simulation results, the validity of the proposed navigation approach is approved.

Chapter 7

General Conclusion & Perspectives

Autonomous navigation is the fundamental element to ensure collision-free motion in any type of environment. In case of quadrotors, the autonomous navigation techniques require suitable performances in terms of reduced time design, low computational requirements and optimal trajectory generation and control capabilities. The main objective of the research work presented in this thesis was to design a complete and efficient decoupled solution for autonomous navigation problem of quadrotors using a combination of algorithms exploited for the first time in Precision Agriculture scenarios. The proposed design was constructed using four different components: offline path planning, offline trajectory generation and optimization, real-time trajectory control and tracking and real-time trajectory re-planning. All of these components were utilized in an innovative cooperative architecture aiming to complete the quadrotor scheduled mission successfully.

The whole research was conducted around the quadrotors autonomous navigation problem. Earlier in Chapter 2, we investigated the background of UAVs in general including their classifications based on the vast explored literature which showed the different advantages and drawbacks of each common UAV system. Such classification has narrowed this research work into focusing on quadrotors thanks to their great advantages in terms higher degree of maneuverability, hovering ability, higher payload capacity and precise movements abilities. However, quadrotors are inherently nonlinear and non-stable. Hence, assigning them motion commands directly would usher to catastrophic results and mission failure. This has led us to survey the literature of the different autonomous navigation strategies. After that, we grouped them into two main categories: coupled and decoupled autonomous navigation structures. Focusing on the latter structure, we divided it into mainly three elements: path planning, trajectory generation and trajectory control. Then, we conducted an extensive literature review of each element where the advantage and drawbacks of each approach of each element were presented (Chapter 3 and 4).

In Chapter 5, we presented the mathematical developments of each component that build up our proposed autonomous navigation strategy. In the path planning component, we ran the ISO algorithm that used artificial OGMs to generate safe and optimal geometric paths. Then, we used the DEMs to locate the reference position local goals (way-points). We augmented the extracted positions with velocities/accelerations and formed constraints at these way-points. In the trajectory generation component, we built an LQR optimal problem that was used to minimize the snap of the quadrotor when subjected to its kinematics. The inputs to the LQR generator were the way-points constraints while the outputs were the global minimum snap trajectories with high degree

of smoothness and stability. In the trajectory re-planning component, we adopted the Improved APF algorithm to re-plan the vehicle's trajectories locally in real-time whenever it sensed the unexpected obstacles. This component made use of the artificial forces built by the Improved APF algorithm around the obstacles and at the target to move the quadrotor away from those obstacles. Finally, we implemented a geometric controller in the Special Euclidean $SE(3)$ group to generate the quadrotor's attitude trajectories while pointing toward a predefined direction. This was achieved using noisy vector measurements provided by the IMU. The geometric controller was also used to track both the generated position and attitude trajectories. We validated the proposed autonomous navigation strategy through several numerical simulations to show its effectiveness. The strategy presents an innovative solution for autonomous navigation especial in PA scenarios, as it is specifically designed to comply the unique characteristics of both the scenario and the quadrotor. Besides, the proposed strategy shows its capability of optimal and efficient generating trajectories, while also demonstrating optimal tracking abilities, making it a highly suitable choice for fully autonomous quadrotor navigation in any chosen outdoor scenario. However, the startegy has certain limitations that need to be considered, such as being vulnerable to various weather conditions such as rain, fog, and dust, as well as having difficulty maneuvering in windy or turbulent conditions. These challenges are particularly pronounced in large, unstructured agricultural environments, where issues related to vehicle localization and endurance become even more prominent.

In terms of perspective of this work, there are several lines of research that could be pursued. Firstly, the main limitation of the work presented in this thesis is that we were only able to perform computer simulations to validate the proposed autonomous navigation strategy. Hence, the experimental validation on a real quadrotor as presented herein will support the obtained results. Hardware In Loop (HIL) simulation environment represents a great alternative to test the strategy algorithms in real-time. Secondly, the proposed strategy could be extended to deal with external disturbances. Besides, in order to increase the reliability of such strategy for practical applications, further work will be conducted to assist the quadrotor in environments with dynamic obstacles. Finally, it would be beneficial to extend this strategy to collaborative systems where multiple quadrotors can be used to perform the mission they will be assigned to. The problem at hand is fraught with a multitude of technical difficulties, particularly with regards to effectively coordinating movement and transmitting information between different systems. At the same time, such system offers the potential to explore environments with very vast areas while reducing the required amount of time to do that.

Appendix A

List of Publications

A.1 International Conference Papers

- MOKRANE, Adel, KADOUCI, Abenasser, CHOUKCHOU-BRAHAM, Amal, & CHERKI, Brahim. Detection and Counting of Fruit from UAV RGB Images Using Computer Vision. In : Computational Vision and Bio-Inspired Computing: Proceedings of ICCVBIC 2021. Singapore : Springer Singapore, 2022. p. 761-777.
- MOKRANE, Adel, BRAHAM, Amal CHOUKCHOU, & CHERKI, Brahim. UAV path planning based on dynamic programming algorithm on photogrammetric DEMs. In : 2020 International Conference on Electrical Engineering (ICEE). IEEE, 2020. p. 1-5.
- MOKRANE, Adel, CHOUKCHOU-BRAHAM, Amal, & CHERKI, Brahim. Coverage Path Planning Of Autonomous Marsupial Systems For Supporting Fruit Counting Process. In : 2020 International Conference on Electrical Engineering (ICEE). IEEE, 2020. p. 1-6.
- MOKRANE, Adel, CHOUKCHOU-BRAHAM, Amal, & CHERKI, Brahim. DEM Generation Based On UAV Photogrammetry. In : 2019 International Conference on Advanced Electrical Engineering (ICAEE). IEEE, 2019. p. 1-5.
- MOKRANE, Adel, BRAHAM, Amal Choukchou, & CHERKI, Brahim. UAV coverage path planning for supporting autonomous fruit counting systems. In : 2019 International Conference on Applied Automation and Industrial Diagnostics (ICAAID). IEEE, 2019. p. 1-5.

A.2 International Journal Articles

- MOKRANE, Adel, BENALLEGUE, Abdelaziz, CHOUKCHOU-BRAHAM, Amal, El HADRI, Abdelhafid, & CHERKI, Brahim. Guidance, Navigation and Control for Autonomous Quadrotor Flight in an Agricultural Field: The Case of Vineyards. Sensors, 2022, vol. 22, no 22, p. 8865.

Bibliography

Aghababa, Mohammad Pourmahmood (2012). "3D path planning for underwater vehicles using five evolutionary optimization algorithms avoiding static and energetic obstacles". In: *Applied Ocean Research* 38, pp. 48–62.

Agrawal, Kanaiya and Punit Shrivastav (2015). "Multi-rotors: A revolution in unmanned aerial vehicle". In: *International Journal of Science and Research* 4.11, pp. 1800–1804.

Al-Hiddabi, Saif A (2009). "Quadrotor control using feedback linearization with dynamic extension". In: *2009 6th International Symposium on Mechatronics and its Applications*. IEEE, pp. 1–3.

Alaliyat, Saleh, Rachid Oucheikh, and Ibrahim Hameed (2019). "Path Planning in Dynamic Environment Using Particle Swarm Optimization Algorithm". In: *2019 8th International Conference on Modeling Simulation and Applied Optimization (ICMSAO)*. IEEE, pp. 1–5.

Alanezi, Mohammed A et al. (2022). "Obstacle Avoidance-Based Autonomous Navigation of a Quadrotor System". In: *Drones* 6.10, p. 288.

Amir, M Yasir and Valiuddin Abbass (2008). "Modeling of quadrotor helicopter dynamics". In: *2008 International Conference on Smart Manufacturing Application*. IEEE, pp. 100–105.

Andersson, Joel, Johan kesson, and Moritz Diehl (2012). "CasADi: A symbolic package for automatic differentiation and optimal control". In: *Recent advances in algorithmic differentiation*. Springer, pp. 297–307.

Arafat, Muhammad Yeasir, Muhammad Morshed Alam, and Sangman Moh (2023). "Vision-Based Navigation Techniques for Unmanned Aerial Vehicles: Review and Challenges". In: *Drones* 7.2, p. 89.

Atiyah, Anis Naema, Noraziah Adzhar, and Nor Izzati Jaini (2021). "An overview: on path planning optimization criteria and mobile robot navigation". In: *Journal of Physics: Conference Series*. Vol. 1988. 1. IOP Publishing, p. 012036.

Bekhti, Mustapha et al. (2017). "Drone Package Delivery: A Heuristic approach for UAVs path planning and tracking". In: *EAI endorsed transactions on Internet of Things* 3.9, e1.

Bellman, Richard E and Stuart E Dreyfus (2015). *Applied dynamic programming*. Vol. 2050. Princeton university press.

Bett, Christopher J and WK Chen (2005). "Gain-scheduled controllers". In: *The electrical engineering handbook*. Academic Press Burlington, pp. 1107–1114.

Böhme, Thomas J et al. (2017). "Indirect methods for optimal control". In: *Hybrid systems, optimal control and hybrid vehicles: Theory, methods and applications*, pp. 215–231.

Bonin-Font, Francisco, Alberto Ortiz, and Gabriel Oliver (2008). "Visual navigation for mobile robots: A survey". In: *Journal of intelligent and robotic systems* 53, pp. 263–296.

Bonna, R and JF Camino (2015). "Trajectory tracking control of a quadrotor using feedback linearization". In: *International Symposium on Dynamic Problems of Mechanics*. University of Campinas Sao Paulo, Brazil.

Boretti, Alberto and Lorenzo Rosa (2019). "Reassessing the projections of the world water development report". In: *NPJ Clean Water* 2.1, p. 15.

Bouabdallah, Samir (2007). *Design and control of quadrotors with application to autonomous flying*. Tech. rep. Epfl.

Boyd, Stephen, Stephen P Boyd, and Lieven Vandenberghe (2004). *Convex optimization*. Cambridge university press.

Byrnes, Christopher I and Alberto Isidori (1991). "Asymptotic stabilization of minimum phase nonlinear systems". In: *IEEE Transactions on Automatic Control* 36.10, pp. 1122–1137.

Cabreira, Tauã M, Lisane B Brisolara, and Ferreira Jr Paulo R (2019). "Survey on coverage path planning with unmanned aerial vehicles". In: *Drones* 3.1, p. 4.

Cao, Zuo Llang, Yuyu Huang, and Ernest L Hall (1988). "Region filling operations with random obstacle avoidance for mobile robots". In: *Journal of Robotic systems* 5.2, pp. 87–102.

Carvalho, Kevin B de et al. (2022). "Q-learning based path planning method for uavs using priority shifting". In: *2022 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, pp. 421–426.

Castellanos, José A, José Neira, and Juan D Tardós (2018). "Map building and SLAM algorithms". In: *Autonomous Mobile Robots*. CRC Press, pp. 335–372.

Cekmez, Ugur, Mustafa Ozsiginan, and Ozgur Koray Sahingoz (2014). "A UAV path planning with parallel ACO algorithm on CUDA platform". In: *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, pp. 347–354.

Chakraborty, Suprava et al. (2022). "A comprehensive review of path planning for agricultural ground robots". In: *Sustainability* 14.15, p. 9156.

Chen, Yong-bo et al. (2016). "UAV path planning using artificial potential field method updated by optimal control theory". In: *International Journal of Systems Science* 47.6, pp. 1407–1420.

De Filippis, Luca, Giorgio Guglieri, and Fulvia Quagliotti (2011). "A minimum risk approach for path planning of UAVs". In: *Journal of Intelligent & Robotic Systems* 61, pp. 203–219.

— (2012). "Path planning strategies for UAVS in 3D environments". In: *Journal of Intelligent & Robotic Systems* 65, pp. 247–264.

Dhullipalla, Mani H et al. (2019). "Trajectory generation on SE (3) for an underactuated vehicle with pointing direction constraints". In: *2019 American Control Conference (ACC)*. IEEE, pp. 1930–1935.

Dijkstra, Edsger W (2022). "A note on two problems in connexion with graphs". In: *Edsger Wybe Dijkstra: His Life, Work, and Legacy*, pp. 287–290.

Doyle, John et al. (1988). "State-space solutions to standard H 2 and H control problems". In: *1988 American Control Conference*. IEEE, pp. 1691–1696.

Dubins, Lester E (1957). "On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents". In: *American Journal of mathematics* 79.3, pp. 497–516.

Dudek, Gregory and Michael Jenkin (2010). *Computational principles of mobile robotics*. Cambridge university press.

Duggal, Vishakh et al. (2016). "Plantation monitoring and yield estimation using autonomous quadcopter for precision agriculture". In: *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, pp. 5121–5127.

Durrant-Whyte, Hugh and Tim Bailey (2006). "Simultaneous localization and mapping: part I". In: *IEEE robotics & automation magazine* 13.2, pp. 99–110.

Eade, Ethan (2013). "Lie groups for 2d and 3d transformations". In: *URL http://ethaneade. com/lie. pdf, revised Dec* 117, p. 118.

Elfes, Alberto (1989). "Using occupancy grids for mobile robot perception and navigation". In: *Computer* 22.6, pp. 46–57.

Elmokadem, Taha and Andrey V Savkin (2021). "Towards fully autonomous UAVs: A survey". In: *Sensors* 21.18, p. 6223.

Eslamiat, Hossein et al. (2022). "Geometric Integral Attitude Control on SO (3)". In: *Electronics* 11.18, p. 2821.

Fan, Xiaojing et al. (2020). "Improved artificial potential field method applied for AUV path planning". In: *Mathematical Problems in Engineering* 2020, pp. 1–21.

Faria, Margarida, Ivan Maza, and Antidio Viguria (2019). "Applying frontier cells based exploration and Lazy Theta* path planning over single grid-based world representation for autonomous inspection of large 3D structures with an UAS". In: *Journal of Intelligent & Robotic Systems* 93, pp. 113–133.

Faria, Margarida et al. (2019). "Efficient lazy theta* path planning over a sparse grid to explore large 3d volumes with a multirotor uav". In: *Sensors* 19.1, p. 174.

Faulwasser, Timm and Rolf Findeisen (2015). "Nonlinear model predictive control for constrained output path following". In: *IEEE Transactions on Automatic Control* 61.4, pp. 1026–1039.

Ferrin, Jeff et al. (2011). "Differential flatness based control of a rotorcraft for aggressive maneuvers". In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Ieee, pp. 2688–2693.

Filliat, David and Jean-Arcady Meyer (2003). "Map-based navigation in mobile robots:: I. a review of localization strategies". In: *Cognitive systems research* 4.4, pp. 243–282.

Gabriely, Yoav and Elon Rimon (2002). "Spiral-STC: An on-line coverage algorithm of grid environments by a mobile robot". In: *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*. Vol. 1. IEEE, pp. 954–960.

Galceran, Enric and Marc Carreras (2013). "A survey on coverage path planning for robotics". In: *Robotics and Autonomous systems* 61.12, pp. 1258–1276.

Galvez, Reagan L, Elmer P Dadios, and Argel A Bandala (2014). "Path planning for quadrotor UAV using genetic algorithm". In: *2014 International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM)*. IEEE, pp. 1–6.

Gao, Zhiqiang (1996). "Techniques in reconfigurable control system design". In: *Control and Dynamic Systems* 79, pp. 89–116.

Geisert, Mathieu and Nicolas Mansard (2016). "Trajectory generation for quadrotor based systems using numerical optimal control". In: *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, pp. 2958–2964.

Gerdes, John W, Satyandra K Gupta, and Stephen A Wilkerson (2012). "A review of bird-inspired flapping wing miniature air vehicle designs". In.

Giesbrecht, Jared (2004). *Global path planning for unmanned ground vehicles*. Tech. rep. Defence Research and Development Suffield (ALBERTA).

Giordano, Paolo Robuffo, Quentin Delamare, and Antonio Franchi (2018). "Trajectory generation for minimum closed-loop state sensitivity". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 286–293.

Gnip, Pavel, Karel Charvat, Matej Krocan, et al. (2008). "Analysis of external drivers for agriculture". In: *IALD WCCA INFITA congress.*

Goodarzi, Farhad, Daewon Lee, and Taeyoung Lee (2013). "Geometric nonlinear PID control of a quadrotor UAV on SE (3)". In: *2013 European control conference (ECC).* IEEE, pp. 3845–3850.

Goodarzi, Farhad A, Daewon Lee, and Taeyoung Lee (2015). "Geometric adaptive tracking control of a quadrotor unmanned aerial vehicle on SE (3) for agile maneuvers". In: *Journal of Dynamic Systems, Measurement, and Control* 137.9, p. 091007.

Hao, Wenjian et al. (2020). "Cell a* for navigation of unmanned aerial vehicles in partially-known environments". In: *arXiv preprint arXiv:2009.07404.*

Hart, Peter E, Nils J Nilsson, and Bertram Raphael (1968). "A formal basis for the heuristic determination of minimum cost paths". In: *IEEE transactions on Systems Science and Cybernetics* 4.2, pp. 100–107.

Horvath, Erno, Claudiu Pozna, and Radu-Emil Precup (2018). "Robot coverage path planning based on iterative structured orientation". In: *Acta Polytechnica Hungarica* 15.2, pp. 231–249.

Hoy, Michael, Alexey S Matveev, and Andrey V Savkin (2015). "Algorithms for collision-free navigation of mobile robots in complex cluttered environments: a survey". In: *Robotica* 33.3, pp. 463–497.

Huang, Hui-Min (2004). "Autonomy levels for unmanned systems (ALFUS) framework volume I: Terminology version 2.0". In.

Hunter III, Joseph E et al. (2020). "Integration of remote-weed mapping and an autonomous spraying unmanned aerial vehicle for site-specific weed management". In: *Pest Management Science* 76.4, pp. 1386–1392.

Idrissi, Moad, Mohammad Salami, and Fawaz Annaz (2022). "A review of quadrotor unmanned aerial vehicles: applications, architectural design and control algorithms". In: *Journal of Intelligent & Robotic Systems* 104.2, p. 22.

Ignizio, James P and Tom M Cavalier (1994). *Linear programming.* Prentice-Hall, Inc.

Jiao, Qingsong et al. (2018). "Analysis and design the controller for quadrotors based on PID control method". In: *2018 33rd Youth Academic Annual Conference of Chinese Association of Automation (YAC).* IEEE, pp. 88–92.

Judd, Kevin and Timothy McLain (2001). "Spline based path planning for unmanned air vehicles". In: *AIAA Guidance, Navigation, and Control Conference and Exhibit*, p. 4238.

Kanellakis, Christoforos and George Nikolakopoulos (2017). "Survey on computer vision for UAVs: Current developments and trends". In: *Journal of Intelligent & Robotic Systems* 87, pp. 141–168.

Karaman, Sertac and Emilio Frazzoli (2011). "Sampling-based algorithms for optimal motion planning". In: *The international journal of robotics research* 30.7, pp. 846–894.

Kavraki, Lydia E et al. (1996). "Probabilistic roadmaps for path planning in high-dimensional configuration spaces". In: *IEEE transactions on Robotics and Automation* 12.4, pp. 566–580.

Kellermann, Robin, Tobias Biehle, and Liliann Fischer (2020). "Drones for parcel and passenger transportation: A literature review". In: *Transportation Research Interdisciplinary Perspectives* 4, p. 100088.

Kermani, Pedram and Ali A Afzalian (2014). "Flight path planning using GA and fuzzy logic considering communication constraints". In: *7'th International Symposium on Telecommunications (IST'2014).* IEEE, pp. 6–11.

Khatib, Oussama (1986). "Real-time obstacle avoidance for manipulators and mobile robots". In: *The international journal of robotics research* 5.1, pp. 90–98.

Killian, Lars and Jan Backhaus (2021). "Utilizing the RRT*-Algorithm for Collision Avoidance in UAV Photogrammetry Missions". In: *arXiv preprint arXiv:2108.03863*.

Koenig, Sven and Maxim Likhachev (2001). "Incremental a". In: *Advances in neural information processing systems* 14.

Koutsoukos, Xenofon D and Panos J Antsaklis (1999). "Computational issues in intelligent control: Discrete-event and hybrid systems". In: *Soft Computing and Intelligent Systems: Theory and Practice, NK Sinha and MM Gupta, Eds*, pp. 39–69.

Laghmara, Hind et al. (2019). "Obstacle avoidance, path planning and control for autonomous vehicles". In: *2019 IEEE intelligent vehicles symposium (IV)*. IEEE, pp. 529–534.

LaValle, Steven M (2006). *Planning algorithms*. Cambridge university press.

Lee, Hyeonbeom et al. (2013). "Backstepping control on se (3) of a micro quadrotor for stable trajectory tracking". In: *2013 IEEE international conference on systems, man, and cybernetics*. IEEE, pp. 4522–4527.

Lee, Taeyoung, Melvin Leok, and N Harris McClamroch (2010). "Geometric tracking control of a quadrotor UAV on SE (3)". In: *49th IEEE conference on decision and control (CDC)*. IEEE, pp. 5420–5425.

Leondes, Cornelius T (1998). *Neural network systems techniques and applications: Advances in theory and applications*. Academic Press.

Li, Jun and Yuntang Li (2011). "Dynamic analysis and PID control for a quadrotor". In: *2011 IEEE International Conference on Mechatronics and Automation*. IEEE, pp. 573–578.

Li, Lebao, Lingling Sun, and Jie Jin (2015). "Survey of advances in control algorithms of quadrotor unmanned aerial vehicle". In: *2015 IEEE 16th international conference on communication technology (ICCT)*. IEEE, pp. 107–111.

Li, You and Yassine Ruichek (2014). "Occupancy grid mapping in urban environments from a moving on-board stereo-vision system". In: *Sensors* 14.6, pp. 10454–10478.

Lin, Yucong and Srikanth Saripalli (2017). "Sampling-based path planning for UAV collision avoidance". In: *IEEE Transactions on Intelligent Transportation Systems* 18.11, pp. 3179–3192.

Liu, Haiying et al. (2022). "An Improved RRT* UAV Formation Path Planning Algorithm Based on Goal Bias and Node Rejection Strategy". In: *Unmanned Systems*, pp. 1–10.

Lukmana, Muhammad Arifudin and Hendro Nurhadi (2015). "Preliminary study on unmanned aerial vehicle (uav) quadcopter using pid controller". In: *2015 International Conference on Advanced Mechatronics, Intelligent Manufacture, and Industrial Automation (ICAMIMIA)*. IEEE, pp. 34–37.

Madani, Tarek and Abdelaziz Benallegue (2007). "Sliding mode observer and backstepping control for a quadrotor unmanned aerial vehicles". In: *2007 American control conference*. IEEE, pp. 5887–5892.

Martinez Martinez, Vicente (2007). "Modelling of the flight dynamics of a quadrotor helicopter". In.

Masehian, Ellips and MR Amin-Naseri (2004). "A voronoi diagram-visibility graph-potential field compound algorithm for robot path planning". In: *Journal of Robotic Systems* 21.6, pp. 275–300.

Masehian, Ellips and Davoud Sedighizadeh (2007). "Classic and heuristic approaches in robot motion planning-a chronological review". In: *World Academy of Science, Engineering and Technology* 23.5, pp. 101–106.

Mellinger, Daniel and Vijay Kumar (2011). "Minimum snap trajectory generation and control for quadrotors". In: *2011 IEEE international conference on robotics and automation*. IEEE, pp. 2520–2525.

Mellinger, Daniel, Nathan Michael, and Vijay Kumar (2012). "Trajectory generation and control for precise aggressive maneuvers with quadrotors". In: *The International Journal of Robotics Research* 31.5, pp. 664–674.

Mokrane, Adel et al. (2022). "Guidance, Navigation and Control for Autonomous Quadrotor Flight in an Agricultural Field: The Case of Vineyards". In: *Sensors* 22.22, p. 8865.

Montazeri, Allahyar, Aydin Can, and Imil Hamda Imran (2021). "Unmanned aerial systems: autonomy, cognition, and control". In: *Unmanned Aerial Systems*. Elsevier, pp. 47–80.

Montemerlo, Michael et al. (2002). "FastSLAM: A factored solution to the simultaneous localization and mapping problem". In: *Aaai/iaai* 593598.

Nasirian, B, Mehran Mehrandezh, and Farrokh Janabi-Sharifi (2021). "Efficient coverage path planning for mobile disinfecting robots using graph-based representation of environment". In: *Frontiers in Robotics and AI* 8, p. 624333.

Nurimbetov, Birzhan et al. (2017). "Motion planning for hybrid UAVs in dense urban environments". In: *2017 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*. IEEE, pp. 1627–1632.

Nuss, Dominik et al. (2018). "A random finite set approach for dynamic occupancy grid maps with real-time application". In: *The International Journal of Robotics Research* 37.8, pp. 841–866.

Oh, Joon Seop et al. (2004). "Complete coverage navigation of cleaning robots using triangular-cell-based map". In: *IEEE Transactions on Industrial Electronics* 51.3, pp. 718–726.

Özbek, Necdet Sinan, Mert Önkol, and Mehmet Önder Efe (2016). "Feedback control strategies for quadrotor-type aerial robots: a survey". In: *Transactions of the Institute of Measurement and Control* 38.5, pp. 529–554.

Partovi, Ali Reza et al. (2012). "Development of a cross style quadrotor". In: *AIAA Guidance, Navigation, and Control Conference*, p. 4780.

Patel, Parth N et al. (2013). "Quadcopter for agricultural surveillance". In: *Advance in Electronic and Electric Engineering* 3.4, pp. 427–432.

Penin, Bryan (2018). "Contributions to optimal and reactive vision-based trajectory generation for a quadrotor UAV". PhD thesis. Rennes 1.

Perez-Alcocer, Ricardo and Javier Moreno-Valenzuela (2019). "Adaptive control for quadrotor trajectory tracking with accurate parametrization". In: *IEEE Access* 7, pp. 53236–53247.

Pizetta, Igor Henrique Beloti, Alexandre Santos Brandão, and Mário Sarcinelli-Filho (2019). "Avoiding obstacles in cooperative load transportation". In: *ISA transactions* 91, pp. 253–261.

Raiesdana, Somayeh (2020). "Control of quadrotor trajectory tracking with sliding mode control optimized by neural networks". In: *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering* 234.10, pp. 1101–1119.

Raja, Purushothaman and Sivagurunathan Pugazhenthi (2012). "Optimal path planning of mobile robots: A review". In: *International journal of physical sciences* 7.9, pp. 1314–1320.

Rashad, Ramy, Ahmed Aboudonia, and Ayman El-Badawy (2015). "Backstepping trajectory tracking control of a quadrotor with disturbance rejection". In: *2015 XXV International Conference on Information, Communication and Automation Technologies (ICAT)*. IEEE, pp. 1–7.

Rezwan, Sifat and Wooyeol Choi (2022). "Artificial intelligence approaches for UAV navigation: Recent advances and future challenges". In: *IEEE Access*.

Rodriguez, Samuel et al. (2006). "An obstacle-based rapidly-exploring random tree". In: *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.* IEEE, pp. 895–900.

Roussos, Giannis, Dimos V Dimarogonas, and Kostas J Kyriakopoulos (2010). "3D navigation and collision avoidance for nonholonomic aircraft-like vehicles". In: *International Journal of Adaptive Control and Signal Processing* 24.10, pp. 900–920.

Rubí, Bartomeu, Ramon Pérez, and Bernardo Morcego (2020). "A survey of path following control strategies for UAVs focused on quadrotors". In: *Journal of Intelligent & Robotic Systems* 98.2, pp. 241–265.

Sabo, Chelsea and Kelly Cohen (2012). "Fuzzy logic unmanned air vehicle motion planning". In: *Advances in Fuzzy Systems* 2012, pp. 13–13.

Saccon, Pierpaolo (2018). "Water for agriculture, irrigation management". In: *Applied soil ecology* 123, pp. 793–796.

Santos, Milton Cesar Paes et al. (2017). "A novel null-space-based UAV trajectory tracking controller with collision avoidance". In: *IEEE/ASME Transactions on Mechatronics* 22.6, pp. 2543–2553.

Sanyal, Amit, Nikolaj Nordkvist, and Monique Chyba (2010). "An almost global tracking control scheme for maneuverable autonomous vehicles and its discretization". In: *IEEE Transactions on Automatic control* 56.2, pp. 457–462.

Sawyer, Shaun (2015). "Gain-scheduled control of a quadcopter UAV". MA thesis. University of Waterloo.

Schauwecker, Konstantin et al. (2012). "Markerless visual control of a quad-rotor micro aerial vehicle by means of on-board stereo processing". In: *Autonomous Mobile Systems 2012: 22. Fachgespräch Stuttgart, 26. bis 28. September 2012.* Springer, pp. 11–20.

Seborg, Dale E et al. (2016). *Process dynamics and control.* John Wiley & Sons.

Sencer, Burak and Kosuke Ishizaki (2015). "Smooth polynomial interpolation for point-to-point trajectories with vibration avoidance". In: *IECON 2015-41st Annual Conference of the IEEE Industrial Electronics Society.* IEEE, pp. 002070–002075.

Siegwart, Roland, Illah Reza Nourbakhsh, and Davide Scaramuzza (2011). *Introduction to autonomous mobile robots.* MIT press.

Singhal, Gaurav, Babankumar Bansod, and Lini Mathew (2018). "Unmanned aerial vehicle classification, applications and challenges: A review". In.

Sivakumar, Mithra and Naga Malleswari TYJ (2021). "A literature survey of unmanned aerial vehicle usage for civil applications". In: *Journal of Aerospace Technology and Management* 13.

Subirana, Jaime Sanz, Manuel Hernandez-Pajares, and Jos℗ e Miguel Juan Zornoza (2013). *GNSS data processing: fundamentals and algorithms.* European Space Agency.

Sun, Sihao et al. (2020). "Upset recovery control for quadrotors subjected to a complete rotor failure from large initial disturbances". In: *2020 IEEE International Conference on Robotics and Automation (ICRA).* IEEE, pp. 4273–4279.

Swaroop, D et al. (2000). "Dynamic surface control for a class of nonlinear systems". In: *IEEE transactions on automatic control* 45.10, pp. 1893–1899.

Taketomi, Takafumi, Hideaki Uchiyama, and Sei Ikeda (2017). "Visual SLAM algorithms: A survey from 2010 to 2016". In: *IPSJ Transactions on Computer Vision and Applications* 9.1, pp. 1–11.

Tan, J et al. (2020). "Improved PRM algorithm for path planning of UAV". In: *Transducer Microsyst. Technol* 39, pp. 38–41.

Tanzi, Tullio Joseph et al. (2016). "Towards" drone-borne" disaster management: future application scenarios". In: *XXIII ISPRS Congress, Commission VIII (Volume III-8)*. Vol. 3. Copernicus GmbH, pp. 181–189.

Tassa, Yuval, Tom Erez, and Emanuel Todorov (2012). "Synthesis and stabilization of complex behaviors through online trajectory optimization". In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 4906–4913.

Tayebi, Abdelhamid, Andrew Roberts, and Abdelaziz Benallegue (2013). "Inertial vector measurements based velocity-free attitude stabilization". In: *IEEE Transactions on Automatic Control* 58.11, pp. 2893–2898.

Taylor, Brett and Anthony Choi (2014). "Fuzzy ant colony algorithm for terrain following optimization". In: *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, pp. 3834–3839.

Teleweck, PE and B Chandrasekaran (2019). "Path planning algorithms and their use in robotic navigation systems". In: *Journal of Physics: Conference Series*. Vol. 1207. 1. IOP Publishing, p. 012018.

Theile, Mirco et al. (2021). "UAV path planning using global and local map information with deep reinforcement learning". In: *2021 20th International Conference on Advanced Robotics (ICAR)*. IEEE, pp. 539–546.

Thomas, Justin et al. (2015). "Planning and control of aggressive maneuvers for perching on inclined and vertical surfaces". In: *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. Vol. 57144. American Society of Mechanical Engineers, V05CT08A012.

Thrun, Sebastian (2003). "Learning occupancy grid maps with forward sensor models". In: *Autonomous robots* 15, pp. 111–127.

Thrun, Sebastian and Michael Montemerlo (2006). "The graph SLAM algorithm with applications to large-scale mapping of urban structures". In: *The International Journal of Robotics Research* 25.5-6, pp. 403–429.

Thrun, Sebastian et al. (2002). "Robotic mapping: A survey". In.

Tsouros, Dimosthenis C, Stamatia Bibi, and Panagiotis G Sarigiannidis (2019). "A review on UAV-based applications for precision agriculture". In: *Information* 10.11, p. 349.

Udeanu, Gheorghe, Alexandra Dobrescu, and Mihaela Oltean (2016). "Unmanned aerial vehicle in military operations". In: *Sci. Res. Educ. Air Force* 18.1, pp. 199–206.

Vaidyanathan, Sundarapandian and Ahmad Taher Azar (2016). "A novel 4-D four-wing chaotic system with four quadratic nonlinearities and its synchronization via adaptive control method". In: *Advances in chaos theory and intelligent control*. Springer, pp. 203–224.

Valavanis, Kimon P and George J Vachtsevanos (2015). *Handbook of unmanned aerial vehicles*. Vol. 1. Springer.

Wang, Ning, Jiyang Dai, and Jin Ying (2021). "UAV formation obstacle avoidance control algorithm based on improved artificial potential field and consensus". In: *International Journal of Aeronautical and Space Sciences* 22.6, pp. 1413–1427.

Yan, Chao and Xiaojia Xiang (2018). "A path planning algorithm for uav based on improved q-learning". In: *2018 2nd international conference on robotics and automation sciences (ICRAS)*. IEEE, pp. 1–5.

Yan, Fei, Yi-Sha Liu, and Ji-Zhong Xiao (2013). "Path planning in complex 3D environments using a probabilistic roadmap method". In: *International Journal of Automation and computing* 10, pp. 525–533.

Yang, Liang et al. (2016). "Survey of robot 3D path planning algorithms". In: *Journal of Control Science and Engineering* 2016.

Yanyang, WANG, WEI Tietao, and QU Xiangju (2012). "Study of multi-objective fuzzy optimization for path planning". In: *Chinese Journal of Aeronautics* 25.1, pp. 51–56.

Yu, Jing, Zhihao Cai, and Yingxun Wang (2016). "The minimum jerk trajectory generation of a quadrotor based on the differential flatness". In: *2016 IEEE Chinese Guidance, Navigation and Control Conference (CGNCC)*. IEEE, pp. 1061–1066.

Yu, Jinglun, Yuancheng Su, and Yifan Liao (2020). "The path planning of mobile robot by neural networks and hierarchical reinforcement learning". In: *Frontiers in Neurorobotics* 14, p. 63.

Zakaria, MY, Moatassem M Abdallah, and M Adnan Elshafie (2012). "Design and production of small tailless unmanned aerial vehicle". In: *The International Conference on Applied Mechanics and Mechanical Engineering*. Vol. 15. 15th International Conference on Applied Mechanics and Mechanical Engineering. Military Technical College, pp. 1–26.

Zakariah, Azwaan et al. (2015). "Medium size dual-axis solar tracking system with sunlight intensity comparison method and fuzzy logic implementation". In: *Jurnal Teknologi* 77.17.

Zhang, Zhe et al. (2020). "A novel real-time penetration path planning algorithm for stealth UAV in 3D complex dynamic environment". In: *IEEE Access* 8, pp. 122757–122771.

Zhu, Lihua, Xianghong Cheng, and Fuh-Gwo Yuan (2016). "A 3D collision avoidance strategy for UAV with physical constraints". In: *Measurement* 77, pp. 40–49.

Zhuoning, Dong et al. (2010). "Study on UAV path planning approach based on fuzzy virtual force". In: *Chinese Journal of Aeronautics* 23.3, pp. 341–350.

Zulu, Andrew and Samuel John (2016). "A review of control algorithms for autonomous quadrotors". In: *arXiv preprint arXiv:1602.02622*.