POLITECNICO DI TORINO

**Master's Degree in
Mechatronic Engineering**

Master Thesis

3D Real-Time Energy Efficient Path Planning for a Fleet of Fixed-Wing UAVs

Relators
Prof. Alessandro Rizzo
Prof. Kimon P. Valavanis

Candidate
Giuseppe Aiello

Ai miei genitori.

Abstract

Path planning of unmanned aerial vehicle (UAV) targets at finding an optimal and collision free path in a cluttered environment while taking into account the geometric, physical and temporal constraints. The goal is to reach fully autonomous UAVs robust enough to work in every possible situation and under every possible source of uncertainty. The scientific community is increasingly focusing on this topic and a lot of works has been done in the last decades. It is important to try to go ahead in the research to understand what is the state of the art in order to locate which are the steps to be done to move forward.
This Thesis work starts showing the state of the art and detecting the challenges still unsolved. After that, a novel node-based algorithm based on the A* Search algorithm has been developed. The algorithm called EEA* has been developed in order to deal with 3D environments, being robust, energy efficient and working in real-time.
Along with the EEA*, a local path planner has been developed in order to cope with unknown dynamic threats in the environment. The UAV have been tested in different mountain-wise 3D environments, with different numbers of unknown dynamic obstacles.
Then, the algorithm has been employed in a multi agent environment, dealing with three UAVs working altogether, making them to perform their path safely. Lastly, the energy efficiency of the path planning algorithm has been tested and compared with the A* algorithm.

Contents

List of Tables

List of Figures

Part I

Introduction

Unmanned Aerial Vehicle (UAV) applications have been increasing in recent years, including surveillance, reconnaissance, search/destroy missions, aerial photography, and disaster monitoring. In this field of study a large set of problems has to be addressed; some of the most considerable ones are related to planning the optimal path, estimating the plume concentration, swarm control or ensuring fault detection in a manner that safety is guaranteed. Furthermore, many of these applications require the use of multiple drones together, either collaborating or working on different tasks at the same time. To make the UAVs able to complete those tasks it is necessary to control them in a proper way, possibly autonomously. Still autonomously, they have to be able to compute an optimal path to pass from a position to another without ending up in collisions either against other employed drones or against environment obstacles. To do this, path planning algorithms must cope with an environment as much realistic as possible.
Such an environment model has to be made by:

- **3D space**: Real world is a three dimensional world and since UAVs, differently from other kinds of robots, can move freely in all the three dimensions of the space, it is mandatory to develop something that take into account of the three dimensions of the space.

- **Static obstacles**: the environment in which the drones work is made by static objects, like mountains, rocks, trees, buildings for outdoor maps, and from walls, furniture and other architectural components for what concerns indoor maps. Static threats can be either already known or unknown.

- **Dynamic obstacles**: They are the all possible unpredictable and dynamically changing threats in the map. In some approaches other UAVs working in the same environment might be part of the dynamic obstacles. Dynamic threats are reasonably unknown and are avoided by some form of local path planning logic.

This research is focused on a specific kind of UAVs: fixed-wing UAVs. For fixed-wings further considerations has to be done. In fact, a path planning algorithm for fixed wing drones, unlike the ones intended for multi rotor UAVs, must compute a path smoothed enough to be followed by a drone not able to perform sharp turnings. Hence, the problem of respecting the kinematics and dynamics constraints of the system becomes even more relevant for those sort of UAVs.

1 Motivation and Rationale

Recently, path planning for UAVs has become a hot topic in the scientific community, due to the huge number of applications UAVs can be employed. Therefore, an increasing number of scientists focused their effort on developing effective and robust path planning algorithms to solve the problem of the optimal and collision free path in a cluttered environment. Nevertheless, from the optimization theory point of view, finding a 3D complete path is NP-hard problem, thus there exist no common solutions.
Over the past decades, a couple of methods are proposed to solve such problems. Algorithms implemented in 3D environments includes Visibility Graph which is developed from computer science; randomly sampling search algorithms such as Rapidly-exploring Random Tree, and Probabilistic Roadmap; optimal search algorithms like Dijsktra's algorithm, A*, and D*; bio-inspired planning algorithms and etc [20].
In the myriad of works and proposed solutions in this field, it is important to look at the state of the art and through a literature review, trying to understand what might be the limits of the current implementations. Detecting common drawbacks and common challenges it is intended to develop a solution which is a step forward to the current state of the art.

2 Summary of Contributions

The main contributions of this Thesis work is a literature review on the state of the art of UAV path planning algorithms and the development, implementation and testing of a robust real-time path planning algorithm with obstacle avoidance capabilities in a 3D space that minimizes the energy consumption of the UAV. The developed algorithm is implemented and run in MATLAB, in a 3D partly unknown cluttered environment with dynamically changing threats. The simulation is made for a fleet of UAVs, where every drone consider other ones as dynamic obstacles to be avoided along the path. The achievements and contributions are summarized as follows:

- Literature review about solutions to the problem of computing the optimal collision free path for UAVs, detecting advantages and limits of each one.
- Development and implementation of an A*-based global path planner that is an energy efficient kind of the A*, called EEA* that stands for Energy Efficient A*.
- Development and implementation of a local path planning logic to perform collision avoidance in real-time.
- Simulation of the proposed algorithm and comparison with the standard A* star version.
- Implementation and simulation of the algorithm as a decentralized multi-agent path planning algorithm.

3 Thesis Outline

The remainder of this dissertation is organized as follows: Part II presents a literature review on path planning algorithms for UAVs developed in the last 10-15 years, classifying the algorithms and detecting their advantages and limitations. Part III explains the problem statement, showing its importance and its complexity. Part IV shows a proposed solution with a detailed explanation on the work done and the tools used whereas Part V focuses on results of the simulations performed. In particular, the algorithm is simulated firstly for a single UAV, then for a fleet of three UAVs. Part VI summarizes the work performed in this dissertation and describes its limits and future work to move forward in the research.

Part II
Literature Review on Path Planning for UAVs

4 Introduction

In the last decades, the problem of path planning in the field of Robotics and in particular for Unmanned Aerial Vehicles (UAV) has had a growing relevance in the scientific community.
The following literature review about the work done could be useful to give an overview about what is the undertook way in this field of research, which methods are the more promising ones, and which are the possible challenges to deal with in order to do others step further.

5 Literature Review

5.1 Classification of the algorithms

The path planning problem for UAVs is implemented as an optimization problem such that it gains an optimal solution among all the possible ones. According to [37] (2019), two main steps for the process are the representation of the UAV in a 2D or 3D environment for the object and obstacle avoidance and the creation of a map or a graph that takes into account the specifications and the configuration of the UAV in the environment.
According to [28] (2016), the path planning algorithms can be divided in five main categories as depicted in 1:

- Sampling based algorithms
- Node based optimal algorithms
- Mathematical model based algorithms
- Bio-inspired algorithms
- Multi-fusion based algorithms

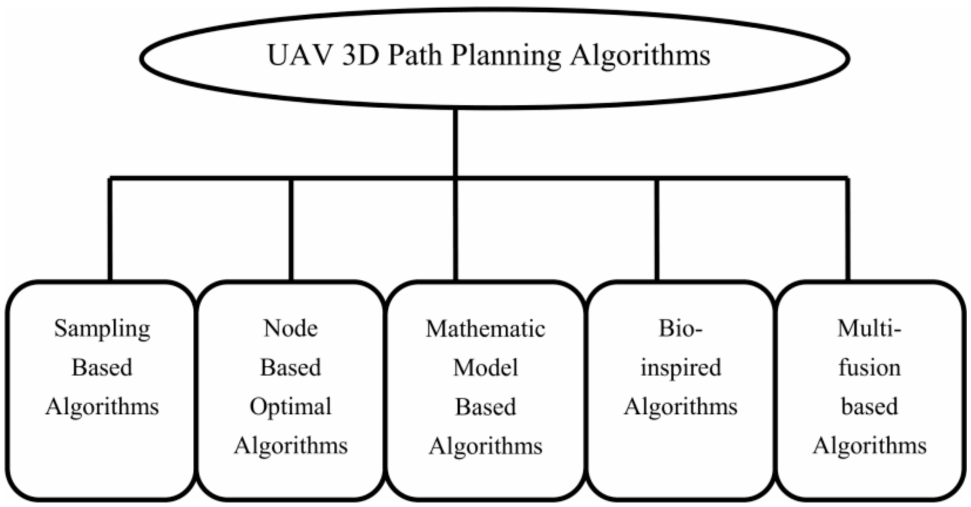

UAV 3D Path Planning Algorithms
Sampling Based Algorithms
Node Based Optimal Algorithms
Mathematic Model Based Algorithms
Bio-inspired Algorithms
Multi-fusion based Algorithms

Figure 1: Path planning taxonomy

Sampling based algorithms: The algorithms belonging to this method need an a-priori knowledge about the whole workspace, that is a mathematical representation of the workspace. The reason why they are called sampling based algorithms is that these kind of approaches divide the environment as a set of nodes, or cells, or other shapes. These approaches are appropriate for on-line implementation as they have a high time efficiency, with the ability to handle dynamic and static threats. Rapidly exploring Random Trees algorithm, Voronoi Algorithm, Probabilistic Roadmaps, Artificial Potential Fields are examples of algorithms belonging to this family of path planning approaches.
Sampling base algorithms can be further divide into two categories: *active* and *passive*. Active means algorithms such as RRT which can achieve the best feasible path to the goal by its own processing procedure. Passive, like PRM, are algorithms that only generate a road net from start to the goal, thus they need a combination of search algorithms to pick up the best feasible path in the net map in which there are many feasible paths.

Node based optimal algorithms: Node based optimal algorithms explore through the decomposed graph. From the search mechanism point of view, this family of algorithms share the same property of exploring among a set of nodes in the map, where information sensing and processing are already executed. This kind of approaches are able to always find an optimal solution to certain decomposition. Their results rely much on the preconstructed graph and can be combined with other methods to achieve global optimal. Some algorithms belonging to this category are Dijkstra's algorithm, A*, LPA*, D*, D*-Lite, Theta*, Lazy Theta* and so forth.

Mathematical model based algorithms: Mathematical model based algorithms include linear algorithms and optimal control. These methods model the environment as well as the system and then bound the cost function with all the kinematic and dynamic constraints bounds which are inequalities or equations to achieve an optimal solution. This method employs differential flatness to ensure control flatness along the reference path; it linearizes the nonlinear kinematic and dynamic constraints to form a rather simple form. This kind of algorithms gives an overall consideration to safety, reliability, and efficiency and then bounds the cost function tightly. Although this kind of methods loads a heavy computational burden on computer, it can perform well enough with the improvement of computer technology. Among these algorithms it can be found Mixed-Integer Linear Programming (MILP) and the Binary Linear Programming (BLP).

Bio-inspired algorithms: Bio-inspired algorithms originate from mimicking biological behavior to solve problems. This kind of planning methods leaves out the process of constructing complex environment models to search a near optimal path based on stochastic approaches. They are able to overcome the weakness where general mathematical model based algorithms often fails in solving in a heuristic approach NP-hard problems with a great amount of variables and nonlinear objective functions. Almost all bio-inspired algorithms have the shortcoming of having a long dealing period. Thus this kind of algorithms is suggested to work off-line, even though they can handle dynamic threats.
Bio-inspired algorithms can be divided in *Evolutionary Algorithms* and *Neural Networks*. Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Shuffled Frog Leaping Algorithm (SFLA), Grey Wolf Optimization (GWO) are all examples from this category.

Multi-fusion based algorithms: Multi-fusion base algorithms is an approach that combines several algorithms together to achieve a better performance. It is a recently arising approach. Fusing more algorithms together make them able to benefit each other. Algorithms implemented tend to fuse in a layer by

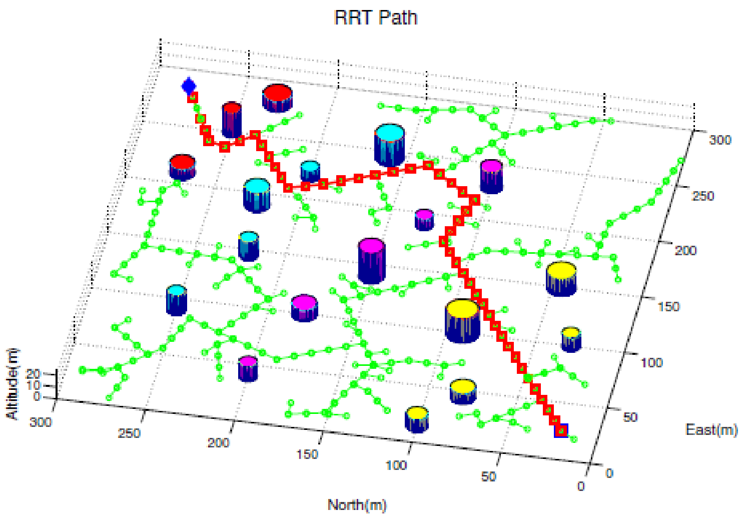


Figure 2: An example of the RRT algorithm from [6]

layer way, aiming to plan an optimal path. For example, Artificial Potential Field algorithms usually tend to drop into local minima without navigation function or other tricks. Probabilistic Road Maps also cannot generate an optimal single path by itself. These algorithms altogether are designed to work real time, with strong environmental adaption.

5.2 Sampling Based Algorithms

In [6] (2008), a UAV path planner has been developed using the RRT algorithm, combining biased sampling and greedy extension of nodes. Since this path consists of piece-wise linear segment, smoothing processes has been needed. A continuous curvature path smoothing algorithm has been developed satisfying non-holonomic constraints. The angle between two waypoints was the only required information for the generation of continuous curvature path. Thus, the suggested algorithm was very simple and fast enough to be executed online path planning. Finally, this algorithm has been extended to 3-dimensional space.
In [22] (2015), a new 3D structural inspection path planning algorithm has

been proposed. The algorithm is capable of computing optimized paths for both holonomic and non-holonomic vehicles and supports multiple cameras while respecting further mission constraints. Employing two different aerial robots, a rotorcraft as swell as a fixed-wing vehicle, the algorithm has been practically tested in four different scenarios relevant with both holonomic and non-holonomic configurations, nadir- and oblique-camera views for close proximity as well as large-scale 3D mapping applications. With the support of state-of-the-art 3D reconstruction software, the recorded inspection datas have been postprocessed and practical full coverage, very dense and high-quality point clouds and triangular meshes have been derived.
In [23] (2015), a hybrid approach has been employed for near-optimal solution in a 2D environment. Then a third dimension is managed in an online manner with a fuzzy controller working simultaneously with a Lazy Theta* algorithm. They have been able to find the best path to overcome obstacle changing the UAV's altitude online using data from the sensors and topographical database. The author have tested the algorithm in 500 randomly generated maps and compared it to the Lazy Theta* algorithm, showing a very little improvement, but according to them, it is always convenient to choose it between the two due to its significantly better performance in some maps and since the cost of implementing and running the algorithm is low.

5.3 Node Based Optimal Algorithms

In [7] (2010), a Fuzzy Virtual Forces (FVF) algorithm is used. To meet real-time requirements a fixed-step method has been implemented and to solve the local minima problem, that happens often in VF algorithms, the combination of threats and adjacent matrix have been integrated in the algorithm. Furthermore, they added the adaptive proportion coefficient based on Bayesian belief network and fuzzy logic reasoning, that can be adapted to the change of the environment, in this case a battlefield. Finally, the algorithm has been compared with the A* algorithm, showing better performance than A*.
In [34] (2019), it has been proposed a modified A* algorithm to be combined with the Global Navigation Satellite System (GNSS) error distribution obtained by implementing the GNSS. Even if the UAV is able to move in a 3D environment, actually the A* algorithm has been employed for the 2D path planning, whereas for the third dimension it has been added an height selection.

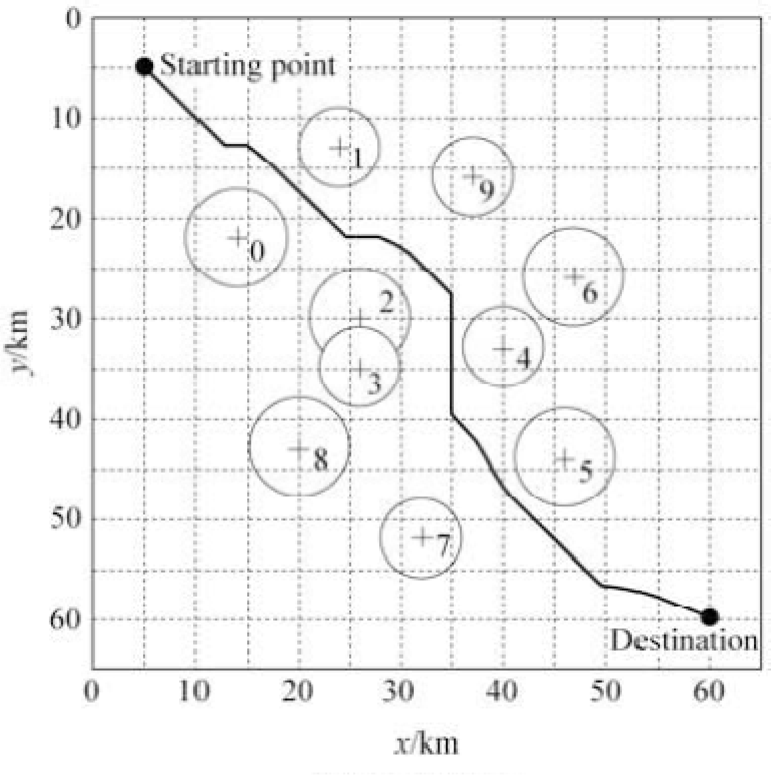


Figure 3: An example of the Fuzzy Virtual Forces algorithm from [7]

In [38] (2019), researchers have proposed an Evolutionary Optimization Algorithm based on improved t-distribution to deal with high computational complexity. The experiments have been conducted in a 3D unknown complex environment.

5.4 Mathematical Model Based Algorithms

In [11] (2012), to calculate paths of UAVs to form a communication chain between the start and end nodes, mixed-integer programming for control (MILP) has been used due to its ability to give a global solution and to take into account non-convex constraints. It has been successfully demonstrated the ability to plan paths that achieve the desired communication topology, while minimizing fuel consumption, avoiding collision and no-fly zones, and

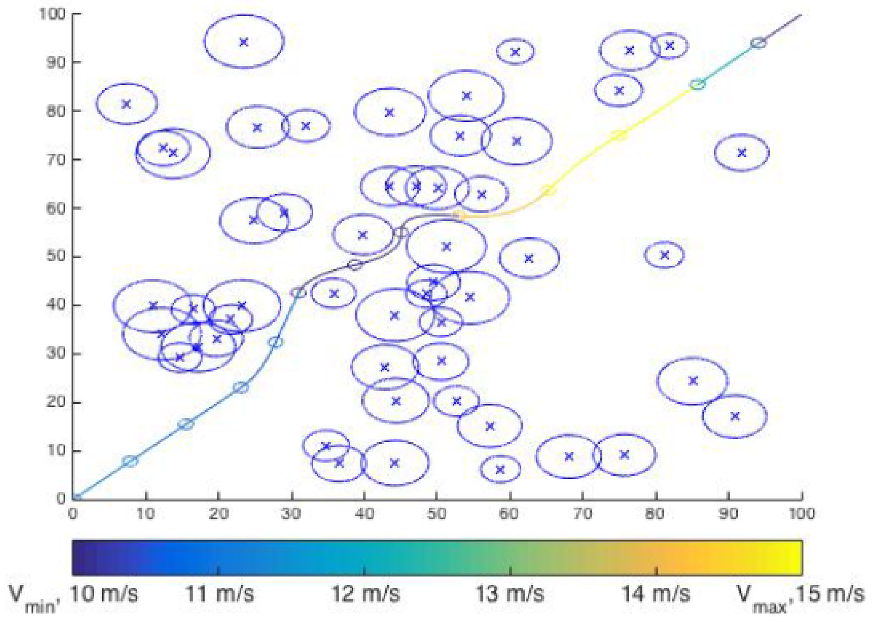


Figure 4: An example of Receding Horizon Technique from [25]

satisfy altitude constraints with respect to the terrain. Furthermore, it has had an iterative manner improved the accuracy of the achievable communication rate by simulating the planned path in the radio propagation program SPLAT!.

In [12] (2012), flatness techniques have been used to formulate the problem of trajectory planning into an optimization problem in order to determine a-priori if a specified trajectory is feasible and to change the trajectory parameters according to the system constraints. In this work it has been proposed a trajectory planning/replanning technique that can be implemented on real-time applications. To reduce calculation requirements and computational load, the simplified model has been employed to replace the constraints of the optimization problem by their extrema. Replacing the constraints with their algebraic extrema drastically reduced the computation times. Nevertheless, using the simplified models in the trajectory planning increased the uncertainties and the mismatch with the system.

In [14] (2013), a BLP based approach is proposed, including a BLP model and a solution algorithm embedded with heuristic strategies. The model

divides the planning objective into two levels to improve the presented abilities in a cooperative and competitive manner. In the planning process, threat environment is detected in real-time, and the path is updated at variable time intervals under the constraints of the UAV's relative kinematic properties.
In [15] (2013), it has been given an adaptability description of path planning. The adaptability plays a crucial part in real-time path planning, considering the changes or fluctuations of a UAV's performances in different flight missions or at different stages of a mission. It has been introduced its description in the proposed planner to meet more planning requirements in practice. Then, it has been proposed a BLP-based real-time path planner to generate reference waypoints and control inputs at variable planning time intervals. It can address the performance variations and adaptively plan smooth flight paths only when necessary. Lastly, discrete flight path search algorithm for the BLP-based planner is showed. The characteristics of BLP and path planning procedure are both considered in the planner's implementation process, so as to construct a feasible and efficient discrete search algorithm for the adaptive planner.
In [24] (2016), a path planning scheme has been introduced that enables the online planning of good paths to explore a given bounded volume in a receding horizon manner. It has been shown that with minor adaptation of the objective function, the proposed planning method finds good paths for the inspection of an a-priori known surface in either known or unknown environment. These different scenarios have been evaluated and further validated in real world experiments using rotorcraft MAVs. Moreover, analysis on the computational complexity has been provided and the good scaling properties with respect to the scenario size have further been highlighted in the simulation scenarios of different scale.
In [25] (2016), UAV path planning has been modeled as a single objective optimization problem that utilizes a receding horizon approach and quadratic Bézier curves, where the path has been constrained to avoid obstacle collision and to account for flight aerodynamic constraints. The proposed method is gradient based, allowing for quick and robust convergence to a near optimal solution. This heuristic method converges closely to full-knowledge optimal solutions, lessening the risk to the safety of others and the safety of the UAV.
In [32] (2018), it has been proposed to integrate active perception in a receding horizon setting for a goal reaching task. In particular, it has been designed a perception-aware receding horizon navigation system using a single forward looking camera for MAVs. It has been used a monocular visual odometry

SVO and a dense reconstruction algorithm REMODE to provide the essential information for navigation. Using the information, it has been generated a library of trajectories and evaluated them in terms of collision probability, perception quality and goal progress to select the next motion for MAVs, which naturally combines different performance metrics. In addition of the capability of avoiding obstacles, the perception-aware receding horizon navigation system it has been able to select motion to favor the state estimation accuracy, which is especially advantageous in environments with visually degraded regions.
In [35] (2019), it has been proposed the approach of belief uncertainty-aware planning to achieve consistent exploration and it has been presented an architecture to achieve this goal. More specifically, it has been elaborated the steps of such a process, formulated the framework to properly interconnect them, it has been analyzed the computational complexity of such an algorithm. Furthermore, it has been also presented a complete robotic implementation that can support the proposed functionality, in cases of simple mockups as well as field-deployments in DVE conditions. Finally, it has been experimentally demonstrated that the proposed receding horizon, two-step, planning paradigm manages to explore different unknown environments with consistency, by following uncertainty optimizing trajectories that are derived through a belief-space propagation process operating onboard a Micro Aerial Vehicle robot.

5.5 Bio-Inspired Algorithms

In [5] (2007), an evolutionary algorithm-based path planner for UAVs has been presented to find a path line with desired features in different 3D terrain models. The proposed method uses Bézier curves to represent a continuous path line, which is described by a small set of parameters - the coordinates of the control points. The construction of Bézier curves based on control points, are suitable for coupling with a Genethic Algortithm (GA). The resulting path is smooth and assumed to be easily generated by autonomous navigation controllers. The Vibrational Genethic Algorithm (VGA) used is able to construct feasible path lines under the prescribed constraints such as the path length, the turn angle, and the clearance between the path and the boundary (terrain) within an acceptable time period, for the online planner as well, where the execution time is important. The added vibrational mutation

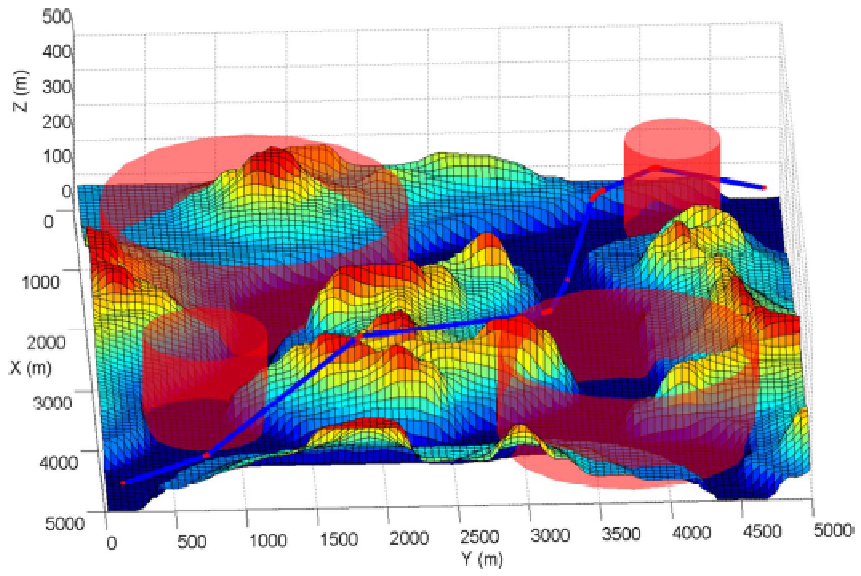


Figure 5: An example of the Genetic Algorithm from [16]

operator results in the diversity within population during the reproduction process, that normally is obtained in a high population number, but generating more cost function calculations. VGA shows more efficiency in low population numbers. Therefore, vibrational mutation technique is clearly an efficient technique especially in low population rates.
In [16] (2013), it has been used two non-deterministic algorithms, the Genetic Algorithm(GA) and the Particle Swarm Optimization(PSO), to produce solutions in a relatively short computation time. The researchers further reduced the execution time by developing parallel versions of these algorithms. After achieving a quasi-linear speedup of 7.3 on 8 cores and an execution time of 10 s for both algorithms, it has been concluded that by using a parallel implementation on standard multicore CPUs, real-time path planning for UAVs is possible. Moreover, rigorous comparison of the two algorithms shows, with statistical significance, that the implementation of the GA produces superior trajectories than the implementation of the PSO when using the same encoding.
In [18] (2013), a new approach to improve coverage trajectories has been employed. A novel algorithm denoted by Harmony Search has been studied

to be applied to the optimization of agricultural management tasks. The key feature of this approach is that it is able to reduce the number of turns of the coverage trajectories by holding the former start and goal positions set previously. Although, the computation time is high, the mission planner's aim is not to work on-line. The mission completion time has been reduced by minimizing the number of turns, improving safety for the operator and UAVs.
In [27] (2016), it has been proposed a Hybrid Genetic Algorithm (HGA) to solve the path planning problem in non-convex environment. The method proposed a Multi-Population Genetic Algorithm (MPGA) with visibility graph. A total of 50 maps were generated to analyze the performance of HGA, then the computational results have been compared against an exact (CPLEX) and heuristic (CSA) methods. The simulations showed good robustness and the capability to find a solution within 10 seconds. In spite of being run for a short time, the quality of HGA solutions are quite similar to the solutions reached in much more time by CPLEX and CSA for many maps. Nevertheless HGA have given back ineffective results in terms of fuel consumption due to the high accelerations required to follow the path.
In [39] (2019), it has been designed a complete UAV surveillance system. An Artificial Neural Network (ANN) is used to make the control structure easier to be used, to reject the disturbances and to reduced the control parameters to be controlled. Furthermore, with a combination of K-agglomerative clustering, a Set-based Particle Swarm Optimization (S-PSO) and an A* algorithm, a path can be efficiently planned predicting and reducing the energy consumption as well.
In [42] (2020), three different path planning algorithms has been implemented for biological control. In more details, the biological control in this case concerns the capsule depositions sites of areas of any shape, even non-convex, in agricultural fields, in order to cover the entire area. Such capsules contains natural enemies used to fight against exotic pests that causes huge damages to the agricultural sector. The algorithm implemented have been Ant Colony Optimization (ACO), Guided Local Search (GLS) and Lin-Kernighan (LKH). LKH is the one that gave better results in term of RAM consumption, execution time, number of memory used and distance travelled.

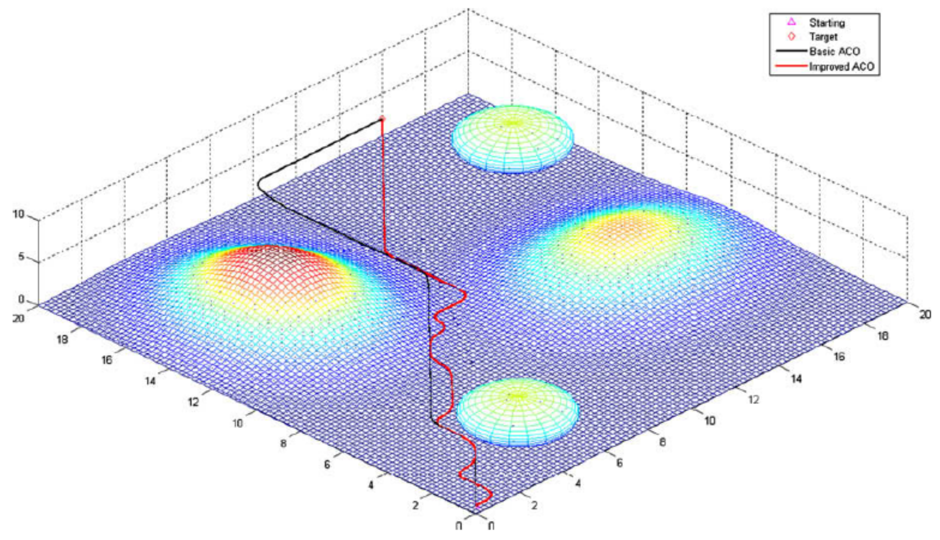


Figure 6: An example of a hybrid algorithm using Ant Colony Optimization integrated with Differential Evolution algorithm from [8]

5.6 Multi-Fusion Based Algorithms

In [8] (2010), it has been presented a hybrid meta-heuristic Ant Colony Optimization (ACO) and Differential Evolution (DE) algorithm approach for Uninhabited Combat Aerial Vehicles (UCAV) three-dimension path planning in combat field environments. More precisely, a novel type of ACO model has been described and the DE is applied to optimize the pheromone trail of the improved ACO model during the process of ant pheromone updating. A k-trajectory has been added to smooth the computed path to make it more feasible.
In [9] (2011), it has been presented an hybrid algorithm to deal with path re-planning problem. The algorithm chosen are the Virtual Force (VF) algorithm for its simplicity and the A* algorithm for its robustness and efficiency. Merging them, authors have been able to combine the advantages of the two and comparing it with an usual A* algorithm they highlighted the more effectiveness of the developed hybrid approach.
In [17] (2013), it has been proposed a 3D path planning method for UAV navigation in complex environments. In this study, it has been used octree

algorithm to divide the work space into voxels. The free voxels, which have enough space tolerance to satisfy the safety needs of the UAV, are sorted to represent the free space of the environment. In order to obtain the connectivity of free voxels effectively, the environment has been divided into several regions called bounding boxes, and the evaluation of free voxels connectivity has been carried out in each bounding box instead of in the whole space. Since the basic PRM uses random sampling strategy to choose nodes in the entire free space, which causes the narrow-passage problem that may lead to failure of the path planning algorithm, the authors have improved the PRM by random sampling in bounding boxes to ensure a more reasonable distribution in the 3D space. Finally, based on the voxel connectivity, the selected nodes composed a roadmap, which has been applied for path searching by A* algorithm for feasible path.
In [19] (2014), it has been implemented an hybrid metaheuristic approach combining together the Genethic Algorithm (GA) and the Particle Swarm Optimization (PSO) to obtain the advantages of both the algorithms. The disadvantage that has to be mentioned is that this approach modifies the convergence properties of the single algorithm and the convergence of the new algorithm remains unproven. The implementation of the algorithms has been carried out in a parallel manner and they are able, according to the author, to be employed in real-time. Lastly, a comparison with a GA and a PSO against the GA-PSO implementation has been done, showing a better performance of the proposed one almost in every map.
In [26] (2016), it has been proposed a novel approach that utilizes the formulation of dynamic Bayesian, Distance Based Value Function(DBVF) and Grey Wolf Optimization(GWO) Algorithm to solve the problem of path planning and collision avoidance for UAV in presence of fixed and moving obstacles in uncertain environment. A probabilistic framework has been used for solving the problem because of the fact that the trajectories of the intruder aircraft may not be known to the path planner. The problem solved in this paper is relevant to the problem of UAV integration in the National Airspace System (NAS) in the presence of fixed obstacles and intruder aircraft. The paper assumes the knowledge of location of obstacles and other aircraft, which in real world scenario, would be obtained from ADS-B and ground-based radar. Upon intruding in the region close to the UAV, cells around the UAV would change their weights based on the probability values representing the risks of collision. The path planning problem is then formulated as an optimization problem with obstacles (both stationary and dynamic ones) incorporated as

constraints. The cost function used for the optimization comprises of two components: total distance of flight and a safety metric modeled as weights of cells representing the risk of collision.
In [29] (2017), four algorithms have been selected to study their robustness in the case of uncertain position, performing experiments under the uncertainty condition. The authors selected one algorithm of each different method. For the random sample class they chose RRT algorithm, for the fixed sample class it has been chosen A* algorithm, for the mathematical model based class BLP has been chosen and lastly the PSO for the computational intelligence class. Generally speaking, RRT and BLP performs better considering position uncertainty. In the case of low variance, BLP's planning result is the best among all 4 algorithms. RRT behaves better in the case of high variance and low complexity of scenario.
In [30] (2017), a Genethic Algorithm (GA) has been implemented along with a Multi-Objective Path Planning (MOPP) Algorithm. They have been used for area coverage and target detection. The GA aims to minimize the completion time, which includes the time to find the target and the time to set up a communication path. Moreover, depending on the mission demand, the algorithm can be tuned in such a way to prioritize coverage or connectivity.
In [33] (2018), it has been implemented a hybrid approach to a quadrotor. With the use of the mathematical model enriched by a Q-learning method they have been able to reach the goal after a reinforcement learning phase. The method has been implemented only for a 2D environment, but according to the authors it can be easily enough implemented for a 3D one.
In [36] (2019), the authors calculated the shortest path for UAVs with two-dimensional(2D) path planning algorithms in the environment including obstacles, in order to make the robots able to perform their tasks as soon as possible in the environment. Therefore, the aim of this paper it has been to avoid obstacles and to find the shortest way to the target point. The chosen algorithm have been A* algorithm and Dijkstra algorithm. Real time tests have been performed on UAV obtaining that the two path length algorithms show the same path length but only in time saving A* algorithm has been showed better performances than Dijkstra algorithm.
In [40] (2019), the authors have realized a Lazy Theta* implementation for real time usage for autonomous exploration in a large environment. To achieve this goal, two optimization have been introduced: the voxels composing the neighborhood of another voxel are calculated taking into account the sparse grid that represents the world and the obstacle detection calculations are

reduced by restricting the space discretization of the flight corridor to two dimensions.
In [43] (2020), a Grey Wolf Optimization algorithm has been enriched with a Reinforcement Learning method. This RLGWO algorithm is able to cope with a 3D flight environment and the operations has been developed for each individuals: exploration, exploitation, geometry adjustment and optimal adjustment. According to the accumulated performance controlled by the reinforcement learning, the individual switches operation adaptively. A cubic B-spline curve has been used to smooth the generated flight path. The authors compared the developed algorithm with other kind of GWO algorithm, showing its statistically superiority.

5.7 Challenges and Opportunities

It is useful, to go further in the research, to classify and detect possible problems and drawbacks related to the implementations presented in the literature, in order to understand what might be the challenges and the opportunities in this field of research.
Some main challenges has been identified:

1. *3D*: since it is aimed to use the proposed solution in a 3D real world environment, to have a satisfying work, it is needed to develop and test it in such a situation. Many of the works done in the near future are able to manage with 3D environments using a real 3D approach or subdividing the problem using a 2.5D, though some algorithms do not cope with this issue yet.

2. *Real Time*: the environment in which the UAV is intended to move is a dynamically changing one. Unexpected situations may occur and the world in which the UAVs fly may change in a significant way during the flight. Therefore it is necessary to aim to an algorithm capable of being implemented in an online fashion. Despite the relevance of obtaining this feature, not all the algorithms proposed are intended to be run online due to their high computational burden.

3. *Energy Efficiency*: dealing with fuel consumption is mandatory and path planning algorithms that, for issue, require high accelerations to be accomplished are more ineffective in terms of energy consumption.

Managing the energy is probably one of the least treated arguments, but by no means it is less important than other aspects.

4. *Robustness*: a path planning algorithm has to be robust and has to provide always reliable solutions, such that to not ending in local minima, having a convergence property, giving at least a near optimal result. Furthermore, the UAVs have to show the capabilities of tolerating position sensitive device errors, the rotation driving errors, linear driving errors during path planning.

5.8 Remarks

The works done in the last decade showed the will of the Scientific Community to realize 3D path planning algorithms due to their ability to cope with a real world environment, even though the computational burden often does not permit a real-time performance since it is an NP-hard problem. That is why an increasing interest has been manifested to heuristic approaches like bio-inspired algorithms, achieving an online performance at the expense of an optimal solution. Other researches opted for 2.5D solution able to reasonably deal with a 3D environment, reducing the computational cost.
One of the least treated aspect by many works but not less important than others is the energy efficiency issue. Fuel consumption is a crucial issue for a UAV, mostly for Fixed-Wing UAVs which are often intended to cover long distances. Most of the research that tried to manage this problem treated it as a value to be optimized in an optimization function.
In recent years multi-fusion algorithms look like the main trend, thanks to their ability to merge the advantages of more kind of algorithms, achieving a better performance than if they were used singularly. It is also true that this kind of approach often might lack the robustness and the reliability of a single algorithm by its own. More researches has to be conducted in this sense for some certain multi-fusion algorithm to guarantee certain necessary properties, like convergence.

As far as we know, none of the paper done met all the challenges proposed in 5.7 in one path planning algorithm.

REF	UAV	ALG	2D	3D	REAL-TIME	LIMITATIONS
[5] (2007)	Generic	VGA		✓	✓	3
[6] (2008)	Helicopter	RRT	✓	✓	✓	3,4
[7] (2010)	Generic	FVF	✓		✓	1
[8] (2010	Generic	ACO & DE		✓	✓	2
[9] (2011)	Generic	HVFA	✓		✓	1,3
[11] (2012)	Generic	MILP	✓			1,2
[12] (2012)	Quadrotor	Flatness-based	✓		✓	1,4
[14] (2013)	Fixed-Wing	BLP	✓		✓	1
[16] (2013)	Fixed-Wing	GA, PSO		✓	✓	3,4
[17] (2013)	Quadrotor	PRM & A*	✓	✓	✓	3,4
[18] (2013)	Quadrotor	HS	✓			1,2,3
[19] (2014)	Generic	GA & PSO		✓	✓	4
[22] (2015)	Fixed-Wing, Quadrotor	RRT*		✓	✓	3
[23] (2015)	Generic	BF Lazy Theta*	✓	✓	✓	3
[24] (2016)	Quadrotor	RRT		✓	✓	3
[25] (2016)	Quadrotor	Receding Horizon	✓		✓	1
[26] (2016)	Quadrotor	GWO	✓			1,2,3
[27] (2016)	Fixed-Wing	HGA	✓			1,2,3
[30] (2017)	Generic	MOPP & GA	✓			1,3,4
[33] (2018)	Quadrotor	Q-Learning	✓			1,2,3,4
[34] (2018)	Quadrotor	Modified A*	✓	✓		2,3
[35] (2019)	Hexarotor	Receding Horizon		✓	✓	3
[36] (2019)	Quadrotor	A*, Dijkstra	✓		✓	1,3
[38] (2019)	Generic	EA		✓	✓	3
[39] (2019)	Hexarotor	A* & S-PSO		✓	✓	4
[42] (2020)	Quadrotor	ACO, GLS, LKH	✓			1,2,3
[43] (2020)	Generic	RLGWO		✓		2,4

Table 1: Summary of the review of path planning for UAVs. The columns represent the referenced work and the year published (REF), the type of UAV under control (UAV), the algorithms employed (ALG), whether it is applied for a 2D environment (2D) and/or for a 3D one (3D), if it has real-implementation (REAL-TIME), the challenges not satisfied in the research (LIMITATIONS).

Part III
Problem Statement

6 Mathematical Model of a Fixed-Wing UAV

Firstly, it is needed to describe in a rigorous and mathematical way the model of a Fixed-Wing UAV, the coordinate frames used and the equations governing the system to be controlled. Starting from the coordinate frames and their description, it will be showed and explained the concepts of airspeed, wind speed and ground speed, followed by the kinematics and dynamics of the fixed-wing UAV, deriving their equation systems, as done in [13].

6.1 Coordinate Frames

In this part, we are going to derive the dynamic behavior of a fixed-wing Unmanned Aerial Vehicles (UAV). To accomplish this task, several coordinate frames are of interest. Assuming a flat, non-rotating earth, valid assumption for small UAVs, we are going to define: the inertial frame, the vehicle frame, the vehicle-1 frame, the vehicle-2 frame, the body frame, the stability frame and the wind frame. The first two are related by a translation, whereas the others by rotations.

6.1.1 The Inertial Frame F^i

The inertial coordinate system is an earth-fixed coordinate system. As shown in figure 7, the unit vector i^i is directed north, j^i is directed east, k^i is directed towards the center of the earth, or in general downwards. This reference frame is also called north-east-down (NED) reference frame.

6.1.2 The Vehicle Frame F^v

The only difference between the inertial frame and the vehicle frame is that the second one is centered in the center of mass of the UAV, while the axis of F^v are aligned to the axis of F^i. The vehicle frame is shown in figure 8

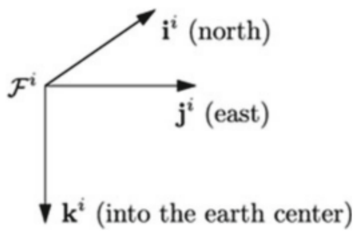


Figure 7: The inertial reference frame

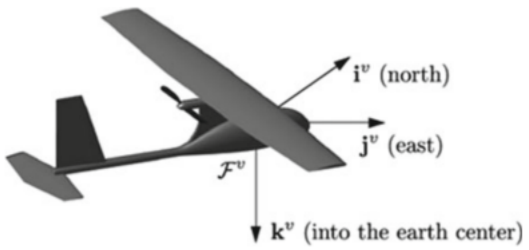


Figure 8: The vehicle reference frame

6.1.3 The Vehicle-1 Frame F^{v1}

The origin of the vehicle-1 frame is the same of the vehicle frame, the center of mass of the aircraft. Nevertheless, the F^{v1} frame is rotated in the positive right-handed direction about k^v by the *heading* (yaw) angle ψ, in such a way that i^{v1} points out of the nose of the airframe, j^{v1} out of the right wing and k^{v1} is aligned with k^v pointing into the earth, as depicted in figure 9.

The transformation from F^v to F^{v1} is given by

$$p^{v1} = R_v^{v1}(\psi)p^v \tag{1}$$

where

$$R_v^{v1}(\psi) = \begin{pmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2}$$

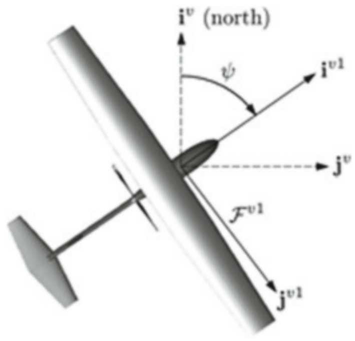


Figure 9: The vehicle-1 reference frame

6.1.4 The Vehicle-2 Frame F^{v2}

The origin of this reference frame is always the center of mass of the UAV and it is obtained by rotating the vehicle-1 frame in a right-handed rotation about the j^{v1} axis by the *pitch* angle θ. The unit vector i^{v2} points out of the nose of the aircraft, j^{v2} points out of the right wing and k^{v2} points out of the belly, as depicted in figure 10.

The transformation from F^{v1} to F^{v2} is given by

$$p^{v2} = R_{v1}^{v2}(\theta)p^{v1} \tag{3}$$

where

$$R_{v1}^{v2}(\theta) = \begin{pmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{pmatrix} \tag{4}$$

6.1.5 The Body Frame F^b

The body frame is obtained rotating the vehicle-2 frame in a right-handed rotation about i^{v2} by the *roll* angle ϕ. Therefore, i^b points out if the nose airframe, j^b points out of the right wing and k^b points out of the belly. The body frame is shown in figure 11.

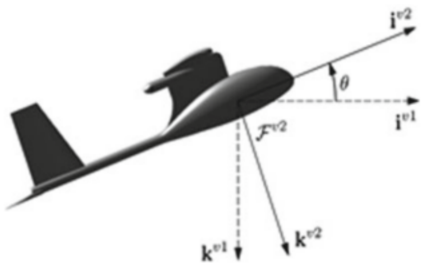


Figure 10: The vehicle-2 reference frame

The transformation from F^{v2} to F^b is obtained by

$$p^b = R^b_{v2}(\phi)p^{v2} \tag{5}$$

where

$$R^b_{v2}(\phi) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{pmatrix} \tag{6}$$

The transformation from the vehicle frame to the body frame is given by

$$\begin{aligned} R^b_v(\phi,\theta,\psi) &= R^b_{v2}(\phi)R^{v2}_{v1}(\theta)R^{v1}_v(\psi) \\ &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{pmatrix} \begin{pmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{pmatrix} \begin{pmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \\ &= \begin{pmatrix} c_\theta c_\psi & c_\theta s_\psi & -s_\theta \\ s_\phi s_\theta c_\psi - c_\phi s_\psi & s_\phi s_\theta s_\psi + c_\phi c_\psi & s_\phi c_\theta \\ c_\phi s_\theta c_\psi + s_\phi s_\psi & c_\phi s_\theta s_\psi - s_\phi c_\psi & c_\phi c_\theta \end{pmatrix} \end{aligned} \tag{7}$$

where $c_\phi \triangleq \cos\phi$ and $s_\phi \triangleq \sin\phi$. The angles ϕ, θ and ψ are commonly referred to as Euler angles.

6.1.6 The Stability Frame F^s

We refer to the velocity of the aircraft relative to the surrounding air as V_a, the *airspeed vector*, and its magnitude is called airspeed. To generate lift the wings of the airframe has to fly at a positive angle with respect to the airspeed

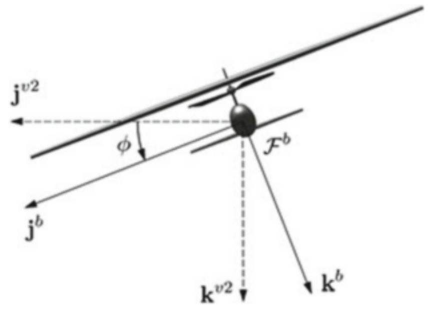


Figure 11: The body reference frame

vector. Such angle is called *angle of attack*, denoted by α. As depicted in figure 12 as a left-handed rotation about j^b and is such that i^s aligns with the projection of V_a onto the plane spanned by i^b and k^b.

Since α is given by a left-handed rotation, the transformation from F^b to F^s is given by

$$p^s = R_b^s(\alpha)p^b \tag{8}$$

where

$$R_b^s(\alpha) = \begin{pmatrix} cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{pmatrix} \tag{9}$$

6.1.7 The Wind Frame F^w

The angle between the airspeed vector and the $i^b - k^b$ plane is called the *side-slip angle*, denoted by β. As shown in figure 13, the wind frame is obtained by rotating the stability frame by a right-handed rotation of β about k^s. The unit vector i^w is aligned with the airspeed vector V_a.

The transformation from F^s to F^w is obtained doing

$$p^w = R_s^w(\beta)p^s \tag{10}$$

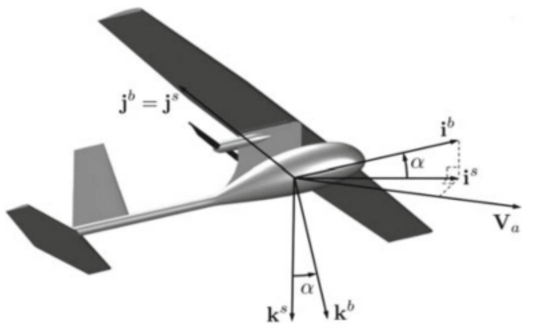


Figure 12: The stability reference frame

where

$$R_b^s(\beta) = \begin{pmatrix} cos\beta & \sin\alpha & 0 \\ -\sin\beta & \cos\beta & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{11}$$

The total transformation from the body frame to the wind frame is given by

$$\begin{aligned} R_b^w(\alpha,\beta) &= R_s^w(\beta) R_b^s(\alpha) \\ &= \begin{pmatrix} \cos\beta & \sin\beta & 0 \\ -\sin\beta & \cos\beta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{pmatrix} \\ &= \begin{pmatrix} c_\beta c_\alpha & s_\beta & c_\beta s_\alpha \\ -s_\beta c_\alpha & c_\beta & -s_\beta s_\alpha \\ -s_\alpha & 0 & c_\alpha \end{pmatrix} \end{aligned} \tag{12}$$

Alternatively, the transformation from the wind frame to the body frame is

$$R_w^b(\alpha,\beta) = (R_b^w)^T(\alpha,\beta) = \begin{pmatrix} c_\beta c_\alpha & -s_\beta c_\alpha & -s_\alpha \\ s_\beta & c_\beta & 0 \\ c_\beta s_\alpha & -s_\beta s_\alpha & c_\alpha \end{pmatrix} \tag{13}$$

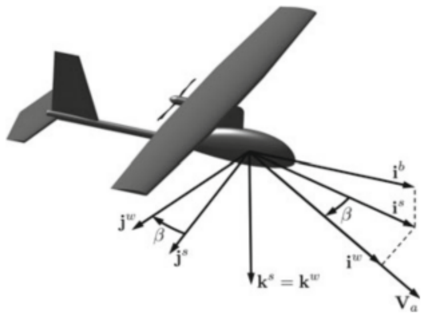


Figure 13: The wind reference frame

6.2 Airspeed, Wind Speed and Ground Speed

Dealing with small UAVs, the wind velocity can easily be in the range of $20-50\%$ of the airspeed velocity. Since wind is almost always present, we must carefully distinguish between airspeed, represented by the velocity with respect to the surrounding air V_a and the *ground speed*, represented by the velocity with respect to the inertial frame V_g. These velocities are related by the expression

$$V_a = V_g - V_w \tag{14}$$

where V_w is the wind velocity relative to the inertial frame.
The relationship among the the groundspeed vector, the airspeed vector and the wind vector is called the *wind triangle*.
The UAV velocity V_g can be expressed in the body frame in terms of components along the i^b, j^b and k^b axes.

$$V_g^b = \begin{pmatrix} u \\ v \\ w \end{pmatrix} \tag{15}$$

The direction of the ground speed vector relative to an inertial frame is specified using two angles, the *course angle* χ and the *flight path angle* γ. The flight path angle γ is the angle between the horizontal plane and the ground velocity vector V_g, whereas the course angle χ is the angle between the projection of the ground velocity vector onto the horizontal plane and the true north, as shown in figure 14.

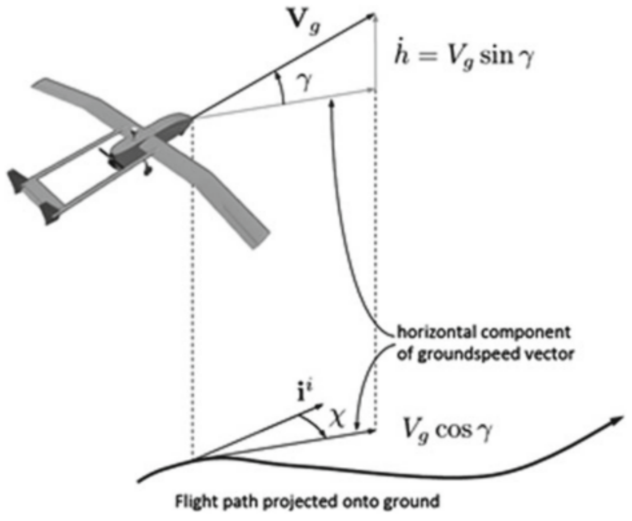

$\mathbf{V}_g$
γ
$\dot{h} = V_g \sin\gamma$
horizontal component of groundspeed vector
$\mathbf{i}^i$
χ
$V_g \cos\gamma$
Flight path projected onto ground

Figure 14: The flight path angle γ and the course angle χ

6.3 Kinematics and Dynamics

In developing the equations of motion for a UAV, twelve state variables will be introduced. There are three position states and three velocity states associated with the translational motion of the UAV. Similarly, three angular position and three angular velocity states associated with the rotational motion. The states variables are listed in the table 2 and shown schematically in figure 15.

The north-east-down positions (p_n, p_e, p_d) are defined relative to the inertial frame. The linear velocities (u,v,w) and the angular velocities (p,q,r) of the UAV are defined with respect to the body frame. Euler angles, roll ϕ, pitch θ and heading(yaw) ψ, are defined with respect to the vehicle-2, vehicle-1 and vehicle frames, respectively. Due to the fact they are defined relative to intermediate frames of reference, it cannot be said that angular rates (p,q,r) are simply the time derivatives of the attitude angles (ϕ,θ,ψ).

6.3.1 Kinematic Equations

To derive the equations we start doing some considerations. As previously said, the components u,v and w correspond to the inertial velocity of the

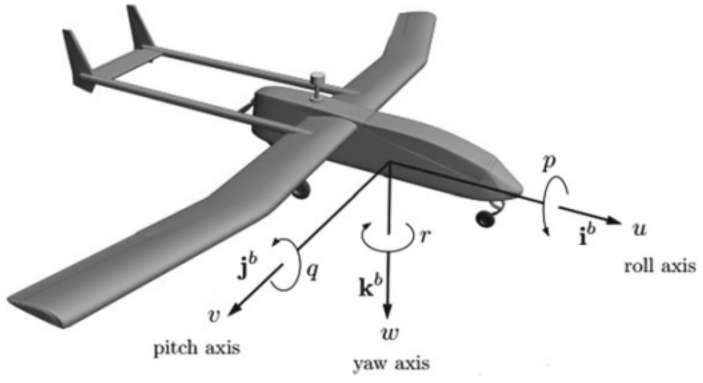


Figure 15: Definition of axes of motion

Table 2: State variables for UAV equations of motions

Variable	Description
p_n	Inertial north position of the UAV
p_e	Inertial east position of the UAV
p_d	Inertial south position of the UAV
u	Velocity along i^b
v	Velocity along j^b
w	Velocity along k^b
ϕ	Roll angle of the UAV
θ	Pitch angle of the UAV
ψ	Heading (yaw) angle of the UAV
p	Roll rate
q	Pitch rate
r	Yaw rate

vehicle projected onto the i^b,j^b and k^b axes, respectively. On the other hand, the translational position of the UAV is usually measured and expressed in an inertial frame, therefore, relating the translational velocity and position requires differentiation and a rotational transformation

$$\frac{d}{dt}\begin{pmatrix} p_n \\ p_e \\ p_d \end{pmatrix} = R_b^v \begin{pmatrix} u \\ v \\ w \end{pmatrix} = (R_v^b)^T \begin{pmatrix} u \\ v \\ w \end{pmatrix} \tag{16}$$

which using equation (7) gives

$$\begin{pmatrix} \dot{p_n} \\ \dot{p_e} \\ \dot{p_d} \end{pmatrix} = \begin{pmatrix} c_\theta c_\psi & s_\phi s_\theta c_\psi - c_\phi s_\psi & c_\phi s_\theta c_\psi + s_\phi s_\psi \\ c_\theta s_\psi & s_\phi s_\theta s_\psi + c_\phi c_\psi & c_\phi s_\theta s_\psi - s_\phi c_\psi \\ -s_\theta & s_\phi c_\theta & c_\phi c_\theta \end{pmatrix} \begin{pmatrix} u \\ v \\ w \end{pmatrix} \tag{17}$$

This is the kinematic relation that relates the derivative of position to velocity.
The relationship among angular positions ϕ, θ and ψ and the angular rates p, q and r is complicated by the fact that these quantities are defined in different reference frames. Therefore, the body-frame angular rates can be expressed in terms of the derivatives of the Euler angles, provided that the proper rotational transformations are carried out as follows:

$$\begin{aligned} \begin{pmatrix} p \\ q \\ r \end{pmatrix} &= \begin{pmatrix} \dot\phi \\ 0 \\ 0 \end{pmatrix} + R_{v2}^b(\phi) \begin{pmatrix} 0 \\ \dot\theta \\ 0 \end{pmatrix} + R_{v2}^b(\phi) R_{v1}^{v2}(\theta) \begin{pmatrix} 0 \\ 0 \\ \dot\psi \end{pmatrix} \\ &= \begin{pmatrix} \dot\phi \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{pmatrix} \begin{pmatrix} 0 \\ \dot\theta \\ 0 \end{pmatrix} \\ &+ \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{pmatrix} \begin{pmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ \dot\psi \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \sin\phi\cos\theta \\ 0 & -sin\phi & \cos\phi\cos\theta \end{pmatrix} \begin{pmatrix} \dot\phi \\ \dot\theta \\ \dot\psi \end{pmatrix} \end{aligned} \tag{18}$$

Inverting this expression yields

$$\begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} = \begin{pmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -sin\phi \\ 0 & sin\phi\sec\theta & \cos\phi\sec\theta \end{pmatrix} \begin{pmatrix} p \\ q \\ r \end{pmatrix} \tag{19}$$

which express the derivatives of the three angular position states in terms of angular positions ϕ and θ and the body rates p,q and r.

6.3.2 Dynamic Equations

To derive the dynamic equation of motion for the UAV, we will apply Newton's second law to both the translational and the rotational degrees of freedom. Newton's laws hold in inertial reference frames, meaning the motion of the body of interests must be referenced to a fixed frame of reference, which in our case is the ground. Nevertheless, motion can be expressed using vector components associated with other frame, such as the body frame.

Newton's second law applied to a body undergoing translational motion can be stated as

$$m\frac{dV_g}{dt_i} = f \tag{20}$$

where m is the mass of the fixed-wing UAV, $\frac{d}{dt_i}$ is the time derivative in the inertial frame, and f is the sum of all external forces acting on the aircraft, such as gravity, aerodynamic forces and propulsion forces.
The derivative of the velocity taken in the inertial frame can be written in terms of the derivative in the body frame and the angular velocity as

$$\frac{dV_g}{dt_i} = \frac{dV_g}{dt_b} + \omega_{b/i} \times V_g \tag{21}$$

where $\omega_{b/i}$ is the angular velocity of the UAV with respect to the inertial frame. Combining the equations (20) and (21) results in an alternative representation of Newton's second law with differentiation carried out in the body frame:

$$m(\frac{dV_g}{dt_b} + \omega_{b/i} \times V_g) = f \tag{22}$$

In the case of maneuvering aircraft, we can most easily apply Newton's second law by expressing the forces and velocities in the body frame as

$$m(\frac{dV_g^b}{dt_b} + \omega_{b/i}^b \times V_g^b) = f^b \tag{23}$$

where $V_g^b = (u, v, w)^T$ and $\omega_{b/i}^b = (p, q, r)^T$. The vector f^b represents the sum of the externally applied forces and is defined in terms of its body-frame components as $f^b \triangleq (f_x, f_y, f_z)^T$.
The expression $m\frac{dV_g^b}{dt_b}$ is the rate of change of the velocity expressed in the body frame, as viewed by an observer on the moving body. Since u, v and w are the instantaneous projections of V_g^b onto the i^b,j^b and k^b axes, it follows that

$$m\frac{dV_g^b}{dt_b} = \begin{pmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{pmatrix} \tag{24}$$

Finally, expanding the cross product and rearranging terms we get

$$\begin{pmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{pmatrix} = \begin{pmatrix} rv - qw \\ pw - ru \\ qu - pv \end{pmatrix} + \frac{1}{m} \begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix} \tag{25}$$

For rotational motion, Newton's second law states that

$$\frac{dh}{dt_i} = \mathbf{m} \tag{26}$$

where h is the angular momentum in vector form and $\mathbf{m}$ is the sum of externally applied moments. This expression is true provided that moments are summed about the center of the mass of the UAV. Following the same procedure used for the translational motion, we get

$$\frac{dh^b}{dt_b} + \omega_{b/i}^b \times h^b = \mathbf{m}^b \tag{27}$$

For a rigid body, angular momentum is defined as the defined as the product of the *inertia matrix* J and the angular velocity vector: $h \triangleq J\omega_{b/i}^b$ where J is given by

$$\begin{aligned} J &= \begin{pmatrix} \int (y^2+z^2)dm & -\int xydm & -\int xzdm \\ -\int xydm & \int (x^2+z^2)dm & -\int yzdm \\ -\int xzdm & -\int yzdm & \int (x^2+y^2)dm \end{pmatrix} \\ &= \begin{pmatrix} J_x & -J_{xy} & -J_{xz} \\ -J_{xy} & J_y & -J_{yz} \\ -J_{xz} & -J_{yz} & J_z \end{pmatrix} \end{aligned} \tag{28}$$

The diagonal terms of J are called the *moments of inertia*, whereas the off-diagonal terms are the *product of inertia*. The moments of inertia are measures of the aircraft's tendency to oppose acceleration about a specific axis of rotations.
Since integrals in equation (28) are computed with respect to the i^b, j^b and k^b axes fixed in the rigid body, J is constant when viewed from the body frame, hence $\frac{dJ}{dt_b} = 0$.
Thus, we can rewrite the equation (27) as

$$J\frac{d\omega^b_{b/i}}{dt_b} + \omega^b_{b/i} \times (J\omega^b_{b/i}) = \mathbf{m}^b \tag{29}$$

The expression $\frac{d\omega^b_{b/i}}{dt_b}$ is the rate of change of the angular velocity expressed in the body frame, as viewed by an observer on the moving body. Since p, q and r are the instantaneous projections of $\omega^b_{b/i}$ onto the i^b, j^b and k^b axes, it follows that

$$\dot{\omega}^b_{b/i} = \frac{d\omega^b_{b/i}}{dt_b} = \begin{pmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{pmatrix} \tag{30}$$

Rearranging the equation

$$\dot{\omega}^b_{b/i} = J^{-1}[-\omega^b_{b/i} \times (J\omega^b_{b/i}) + \mathbf{m}^b] \tag{31}$$

Since aircraft are often symmetric about the plane spanned by i^b and k^b, $J_{xy} = J_{yz} = 0$.
Under this symmetry assumption, the inverse of J is given by

$$\begin{aligned}
J^{-1} &= \frac{adj(J)}{det(J)} \\
&= \frac{\begin{pmatrix} J_yJ_z & 0 & J_xJ_{xz} \\ 0 & J_xJ_z - J_{xz}^2 & 0 \\ J_{xz}J_y & 0 & J_xJ_y \end{pmatrix}}{J_xJ_yJ_z - J_{xz}^2J_y} \\
&= \begin{pmatrix} \frac{J_z}{\Gamma} & 0 & \frac{J_{xz}}{\Gamma} \\ 0 & \frac{1}{J_y} & 0 \\ \frac{J_{xz}}{\Gamma} & 0 & \frac{J_z}{\Gamma} \end{pmatrix} \qquad (32)
\end{aligned}$$

where $\Gamma \triangleq J_xJ_z - J_{xz}^2$.
Defining the component of the externally applied moment about i^b, j^b and k^b axes as $\mathbf{m}^b \triangleq (l, m, n)^T$, we can write the equation (31) as

$$\begin{aligned}
\begin{pmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{pmatrix} &= \begin{pmatrix} \frac{J_z}{\Gamma} & 0 & \frac{J_{xz}}{\Gamma} \\ 0 & \frac{1}{J_y} & 0 \\ \frac{J_{xz}}{\Gamma} & 0 & \frac{J_z}{\Gamma} \end{pmatrix} \left[\begin{pmatrix} 0 & r & -q \\ -r & 0 & p \\ q & -p & 0 \end{pmatrix} \begin{pmatrix} Jx & 0 & -J_{xz} \\ 0 & J_y & 0 \\ -J_{xz} & 0 & J_z \end{pmatrix} \begin{pmatrix} p \\ q \\ r \end{pmatrix} + \begin{pmatrix} l \\ m \\ n \end{pmatrix} \right] \\
&= \begin{pmatrix} \Gamma_1 pq - \Gamma_2 qr + \Gamma_3 l + \Gamma_4 n \\ \Gamma_5 pr - \Gamma_6(p^2 - r^2) + \frac{1}{J_y} m \\ \Gamma_7 pq - \Gamma_1 qr + \Gamma_4 l + \Gamma_8 n \end{pmatrix} \qquad (33)
\end{aligned}$$

where

$$\Gamma_1 = \frac{J_{xz}(J_x - J_y + J_z)}{\Gamma} \tag{34}$$

$$\Gamma_2 = \frac{J_z(J_z - J_y) + J_{xz}^2}{\Gamma} \tag{35}$$

$$\Gamma_3 = \frac{J_z}{\Gamma} \tag{36}$$

$$\Gamma_4 = \frac{J_{xz}}{\Gamma} \tag{37}$$

$$\Gamma_5 = \frac{J_z - J_x}{J_y} \tag{38}$$

$$\Gamma_6 = \frac{J_{xz}}{J_y} \tag{39}$$

$$\Gamma_7 = \frac{J_x(J_x - J_y) + J_{xz}^2}{\Gamma} \tag{40}$$

$$\Gamma_8 = \frac{J_x}{\Gamma} \tag{41}$$

The complete set of the navigation, force, kinematic and moment equations that govern the dynamic behavior of the UAV during flight, given by equations (17), (25), (19) and (33), is formed by 12 first-order ordinary differential equations:

Navigation equations

$$\begin{aligned}\dot{p_n} &= (\cos\theta\cos\psi)u + (\sin\phi\sin\theta\cos\psi)v \\ &\quad +(\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi)w \end{aligned} \tag{42}$$

$$\begin{aligned}\dot{p_e} &= (\cos\theta\sin\psi)u + (\sin\phi\sin\theta\sin\psi)v \\ &\quad +(\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi)w \end{aligned} \tag{43}$$

$$\dot{p_d} = u\sin\theta + v\sin\phi\cos\theta - w\cos\phi\cos\theta \tag{44}$$

Force equations

$$\dot{u} = rv - qw - g\sin\theta + F_{i^b}/m \tag{45}$$

$$\dot{v} = pw - ru + g\cos\theta\sin\phi + F_{j^b}/m \tag{46}$$

$$\dot{w} = qu - pv + g\cos\theta\cos\phi + F_{k^b}/m \tag{47}$$

Table 3: UAV aerodynamics forces and moments

Parameter	Description
F_{i^b}	Total force along i^b axes
F_{j^b}	Total force along j^b axes
F_{k^b}	Total force along k^b axes
l	Rolling moment
m	Pitching moment
n	Yawing moment
C_L	Lift coefficient
C_D	Drag coefficient
C_Y	Sideforce coefficient
C_l	Rolling moment coefficient
C_m	Pitching moment coefficient
C_n	Yawing moment coefficient

Kinematic equations

$$\dot{\phi} = p + q \sin\phi \tan\theta + r \cos\phi \tan\theta \tag{48}$$

$$\dot{\theta} = q \cos\phi + r \sin\phi \tag{49}$$

$$\dot{\psi} = q \sin\phi \sec\theta + r \cos\phi \sec\theta \tag{50}$$

textbfMoment equations

$$\dot{p} = \Gamma_1 pq - \Gamma_2 qr + \Gamma_3 l + \Gamma_4 n \tag{51}$$

$$\dot{q} = \Gamma_5 pr - \Gamma_6 (p^2 - r^2) + m/J_y \tag{52}$$

$$\dot{r} = \Gamma_7 pq - \Gamma_1 qr + \Gamma_4 l + \Gamma_8 n \tag{53}$$

7 Problem Statement

Scientific researches keep moving steps forward in controlling, maneuvering and planning the paths of the UAVs. The relevance of overcoming the limitations of the approaches proposed is huge since UAVs have to be able to fly autonomously and safely, dealing with unpredictable situations, uncertainties, a dynamically changing 3D environment, and so on and so forth. The

Table 4: Fixed-Wing UAV Equations of Motion

Navigation Equations
$\dot{p_n} = (\cos\theta\cos\psi)u + (\sin\phi\sin\theta\cos\psi)v + (\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi)w$
$\dot{p_e} = (\cos\theta\sin\psi)u + (\sin\phi\sin\theta\sin\psi)v + (\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi)w$
$\dot{p_d} = u\sin\theta + v\sin\phi\cos\theta - w\cos\phi\cos\theta$
Force Equations
$\dot{u} = rv - qw - g\sin\theta + F_{i^b}/m$
$\dot{v} = pw - ru + g\cos\theta\sin\phi + F_{j^b}/m$
$\dot{w} = qu - pv + g\cos\theta\cos\phi + F_{k^b}/m$
Kinematic equations
$\dot{\phi} = p + q\sin\phi\tan\theta + r\cos\phi\tan\theta$
$\dot{\theta} = q\cos\phi + r\sin\phi$
$\dot{\psi} = q\sin\phi\sec\theta + r\cos\phi\sec\theta$
Moments Equations
$\dot{p} = \Gamma_1 pq - \Gamma_2 qr + \Gamma_3 l + \Gamma_4 n$
$\dot{q} = \Gamma_5 pr - \Gamma_6(p^2 - r^2) + m/J_y$
$\dot{r} = \Gamma_7 pq - \Gamma_1 qr + \Gamma_4 l + \Gamma_8 n$

challenges identified in 5.7 to overcome those limitations have been taken up by many of the researchers and overcome with different approaches, nevertheless as far as we know none of them solved those limitations altogether. The main goal of this Thesis work is to propose a path planning algorithm for UAVs able to guarantee a robust solution in a real time manner inside a 3D real world environment, while minimizing the energy consumption.

7.1 3D Path Planning

Before introducing the proposed solution, it is important to define what the path planning is. Based on the work in [1], [2], [4], [10], a clear and mathematical definition of the 3D path planning problem is presented.

In this dissertation fixed-wing UAVs are assumed to fly in a 3 dimension space (R^3), called the workspace w. The workspace has normally obstacles, so let wo_i be the ith obstacle. The free workspace without obstacles is the overall area represented by

$$w_{free} = w \setminus U_i wo_i \tag{54}$$

The initial point x_{init} and the goal point x_{goal} are all elements in the free workspace w_{free}. Thus, a path planning problem is defined by a triplet $(x_{init}, x_{goal}, w_{free})$.

Definition 1: Given a function $\delta : [0, T] \to R^3$ of bounded variation, where $\delta(0) = x_{init}$ and $\delta(T) = x_{goal}$. If there exists a process Φ which can guarantee $\delta(\tau) \in w_{free}$ for all $\tau \in [0, T]$, then Φ is called *path planning*.

Definition 2: Given a path planning problem $(x_{init}, x_{goal}, w_{free})$ and a cost function $c : \Sigma \to R \geq 0$, where Σ denotes the set of all paths, if a process fulfill the *definition 1* to find a δ', and $c(\delta') = min\{c(\delta), \delta$ set of all feasible path$\}$, then δ' is the *optimal path* and Φ' is *optimal path planning*.

7.2 Real Time Path Planning

Real-Time Path Planning is a term used in robotics that consists of motion planning methods that can adapt to real time changes in the environment, as explained in [3]. This includes everything from primitive algorithms that stop a robot when it approaches an obstacle to more complex algorithms that continuously takes in information from the surroundings and creates a plan to avoid obstacles.

When an obstacle in the environment is on a robot trajectory planned, or will crash the robot at any point of the trajectory when the robot moves, it is called as an *active obstacle*.

The ability of a robotic system to avoid in real time a dynamically changing object is strictly related to a concept of local path planning algorithm, since this feature takes into account either a totally or partially unknown environment and the possibility to have active obstacles that cannot be considered or coped by a global path planner.

7.3 Robust Path Planning

The robustness of a path planning algorithm covers different aspects of the trajectory planning problem.
To be robust an algorithm has to deal with all the possible situations in which the fixed-wing UAVs might be. Furthermore, the algorithm has to converge quickly enough to the optimal path without falling into local minima during the computation of the trajectory. Lastly, the algorithm has to guarantee a robustness of position estimation of the drones.
In [29] robustness of position estimation is defined as the tolerance of the algorithm for position uncertainty. When the robustness of an algorithm is strong, it implies that in uncertainty condition, this algorithm's performance is well. The safety degree is a good measurement to present the Robustness. Let $Q(Q = w_i | i \in [1, ...N])$ denote as the path sequence of the UAV from the start to the end of the flight. The path safety degree is described by two indicators, defined as follows:

$$Avg(Q) = \frac{1}{N} \sum_{i=1}^{N} P(w_i) \tag{55}$$

This formula represents the average of the threat degree for all path nodes to describe the safety.

$$Max(Q) = max(P(w_i)), i \in [1, ...N] \tag{56}$$

This formula represents the maximum degree of the threat of the flight to describe the safety.
Similar to the position, let $Q'(Q' = w'_i | i \in [1, ...N))$ denote as the path sequence considering the position uncertainty. The difference of Q and Q' is defined as follow:

$$\Delta Avg(Q') = Avg(Q') - Avg(Q) \tag{57}$$

Finally, the robustness could be calculated as follow:

$$Robust(Q') = \Delta Avg(Q') \cdot Max(Q') \tag{58}$$

The formula listed above represents the tolerance of the uncertainty. Generally, robust should be depicted as variance.. The formula similar to standard deviation multiplies range. It could be negative for $\Delta Avg(Q')$ which may give more interesting information. If the result of uncertainty is as well as the one in precision. That means the algorithm's ability of tolerance is high. So, high $Robust(Q')$ value means low robustness.

7.4 Energy Efficient Path Planning

The energy efficiency problem has to be adapted to a node-base path planning algorithm, still being consistent with the physics of the model.
The total energy for traveling between a specified start node and the destination node would be the sum of the total energy spent in traveled edges. Hence, in order to calculate the energy of a path which consists of multiple straight flight paths, it is needed to compute energy consumption $\Delta E_{i,j}$ going through each straight line with distance Δd from node i to node j, as depicted in 16. The difference in the total energy consumption should be the sum of differences in potential energy and difference in kinetic energy plus the energy used to turn the fixed-wing UAV between arcs.

$$\Delta E_{i,j} = \Delta E_p + \Delta E_k + \Delta E_{turn} \tag{59}$$

Assuming there is no energy gain for UAV during decreasing altitude, the difference of potential energy is given as

$$\Delta E_p = max(W\Delta h), 0) = max(mg(h_i - h_j), 0) \tag{60}$$

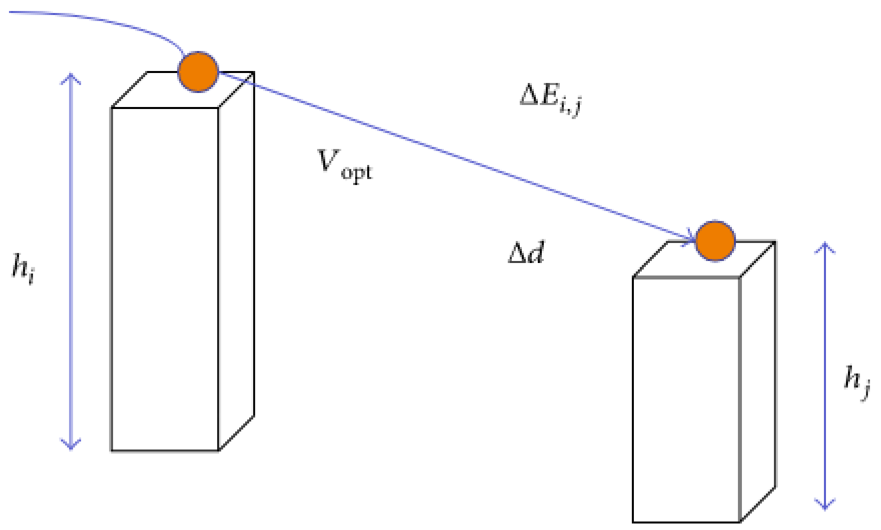


Figure 16: Energy consumption between two nodes (picture showed in [31])

To calculate the kinetic energy, it has to be focused on optimizing the velocity which depends on the drag-to-lift ratio of the UAV. In [46] the drag-to-lift ratio is represented as

$$\frac{D}{L} = AV^2 + \frac{B}{V^2} \tag{61}$$

where V is the flight speed and A and B are variables calculated by air density; UAV's parameters are given as

$$A = \frac{\rho f}{2W} \tag{62}$$

$$B = \frac{2W}{\rho b^2 \pi e} \tag{63}$$

Most of the parameters are dependent on the structure of UAV; $W = mg$ is the aircraft's weight, b is the wing span, f is the parasite area of the aircraft, and e is Oswald's efficiency factor of the UAV. In addition, $\rho(h)$ is the air density at altitude h given in [48]:

$$\rho(h) = \frac{p(h)}{RT(h)} = \frac{P_0(1 - 0.0065(h/T_0))^{5.2561}}{R(T_0 - 6.5(h/1000))} \tag{64}$$

where $P_0 = 101,325 N/m^2$ is the sea level atmospheric pressure, $T_0 = 288.15K$ is the sea level standard temperature, and $R = 287.04 m^2/Ks^2$ is the universal gas constant.

We assume that the fixed-wing UAV flies with a constant optimum speed $V_{opt} = (B/A)^{1/4}$, occurring when the D/L ratio is optimum ($D/L_{opt} = 2\sqrt{(AB)}$). If we assume that the consumption efficient of the drone is 100%, then this ratio is proportionate to the energy consumption, with or without the effect of wind force.
For a zero-wind scenario, the kinetic energy ΔE_k consumed will only be used to overcome the drag force D on a straight flight.

$$\Delta E_k = \int D dxyz = D\Delta d \tag{65}$$

Because the lift force L is equal to UAV's weight, if the UAV flies with the constant optimum Carson's speed, we have

$$\Delta E_k = W(D/L)_{opt}\Delta d = 2W\sqrt{AB}\Delta d = 2W\sqrt{\frac{f}{\bar{\rho}b^2\pi e}}\Delta d \tag{66}$$

where $\bar{\rho} = (\rho(h_i) + \rho(h_j))/2$.

Following the calculation from [49], the consumed energy for turning between two arcs could be approximated by

$$E_{turn} \approx m\frac{D}{L}sin\sigma|V^2| \tag{67}$$

where σ is the steepest rolling angle that the drone needs to take for turning.

Hence, the energy consumption for travelling between two nodes i and j without a wind force is calculated as follows:

$$\begin{aligned}\Delta E_{i,j} = {} & max(mg(h_i - h_j), 0) \\ & + 2W\sqrt{\frac{f}{\bar{\rho}b^2\pi e}}\Delta d \\ & + min(m\frac{D}{L}sin\sigma|V^2|, 0)\end{aligned} \tag{68}$$

The next step includes the effect of the wind field. Practically, since the operation range of the fixed-wing UAV is limited in a restricted area, it can be applied a constant wind field in the configuration space. Considering a reference frame based on the plane comprising of wind velocity vector V_{wind} and optimum velocity vector V_{opt} between node i and node j, let us call β and γ as the angle of V_{opt} and V_{wind} to this reference's horizontal axis, respectively. In order to keep the optimum speed, assuming that the minimum velocity of the UAV can overcome the wind force, the UAV needs to make a turn with an angle α and a controlled speed V_c that satisfy $V_{opt} = V_{wind} + V_c$. An example is showed in 17.

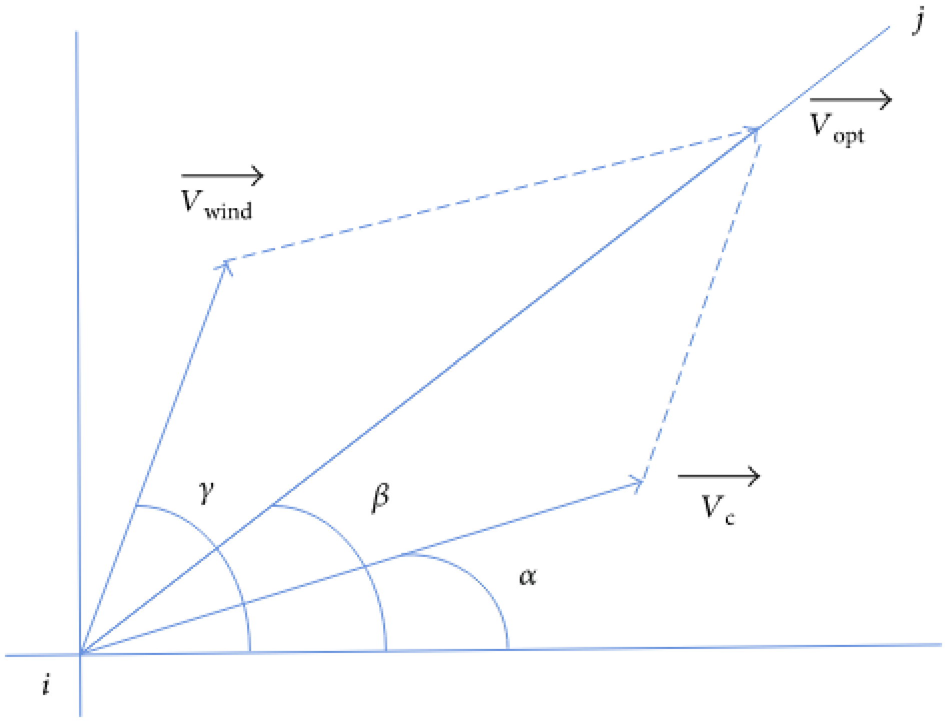


Figure 17: Speed adjustment under effect of wind velocity vector.

The duration for giving the controlled speed V_c equals to the duration used to travel with optimum speed from node i to node j, which means $t = \Delta d/V_{opt}$. The new equation for kinetic energy was derived as

$$\Delta E_k = D\Delta d = W(D/L)_c V_c t = W(AV_c^2 + \frac{B}{V_c^2})V_c t \tag{69}$$

The V_{opt}, in this case, can be computed by

$$V_{opt} = V_{wind}cos(\gamma - \beta) + \sqrt{V_c^2 + V_{wind}^2 sin^2(\gamma - \beta)} \tag{70}$$

In the end, from 64, 69, and 70, the energy consumption for travelling between two nodes i and j in presence of a wind force is calculated as follows:

$$\begin{aligned}\Delta E_{i,j} &= max(mg(h_i - h_j), 0) \\ &+ \frac{W(AV_c^2 + B/V_c^2)V_c \Delta d}{V_{wind}cos(\gamma - \beta) + \sqrt{V_c^2 + V_{wind}^2 sin^2(\gamma - \beta)}} \\ &+ min(m\frac{D}{L}sin\sigma|V^2|, 0)\end{aligned} \tag{71}$$

Since one of the assumption made in this Thesis work is that the environment is not affected by wind forces, the equation 68 is used to compute the total energy which the planned paths need to be traversed.

7.5 Multi Agent Path Planning

Multi-agent path planning (MAPP) is a challenging task that aims to find conflict free paths for all the agents in a given domain. Multi-agent path planning is important in a variety of fields, ranging from games to robotics and warehouse management.
A multi-agent path planning can be divided in two categories:

- *centralized*
- *decentralized or distributed*

As explained in [21], although centralized control in the joint action space can provide optimal plans, this often is computationally infeasible. Decoupled planning is much more scalable. Traditional decoupled approaches perform a unit-centric decomposition, replacing a multi-agent search with a series of single-agent searches, one for each mobile unit.

In [41] a Multi-Agent Path Finding Problem problem is defined as a triple $P = (G, R, T)$, where $G = (V, E)$ is an undirected connected graph, where V and E correspond to locations and ways of moving between them for the

agents, R is a set of robots, and T is a set of orders. For each robot $r \in R$, the starting location of r is specified by a location $l_r \in V$. Each order $o \in T$ is specified by a location $l_o \in V$.
Robots can move between the vertices along the edges of G,one edge at a time, under the restrictions:

- two robots cannot swap locations in a single time step
- each location can be occupied by at most one robot at any time

A path for a robot r is a sequence of vertices $\alpha = <v1, ..., vn>$ if

- the robot starts at $v1$
- for any two subsequent vertices v_i and v_{i+1}, there is an edge between them (i.e., $(v_i, v_{i+1}) \in E$) or they are the same vertex (i.e., $v_i = v_{i+1}$).

A robot r completes an order o via a path $\alpha = <v1, ..., vn>$ if $l_o \in v1, ..., vn$. A solution of a Multi Agent Path Finding problem P is a collection of paths $S = \alpha_r | r \in R$ for the robots in R so that all orders in T are completed.

7.6 MATLAB

MATLAB (an abbreviation of "matrix laboratory") is a multi-paradigm numerical computing environment and proprietary programming language developed by MathWorks. MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages.

Although, as explained in [44], MATLAB is intended primarily for numerical computing, an optional toolbox uses the MuPAD symbolic engine allowing access to symbolic computing abilities. An additional package, Simulink, adds graphical multi-domain simulation and model-based design for dynamic and embedded systems.

The simulation environment and the simulations themselves are performed entirely in the MATLAB environment, due to its huge versatility and simplicity even for complex problems like a 3D path planning for a fleet of UAVs.

Part IV
Proposed Solution

8 Introduction

As previously mentioned, the four challenges to be faced in the path planning problem for a fleet of fixed-wing UAVs are:

- *Robustness*
- *Three dimensional space*
- *Real-Time*
- *Energy efficiency*

Hence, it is looked for a widely studied algorithm, in order to be sure of the robustness of the proposed solution. Moreover, an algorithm fast enough to deal with real-time solution in a cluttered 3D environment had to be chosen. For this aim, the literature review itself has been very useful in order to understand the effectiveness, the computational burden and the robustness of every group of path planning algorithms and their state of the art.

9 Details of the Proposed Methodology

The choice has fallen on Node-based optimal algorithms, due to their reliability and velocity in finding optimal path. Those characteristics make them very appropriate for a 3D path planning problem.
Among them, A* is one of the most effective choice. Indeed, thanks to its heuristic function, A* search algorithm is noticeably faster than other Node-based ones. Choosing the A* algorithm two of the four challenges to be faced are intrinsically solved: it can deal with 3D environment and it is robust.
The third challenge to be faced is the energy consumption problem. That is why an energy efficient version of the A* has been developed, called EEA* search algorithm.
Finally, to wholly fulfill the real-time requirement, a local path planner logic

has been added in order to deal with unknown dynamic obstacles. The logic is to generates intermediate waypoints along the path when an obstacle is sensed by the fixed-wing UAVs.
The assumption made are:

- Part of the map, in particular the static obstacles of the environment are known a-priori.
- No wind or disturbances during the flight.

9.1 A* Search Algorithm

The A* search algorithm is a node-based algorithm born as an extension of the Dijkstra's algorithm. As explained in [45], A* was created as part of the Shakey project, which had the aim of building a mobile robot that could plan its own actions. Nils Nilsson originally proposed using the Graph Traverser algorithm for Shakey's path planning. Graph Traverser is guided by a heuristic function $h(x)$, the estimated distance from node x to the goal node: it entirely ignores $g(x)$ the distance from the start node to x. Bertram Raphael suggested using the sum, $g(x) + h(x)$. Peter Hart invented the concepts we now call *admissibility* and *consistency* of heuristic functions. A* was originally designed for finding least-cost paths when the cost of a path is the sum of its edge costs, but it has been shown that A* can be used to find optimal paths for any problem satisfying the conditions of a cost algebra.
Ultimately, Node based optimal algorithms are classified by the reason that they deal with nodes and arcs weight information; they calculate the cost by exploring through the nodes, thus to find the optimal path. Unlike Dijkstra's algorithm, A* reduces the number of states by introducing the heuristic estimation of the cost from the current state to the goal state. Thanks to the heuristic function, A* can converge very quickly and to an optimal solution.

A* is proposed by introducing an evaluation function, which consists of post-calculation toward the initial state and heuristic estimation toward the goal,

$$f(x) = g(x) + h(x) \tag{72}$$

where g(x) is the cost from the initial state x_{init} to the current state x, $h(x)$ is the heuristic estimation of the cost of an optimal path from the current

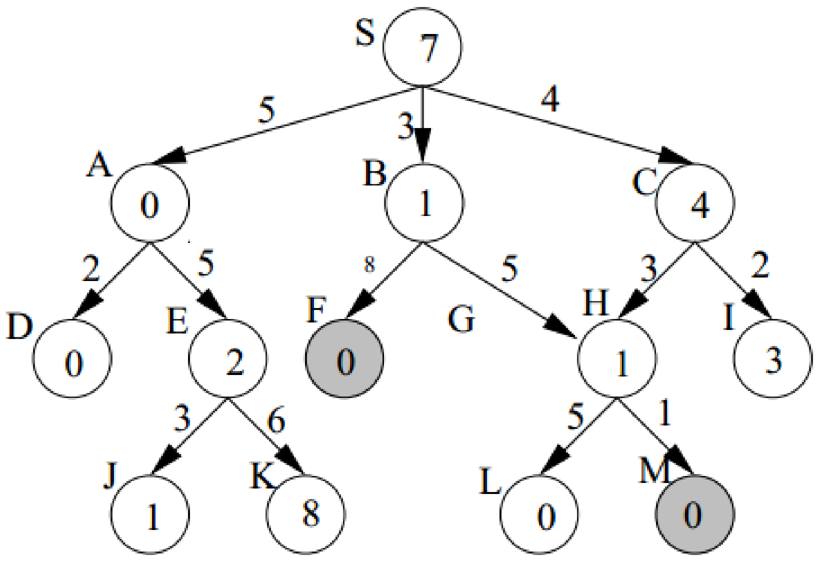


Figure 18: Example of A* Search Algorithm. The numbers on the edge are $g(x)$, the number in the nodes are the estimated distance from the goal $h(x)$. The grey nodes represent the goal node.

state x to the goal state x_{goal}.
On finite graphs with non-negative edge weights, A* is guaranteed to terminate and is complete, i.e. it will always find a solution (a path from start to goal) if one exists. On infinite graphs with a finite branching factor and edge costs that are bounded away from zero $d(x, y) > \varepsilon > 0$ for some fixed ε, A* is guaranteed to terminate only if there exists a solution.

9.1.1 Admissibility

A search algorithm is said to be admissible if it is guaranteed to return an optimal solution. If the heuristic function used by A* is admissible, then A* is admissible. An intuitive proof of this is as follows:
when A* terminates its search, it has found a path from start to goal whose actual cost is lower than the estimated cost of any path from start to goal through any open node (the node's f value). When the heuristic is admissible, those estimates are optimistic, so A* can safely ignore those nodes because they cannot possibly lead to a cheaper solution than the one it already has. In other words, A* will never overlook the possibility of a lower-cost path

from start to goal and so it will continue to search until no such possibilities exist. The actual proof is a bit more involved because the f values of open nodes are not guaranteed to be optimistic even if the heuristic is admissible. This is because the g values of open nodes are not guaranteed to be optimal, so the sum $g + h$ is not guaranteed to be optimistic.

Algorithm 1 A* Search Algorithm Pseudocode

```
Result: Waypoints
put $x_{init}$ in the $OPEN$ list
while $OPEN$ list is not empty do
    take from the $OPEN$ list the $x$ with the lowest $f(x) = g(x) + h(x)$
    if $x = x_{goal}$ then
        break
    end
    generate each node successors $x_{succ}$ that come after $x$
    for each $x_{succ}$ of $x$ do
        successor current cost $= g(x) + w(x, x_{succ})$
        if $x_{succ}$ is in the $OPEN$ list then
            if successor current cost $\leq g(x_{succ})$ then
                jump outside the for
            end
        else if $x_{succ}$ is in the $CLOSED$ list then
            if successor current cost $\leq g(x_{succ})$ then
                move $x_{succ}$ from the $CLOSED$ list to the $OPEN$ list
            end
        else
            add $x_{succ}$ to the $OPEN$ list
            set the parent of $x_{succ}$ to $x$
        end
        $g(x_{succ})$ = successor current cost
        set the parent of $x_{succ}$ to $x$
    end
    add $x$ to the $CLOSED$ list
end
if $x$ different from $x_{goal}$ then
    exit with error, the $OPEN$ list is empty
end
```

9.1.2 Optimal Efficiency

Algorithm A is optimally efficient with respect to a set of alternative algorithms Alts on a set of problems P if for every problem P in P and every algorithm A' in Alts, the set of nodes expanded by A in solving P is a subset (possibly equal) of the set of nodes expanded by A' in solving P.
The definitive study of the optimal efficiency of A* is due to Rina Dechter and Judea Pearl. In [47] they considered a variety of definitions of Alts and P in combination with A*'s heuristic being merely admissible or being both consistent and admissible. The most interesting positive result they proved is that A*, with a consistent heuristic, is optimally efficient with respect to all admissible A*-like search algorithms on all non-pathological search problems. Roughly speaking, their notion of non-pathological problem is what we now mean by "up to tie-breaking". This result does not hold if A*'s heuristic is admissible but not consistent. In that case, Dechter and Pearl showed there exist admissible A*-like algorithms that can expand arbitrarily fewer nodes than A* on some non-pathological problems.

Optimal efficiency is about the set of nodes expanded, not the number of node expansions (the number of iterations of A*'s main loop). When the heuristic being used is admissible but not consistent, it is possible for a node to be expanded by A* many times, an exponential number of times in the worst case. In such circumstances Dijkstra's algorithm could outperform A* by a large margin.

9.1.3 Complexity

The time complexity of A* depends on the heuristic function. In the worst case of an unbounded search space, the number of nodes expanded is exponential in the depth of the solution (the shortest path) $d : O(b^d)$, where b is the branching factor (the average number of successors per state). This assumes that a goal state exists at all, and is reachable from the start state; if it is not, and the state space is infinite, the algorithm will not terminate.

The heuristic function has a major effect on the practical performance of A* search, since a good heuristic allows A* to prune away many of the b^d nodes that an uninformed search would expand. Its quality can be expressed in terms of the effective branching factor b^*, which can be determined empirically for a problem instance by measuring the number of nodes expanded N, and the depth of the solution, then solving:

$$N + 1 = 1 + b^* + (b^*)^2 + ... + (b^*)^d \tag{73}$$

Good heuristics are those with low effective branching factor (the optimal being $b^* = 1$).
The time complexity is polynomial when the search space is a tree, there is a single goal state, and the heuristic function h meets the following condition:

$$abs(h(x) - h^*(x)) = Olog(h^*(x)) \tag{74}$$

where h^* is the optimal heuristic, the exact cost to get from x to the goal. In other words, the error of h will not grow faster than the logarithm of the perfect heuristic h^* that returns the true distance from x to the goal.

The space complexity of A* is roughly the same as that of all other graph search algorithms, as it keeps all generated nodes in memory. In practice, this turns out to be the biggest drawback of A* search, leading to the development of memory-bounded heuristic searches, such as Iterative deepening A*, memory bounded A*, and SMA*.

9.2 EEA* Search Algorithm

Starting from the implementation of the A* search algorithm, it has been studied a way to reduce as much as possible the usage of the energy. The new algorithm has to own all the advantages of an A* search algorithm and concurrently giving back a route which is more cost-effective than the optimal path in terms of energy consumption. Hence, such a variant of the A* algorithm, choosing a better path in terms of energy efficiency, does not give in output the optimal path in terms of distance anymore. Therefore, the EEA* algorithm must reach a trade off between energy efficiency and distance.

As previously said in 7.4, since the assumption made is that we have an environment with no wind forces, the equation to be referred to is 68. It has been assumed a constant velocity along the route V. The aircraft' weight W, its mass m, the wing span b, the parasite area of the UAV f, the drag-to-lift ratio D/L and the Oswald's efficiency factor of the UAV e are given. The distance from the node i to the node j Δd is kept constant.

Algorithm 2 EEA* Search Algorithm Pseudocode

```
Result: Waypoints
put $x_{init}$ in the $OPEN$ list
while $OPEN$ list is not empty do
    while not done do
        take from the $OPEN$ list the $x$ with the lowest $f(x) = g(x) + h(x)$
        if $x = x_{goal}$ then
            break
        end
        if z component of $x$ is lower or equal to the z component of its parent node then
            if auxArray is not empty then
                set the previously modified cost in the $OPEN$ list back to their values
                empty $auxArray$
            end
            done
        end
        if z component of $x$ is greater than the z component of its parent node then
            if $x$ is in auxArray then
                set the previously modified cost in the $OPEN$ list back to their values
                empty $auxArray$
                done
            else
                put $x$ in $auxArray$
                update the cost of $x$ to the one considering the ascending manoeuvre
            end
        end
    end
    same operations of the A* from now on
end
if $x$ different from $x_{goal}$ then
    exit with error, the $OPEN$ list is empty
end
```

Therefore, the parameters that affect the total energy computation are the difference of altitude Δh in the potential energy ΔE_p, the air density at a certain altitude $\rho(h)$ for what concerns the kinetic energy ΔE_k, and the steepest rolling angle the drone needs to take for turning σ affects the energy consumed for turning, called ΔE_{turn}.

In this Thesis the parameter on which the research has been focused and at the same time the one it seemed more logical to work with has been the difference of altitude Δh between one node and its successor. In particular limiting as much as possible the ascending movement of the fixed-wing UAV, so that the total potential energy ΔE_p is as lower as possible to fly from the starting node to the goal node.
Working on the kinetic energy is less relevant since the air density is strictly related to the altitude to which the UAV flies in that particular moment. Working on this parameter is harder due to some constraints in the flight. Firstly, a safety constraint on the minimum altitude to avoid collisions with the ground. Secondly, constraints inherent in laws and legislations that give certain ranges of altitudes for certain type of aircraft.
An improvement to the algorithm might be working also with the energy needed to turning, nonetheless the energy required to turn the fixed-wing is less costly than the one required to ascend. Moreover, every constraint added to improve the energy consumption diverts the final path computed by the algorithm from the optimal path. Then, having a longer path in terms of distance implies consuming more energy because the UAV takes longer to follow it. That is why having a trade off between optimal path and energy consumption is important and it has been chosen to operate only on the potential energy term of the equation 68.

10 Implementation Details

As discussed in 7.6, the algorithm has been developed by means of MATLAB scripts. The simulations are built and prepared using the same development environment, as well. In the following parts, the implementation details are discussed.

10.1 Environment

For simulation purposes an environment has been built. Since fixed-wing UAVs are generally made for long outdoor distances, the map built is a mountain-wise environment.
The model of the environment is $100x100$ matrix representing the $x-y$ plane. The values inside each element of this $100x100$ matrix corresponds to the height of the mountain in that point on the z axis. Finally, the map is included in a tensor-wise 3D space wide $100x100x50$. Every cell is $100m$ along the x axis, $100m$ along the y axis and $100m$ along the z axis, obtaining a map that has a width of $10km$, a length of $10km$ and a height of $5km$.
The mountain is part of the static obstacles and it is known a-priori. Hence, every point of the map which falls inside the mountain is automatically inserted in the $CLOSED$ list used by the EEA* search algorithm later on to compute the optimal path. The $CLOSED$ list is interpreted by the algorithm as the list of forbidden nodes, where the fixed wing UAVs are not allowed to fly through.

To enhance the complexity of the environment two areas has been added: a *no-fly zone* and a *dangerous weather zone*.
The no-fly zone, depicted in red in 20, is a territory established by a military power where certain kind of aircraft are not permitted to fly. The zone is a circular area having diameter of $2000m$ and area of $3.141.593m^2$.
The dangerous weather area, depicted in white in 20, is instead an area where the flight is strongly discouraged due to hostile climatic conditions. This area has a diameter of $1500m$ and consequently an area of $1.767.145m^2$.

These two areas are modeled differently in the code for the scope of the algorithm. Indeed, the no-fly zone area is treated as an obstacle area, which means inserting each point that falls into the red volume in the 20 inside the $CLOSED$ list. Differently, the nodes inside the dangerous weather zone are not added by the algorithm inside the $CLOSED$ list. Those nodes are actually flyable by the UAVs, but extremely not suggested, so they are modeled as flyable nodes to whom is associated a heavily high cost to fly through. In practice, the UAVs will always avoid those nodes, unless it is really strictly necessary to fly through them to reach the goal.
The reason behind this choice is simple. The no-fly zone are territories in which is totally forbidden to fly, whereas, at least in principle, dangerous

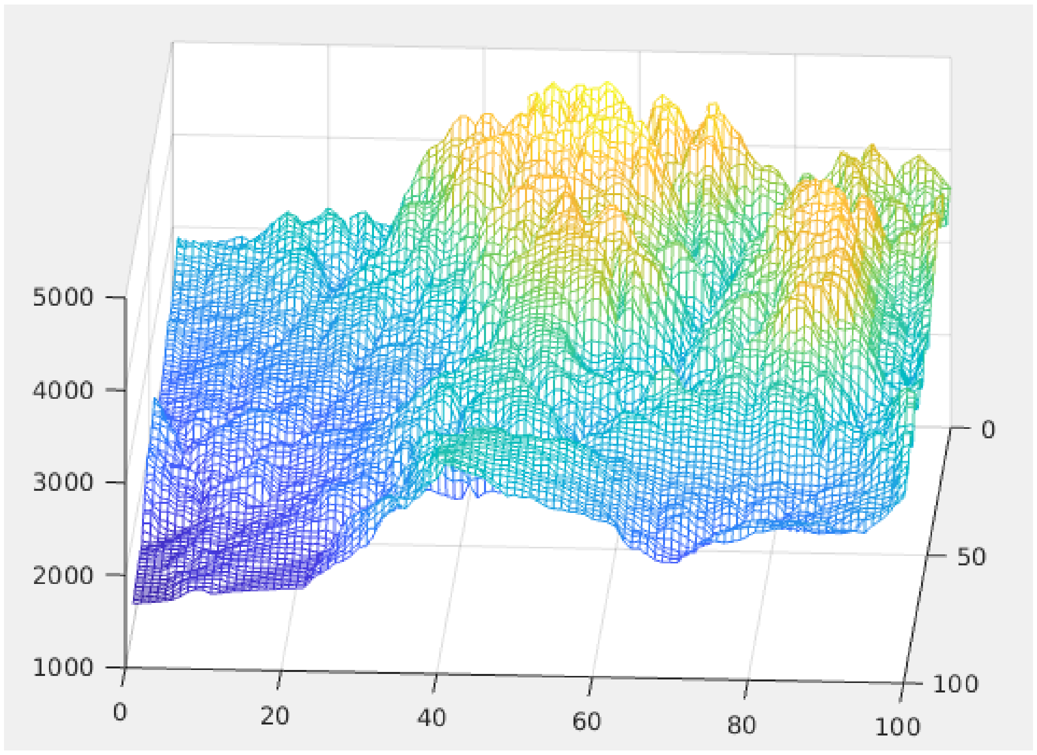


Figure 19: Model of the world used for the simulations

weather areas are not forbidden but only strongly discouraged. For that reason, the cost of passing through them is very high.

Moreover, to test the local path planner logic, dynamic objects are put into the environment. The objects are represented as spheres whose only the head is a real obstacle, while the tail is only to visualize the trajectory performed by the sphere. Such spheres are totally unknown by the UAVs and has to be sensed and avoided in real-time. Two spheres are located inside the map.

Lastly, even if the UAVs are not actually part of the map itself, they deserved to be mentioned for what concern the fleet simulation since for the local path planning algorithm they consider each other as dynamically changing obstacles to be avoided in real time.

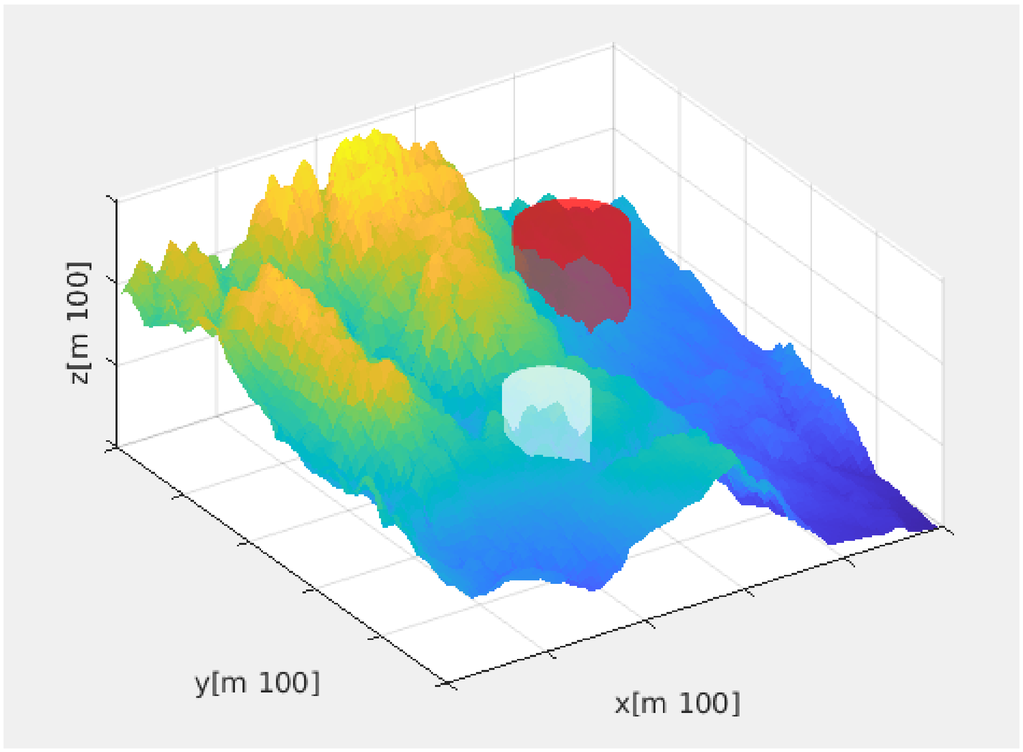


Figure 20: Cluttered 3D environment in which the simulations are performed. The red cylinder is the no-fly zone. The white cylinder represents the dangerous weather zone.

10.2 Global Path Planner

The novel EEA* Search Algorithm is used as global path planner. The algorithm using the same principles and working in the same way of the A*, is also able to cope with the energy efficiency. As discussed in 9.2, the work has been focused on minimizing in particular the potential energy term of the total energy computation. To do that, it has been operated on avoiding useless or too much costly ascension of the fixed-wing UAV.

Going into details, at each computational step of the algorithm, performed in the same way as the A*, the EEA* looks for the node with the lowest $f(x)$ in the OPEN list. Then, it confronts z_j, the z component of the node j, with z_i, the z component of the node i (the parent node of j), obtaining two possible branches:

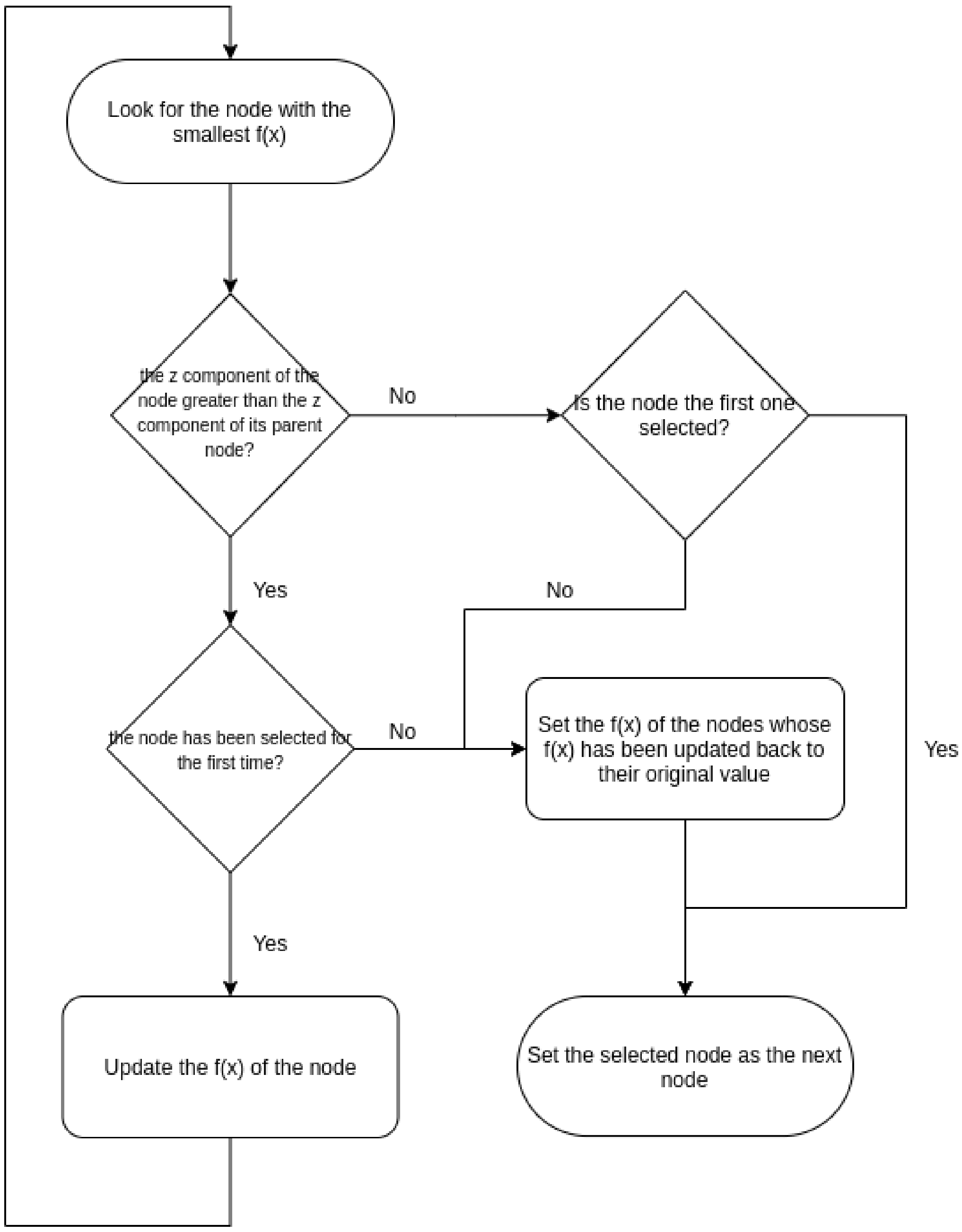

Look for the node with the smallest f(x)
the z component of the node greater than the z component of its parent node?
No
Is the node the first one selected?
Yes
No
the node has been selected for the first time?
No
Set the f(x) of the nodes whose f(x) has been updated back to their original value
Yes
Yes
Update the f(x) of the node
Set the selected node as the next node

Figure 21: Flow Chart of the EEA* Search Algorithm. This part of the algorithm is the one which takes over the A* after it has computed the Expanded Array of the particular time step and it has updated the OPEN list accordingly.

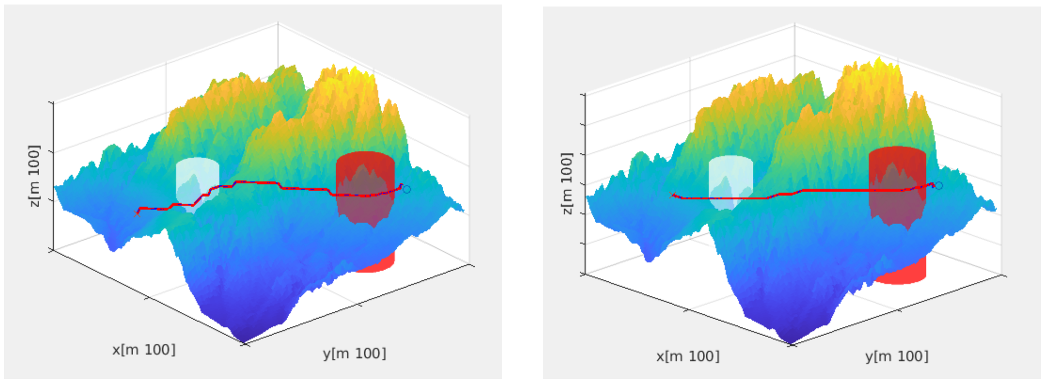


Figure 22: Comparison between the A* Search Algorithm and the EEA* Search Algorithm in terms of the shape of the path computed by the two algorithms. On the left it is depicted the path computed by the A*; on the right it is depicted the path computed by the EEA*. It is evident that the path given back by the EEA* reaches the goal avoiding ascending maneuvers.

- if z_j is equal or lower than z_i then the next node has been found (the UAV is keeping is altitude or is descending).
- if z_j is greater than z_i then update the jth node with a new greater $f(x)$ value which takes into account of the energy spent to ascend to a new altitude and repeat the process looking back in the OPEN list.

If a chosen node has been already picked up before and its $f(x)$ has already been updated, that is going to be the next node since every node can be rejected only once.
Once the next node is selected, every nodes whose $f(x)$ has been changed during this phase has brought back to its original value. This operation is needed to take into account the fact that the value of $f(x)$ changes according to the parent node where the algorithm comes from and does not depend only on the node itself. Hence, the operation has to be done again every time the algorithm has to compute a new node.

10.3 Local Path Planner

For implementing the local path planning, the sensor to detect the dynamic obstacles might be implemented in a real world simulation as Lidars, Radars,

monocular or stereo cameras and so on so forth. For the sake of simplicity it has been thought to abstract the problem without using a specific sensor whatsoever. Indeed, the sensor is actually modeled as a function that time by time computes the distance in a three dimensional space between the UAVs and the dynamic obstacles in the environment.
The assumption made is that the sensors on-board have to be able to recognize if the obstacle is a generic dynamic obstacle or another UAV belonging to the fleet. The detection happen when the distance of the fixed-wing goes below a certain distance with respect to an obstacle. When this condition is verified, two different situation can be triggered:

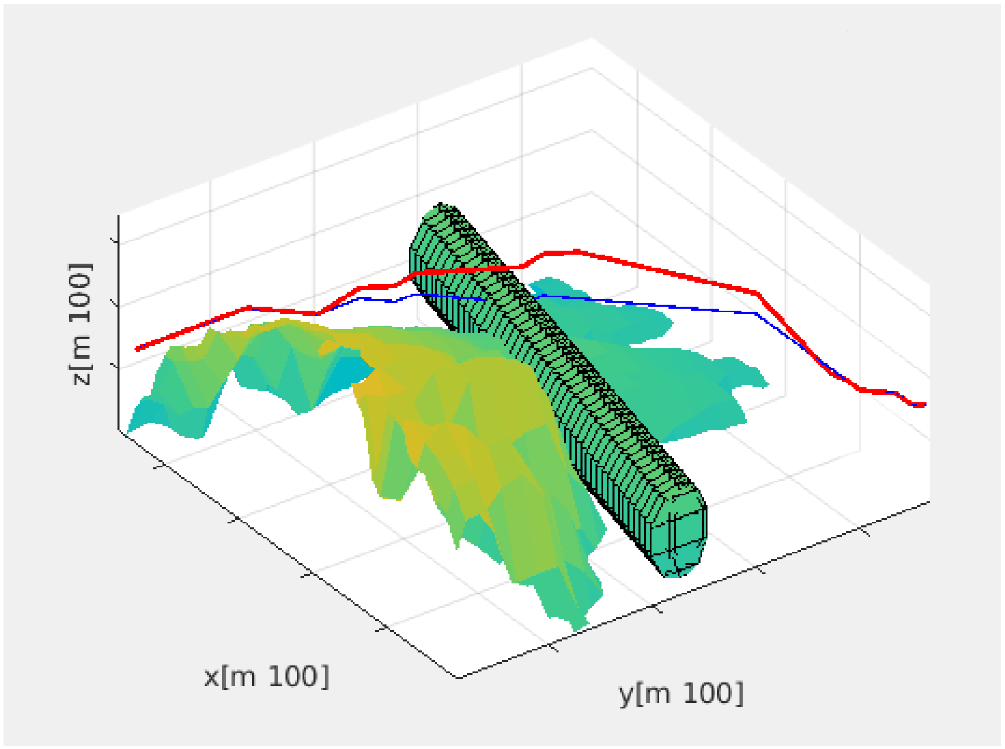


Figure 23: The local path planner performing the dynamic obstacle avoidance. The green spherical object with the tail is the dynamic threatening obstacle, the blue line is the nominal optimal path computed by the global path planner, the red line is the actual path performed by the fixed-wing UAV.

- the UAV detects a generic dynamic obstacle: the local path planner generates an intermediate waypoint above the obstacle in order to guarantee the safety of the flight. The algorithm keep generating intermediate waypoints above the trajectory planned by the global path planner as long as the object is sensed by the sensors of the UAV.

- the UAV detects another UAV: the local path planner generates an intermediate waypoint rightwards with respect to the planned algorithm and keep doing that until the other UAV is not sensed anymore.

In the simulations a blue line is drawn to show the optimal path planned by the EEA* search algorithm, whereas a red line represents the actual trajectory performed by the fixed-wing UAV. As depicted in 23, the drone avoids in a fairly smooth manner the dynamic obstacle after sensed it, generating intermediate waypoints above the actual path computed by the global path planner. The sensing of the dynamic unknown obstacles and the computation of new waypoints is carried out in real-time.

Algorithm 3 Local Path Planning Pseudo-code

```
Result: New Waypoint
start the simulation at the first time step
while $x$ different from $x_{goal}$ do
    while an obstacle is detected do
        if the obstacle is another UAV then
            generate an intermediate waypoint rightwards
        else
            generate an intermediate waypoint upwards
        end
        reach the next waypoint
        move to the next time step
    end
end
```

In 24, instead, is depicted the behaviour of the local path planner in the case of two fixed-wing UAVs belonging to the fleet who detect each other as dynamically changing obstacles to be avoided. As explained previously, once they detect one another, thanks to the sensors and the assumption made

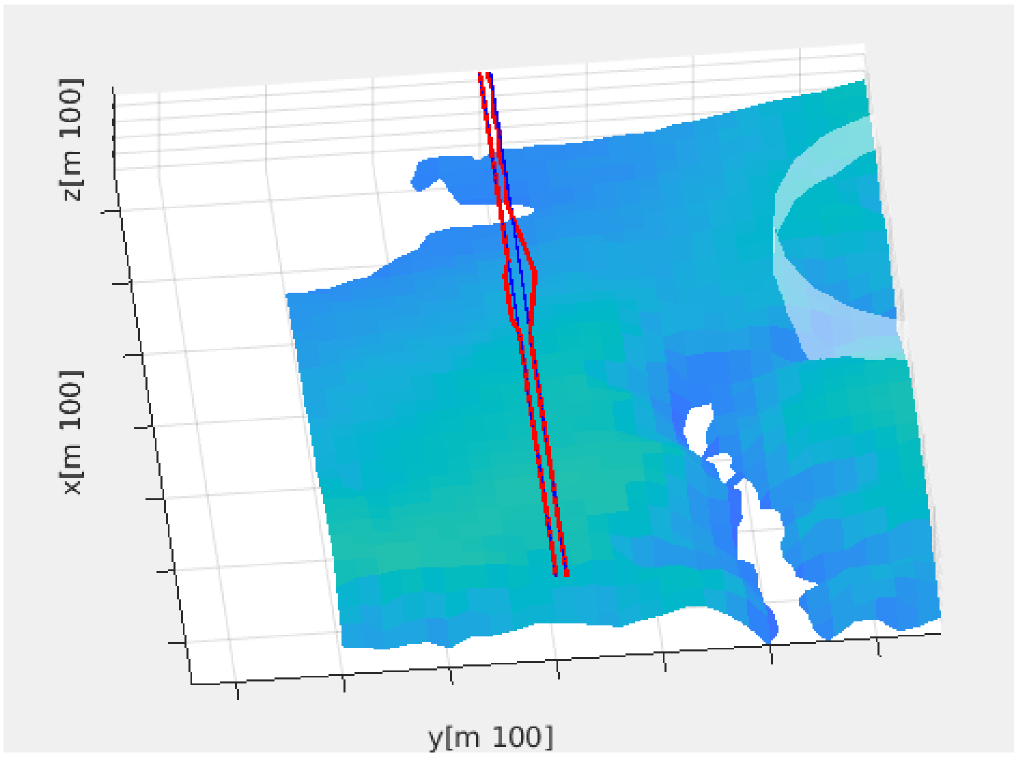


Figure 24: The local path planner performing the dynamic obstacle avoidance in the case of two UAVs of the fleet risking a collision. Once they detect one another, they respectively generate intermediate waypoints rightward.

before that they are able to distinguish another UAV from a generic dynamic obstacle, the algorithm computes intermediate waypoints rightward for both the fixed-wings. Following those new waypoints, generated until the distance does not come back to a safety value, guarantees the two UAVs to do not crash.

10.4 Multi-Agent Implementation

The implementation of the fleet of UAVs has been made to show that the constellation of algorithms and functions employed by each fixed-wing make them able to work properly together in a multi-agent situation. Each drone, receiving the coordinates of the starting point and of the goal point, is able to compute its own nominal path deploying the EEA* Search Algorithm; then

as presented in 32, when the UAVs start following their route, they use their local path planning algorithm to avoid each other treating the other agents as dynamically changing obstacles. Hence an UAV does not know during the flight where the other UAVs are in a specific moment unless they are near enough to be sensed.
Furthermore, the local path planner applies different actions on the computation of the intermediate waypoint depending on if the obstacle is an unrecognized object or another agent. To do that, the agents are able to recognize the other agents once they sense each other. In 25 some examples are depicted.

The workflow of the simulation is the following one:

- The number of agents requested are spawned in the map in their respective starting node.
- The agents receive their goal node.
- The agents employ the global path planner, in the specific the EEA* Search Algorithm developed in this Thesis work.
- The simulation starts and time by time the agents monitor the area around them to look at possible threats.
- If a threat is sensed, depending on if it is an unknown obstacle or another agent, a control action is taken to make it able to avoid the collision.
- Every agent reach its goal node and the simulation ends.

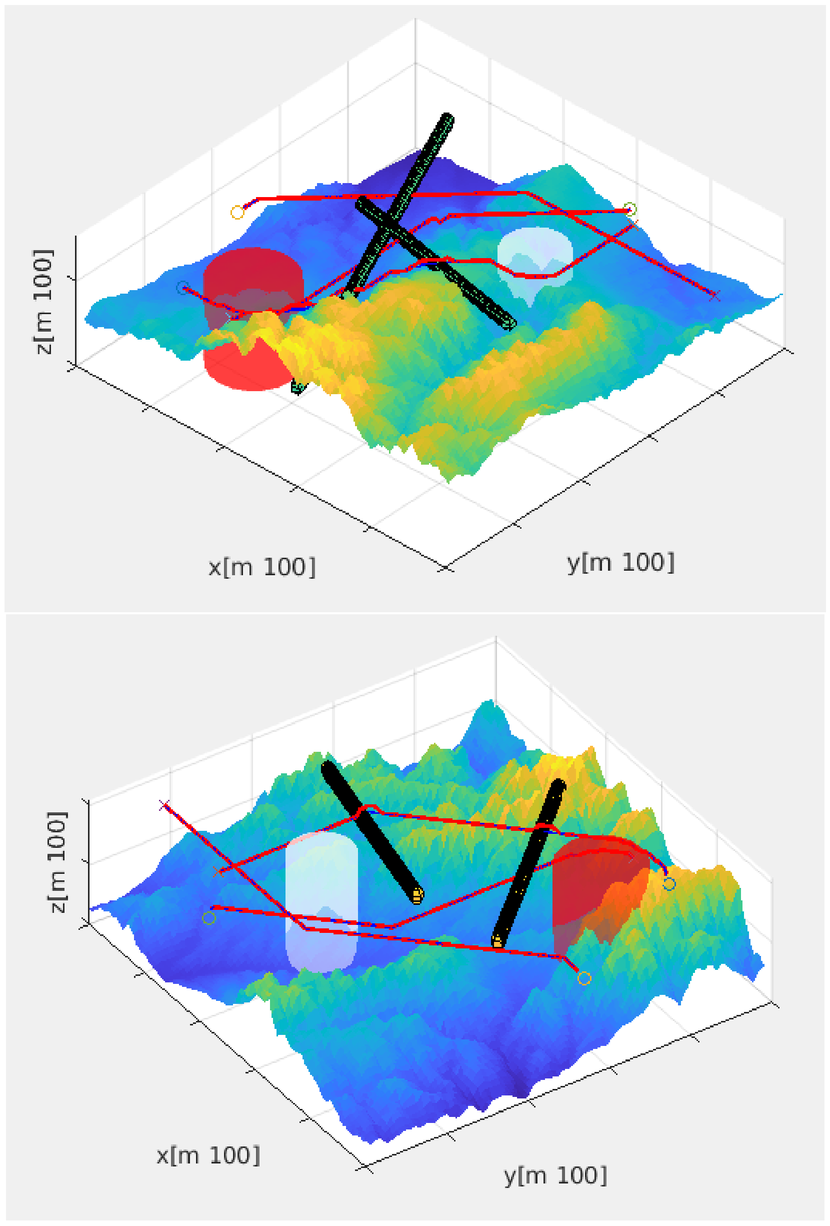


Figure 25: Fleet of UAVs performing their path altogether. When the path intersect the object is not actual a collision but it means the object or the UAV are passed in a different moment in time so they did not collide at all.

Part V
Simulations and Results

11 Lockheed Martin RQ-170 Sentinel

The simulations are performed using the specifications of the RQ-170 Sentinel. The RQ-170 Sentinel is an unmanned aerial vehicle developed by Lockheed Martin and operated by the United States Air Force (USAF) for the Central Intelligence Agency (CIA). While the USAF has released few details on the UAV's design or capabilities, defense analysts believe that it is a stealth aircraft fitted with reconnaissance equipment.
It is a tailless flying wing aircraft, with pods, presumably for sensors or SATCOMs, built into the upper surface of each wing. Estimates of its wingspan range from $13m$ to $26m$. Its takeoff weight is estimated as being greater than the RQ-3 DarkStar's, which is of $3900kg$. The design lacks several elements common to stealth engineering such as zig-zag edged landing gear doors and sharp leading edges, and the exhaust is not shielded by the wing. In [50], it has been postulated that these elements suggest the designers have avoided "highly sensitive technologies" due to the near certainty of eventual operational loss inherent with a single engine design and a desire to avoid the risk of compromising leading edge technology. It has been also suggested that the medium-grey color implies a mid-altitude ceiling, unlikely to exceed $15.000m$ since a higher ceiling would normally be painted darker for best concealment. The postulated weight and ceiling parameters suggests the possible use of a General Electric TF34 engine, or a variant, in the airframe.
On the basis of the few publicly available photographs of the RQ-170, it has been assessed that the UAV is equipped with an electro-optical/infrared sensor and possibly an active electronically scanned array (AESA) radar mounted in its belly fairing.
The "RQ" designation indicates that the RQ-170 Sentinel does not carry weapons.

Known as "The Beast of Kandahar", because Kandahar in Afghanistan was the place where the UAV had been sighted for the first time, RQ-170s have been reported as having operated in Afghanistan as part of Operation Enduring Freedom. It has been confirmed that the UAVs have operated over

Figure 26: A picture of the Lockheed Martin RQ-170 Sentinel, a stealth UAV.

Pakistan and Iran; operations over Pakistan included sorties which collected intelligence before and during the operation which led to the death of Osama Bin Laden in May 2011. In 26 a picture of the RQ-170 Sentinel.

12 Simulations

To recap, the challenges identified and faced in this Thesis work for a path planning algorithm applied on a UAV are:

- 3D environment
- Real-Time performance

- Robustness of the algorithm
- Energy efficiency

The algorithm is developed and works in a 3D environment, it performs its computations in real-time thanks both to the computational lightness of a node-base algorithm and to the help of a local path planner working online to avoid dynamic objects on the route, and lastly it is intrinsically robust, since it is based on an A*.
The feature to be tested is the energy efficiency of the EEA*, that is what makes it different from an A* Search Algorithm. To do that, it has been written a function to calculate the work and the power consumed by the fixed-wing UAV during its flights. 30 simulations have been run both using the A* and the EEA*, varying for each simulation the starting and the goal node. Then, a confront between the two algorithms have been made in terms of energy consumption, power consumption, length of the path.

The specifications of the RQ-170 Sentinel has been used in order to compute the energy spent. Since some of the parameters are not publicly released and only estimated by other researchers, the values has been chosen following the estimated values for the UAV. Since, to calculate the exact value of the energy is not the goal of this thesis, but rather evaluate the improvement in the energy consumption, some parameters have not been obtained by rigorous computations but have been set by estimations in such a way to be as more adherent as possible to the reality. In particular for what concerns the UAV:

- $m = 4000kg$, mass of the UAV
- $b = 20m$, wingspan
- $h_v = 2m$, height of the UAV
- $f = 2000m^2$, equivalent parasitic drag area
- $e = 0.85$, Oswald's efficiency factor4

Then, the other parameters are computed time by time by the function which computes the costs, knowing the altitude h_i to which the UAV is flying in that specific instant and knowing that:

- $g = 9.81 m/s^2$, gravitational acceleration
- $P_0 = 101.325 N/m^2$, sea level atmospheric pressure
- $T_0 = 288,15K$, sea level temperature
- $R = 287.04 m^2/Ks^2$, universal gas constant
- $L = 0.0065K/m$, lapse rate

the following parameters can be computed:

- T, the temperature at the altitude h_i computed as $T_0 - Lh_i$, as long as we are in the troposphere, condition respected since the RQ-170 Sentinel is not apparently intended to flight above $15.000m$
- P, the atmospheric pressure at the altitude h_i, computed as $P = P_0(\frac{T}{T_0})^{\frac{g}{LR}}$
- $\rho(h)$, the air density at the altitude h_i, computed using the equation 64
- A and B, using the equation 62, computed at the specific altitude h_i
- D/L, the drag-to-lift ratio, computed as the optimal one that is $D/L_{opt} = 2\sqrt{AB}$
- V, the velocity of the UAV, kept at its optimal value and computed as $v = (B/A)^{\frac{1}{4}}$
- σ, the steepest rolling angle to be taken to turn, set to a maximum of $0.78rad$ in case of turning

To perform this computation, the UAV has to be able to know the altitude where it is at each moment, hence it has to be equipped by the needed sensors, like a barometer. Then, thanks to a velocity controller, the UAV is kept at the optimal velocity V_{opt} which changes according to the air density at a certain altitude. Since the velocity is not constant and can change to keep the optimal value in every instant, the UAV performs some accelerations and decelerations that have to be taken into account in the computation of the variation of the kinetic energy as

$$\Delta E_k = ma\Delta d \tag{75}$$

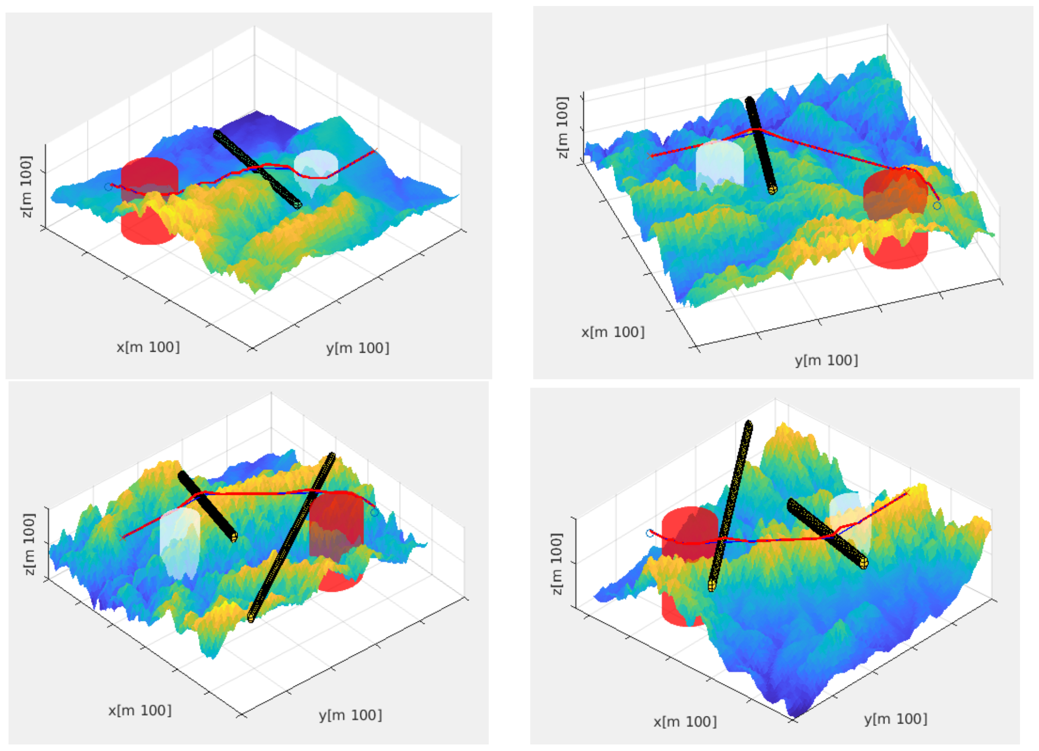


Figure 27: Simulations in different mountain environments. In every map there is a no-fly area in red and a dangerous weather area in white. The blue line is the one computed by the global path planner, the red line is the actual trajectory performed by the fixed-wing UAV. The black and yellow objects are the dynamic threats not known a-priori. The objects are spheres whose the head is the actual threat; the tail is not a threat yet but it is represented to show the trajectory followed by the dynamically changing obstacles. The two examples above are performed in maps with one dynamic obstacle, the two examples below are performed in more challenging environments with two dynamic obstacles. The small circle is the starting node, the small cross is the goal node.

At this point, it has been had all the parameters necessary to compute the total energy consumed employing the equation 16. Nonetheless, given that the developed algorithm has not a path smoothing feature implemented, but deals with the kinematic and dynamic constraints setting the maximum turning angle at $0.78rad$, the dissipated energy for a turning is not reliable and misleading. Such a sharp angle for a fixed-wing UAV at high velocities implies a huge amount of energy dissipated. Turning angles in real life would be much smaller to perform more gentle curves, leading to smaller values in the energy consumed for a turning.
Again, it is not the aim of the Thesis to compute exactly the energy consumed by the aircraft, but rather doing a comparison between the algorithms in terms of amounts of energy consumed, as long as the computation of the work done is as nearer as possible to the actual value and consistent between the algorithms. Therefore, the energy dissipated by turnings maneuvers is neglected, considering only the variation of potential energy ΔE_p and the variation of the kinetic energy ΔE_k.

Once we have the energy spent to follow the path, the power needed is obtained knowing that:

$$[J/s] = [W] \tag{76}$$

After the length of the path d is computed through a proper function, the time t spent to reach the goal is computed as:

$$t = \frac{d}{V} \tag{77}$$

Having t, the power is easily obtained as:

$$P = \frac{\Delta E_{tot}}{t} \tag{78}$$

13 Results

To test the energy efficiency of the EEA* Search Algorithm 30 simulations have been carried out for the EEA* and for the A*. In order to compare the performances, the same start goals and node goals have been given as input. The tests have been made on a HP Pavillion dv6 laptop with a second generation Intel Core i7, an $8GB$ RAM and $500GB$ Hard Disk.

Sim	ΔE_p	ΔE_k	ΔE	P	d	t
1	$27.47MJ$	$1.332GJ$	$1.359GJ$	$30.04MW$	$10291m$	$44.76s$
2	$7.85MJ$	$1.273GJ$	$1.281GJ$	$29.91MW$	$9847m$	$42.83s$
3	$30.50MJ$	$1.081GJ$	$1.111GJ$	$31.63MW$	$7998m$	$35.06s$
4	$3.92MJ$	$1.339GJ$	$1.343GJ$	$29.39MW$	$10510m$	$45.71s$
5	$3.93MJ$	$1.339GJ$	$1.343GJ$	$29.36MW$	$10511m$	$45.7s$
6	$3.92MJ$	$0.9185GJ$	$0.9224GJ$	$33.64MW$	$6470m$	$27.65s$
7	$7.98MJ$	$0.7074GJ$	$0.7154GJ$	$36.09MW$	$4597m$	$19.82s$
8	$3.92MJ$	$0.5406GJ$	$0.5446GJ$	$41.59MW$	$3024m$	$13.09s$
9	$23.54MJ$	$1.5826GJ$	$1.6061GJ$	$29.18MW$	$12600m$	$55.05s$
10	$27.47MJ$	$1.4444GJ$	$1.4719GJ$	$29.91MW$	$11325m$	$49.21s$
11	$35.32MJ$	$1.7567GJ$	$1.7920GJ$	$28.71MW$	$14300m$	$62.40s$
12	$23.54MJ$	$1.2973GJ$	$1.3209GJ$	$30.33MW$	$10061m$	$43.55s$
13	$31.39MJ$	$1.4198GJ$	$1.4512GJ$	$29.64MW$	$11185m$	$48.97s$
14	$39.24MJ$	$1.3935GJ$	$1.3974GJ$	$29.42MW$	$10879m$	$47.50s$
15	$35.32MJ$	$1.6177GJ$	$1.653GJ$	$28.83MW$	$12990m$	$57.33s$
16	$23.54MJ$	$1.1427GJ$	$1.1663GJ$	$31.65MW$	$8504m$	$36.85s$
17	$23.54MJ$	$0.7198GJ$	$0.7433GJ$	$37.3MW$	$4565m$	$19.93s$
18	$0MJ$	$1.5204GJ$	$1.5204GJ$	$28.88MW$	$12168m$	$52.64s$
19	$15.69MJ$	$1.4031GJ$	$1.4188GJ$	$29.53MW$	$10979m$	$48.05s$
20	$19.62MJ$	$1.1810GJ$	$1.2006GJ$	$30.47MW$	$8970m$	$39.40s$
21	$15.70MJ$	$1.3277GJ$	$1.3434GJ$	$29.59MW$	$10348m$	$45.41s$
22	$15.69MJ$	$1.4313GJ$	$1.4470GJ$	$29.21MW$	$11300m$	$49.54s$
23	$31.39MJ$	$1.3421GJ$	$1.3735GJ$	$29.94MW$	$10414m$	$45.87s$
24	$15.69MJ$	$1.744GJ$	$1.7597GJ$	$28.09MW$	$14184m$	$62.65s$
25	$15.69MJ$	$1.3494GJ$	$1.4097GJ$	$29.31MW$	$10915m$	$48.11s$
26	$19.62MJ$	$1.1293GJ$	$1.2789GJ$	$29.82MW$	$9701m$	$42.89s$
27	$19.62MJ$	$1.3601GJ$	$1.3797GJ$	$29.99MW$	$10626m$	$46.10s$
28	$15.7MJ$	$1.1430GJ$	$1.1587GJ$	$31.15MW$	$8513m$	$36.84s$
29	$31.39MJ$	$0.8775GJ$	$0.9098GJ$	$34.46MW$	$6026m$	$26.38s$
30	$31.39MJ$	$1.6788GJ$	$1.7102GJ$	$28.78MW$	$13529m$	$59.42s$

Table 5: Results of the test of the A*. **Sim** stands for the number of simulation, ΔE_p is the variation of the potential energy, ΔE_k is the variation of the kinetic energy, ΔE is the variation of the total energy, P is the average power consumed, d is the total length of the route, t is the time took to arrive at the goal point.

Sim	ΔE_p	ΔE_k	ΔE	P	d	t
1	$0MJ$	$1.352GJ$	$1.352GJ$	$29.07MW$	$10696m$	$46.5s$
2	$0MJ$	$1.340GJ$	$1.340GJ$	$29.32MW$	$10515m$	$45.72s$
3	$5.82MJ$	$1.090GJ$	$1.090GJ$	$30.45MW$	$8067m$	$36.10s$
4	$0MJ$	$1.339GJ$	$1.339GJ$	$29.7MW$	$10510m$	$45.07s$
5	$0MJ$	$1.336GJ$	$1.336GJ$	$29.34MW$	$10470m$	$45.56s$
6	$0MJ$	$0.9114GJ$	$0.9114GJ$	$32.75MW$	$6436m$	$27.83s$
7	$0MJ$	$0.7073GJ$	$0.7073GJ$	$35.56MW$	$4508m$	$19.89s$
8	$0MJ$	$0.5295GJ$	$0.5295GJ$	$41.61MW$	$2941m$	$12.72s$
9	$0MJ$	$1.5100GJ$	$1.5100GJ$	$28.99MW$	$12043m$	$52.09s$
10	$0MJ$	$1.3801GJ$	$1.3801GJ$	$29.37MW$	$10861m$	$46.99s$
11	$15.69MJ$	$1.6858GJ$	$1.7015GJ$	$28.62MW$	$13738m$	$59.45s$
12	$19.62MJ$	$1.2931GJ$	$1.3127GJ$	$29.82MW$	$10051m$	$44.01s$
13	$31.39MJ$	$1.4179GJ$	$1.4493GJ$	$29.46MW$	$11165m$	$49.18s$
14	$11.77MJ$	$1.4052GJ$	$1.4169GJ$	$29.24MW$	$10948m$	$48.44s$
15	$3.92MJ$	$1.5568GJ$	$1.5607GJ$	$28.84MW$	$12507m$	$54.1s$
16	$3.92MJ$	$1.0859GJ$	$1.0898GJ$	$31.29MW$	$8070m$	$34.82s$
17	$0MJ$	$0.657GJ$	$0.657GJ$	$37.06MW$	$4097m$	$17.72s$
18	$0MJ$	$1.5204GJ$	$1.5204GJ$	$28.88MW$	$12168m$	$52.64s$
19	$0MJ$	$1.4417GJ$	$1.4417GJ$	$28.75MW$	$11474m$	$50.14s$
20	$3.92MJ$	$1.1413GJ$	$1.1452GJ$	$30.19MW$	$8677m$	$37.92s$
21	$11.77MJ$	$1.3177GJ$	$1.3295GJ$	$29.51MW$	$10310m$	$45.05s$
22	$7.84MJ$	$1.4073GJ$	$1.4151GJ$	$29.15MW$	$11115m$	$48.53s$
23	$3.92MJ$	$1.2784GJ$	$1.2823GJ$	$29.89MW$	$9914m$	$42.89s$
24	$7.85MJ$	$1.7115GJ$	$1.7193GJ$	$28.04MW$	$13967m$	$61.31s$
25	$0MJ$	$1.3436GJ$	$1.3436GJ$	$28.91MW$	$10581m$	$46.47s$
26	$0MJ$	$1.1933GJ$	$1.1933GJ$	$29.59MW$	$9181m$	$40.33s$
27	$7.84MJ$	$1.3134GJ$	$1.3212GJ$	$29.79MW$	$10250m$	$44.34s$
28	$0MJ$	$1.1198GJ$	$1.1198GJ$	$30.91MW$	$8376m$	$36.22s$
29	$0MJ$	$0.7873GJ$	$0.7873GJ$	$34.07MW$	$5341m$	$23.11s$
30	$0MJ$	$1.5826GJ$	$1.5826GJ$	$28.32MW$	$12787m$	$55.88s$

Table 6: Results of the test of the EEA*. **Sim** stands for the number of simulation, ΔE_p is the variation of the potential energy, ΔE_k is the variation of the kinetic energy, ΔE is the variation of the total energy, P is the average power consumed, d is the total length of the route, t is the time took to arrive at the goal point.

The results have been collected and showed respectively in the table 5 for the A* and in the table 6 for the EEA*. The parameters that have been collected are the variation of potential energy ΔE_p, the variation of the kinetic energy ΔE_k, the variation of the total energy ΔE which is the sum of the first two, the average power consumed P, the total length of the route d, the time required to arrive from the start node to the goal node t.

Then, they have been compared to assess the performance of the EEA* in terms of energy efficiency. Specifically, the variation of the total energy has been compared and the difference between the two has been computed for every simulation done. In this way, the amount of work saved to perform the route with the novel algorithm has been obtained. After that, the same result has been computed in terms of percentage value; hence it has been gotten the percentage of variation of energy saved with the novel algorithm. Lastly, it was also interesting to do that for the average power dissipated per second, though it was not the main goal of the EEA* to minimize that.
In the table 7, those computed datas have been collected and organized as follows: in the first column we have the number of the simulation which the other datas on the raw refers to, in the second column we have the total variation of energy saved, in the third column the percentage of the total variation of energy saved, in the fourth column we have the average power saved per second and in the fifth column the percentage of average power saved per second.

Looking at the table 7, some conclusions can be drawn. The algorithm performs better in terms of energy consumption 86.7% of the time, performs worse 10% of the time and the two algorithm did the same once, equal to the 3.3% of the simulation run.
When the EEA* does better, the work saved by the UAV spans between 0.13% and 13.46% of the one done with the A*, with an average reduction of $41.5 MJ$ per flight corresponding to an average 3.43% of variation of energy saved. The standard deviations are equal to $4.67 MJ$ and 4.04%.

For what concerns the average power dissipated, the EEA* uses less power than the A* in the 86.7% of the times, uses more power in the 10% of the cases and uses the same power in the 3.3% of the cases. Even if the percentage is the same as for the variation of the total energy, the two things does not seem to be correlated. When EEA* performs better, in percentage it consumes from 0.07% to 3.73% less, with an average reduction of 0.95% equal to $310 kW$

Sim	ΔE **Saved**	$\Delta E\%$ **Saved**	P **Saved**	$P\%$ **Saved**
1	$0.007GJ$	0.53%	$0.97MW$	3.22%
2	$-0.059GJ$	-4.63%	$0.59MW$	1.97%
3	$0.020GJ$	1.85%	$1.18MW$	3.73%
4	$0.004GJ$	0.30%	$-0.31MW$	-1.05%
5	$0.007GJ$	0.52%	$0.02MW$	0.07%
6	$0.011GJ$	1.20%	$0.89MW$	2.65%
7	$0.0081GJ$	1.15%	$0.53MW$	1.47%
8	$0.0151GJ$	2.79%	$-0.02MW$	-0.05%
9	$0.0961GJ$	6.07%	$0.19MW$	0.65%
10	$0.0918GJ$	6.36%	$0.54MW$	1.81%
11	$0.0905GJ$	5.15%	$0.09MW$	0.31%
12	$0.0082GJ$	0.63%	$0.51MW$	1.68%
13	$0.0019GJ$	0.13%	$0.18MW$	0.61%
14	$-0.0195GJ$	-1.40%	$0.18MW$	0.61%
15	$0.0923GJ$	5.58%	$-0.01MW$	-0.03%
16	$0.0765GJ$	6.56%	$0.36MW$	1.14%
17	$0.0863GJ$	11.61%	$0.24MW$	0.64%
18	$0GJ$	0%	$0MW$	0%
19	$-0.0229GJ$	-1.61%	$0.78MW$	2.64%
20	$0.0554GJ$	4.61%	$0.28MW$	0.92%
21	$0.0139GJ$	1.03%	$0.08MW$	0.27%
22	$0.0319GJ$	2.20%	$0.06MW$	0.21%
23	$0.0912GJ$	6.64%	$0.05MW$	0.17%
24	$0.0404GJ$	2.30%	$0.05MW$	0.18%
25	$0.0661GJ$	4.69%	$0.40MW$	1.36%
26	$0.0856GJ$	6.69%	$0.23MW$	0.77%
27	$0.0585GJ$	4.24%	$0.20MW$	0.67%
28	$0.0389GJ$	3.36%	$0.24MW$	0.77%
29	$0.1225GJ$	13.46%	$0.39MW$	1.13%
30	$0.1276GJ$	10.90%	$0.46MW$	1.60%

Table 7: Results of the comparison between A* and EEA*. **Sim** stands for the number of simulation, ΔE **Saved** is the work saved employing the EEA*, $\Delta E\%$ **Saved** is the percentage of work saved, P **Saved** is the amount of average power saved, $P\%$ **Saved** is the percentage of average power saved.

ΔE **Saved**
Mean = $41.5MJ$ Standard Deviation= $46.8MJ$ Mean[%] = 3.43% Standard Devation[%] = 4.04%
P **Saved**
Mean= $310kW$ Standard Deviation= $330kW$ Mean[%] = 0.95% Standard Deviation[%] = 1.11%

Table 8: The table above collects the mean value of the variation of the total energy which is saved, its standard deviation and those values expressed in percentage. Below, the mean value of the average power saved, its standard deviation and those values expressed in percentage.

and with standard deviations equal to 1.11% and $330kW$.

In 28, the variation of the total energy of both the algorithm has been plotted in order to see and visualize the differences between the two. The plot above is a diagram which interpolates the results obtained in the 30 simulations and order them in an increasing order for the values obtained by the A*, then at each value corresponds the related value obtained by that simulation for the EEA*. That is why the A* plot seems to be smoother than the EEA* one.
It is evident that the EEA* plot is almost always below the A* plot, but equally evident that the EEA* standard deviation of the work done is pretty high. In fact, the distance between the two plots changes highly among the simulations.
The same datas are then plotted on a histogram to visualize the comparison among each single simulation.

In 29, the same plots have been done highlighting the average power consumption of the two algorithms. Also in this case the values related to the A* have been ordered in a non-decreasing fashion, then matched with the EEA* values.
In the diagram above it can be noticed that for higher values of P the two plots seem to tend to the same values. However, in both the diagram and the

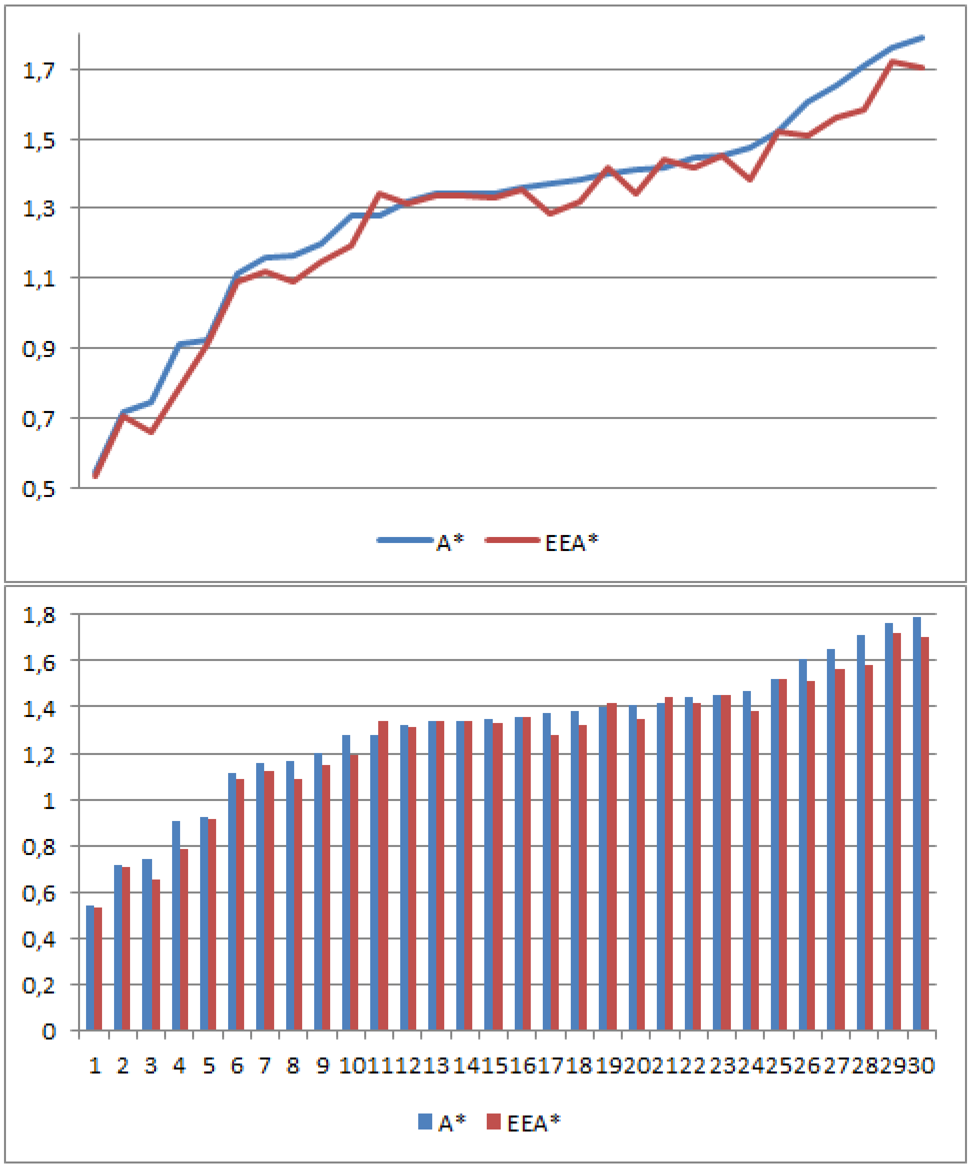
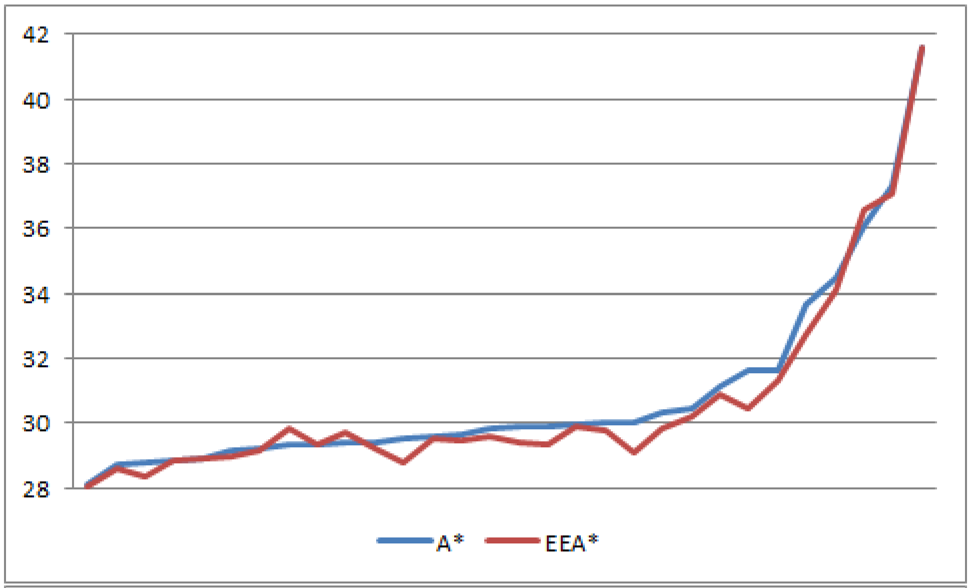


Figure 28: Comparison between A* and EEA* in terms of variation of total energy consumption. The first picture shows the results plotted on a diagram. The second picture shows the results plotted on a histogram.



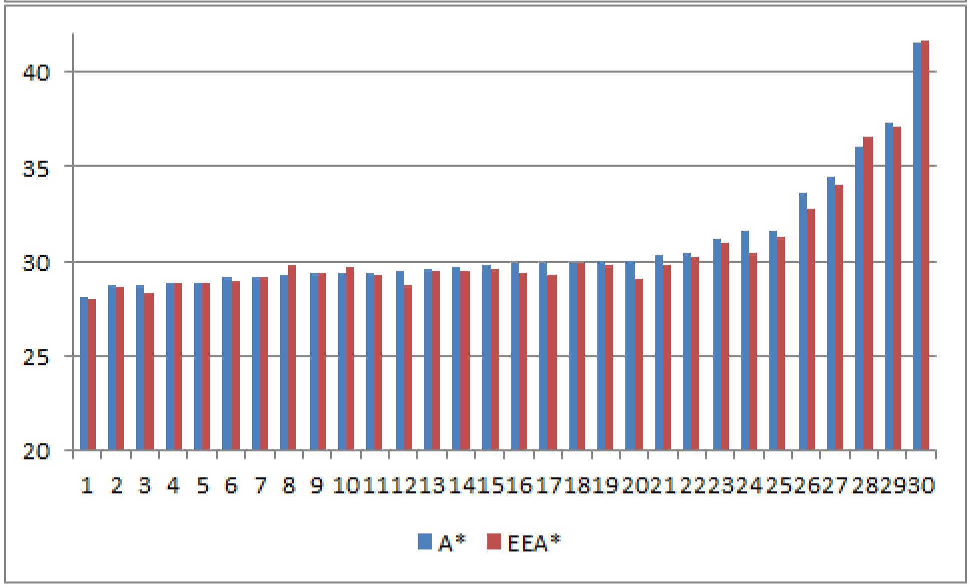


Figure 29: Comparison between A* and EEA* in terms of average power consumed per second. The first picture shows the results plotted on a diagram. The second picture shows the results plotted on a histogram.

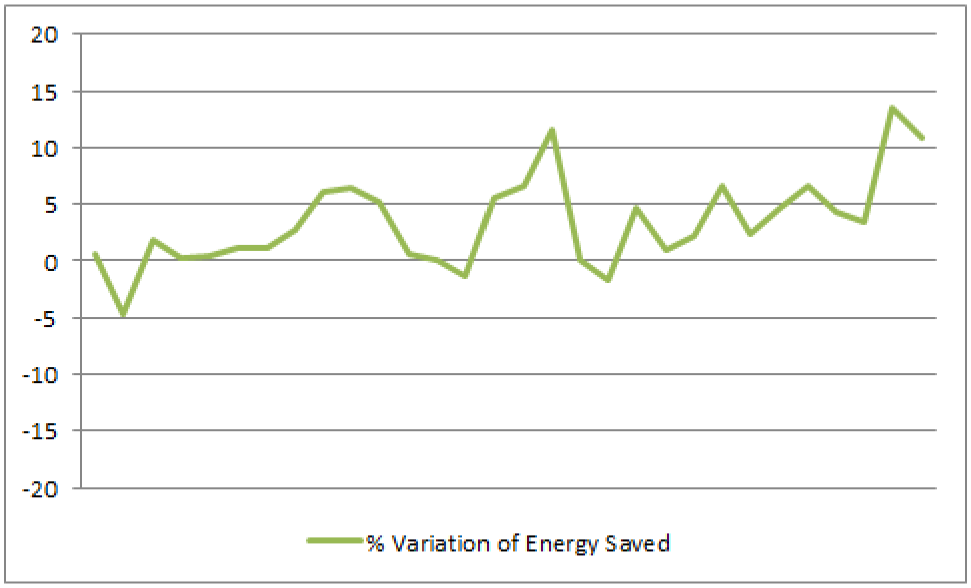


Figure 30: The plot shows the percentage of variation of the total energy saved by means of using the EEA* algorithm instead of the A*.

histogram it can be clearly seen that the real amount of P saved is not so great. Indeed, this result was expected since P is not the parameter to be optimized, but rather it has some little improvement indirectly improving the energy efficiency.

In the figure 30 is depicted the diagram of the percentage of variation of total energy saved by means of using the EEA* instead of the A* to solve the global path planning problem. In this case is even more evident the high standard deviation of the results which, depending highly on the environment in which the algorithm is deployed, can perform much either better than the A* or almost the same as it and in a few cases worse. Nonetheless, with a mean of 3.43% of work saved and with a success rate of 86.7% it is convenient to use this novel algorithm and it seems reasonable to accept a 10% risk of wasting more energy.

On the other hand, for what concerns the percentage of average power dissipated, it can be seen in the figure 31 that the function obtained is near to zero values, having just small savings around the 1%, which considering that

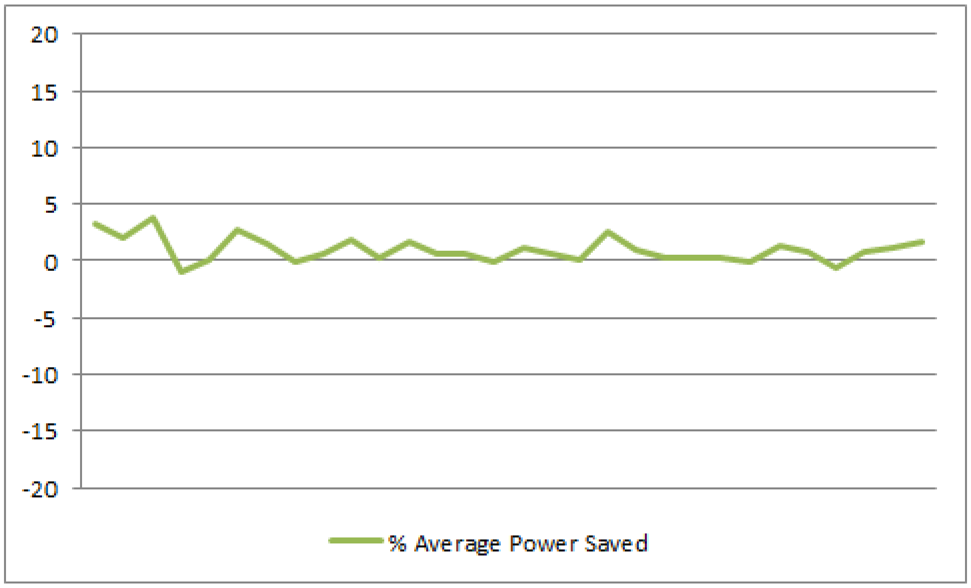


Figure 31: The plot shows the percentage of average power saved by means of using the EEA* algorithm instead of the A*.

is a value it has not been worked on it is a good result, anyway. Given that it is an instantaneous value, its goodness strongly depends on not only the work done but also on the time required to complete the path which depends on the velocity of the UAV. Still, the velocity itself in our case is not constant but computed but depends on the altitude in which the UAV is flying, in order to guarantee the optimal velocity to be kept along the route.

During the 30 simulations using the A* and the 30 simulations of the EEA*, for a total of 60 simulations, no collision has been registered. Both the static obstacles and the dynamic obstacles has been avoided correctly by the algorithms reporting good performances.
In the figure 32 is depicted an example of the control action applied by the local path planner. The diagram depicted shows on the vertical axis the distance expressed in meters between an UAV and one of the dynamic threats present in the map.
As it can be seen, the UAV approaches the object quickly and dangerously in the first part of the plot. Nonetheless, at a certain point, when the distance is more or less near to $500m$, the sensors of the drone detect the moving

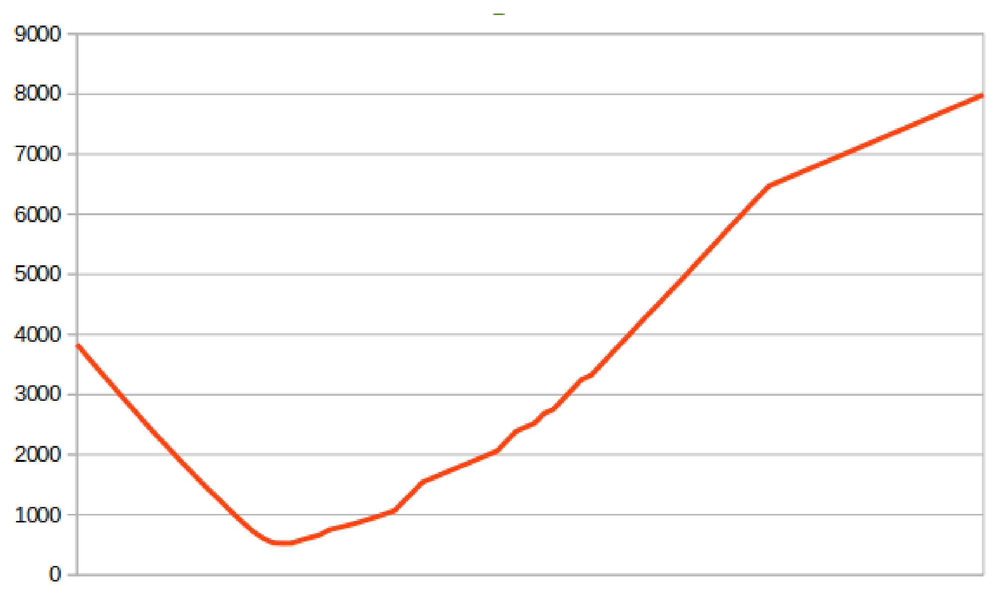


Figure 32: The diagram shows the action of the local path planner with respect to the distance from the UAV and the object. The vertical axis represents the distance in meters between a certain UAV and a dynamic threats in the map.

obstacle approaching and invading the area right in front of the fixed-wing UAV. At that point, the local path planner comes into action. Moving the UAV upwards, the local path planner allows the aircraft to be kept far from the threats avoiding a collision. This behaviour is depicted in the diagram with the sharp turning of the function, with the distance kept for a while at a reasonable distance of half a kilometer. In that area of the plot the UAV and the object are flying in the same region, but thanks to the local path planner, they do that in a safe way. After that, it can be seen an almost linear increase of the distance between the two, signifying a departure since both the UAV and the object keep flying on their own path.

Part VI
Conclusions and Future Works

In this Thesis work has been done a literature review concerning the last decade researches on path planning for UAVs. After a detailed classification of the algorithms employed, limitations have been identified and common challenges have been recognized. In particular four challenges have been identified: algorithm able to deal with 3D environments, algorithm able to work in real-time, robust and able to take into account of the energy consumption. These challenges have been overcome in different ways by researchers, nonetheless, as far as we know, none of them have solved the four problems altogether, at leas for UAV applications.

In the second part of the Thesis, a novel node-based 3D real-time energy efficient path planning algorithm has been developed using MATLAB in order to overcome this four challenges altogether. Specifically, the algorithm, called EEA* Search Algorithm, is based on the A* Search Algorithm and deploys a global path planner and a local path planner finding the most energy efficient route and avoiding dynamic unknown obstacles in the map. The maps are 3D mountain-wise maps built using MATLAB.

For the simulations, a model of a Lockheed Martin RQ-170 Sentinel has been chosen. The parameters of this kind of stealth UAV have been set up and the algorithms have been deployed in different and gradually more complicated environments. Then, three fixed-wings UAVs have been employed to work together without interfere with each other. Some pictures of the models of the UAV have been collected showing the efficiency and the results.

In the end, the energy efficiency has been tested. To do that, 30 simulations have been carried out for both the EEA* and the A* feeding them with the same inputs and looking for the results.
From the datas collected it comes out that the EEA* reduces the work needed to follow the route in 86.7% of the time and of the 3.43% on average. Furthermore, also the average power dissipated has been indirectly decreased in 86.7% of the cases of a 0.95%.

Even if the algorithm seems to show a good behaviour, still a lot of future work has to be done. In future works, we have to aim at increasing the uncertainties adding wind disturbance in such a way to work on an environment as much as similar to the real world as possible.
Another important aspect to be considered, to better validate the algorithm developed, is to add a path smoothing algorithm in order to have more realistic and feasible paths to be performed by a fixed-wing UAV. This feature not only have this pro, but it would permit us to study better the performance of the energy efficiency, taking into account of the work needed for turning maneuvers. Indeed, with more realistic paths, the variation of the energy would be much more accurate, further reducing the approximation of the results.

References

[1] Choset and M. Howie, *Principles of robot motion: theory algorithms and implementations*, MIT press, 2005.

[2] Schøler and Flemming, *3D Path Planning for Autonomous Aerial Vehicles in Constrained Spaces*, Diss. Videnbasen for Aalborg UniversitetVBN Aalborg UniversitetAalborg University Det Teknisk-Naturvidenskabelige FakultetThe Faculty of Engineering and Science Institut for Elektroniske SystemerDepartment of Electronic Systems.

[3] H. Zh. Zhuang, S. X. Du, T. J. Wu, *Real-Time Path Planning for Mobile Robots*, 2005 International Conference on Machine Learning and Cybernetics. IEEE: 526–531, 2005. doi:10.1109/icmlc.2005.1527001. ISBN 0780390911.

[4] S M LAValle, *Planning algorithms*, Cambridge university press, 2006.

[5] Y.V. Pehlivanoglu, O. Baysal, A. Hacioglu, *Path planning for autonomous UAV via vibrational genetic algorithm*, Aircraft Engineering and Aerospace Technology, Vol. 79 Iss 4 pp. 352 - 359, 2007.

[6] K. Yang, S. Sukkarieh, *3D Smooth Path Planning for a UAV in Cluttered Natural Environments*, 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008.

[7] D. Zhuoninga, Z. Rulinb, C. Zongjia, Z. Ruia, *Study on UAV Path Planning Approach Based on Fuzzy Virtual Force*, Chinese Journal of Aeronautics 23(2010) pp. 341-350, 2010.

[8] H. Duan, Y. Yu, X. Zhang, S. Shao, *Three-dimension path planning for UCAV using hybrid meta-heuristic ACO-DE algorithm*, Simulation Modelling Practice and Theory 18 (2010) 1104–1115, 2010.

[9] Z.Dong, Z. Chen, R. Zhou, R. Zhang, *A Hybrid Approach of Virtual Force and A* Search Algorithm for UAV Path Re-Planning*, 2011 6th IEEE Conference on Industrial Electronics and Applications, 2011.

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

[11] E.I. Grøtli, T.A. Johansen, *Path Planning for UAVs Under Communication Constraints Using SPLAT! and MILP*, J Intell Robot Syst (2012) pp. 265–282, 2012.

[12] A. Chamseddine, Y. Zhang, C.A. Rabbath, C.Join, D. Theilliol, *Flatness-Based Trajectory Planning/Replanning for a Quadrotor Unmanned Aerial Vehicle*, IEEE TRANSACTIONS ON AEROSPACE AND ELECTRONIC SYSTEMS VOL. 48, NO. 4 OCTOBER 2012, 2012.

[13] T. W. McLain, R. Beard, *Small Unmanned Aircraft: Theory and Practice*, 2012.

[14] L. Wei, Z. Zheng, C. Kai-Yuan, *Bi-level programming based real-time path planning for unmanned aerial vehicles*, Knowledge-Based Systems 44 (2013) pp. 34–47, 2013.

[15] L. Wei, Z. Zheng, C. Kaiyuan, *Adaptive path planning for unmanned aerial vehicles based on bi-level programming and variable planning time interval*, Chinese Journal of Aeronautics, 2013,26(3) pp. 646–660, 2013.

[16] V. Roberge, M. Tarbouchi, G. Labonté, *Comparison of Parallel Genetic Algorithm and Particle Swarm Optimization for Real-Time UAV Path Planning*, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, VOL. 9, NO. 1, FEBRUARY 2013, 2013.

[17] F. Yan, Y. Liu, J.Z. Xiao, *Path Planning in Complex 3D Environments Using a Probabilistic Roadmap Method*, International Journal of Automation and Computing 10(6), December 2013, pp. 525-533, 2013.

[18] J. Valente, J. Del Cerro, A. Barrientos, David Sanz, *Aerial coverage optimization in precision agriculture management: A musical harmony inspired approach*, Computers and Electronics in Agriculture 99 (2013) pp. 153–159, 2013.

[19] V. Roberge, M. Tarbouchi, F. Allaire, *PARALLEL HYBRID META-HEURISTIC ON SHARED MEMORY SYSTEM FOR REAL-TIME UAV PATH PLANNING*, International Journal of Computational Intelligence and Applications Vol. 13, No. 2 (2014) 1450008 (16 pages), 2014.

[20] L. Yang J. Qi J. Xiao, X. Yong, *A literature review of UAV 3D path planning*, Proceeding of the 11th World Congress on Intelligent Control and Automation, 2014.

[21] C. Wite, A. Botea, *Spatially Distributed Multiagent Path Planning*, Proceedings of the Twenty-Fourth International Conference on Automated Planning and Scheduling, 2014.

[22] A. Bircher, M. Kamel, K. Alexis, M. Burri, P. Oettershagen, S. Omari, T. Mantel, R. Siegwart, *Three-dimensional coverage path planning via viewpoint resampling and tour optimization for aerial robots*, Auton Robot, 2015.

[23] D. Ortiz-Arroyo, *A Hybrid 3D Path Planning Method for UAVs*, 2015 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS) November 23-25, 2015. Cancun, Mexico, 2015.

[24] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, R. Siegwart, *Receding horizon path planning for 3D exploration and surface inspection* Auton Robot (2018) 42 pp. 291–306, 2016.

[25] B. Ingersoll, K. Ingersoll, P. DeFranco, A. Ningx, *UAV Path-Planning using Bézier Curves and a Receding Horizon Approach*, AIAA Modeling and Simulation Technologies Conference, 2016.

[26] M. Radmanesh, M. Kumar, *Grey Wolf Optimization based Sense and Avoid Algorithm for UAV Path Planning in Uncertain Environment using a Bayesian Framework*, 2016 International Conference on Unmanned Aircraft Systems (ICUAS) June 7-10, 2016. Arlington, VA USA, 2016.

[27] M. da Silva Arantes, J. da Silva Arantes C. F. M. Toledo, B. C. Williams, *A Hybrid Multi-Population Genetic Algorithm for UAV Path Planning*, GECCO '16, July 20-24, 2016, Denver, CO, USA, 2016.

[28] L. Yang, J. Qi, D. Song, J. Xiao, J. Han, Y. Xia, *Survey of Robot 3D Path Planning Algorithms*, Hindawi Publishing Corporation Journal of Control Science and Engineering Volume 2016, Article ID 7426913, 22 pages, 2016. http://dx.doi.org/10.1155/2016/7426913, 2016.

[29] M. Kang, Y. Liu, Y. Ren†, Y. Zhao, Z. Zheng, *An Empirical Study on Robustness of UAV Path Planning Algorithms Considering Position*

Uncertainty, 2017 12th International Conference on Intelligent Systems and Knowledge Engineering (ISKE), 2017.

[30] S. Hayat, E. Yanmaz, T. X. Brown, C. Bettstetter, *Multi-Objective UAV Path Planning for Search and Rescue*, 2017 IEEE International Conference on Robotics and Automation (ICRA) Singapore, May 29 - June 3, 2017, 2017.

[31] Z. Ahmad, F. Ullah, C. Tran, S. Lee, *Efficient Energy Flight Path Planning Algorithm Using 3-D Visibility Roadmap for Small Unmanned Aerial Vehicle*, International Journal of Aerospace Engineering, 2017.

[32] Z. Zhang, D. Scaramuzza, *Perception-aware Receding Horizon Navigation for MAVs*, 2018 IEEE International Conference on Robotics and Automation (ICRA) May 21-25, 2018, Brisbane, Australia, 2018.

[33] T. Zhang, X. Huo, S. Chen, B. Yang, G. Zhang, *Hybrid Path Planning of A Quadrotor UAV Based on Q-Learning Algorithm*, Proceedings of the 37th Chinese Control Conference July 25-27, 2018, Wuhan, China, 2018.

[34] G. Zhang, L. Hsu, *A New Path Planning Algorithm Using a GNSS Localization Error Map for UAVs in an Urban Area*, Journal of Intelligent Robotic Systems, 2019.

[35] C. Papachristos, F. Mascarich, S. Khattak, T. Dang, K. Alexis, *Localization uncertainty-aware autonomous exploration andmapping with aerial robots using receding horizon path-planning.*, Autonomous Robots https://doi.org/10.1007/s10514-019-09864-1, 2019.

[36] E. J. Dhulkefl, A. Durdu, *Path Planning Algorithms for Unmanned Aerial Vehicles*, International Journal of Trend in Scientific Research and Development (ijtsrd), ISSN: 2456- 6470, Volume-3 | Issue-4, June 2019, pp. 359-362, URL: https://www.ijtsrd.c om/papers/ijtsrd23 696.pdf, 2019.

[37] S. Aggarwal, N. Kumar, *Path planning techniques for unmanned aerial vehicles: A review, solutions, and challenges*, Computer Communications, 2019.

[38] X. Liu, X. Du, X. Zhang, Q. Zhu, M. Guizani, *Evolution-algorithm-based unmanned aerial vehicles path planning in complex environment*, Computers and Electrical Engineering 80 (2019) 106493, 2019.

[39] R. J. Wai, A. S. Prasetia, *Adaptive Neural Network Control and Optimal Path Planning of UAV Surveillance System With Energy Consumption Prediction*, IEEE Access VOLUME 7, 2019, 2019.

[40] M. Faria, R. Marín, M. Popovic, I. Maza, A. Viguria, *Efficient Lazy Theta* Path Planning over a Sparse Grid to Explore Large 3D Volumes with a Multirotor UAV*, Sensors 2019, 19, 174, 2019.

[41] P. Pianpak, T. C. Son, Z. O. Toups, W. Yeoh, *A Distributed Solver for Multi-Agent Path Finding Problems*, First International Conference on Distributed Artificial Intelligence (DAI '19), October 13–15, 2019.

[42] H. Freitas, B. S. Faiçal, A. Vinicius Cardoso e Silva, J. Ueyama, *Use of UAVs for an efficient capsule distribution and smart path planning for biological pest control*, Computers and Electronics in Agriculture 173 (2020) 105387, 2020.

[43] C. Qu, W. Gai, M. Zhong, J. Zhang, *A novel reinforcement learning based grey wolf optimizer algorithm for unmanned aerial vehicles (UAVs) path planning*, Applied Soft Computing Journal (2020), doi: https://doi.org/10.1016/j.asoc.2020.106099, 2020.

[44] https://en.wikipedia.org/wiki/MATLAB.

[45] https://en.wikipedia.org/wiki/A*_search_algorithm

[46] B. H. Carson, *Fuel efficiency of small aircraft*, Journal of Aircraft, vol. 19, no. 6, pp. 473–479, 1982.

[47] Dechter, Rina, *Generalized best-first search strategies and the optimality of A**, Journal of the ACM. 32 (3): 505–536, 1985.

[48] M. Cavcar, *The international standard atmosphere (ISA)*, Anadolu University, Turkey, vol. 30, 2000.

[49] G. Nachmani, *Minimum-energy flight paths for UAVs using mesoscale wind forecasts and approximate dynamic programming*, DTIC Document, 2007.

[50] Fulghum, David A.; B. Sweetman, *Stealth over Afghanistan*, Aviation Week, McGraw-Hill: 26–27, 2009

Acknowledgements

First of all, I would like to say thank you to Professor Rizzo and Professor Valavanis for giving me the privilege and the honor of studying for my Master Thesis guided by them and for allowing me to crown my dream to live in the United States for a while. Thank you for the suggestions, the material, the hints and the utter support you gave me in order to do this job.
In particular, I would like to thank Kimon for being a friend for me during this experience, giving to me and to my colleague Giancarlo the courage to face the pandemic in another country, so far from home, and for never letting us feel alone. Without your humankind the whole experience would have been way harder than it was. Thank you!
I also would like to say thank you to Professor Matthew Rutherford for all the time spent together in the laboratory, even only for a chat or a suggestion.

Ritengo di utilizzare un eufemismo affermando che questi due anni sono stati i più intensi e incredibili della mia vita. Ho lasciato il nido di casa quello che sembra essere una vita fa. Ho conosciuto persone, visto posti, fatto cose, imparato tanto, allargato i miei orizzonti. E a compimento di questi due anni ho avuto il privilegio di coronare il mio sogno di vivere e studiare negli Stati Uniti, contro ogni probabilità, nonostante il mondo fosse sottosopra. *"Audentes fortuna iuvat"* disse Virgilio. Beh, non so se mi si addica di più la parola audace o imprudente per non avere mollato durante uno dei periodi più folli che si ricordi, quello che è certo è che la fortuna mi ha aiutato. Oltre che dalla fortuna, ogni successo deriva certamente da dedizione e impegno, ma altrettanto dal supporto e dalla presenza delle tante persone senza le quali questo non sarebbe stato possibile. So bene di fare retorica, eppure quanto è vero!

Voglio ringraziare i miei genitori, ai quali dedico questo lavoro di tesi e questo successo. Che sia anche il vostro! Chi sono lo devo a voi e senza il vostro supporto tutto ciò, questa tesi compresa, non sarebbe stato possibile. Grazie!

Ringrazio Giorgia, per avere capito e appoggiato le mie scelte. So bene davanti a quali prove ti ho messo di fronte e malgrado tutto ci sei stata. Non potrò mai essere grato abbastanza per avere trovato una persona come te.

Ringrazio Marco e Francesca, per esserci sempre per me, un solo vostro gesto mi fa sentire tutto il calore fraterno e mi ridona tranquillità. Con voi non ho trovato solo due fratelli ma due complici. Non saprei come fare senza di voi.

Un grazie ai miei fratelli non di sangue, Alessandro, Nino, Gianmarco e Alessio. Nemmeno migliaia di chilometri ci hanno mai separato e mai lo faranno. Non c'è nulla al mondo di più raro e prezioso dell'eterna amicizia che ci lega.

Un grazie particolare a Giancarlo, con il quale ho condiviso questa avventura folle e irripetibile dall'altra parte del mondo. Insieme nemmeno una pandemia è stata in grado di fermarci!

Non ringrazierò mai abbastanza Alex, mio compagno dal primo giorno in questo incredibile viaggio, ormai cinque anni or sono. Chiudiamo insieme questo cerchio e non poteva essere diversamente, come se nella vita a volte tutto sia scritto dalla stessa Mano. A te il meglio che la vita ti possa offrire, te lo meriti!

Sicuramente dimenticherò qualcuno, sarebbe impossibile ringraziare tutti quelli che mi sono stati al fianco in questi due anni e avrei bisogno di un romanzo per ringraziare ognuno di voi personalmente.
Grazie a Diego, Martina, Marco, Marco, Alessio, Marika, Luigi, Kaori, Riccardo, Enzo, Luca, per avere allegerito le fatiche di questi anni con la vostra simpatia; grazie a Gaetano, Piero e agli amici della residenza Borsellino Angelo, Angelo, Francesco, Palma, Tania, Egle, Katia, Hamed, Leo, Paolo, Elisa, per avere riempito di risate e allegria i miei giorni a Torino; grazie a Silvano, Mara, Mauro, Liliana, Carlo, Marcella, Civita, Davide, Enza, Peppe, Mariella, per avermi sempre fatto sentire a casa e parte della vostra famiglia; grazie a zio Franco, zia Bianca, Stefania, per l'infinito affetto che mi date, vi voglio un mondo di bene; grazie ad Alessandra che con la sua sensibilità mi ricorda di non smettere mai di inseguire i miei sogni e di godere delle piccole cose; grazie a Ciro e Barbarba per esserci sempre stati, anche quando nessuno sembrava esserci.

Alcuni affermano che spesso un'infanzia felice e una vita felice e di successo abbiano un filo conduttore che le collega. Per questo voglio ringraziare i miei nonni. La mia infanzia e la mia gioventù sono piene di momenti e ricordi con

voi e se quegli anni sono stati meravigliosi è anche merito vostro. E se è vero quanto detto sopra, e sono fermamente convinto che lo sia, allora di certo chi sono ora è anche merito vostro.
Ciao, Nonna.

[...] illuminate dal chiarore della luna piena e dal candore del deserto, si ergevano maestose e solenni le Piramidi d'Egitto. Il ragazzo cadde in ginocchio e scoppiò a piangere. Ringraziava il Signore per aver creduto nella propria Leggenda Personale e per avere incontrato un giorno un re, un mercante, un inglese e un Alchimista. Ma, soprattutto, per avere incontrato una donna del deserto che gli aveva fatto capire come l'Amore non avrebbe mai separato nessuno dalla propria Leggenda Personale.

Paulo Coelho, L'alchimista