

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO



Control of Multiple Aerial Vehicles in a Mixed-Initiative Environment

Hugo Manuel Soares Oliveira

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Orientador: Sujit Pedda Baliyarasimhuni

Orientador: Joao Tasso Borges de Sousa

23 October 2013

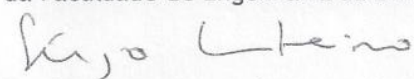
A Dissertação intitulada

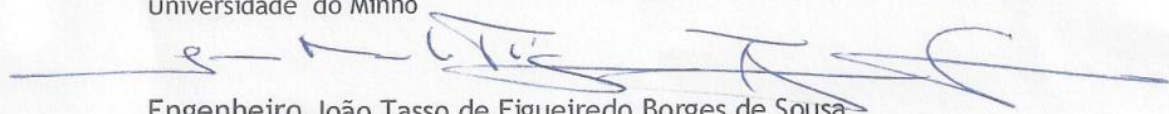
“Control of Multiple Aerial Vehicles in a Mixed-Initiative Environment”

foi aprovada em provas realizadas em 07-10-2013

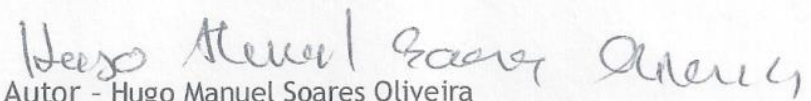
o júri


Presidente Professor Doutor Aníbal Castilho Coimbra de Matos
Professor Auxiliar do Departamento de Engenharia Eletrotécnica e de Computadores
da Faculdade de Engenharia da Universidade do Porto


Professor Doutor Sérgio Paulo Carvalho Monteiro
Professor Auxiliar do Departamento Eletrónica Industrial da Escola de Engenharia da
Universidade do Minho


Engenheiro João Tasso de Figueiredo Borges de Sousa
Assistente Convidado do Departamento de Engenharia Eletrotécnica e de
Computadores da Faculdade de Engenharia da Universidade do Porto

O autor declara que a presente dissertação (ou relatório de projeto) é da sua exclusiva autoria e foi escrita sem qualquer apoio externo não explicitamente autorizado. Os resultados, ideias, parágrafos, ou outros extratos tomados de ou inspirados em trabalhos de outros autores, e demais referências bibliográficas usadas, são corretamente citados.


Autor - Hugo Manuel Soares Oliveira

Resumo

Veículos não tripulados tem vindo a ser adoptados em larga escala para a utilização em missões militares e civis sobre o ar, mar, espaço e no solo. O seu uso massivo é justificado principalmente por evitar a colocação da vida humana em perigo. Além disso, a inexistência operador a bordo permite uma redução significativa de peso no que respeita aos dispositivos de suporte de vida, resultando num menor custo e aumento da *endurance*. O contínuo aumento das capacidades autónomas dos veículos aéreos possibilita a sua utilização em conjunto de equipas, criando um novo paradigma operacional.

Tais características permitem a utilização de pequenos veículos não tripulados em áreas urbanas, permitindo o seguimento de alvos de forma próxima, impossível de se realizar com o uso de veículos de larga escala tripulados, tais como helicópteros. O sistema apresentado foi projectado com o objectivo de realizar tarefas de inteligência persistente, reconhecimento e vigilância de alvos em ambiente urbano.

O uso de sistemas UAV de baixo custo, baixa altitude, baixa *endurance* permite fornecer dados de alta resolução para pessoal de terra mantendo os custos de operação e de logística baixos. Estes UAV's podem ser colocados em operação rapidamente e de forma simples, recorrendo a diferentes combinações de sistemas de lançamento (Mão, catapulta, pista curta). Uma das principais desvantagens deste veículos é a sua limitada capacidade relativo aos sensores e *endurance*. No entanto, isto pode ser superado pela partilha correcta de informações relevantes entre os veículos membros da equipa.

O objectivo deste trabalho é estudar e desenvolver técnicas de cooperação entre equipas de UAVs descentralizadas, a fim de executar tarefas de vigilância persistente numa área urbana densa, com fortes oclusões. Para o efeito, várias técnicas de seguimento e cooperação foram estudados a fim de extrair algumas ideias úteis a adoptar no desenvolvimento do projecto. Tendo em vista o objectivo final desta tese, foi desenvolvido o ambiente de simulação que permite replicar as condições substanciais do cenário de operação, controladores e algoritmos de planeamento de trajectórias para dirigir os veículos aéreos, sistemas de seguimento distribuído com capacidade para multi alvos e multi observação bem como mecanismos de leilão para efeitos de cooperação entre os UAV's.

Abstract

Unmanned Aerial Vehicles have been adopted in large scale for various military and civilians missions over the air, sea, space and the ground. Their massive use are mainly justified by the ability to avoid placing human life in harm's. Also the lack of onboard operator enables significant weight savings regarding life support equipment, resulting the lower cost and weight, increasing endurance.

The continuous increase of autonomous capabilities of aerial vehicles allows their use in a set of teams, creating a new operational paradigm. Such operation scenario is the use of small unmanned vehicles over a urban area, enabling the tracking of targets from a close range impossible to archive with manned large scale vehicles such as helicopters. The system presented is being designed to perform persistence intelligence, reconnaissance and surveillance for targets in a urban environment.

The use of low cost systems, low-altitude, short endurance UAV's can provide high resolution data to ground personal while maintaining a low operation and logistics cost. These UAV's can be deployed quickly and easily over several combination (hand, catapult small track). One of the main disadvantages is their limited sensor capabilities and endurance time. However, this can be overcome by proper sharing of relevant information among the other team of vehicles.

The objective of this thesis is to study and develop cooperation techniques among team of decentralized UAV's in order to perform tasks of persistence surveillance over a urban dense area with strong occlusions. To achieve that, several tracking and cooperation techniques were studied in order to extract some useful ideas and adopt then on the project. Tanking in account the final objective of this thesis, a simulation enviroments was developed in order to replicate the most relevant conditions in the scenario of operation, controllers and path planning algorithms to drive the vehicles, distributed tracking system with multi-target and multi-observation capabilities and a auction mechanism for the cooperation among the UAV's

Acknowledgments

The author would like to thanks to the members of The Underwater Systems and Technology Laboratory (LSTS) at FEUP by the thinking and critical environment presented on the lab and by the support of Sujit P.B. and João Sousa.

Hugo Manuel Soares Oliveira

“To copy others is necessary, but to copy oneself is pathetic.”

Pablo Picasso

Contents

1	Introduction	1
1.1	Introduction and Motivation	1
1.2	Problem Statement	2
1.2.1	Objectives	3
1.2.2	Assumptions	3
1.3	Contributions	4
1.4	Dissertation Layout	4
2	Literature	5
2.1	Introduction to Autonomous Systems	6
2.2	Path Planning	6
2.2.1	Single Vehicle Path Planning	7
2.2.2	Multi-Vehicle Path Planning	12
2.3	Target Pursuit Problem	13
2.3.1	Ground target tracking and detection	14
2.3.2	Estimation	14
2.4	Cooperative Planning	16
2.5	Cooperation Considerations	17
3	Problem Formulation	19
3.1	Introduction	19
3.2	Problem	19
3.3	Problem Decomposition	21
4	Environment	23
4.1	Introduction	23
4.2	Ground Terrain Targets	23
4.3	Obstacle Buildings	24
4.4	Constructing the Observations	25
4.4.1	Constructing the Observation cone	26
4.4.2	Obtain the targets observations	30
4.4.3	Check for occlusions	31
4.4.4	Mechanization of the observations	32
4.4.5	Obstacle Avoidance	33
4.5	Environment Final Result	34

5	Path Following and Management	37
5.1	Introduction to Path Following	37
5.2	Straight-line Path Following	37
5.3	Orbit Following	43
5.4	Path Manager	46
5.4.1	Transition Between Waypoints	46
5.4.2	Dubins Paths	51
5.4.3	Path Length Computation	51
5.5	Implementation	57
6	Tracking	61
6.1	Introduction Tracking	61
6.2	Particle filter	61
6.2.1	The Filtering Problem	62
6.2.2	Particle Filter Method	63
6.2.3	SIR - Sequential Importance Resampling	64
6.2.4	SIS - Sequential Importance Sampling	66
6.2.5	Resampling	67
6.3	Multiple Target Implemented Mechanism	68
6.4	Results Obtained	72
6.5	Single Tracking Implementation using OpenCV	73
6.5.1	Results	76
7	Cooperation	79
7.1	Introduction	79
7.2	Task Allocation	79
7.2.1	Non Faulty Agents	80
7.3	Implementation	82
8	Overview of results obtained and analysis	83
8.1	Simulated experimental plan	83
8.1.1	Simple Target Following	83
8.1.2	Explicit Cooperation	87
8.1.3	Impact of Obstacle Density on Simple Target Following	88
8.1.4	Impact of Speed Ratio on Simple Target Following	89
8.1.5	One Target Following by several UAVs	89
8.1.6	Multiple Targets Following by several UAVs	91
9	Other application areas	93
10	Conclusions and Further Work	97
10.1	Main Contributions	97
10.2	Further Work	98
10.3	Final Remark	98
	References	99

List of Figures

1.1	Operational Concept in Targets Tracking.	2
2.1	Global Hawk (NASA) Storm monitoring over Azores.	5
2.2	Components of rule-based expert systems.	6
2.3	Graph Node and associated cost arcs.	7
2.4	Voronoi Diagram computed over obstacles.	8
2.5	Probabilistic road-map representation.	8
2.6	Path mutation mechanisms.	10
2.7	Overview of Evolutionary Search. From [1]	10
2.8	Expansion diagram from RRT. Image from [2].	11
2.9	Tree diagram from RRT. Image from [2].	11
2.10	Shape formation with virtual center point.	13
2.11	Tracking Boat From Air. REP12.	14
2.12	Proposed architecture of a sensor data fusion system. Data from each sensor should first be converted to a common internal representation before the actual fusion of data is performed [3].	16
3.1	2D projection geometry for an perpendicular zero plane.	20
3.2	UAV to UAV communication range modeling	21
4.1	Several robots traveling over obstacles on the field.	24
4.2	3D world from a 2D gray image.	25
4.3	Projected cone view of the terrain.	26
4.4	Intersected polygon of the cone with the normal plane.	29
4.5	2D Gaussian distribution over a plane $[-1, 1]$ with 20 points definition.	30
4.6	Representation of the Crossing Number Algorithm.	31
4.7	Example of the target inside the footprint but blocked by buildings.	32
4.8	Mechanization of observation.	32
4.9	LIDAR observation model	33
4.10	Two different scenarios.	34
4.11	Example the complete interaction with buildings obstacles	35
4.12	Laser range finder observations	35
5.1	Configuration of the vehicle	38
5.2	Desired altitude calculation for Straight-line path	39
5.3	Vector field for straight-line path following.	42
5.4	Orbital path with center (c_n, c_e)	44
5.5	Ball region around \mathbf{w}_i for the transition point.	46
5.6	Geometry associated with the insertion of the fillet.	48

5.7	Definition of the half-planes between waypoints.	48
5.8	Smooth curve produced with fillets insertion.	49
5.9	Example of one path scenario: R-S-R.	52
5.10	Angular distance between angles ϑ_1 and ϑ_2 for (CW) and (CCW).	52
5.11	Dubins Case R-S-R.	53
5.12	Dubins Case R-S-L.	54
5.13	Dubins Case L-S-R.	55
5.14	Dubins Case L-S-L.	56
5.15	(a) The UAV follow a straight line path (b) The UAV follow a or orbit of radius r	57
5.16	Straight line path and error components	58
5.17	Orbit path and error components	59
5.18	Optimal Dubins Path's with A-Star discovery.	59
5.19	Dubins Path's with A-Star discovery with minor turn rate maneuver.	60
5.20	Dubins Path's with A-Star discovery.	60
6.1	Particle Filter Fluxogram.	68
6.2	Implemented Particle Filter Mechanism	69
6.3	Tracking two target with no crossing paths.	72
6.4	Tracking two target with crossing paths (a) and (b)	72
6.5	Tracking robustness to noise.	73
6.6	The use of sampling to estimate the a posteriori conditional probability (represented by the blue bars).	75
6.7	The object representation and measurement process for active contours. Copyright M. Isard, Oxford University.	75
6.8	The full predict-correct cycle of the condensation algorithm.	76
6.9	Engage of the object by user and correspondent tracking.	77
6.10	Tracking robustness to occlusion.	77
7.1	The decision process of agent A_i which is the auctioneer and the decision process of agent A_j who is a bidder	81
7.2	Capture from <i>MATLAB</i> command window showing the results of the auction assignment of UAV's to targets.	82
8.1	UAV Tracking one target with random controls on the prediction stage.	84
8.2	UAV trajectory (RED), Target true position (BLUE) and Target estimation (GREEN).	84
8.3	UAV Tracking one target with dynamic controls on the prediction stage.	85
8.4	UAV Tracking one target over the buildings obstacles.	86
8.5	Long run test. UAV trajectory (GREEN), Target true position (BLUE) and Target estimation (RED).	86
8.6	Master-Slave configuration tracking a high valuable target. UAV one (GREEN), UAV two (MAGENTA), Target True (BLUE), Target Estimation (PINK).	88
8.7	Impact of increasing the obstacle density on the observation time.	88
8.8	Impact of speed ratio with fixed obstacle density.	89
8.9	Effort of increasing the number of UAV's for a single target with explicit cooperation disabled.	90
8.10	Effort of increasing the number of UAV's for a single target with explicit cooperation enabled.	90
8.11	Effect of the increase of target for fixed UAV number vs increase of the UAV number for fixed target number.	91

9.1	Forest fire spreading example. Image from [4].	93
9.2	ScanEagle’s UAV platform in sea operation.	93
9.3	Image mosaicing of terrain view. Image from [5].	94
9.4	Nasa World Wind examples.	94
9.5	Nasa World Wind Framework.	95

List of Tables

8.1 List of experiments to be conduct.	83
--	----

Abbreviations and Symbols

AGCS	Autonomous Guidance Control System
ASV	Autonomous surface vehicles
AUV	Autonomous Underwater Vehicle
CASE	Computer-Aided Software Engineering
CORBA	Common Object Request Broker Architecture
LiDar	Light Detection And Ranging
MAV	Multi Aerial Vehicle
PDF	Probability Distribution Function
RRT	Rapidly exploring random tree
SUAV	Small Unnamed Aerial Vehicle
UAV	Unnamed Aerial Vehicle
UNCOL	Universal Compiler-Oriented Language
WWW	<i>World Wide Web</i>

Chapter 1

Introduction

This chapter introduces the problem of target tracking using multiple UAV's in a mixed-initiative environments. Further, the motivation in selecting this problem for the dissertation, problem statement and the objectives are described.

1.1 Introduction and Motivation

UAVs (Unmanned Aerial Vehicles) nowadays are of importance for the society. The main advantage of these vehicles is the absence of human operator onboard, minimizing risks and the ability to perform repetitive tasks in remote and unreachable places with reduced operational cost. Recent sensory and computational technologies have accelerated the research and develop of complex unmanned vehicular systems have shown their benefits in both military and civilian scenarios. The primary requirements of autonomy to carry out these missions are the capabilities of detecting internal and external changes, and reacting to them with reduced human intervention in an efficient manner. Furthermore, the UAV operational arena may have obstacles, threats and restricted regions that need to be handled automatically. The main challenge for the UAV operations is being able to carry out assigned task without compromising its integrity or the mission by flying into obstacles or restricted areas or becoming exposed to higher levels of threat risks. This problem can be minimized with the development of robust guidance strategies and their implementation in autonomous guidance and control systems (AGCS) to steer the unmanned vehicles. Several kinds of missions such as target tracking, terrain mapping, surveillance can be performed autonomously by UAV's in adversarial environments. If the autonomous guidance strategies can be formulated in a scalable manner and easy to work framework such as *MATLAB*, many operational scenarios and missions can be easily built enabling the continuous study and development of several AGCS approaches for UAV's operations in these scenarios. In addition to the current degree of autonomy of a single vehicle, coordination and cooperation [6, 7, 8] among several UAV's [9] enables the full utilization of UAV systems. The addition of individual autonomous UAV in a team of UAV's that operate under a coordination strategy defined by user will result in larger capabilities and an increase in the performance and robustness of the mission. In specific missions, such as

tracking of mobile ground targets, the use of several UAV's in close cooperation will benefit the mission, namely when the tracking and pursuit is to be held in a strong adversarial environment. Figure 1.1 shows a dense obstacle scenario where multiple UAV's are guided to perform ground tracking missions while maintain vehicle integrity. For the success of the missions, the cooperative AGCS must lead to feasible decisions regarding the UAV's motions in order to continuously track the ground targets. For example, during a pursuit scenario, the AGCS generates guidance commands for the UAV's to ensure continuous tracking of the target while avoiding obstacles and

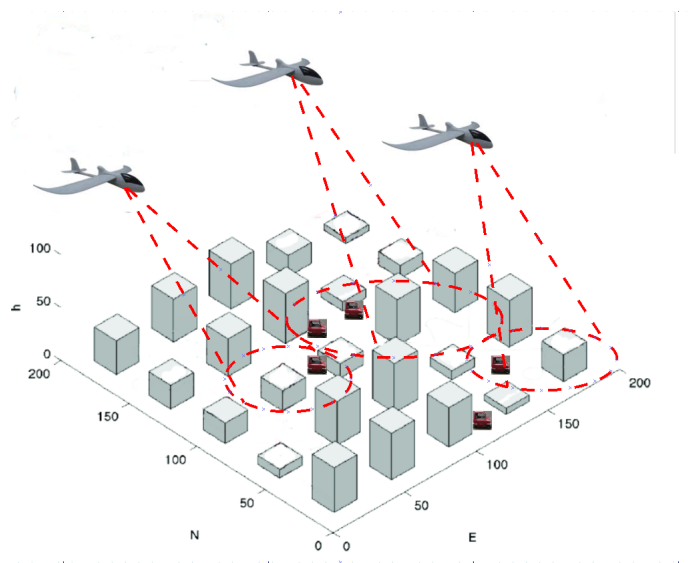


Figure 1.1: Operational Concept in Targets Tracking.

should take into account the constraints regarding dynamics of the vehicles and communications problems between them. In many other mission scenarios, multiple UAV's operate in conjunction with human-operators. Such operations are defined as Mixed-Initiative.

1.2 Problem Statement

In this thesis, we investigate the problem of cooperative target tracking in urban regions. The emphasis is on:

- (i) Development of a framework where non-convex obstacles in the form of buildings can be considered.
- (ii) Development of single UAV target tracking algorithm in the presence of obstacles.
- (iii) Development of multiple UAV target tracking algorithms.

1.2.1 Objectives

Taking in account the vast diversity of applications and the potential of utilizing autonomous and cooperative UAV's, the effort of this thesis is aimed at the development of autonomous and cooperative tracking strategies for UAV's to perform surveillance missions like target tracking, while avoiding unknown obstacles using only the sensed information. For the target tracking, the target motion must be estimated accurately from the sensed information [10]. Due to the presence of obstacle, the target may not be visible which needs to be estimated. The UAV is subject to limited communication and sensing ranges. With these constraints, the UAV must be able to persistently track the target/s.

1.2.1.1 Formulation of the objectives and the strategy

The objectives and requirements of the target tracking strategy are:

- (i) Minimize the target invisibility time during tracking.
- (ii) Obstacle avoidance.
- (iii) Maximization of the number of targets tracked.
- (iv) Avoid risk aware regions, especially in military scenarios.
- (v) Consider UAV kinematics, limited sensor and communication constraints.
- (vi) Estimate the targets states (position, speed, heading).

1.2.1.2 Formulation of cooperation

The cooperation among multiple UAV's to improve the performance of the global system namely,

- (i) Maximize coverage area to increase the visibility of the targets.
- (ii) Increase the flexibility of the UAV's to adopt better trajectories taking in account the obstacles found or potential UAV collisions that might occur while minimizing the target invisibility time.
- (iii) Use of the target information between UAV's to increase the robustness of the target estimated state.

1.2.2 Assumptions

We assume the following for target tracking strategies

- Targets are ground vehicles that are operated in 2D.
- Target motion is random. It can move in any direction subject to the kinematic constraint.

The ground targets have collision avoidance among themselves and avoid colliding with urban structures.

- The initial position of the targets and their total number are not known *a priori*.
- Each tracking vehicle has a 360-degree sensor with limited footprint.
- Each UAV is defined initially with an altitude above the tallest buildings distribution, simplifying the obstacle avoidance behavior and maximizing the field of coverage.
- Each vehicle has its own estimator to track the targets and predict their motion and to fuse data provided by other vehicles or a centralized estimator with for the same purpose.
- Each vehicle is equipped with its own AGCS in order to execute the commanded signals that guide the vehicle to the desired waypoint or trajectory segments.

1.3 Contributions

The following list summarizes the list contributions of this work.

- Multiple Tracking of objects of interest.
- Path Following and Planning with limited sensing and communication constraints .
- Cooperation mechanism among a team of UAV's.

1.4 Dissertation Layout

Chapter 2 will provide an extensive literature survey of related work. Chapter 3 presents the problem statement. Chapter 4, 5, 6, 7 provides the formulation of the environment, path following, tracking and cooperation topics. Chapter 8 provides an overview of results obtained and analysis. Chapter 9 presents the other application areas of the developed target tracking algorithms. Finally, in Chapter 10, the thesis ends with concluding remarks and discussion of related future work.

Chapter 2

Literature

UAV's has been adopted in large scale for the use of various military and civilians missions over the air, sea, space and on the ground. Their massive use are mainly justified by the ability to avoid placing human life in harm. The continuous increase of autonomous capabilities of aerial vehicles allows their use in a set of teams, creating a new paradigm of operation. There are many different kinds of missions for several environments where single or multiple UAV's could be deployed. Civil applications are also benefiting from the use of UAV's today, namely in tasks of search and rescue over hazardous areas, urban surveillance, traffic monitoring, weather and natural phenomenon observation (Figure 2.1), and severe catastrophe mitigation's such as Vulcan's, floods, sites with high levels if radiation, for example Fukushima [11] nuclear plant post-investigations.

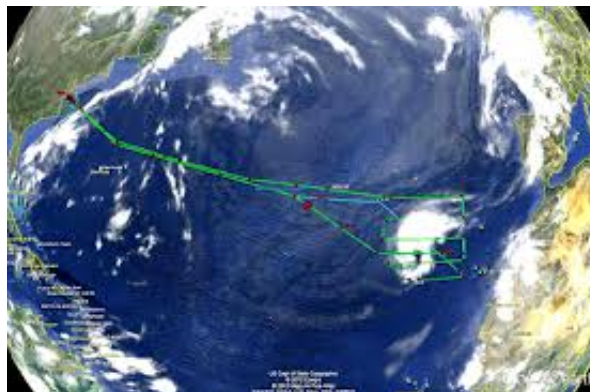


Figure 2.1: Global Hawk (NASA) Storm monitoring over Azores.

One of the common requirements for an effective and successful use of UAV's in this scenarios is the degree of autonomy they present and the capabilities to operate in a hazardous environment while maintaining close range to the object of interest/study.

2.1 Introduction to Autonomous Systems

The primary requirement to achieve autonomy is to react optimally to external environmental changes according to a predefined directive. The design of AGCS to steer the vehicle in a autonomous way towards the objective relies essentially on the use of onboard sensory, navigation and computational capabilities with adequate characteristics according to the profile of the mission and intelligent decision algorithms. With this stand point, autonomous decision making can be developed in order to guide the vehicle towards the assigned task. Several approaches to these algorithms can be considered, namely auction-based [12], rule-based [13], fuzzy logic rules [14], and using nonlinear hybrid dynamical systems for path planning [15].

All these techniques share common concepts and ideas, but are implemented in different manners according to the specificity of the problem to handle. Rule-based are the most popular among the scientific community due to the explanation capabilities enabling the system to review its reasoning taken in the inference mechanism and explain the decisions taken [16], and at same time organize them in an efficient manner with the encapsulation of the procedural knowledge in form of rules. Figure 2.2 presents the main block diagram of a rule-based system.

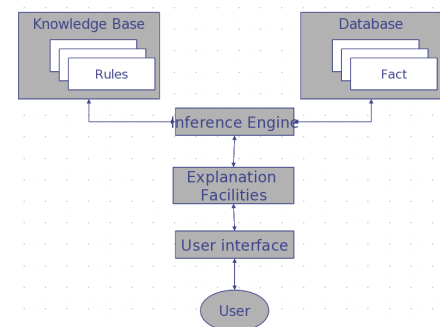


Figure 2.2: Components of rule-based expert systems.

Areas of data mining have benefit substantially from the use of these techniques, namely by increasing the handling, process large amounts of information, extracting relevant information/decisions and sub-dividing a problem to small parts [17]. Autonomous systems also benefit of this technique, namely in the design of intelligence guidance and control systems. Three main mechanics define the rule-based systems, the rule-based itself, a backward chaining of inference and the knowledge-base compiler for optimization.

2.2 Path Planning

Path planning problems have been actively studied by the robotics community. The problem of planning a path in these applications is to find a collision-free path in a environment with static or dynamic obstacles. Many works focus on holonomic and non-holonomic kinematic motion problems with static obstacles. Despite many external differences, most of these methods were based on a few different general approaches: road-map, cell decomposition, and potential field [18].

When moving obstacles are involved in planning problems, the time dimension is added to the configuration space or state-space of the robot, and planning is termed motion planning or trajectory planning instead of path planning. Research has recently been performed in motion planning that takes into account dynamic constraints. All of the mentioned path or motion planning methods focus on obstacle avoidance issues.

2.2.1 Single Vehicle Path Planning

Finding a path between two points (start, final or intermediate) presents several problems [19]. The most advanced and common problem also is the (one-to-many) path planning regarding a single UAV to multiple targets. Most authors determine the best sequence of paths to be executed that allows all targets being visited in a minimal period of time while maximizing UAV's safety. In these two situations, we can consider obstacles or non-fly areas as adversaries. Several techniques used in UAV's are derived from robot path planning using kinematic constraints only. Many robotic path planning techniques can be used for the path planning problems.

2.2.1.1 Graph-based Approaches

In graph-based approaches the partial or complete knowledge of the environment is represented as a graph and the complexity of the graph is strictly related with the obstacles present in the environment. The path graph is represented by nodes that define positions in the environment and the route as arcs. On Figure 2.3, nodes linked them to each other by arcs with some associated cost forming the path.

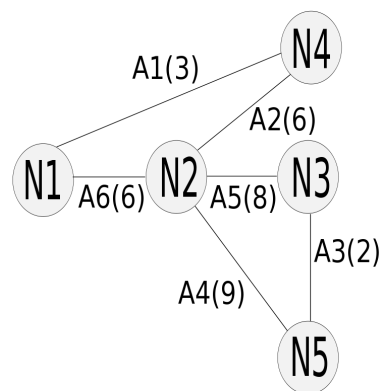


Figure 2.3: Graph Node and associated cost arcs.

The main objective is to find an minimal cost path between two points. Several methods can be applied to represent the environment such as Voronoi diagrams [20], as show in Figure 2.4, Voronoi graphs [21], Delaunay triangulation [22], visibility graphs [18], and topographical mapping [23].

To navigate over the obstacles, these methods must take into account the size of the vehicle. Most Commonly the nodes are considered as points, but in the real-world they have finite space. These space must be considered while deriving the path [24].

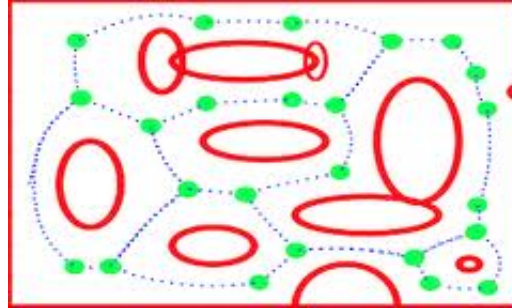


Figure 2.4: Voronoi Diagram computed over obstacles.

After the representation of the environment as graph, search algorithms can be applied for finding the shortest possible path. Those used can be one among many ones available such as Dijkstra [25], Bellman-Ford, A*, D* [26] algorithms used to find the shortest path with different trade offs regarding computational cost and time.

2.2.1.2 Probabilistic Approaches

The voronoi-based methods have high computational cost and are not viable for large maps. On the other hand probabilistic approaches can be used for large maps for example probabilistic road-map [27] or PRM. The PRM works in two stages. The first stage is the planning phase where N random sample points are generated in the free space and each point is connected to the nearest neighbors over a straight line that does not crosses any obstacles. The arcs connecting the nodes are the paths between the two nodes. An example of PRM is shown in Figure 2.5. The path connecting the nodes is determined through a local path planner.

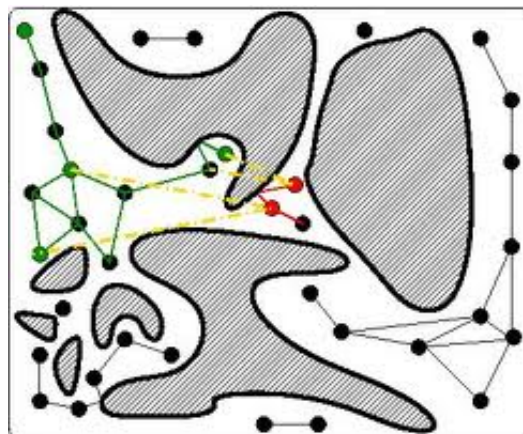


Figure 2.5: Probabilistic road-map representation.

The edge connecting the two nodes has an associated cost which is the Euclidean distance between the nodes. In the second stage, which is called as query phase, a path is found from the start point to the goal. Most of the studies made in path planning deal with full knowledge of the environment and their efficiency is strongly compromised if the queries require to pass in narrow corridors. This phenomenon is also known as narrow passage problem [28] and can be minimized with the use of more powerful local planners. The PRM can be used in unknown environments as suggested by Lee [29] and Lazoni [30]. For unknown environments, initially few nodes are generated over the environment map and paths are created based on lowest cost and those that are within the sensor range of the current node. This enables the erasing of redundant nodes and reduces the number of paths. Old literature refers other probabilistic approaches for robot planning, such as the use of probabilistic cell decomposition [31], building a collision probability between robot and the obstacles [32] on the workspace, which are divided into a finite number of cubes with the probability that each cube may become a dead end. Most of the probabilistic approaches developed for path planning are based on the probabilistic map of the area [33, 19, 34]. Since PRM works on sampling, the computational complexity is low and can deal with the uncertainties regarding the information gathered by sensors about the obstacles and targets during travel.

2.2.1.3 Evolution-based Approaches

Evolution-based approaches are normally classified as optimization techniques that use some kind of stochastic search for optimization. This search generates a random set of feasible paths that are encoded by the use of a predefined sequence of splines, Figure 2.6, and those ones can be primarily defined to include the kinematic constraints present on the vehicles, namely maximum turn rate and speed.

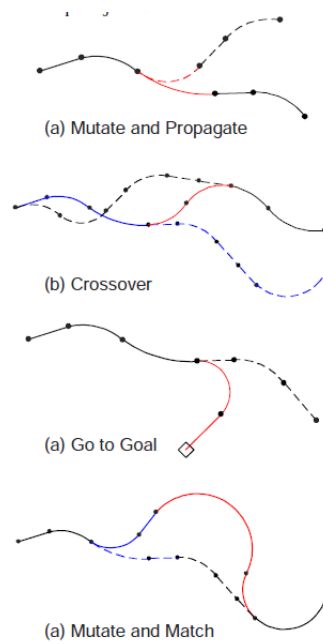


Figure 2.6: Path mutation mechanisms.

The solutions obtained from this search are evaluated using a fitness function, Figure 2.7 and the one that has the best value is the candidate solution. The new solution is then used to generate new candidate solutions with random modifications, and the process continues until a stop criterion is reached.

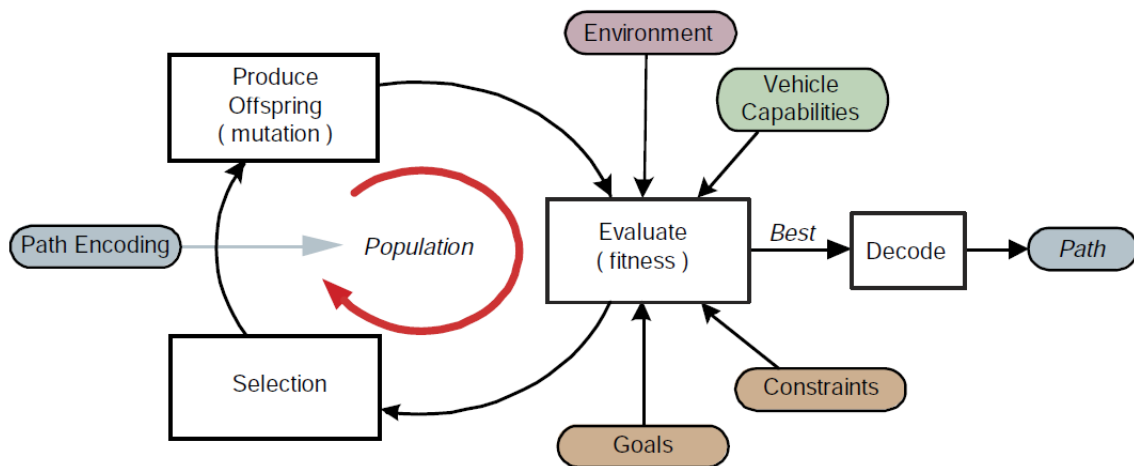


Figure 2.7: Overview of Evolutionary Search. From [1]

Their efficiency to solve combination optimization problems and robustness to uncertainties makes them more reliable than methods that use direct search approaches like graphs search. Problems with the higher number of constraints can be easily implemented and be adapted to special

characteristics of the problem [35]. Several evolutionary path planner are analyzed in [36]. Path planning problems in UAV's and AUV's have been successful solved with the use of offline/online evolutionary approaches [37, 38] and [39]. Near optimal solutions are possible to reach even with dynamical environments [40]. The major drawback that avoids reach optimal solution is the premature convergence to a local minimal. To solve this problem, several evolutionary algorithms can be used in parallel, and the the solutions produced compete with each other [38], simultaneously providing even more solutions and redundancy for the planning stage.

2.2.1.4 RRT- Rapidly Exploring Random Trees

Another method for planning paths through a obstacle field from the starting point to destination is the Rapidly Exploring Random Trees (RRT) method [2]. The RRT scheme is a random exploration algorithm that uniformly, but randomly, explores the search space. It has the advantage of extending for vehicles with complex dynamics. The implementation of the RRT is based on a structured data denominated as *tree*. A tree is a special case of a directed graph. Edges are directed from a child node to its parent and every node has exactly one parent, except the root, Figure 2.9 and Figure 2.8.

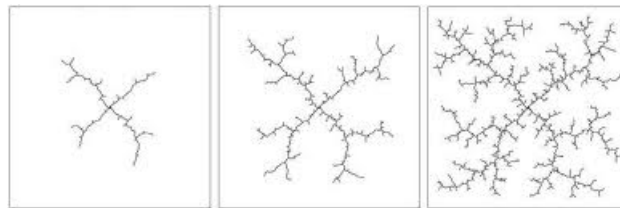


Figure 2.8: Expansion diagram from RRT. Image from [2].

The cost associated with each edge, C_{ij} , is the cost associated with traversing the feasible path between states represented by the nodes. With this a tree that uniformly explores the search space is built using randomly sampling from a uniform probability distribution that results in the shortest feasible path.

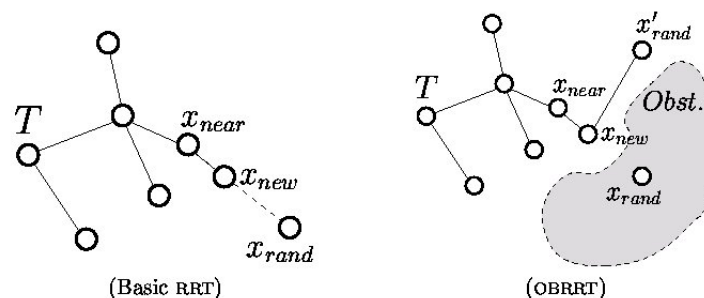


Figure 2.9: Tree diagram from RRT. Image from [2].

Versions of the RRT can also be applied in 3D terrain scenarios [24]. Some recent works have been done to make this approach more dynamic [41, 42]. This method is very attractive due to the overall simplicity of the mechanism and easy adaptation to other variants and easily accommodates kinematics constraints, namely in turning maneuvers.

2.2.2 Multi-Vehicle Path Planning

The use of multi-vehicles in a mission enhances the capabilities of the systems, but at the same time rises new problems regarding path planning requirements for all vehicles. When used in a common mission, path planning should consider the presence of other vehicles into account. In addition to this, targets must be assigned efficiently to maximize the number of tracked targets and minimize the target invisibility time [43], or track the relevant ones while still minimizing the total path length for covering the whole area of operation and improve efficiency. Multi-vehicle path planning can be either centralized or decentralized.

2.2.2.1 Centralized Approaches

Centralized approaches are directly related to the system architecture in which the entire system is managed by one of the agents in the team or by a single ground command and control center. This approach can be easily applicable in a team of UAV's that operate in close range to the center controller. One hierarchical design methodology for centralized path planning of multi-vehicle systems is grouping sequence of available team sources and planning the assignment of feasible routes sequences at higher layers of the architecture [44]. The lower layer is responsible for the generation of feasible trajectories for each vehicle to follow in real-time. Another hierarchical design methodology for centralized approach is to generate the inputs for the team leader where the position and input of the leader is sent to followers in order to maintain the desired formation [45], Figure 2.10. Furthermore, the central controller must account for the probability of loss of UAV during the mission, which should be reflected in the operations of the other vehicles [46]. One example of this can be reconfiguration of the team formation shapes among all vehicles.

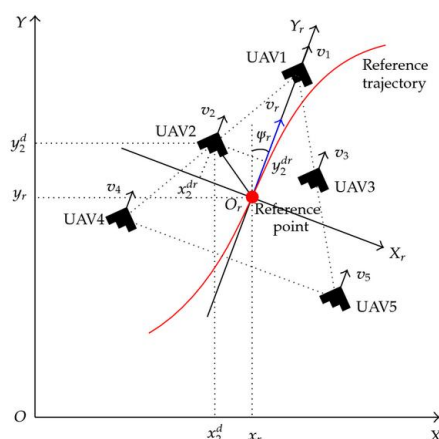


Figure 2.10: Shape formation with virtual center point.

2.2.2.2 Decentralized Approaches

When a team of large number of vehicles is used, several problems raise with the use of centralized approach. These problems are sluggish response to dynamic conditions, intractable solutions for an large team, communications constraints and single point failure [47]. The use of decentralized approaches provide a significant simplification regarding team-level planning while decreasing substantially the volume of information to be exchanged among the agents on unreliable communication infrastructure such as wireless. Decentralization process is frequently seen as the division of complex problems into sub-problems witch can be solved using simple mechanisms. Decentralized approach is extensively used for the path planning problem of multiple UAV's, where each aircraft plans its trajectory using a receding horizon strategy based on mixed integer linear programming [48]. This provides a solution to the conflicts between vehicles in a sequential, decentralized fashion by allowing each aircraft to take into account the latest trajectory and some knowledge of the current dynamics. Decentralized approaches are also applicable for the missions where a team of UAV's operates under timing constraints [49]. In such applications, rather than using UAV state trajectories for all other UAV's on the team centrally, only the critical timing information is determined and used on the coordination of the UAV's.

2.3 Target Pursuit Problem

In many situations the mobile characteristics of the targets to pursuit or track [50, 51] in a mission brings additional constraints to the path planning problem, namely the need to consider the state of the target in the pursuit to maintain a continuous visibility as shown in Figure 2.11.

The main objective in autonomous tracking of a target is to enable the UAV to track the target persistently in a given time and predicting the future states of the target while mapping the area of operation taking into account the dynamic constraints and avoiding obstacles. One of the major military efforts is to use multiple UAV's to track enemy's or escort a friendly convoy [51, 52, 53]



Figure 2.11: Tracking Boat From Air. REP12.

while avoiding potential risk areas such as SAM's (Surface-to-Air-missiles) sites. As a wild life protection effort, animals can be tracked while avoiding any relevant perturbation. In such applications, there are several common features such as (i) mobile objects to be pursued (ii) measurement and estimation of the positions of the moving objects (iii) obstacles to be avoided.

2.3.1 Ground target tracking and detection

In tracking, one of the most challenging problems is to persistently track a moving ground targets by UAV. In the literature, we can find several strategies to accomplish the tracking by changing of heading angle and varying the speed of the vehicle [54, 53, 52]. These strategies assume a pre-defined sinusoidal motion, adding some level of limitation to the strategies to be used. Further, the UAV's are frequently assumed to operate in a friendly environment where threat exposure or restricted regions are not a concern.

Another powerful approach and most common is the use of multiple hypothesis (MHT) [55] algorithms to accomplish the tracking of moving ground targets [51]. These algorithms provide bias estimation, road modeling, stopped target motion and stationary target tracking [56]. Regarding the detection of the ground vehicle, this is also a state-of-art technology, namely the problem of continuous vehicle detection from several camera perspectives. The main challenge in these algorithms is also the data association. This is by far the most challenging problem in continuous tracking algorithm. Nowadays techniques solves the problem of data association with the use Hungarian Algorithm [57], Cluster-based [58] or GNN Methods [59].

2.3.2 Estimation

As referred before in 2.3.1, ground target tracking is much more difficult to accomplish than the aerial target tracking due to the strong variance in terrains patterns and color space, resulting in erroneous interpretation of the measurements gathered. But one tough problem is the occlusion of targets by ground obstacles. Further, ground target tracking presents unique challenges such as high obstacle density and maneuverability, that result in clutter measures and low visibility [56].

In this situations, a robustness estimation and data association are the key players to making use of the data produced by each UAV and fusing all the available information and increase the accuracy of the estimation.

2.3.2.1 Multi-Sensor

In most situations, it is rarely the case that a single sensor can provide sufficient information for the reasoning component in the autonomous system. The use of different kinds of sensors has the objective to gather more information to obtain a more accurate information about the world. When techniques of multi-sensor are applied, two scenarios need to be considered. The first one all sensors operate synchronously. This means that the sampling times occurs all at same instant. In this case no major problems are raised. The second scenario is that sensors operate asynchronously and have different data rates and several communication constraints such as delays. This raises the problem of out-of-sequence information. One example is the radar infrastructure that has several antennas deployed over a large area and a centralized station running the tracking algorithms using the asynchronous information provided by those stations. The tracking mechanisms must adopt special considerations regarding the use of the information from sensors namely in the update stage in order to avoid incorporate erroneous information in that stage.

2.3.2.2 Asynchronous Measurements

Asynchronous measurements are the most usual and common scenario regarding the use of multi-sensors techniques. To handle the asynchronously measurements, most applications relies on the use of particle filters techniques. This technique enables a easy incorporation of out-of-sequence measurements while still very robust to cluttered ones. The main difference regarding common techniques like Kalman Filter based is that doesn't require a batch processing to the initialization of the tracks, being very robustness regarding intermittent observations and initial unknown of the tracks states while handling some non-Gaussian distributions [60] manifested by the targets. By particle filtering, the posterior distribution of interest is represented by a set of weighted particles. The correct choose the weights and with the adequate amount of particles enables the expectations of the posterior distribution to be closer to real distribution that targets presents. Basically, particle filtering is a method of updating these particles as the time progresses and weighting them correctly [61]. Mallick in [62], Nebot [63], Durrant [3] and Lee [64] provide contributions in decentralized estimation using multi-asynchronous sensors. The use of the particle filter have been proved to be powerful tools for image tracking [65, 66, 67]. The strength of these methods lies in their simplicity, flexibility, and systematic treatment of non-linearity and non-Gaussian. The extension of the use of particle filters to multi-target tracking has been for long time work in progress by the researchers over this field. Among others, Hue [68] developed a system for multi-target tracking by expanding the state dimension to include component information, assigned by a Gibbs sampler [69]. Another recent technique that handles asynchronous measurements without performing the sequential processing is the sensor fusion, Figure 2.12. The

asynchronous observations from all sensors are incorporated on the update by the calculation of the conditional probability.

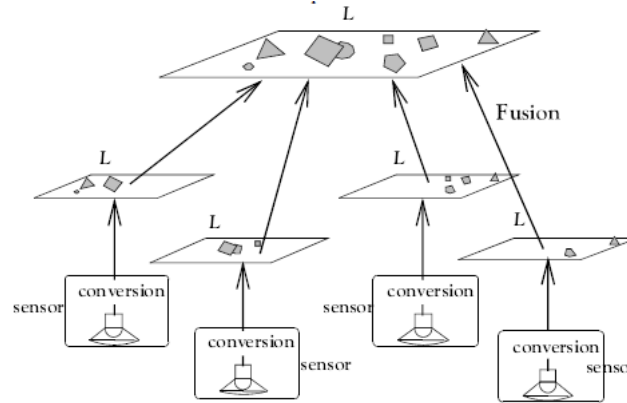


Figure 2.12: Proposed architecture of a sensor data fusion system. Data from each sensor should first be converted to a common internal representation before the actual fusion of data is performed [3].

Essential there are three types of sensor data fusion:

(i) *Complementary fusion*: fusion of several disparate sensors which only give partial information of the environment, e.g. fusion of several range sensors or camera sensors pointed in different directions. Obviously, this type of fusion resolves incompleteness of sensor data.

(ii) *Competitive fusion*: fusion of uncertain sensor data from several sources, e.g. a camera and a range sensor pointed at the same obstacle; through sensor data fusion the distance to the object can be obtained more accurately. This type of fusion is predominantly aimed at reducing the effect of uncertain and erroneous measurements. If a sensor is uncertain about the angle to a certain obstacle it can give a rough estimate of this parameter. This estimate is then refined with other estimates of the same parameter, which is a typical example of competitive fusion.

(iii) *Cooperative fusion*: fusion of different sensors of which one sensor relies on the observations of another sensor to make its own observations. This type of fusion is only mentioned here for completeness. It diminishes uncertainty, measurements errors, as well as incompleteness, but at another level. Durant-White [70] and associates are among the community one of the most relevant persons regarding the techniques of sensor fusion.

2.4 Cooperative Planning

Cooperation among multiple UAV's is a key feature that enables the potential of the use of these vehicles to increase the success of the mission. Cooperation among multi-UAV's can be defined as the coordinated set of actions that optimize the performance of the overhaul mission.

With the increasing level of autonomy, several cooperative planning applications are being deployed such as flying formations [71, 72] for distributed payload transportation [73], cooperative path planning [74, 75], rendezvous [49], target assignment [76] and cooperative target tracking [77, 53, 78]. To accomplish cooperative planning the most common approaches are based in the optimization of a suitably chosen cost function for the mission [75, 79]. In summary the optimization of the defined cost function will reduce the problem of choosing the path segment and velocity among the other UAV's paths in order to minimize the total cost for the whole team. The calculation of these paths to achieve a optimal solution, must be taken in a finite time in order to produce a fast and responsiveness solution to cope with rapidly changes on the environment that might occur during the mission. The cost function can accommodate several parameters of interest such as fuel cost [75, 80], number of UAV's in action over a target and the spread of the targets to intersect [76].

2.5 Cooperation Considerations

The coordination among multiple agents is highly dependable on communications to make cooperative decisions. However, these ones can be subject to partiality or total communications failures and form group clusters. The design of coordination algorithms that take communications limitations into account is hard to accomplish due to the fact in the majority of cases the algorithms need constant information to formulate the decisions. Several mechanism have been developed to address the communications problems and still achieve a high degree of cooperation. Pollini at [81] developed robust mechanisms for routing the information under UAV's communication failures on flight formation, by putting the refereed agent to the back of formations while diminish the impact that may have on other fully functional agents. Estimation of the locations of the agents in a leader-follower structure was addressed by Shin [82] in order to predict the controls to be issued whenever there is a communication issue. Hierarchical control to handle several faults at different levels was proposed by Mehra [83]. Those implementations has good performance in fixed task specifications such as flight formations where a very reliable and accurate model is known, enabling the controller to handle communications failures. In alternative, consensus based task allocation scheme for multiple UAV's under intermittent communications conditions have been proposed by Dionne and Rabbath [84].

Chapter 3

Problem Formulation

In this chapter, we will present the problem formulation, the UAV and ground vehicle kinematic constraints and the performance metrics.

3.1 Introduction

We assume that each UAV is denoted as $A_i, i = 1, \dots, n$ and each target is denoted as $T_j, j = 1, \dots, m$. We assume that the environment vehicles have constraints (kinematics constraints, obstacles collision, limited sensor range, limited communications capabilities and endurance and no prior known of the obstacles, targets position and their number.

3.2 Problem

Given a set of vehicles $A = \{A_1, A_2, \dots, A_n\}$ and a unknown number of m targets $T = \{T_1, T_2, \dots, T_m\}$, maximize the time tracked per target $Total_{time}$ by assigning the trajectories sequences S_i and commands C_i for each vehicle.

Maximize visibility time for the $T_j, = 1, \dots, m$ number of targets

$$Total_{time} = \max T_j(S_i, C_i) \quad (3.1)$$

while minimize exposure to treats from targets $E_{T_j, j = 1, \dots, m}$

$$\sigma_{UAV(i)exposure} = \min E_{T_j}(S_i, C_i) \quad (3.2)$$

taking in consideration the UAV kinematics

$$\begin{aligned}
 \dot{x}_i &= V_i \cos(\theta_i) \\
 \dot{y}_i &= V_i \sin(\theta_i) \\
 \dot{\theta}_i &= \tan \frac{\phi_i}{V_i} \\
 \dot{\phi}_i &= K * (\theta_{fi} - \theta_{vi})
 \end{aligned} \tag{3.3}$$

and constraints, namely tuning rate $-\omega_{max} \leq u_{\phi i} \leq \omega_{max}$, maximum change speed $-V_{max} \leq u_{vi} \leq V_{max}$. Equation 3.3 with the associated heading rate constraint is the Dubin's car model [85] with roll effect incorporated. Subject to these constraints, the UAV's must perform a search task over the field using the cooperative model while maximizing $Total_{time} = \max T_j(S_i, C_u)$ and minimizing the threat exposure $\sigma_{UAV(i)_{exposure}}$ considering the sensory constraints identified, namely the camera resolution and footprint (aperture angle) that results in a maximum area of coverage C_{area} and maximum height for detection C_{height} . The area projection is given by

$$C_{area} = 2\pi * \tan(\theta_{app}) * C_{height} \tag{3.4}$$

represented in 2D and perpendicular to the zero plane,(Figure 3.1).

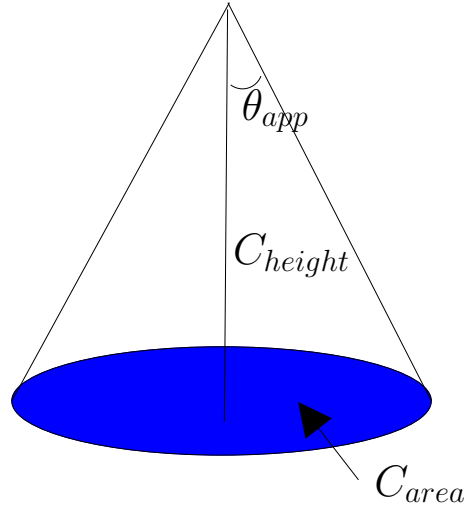


Figure 3.1: 2D projection geometry for an perpendicular zero plane.

For the communication constraints, it must be considered the probability of failure over time using a 2D Gaussian Distribution $F(x,y)$ associated with the distance between UAV that posses valuable information to communicate with those that are the potential receivers. The 2D PDF is

given by

$$F(x,y) = \frac{1}{2\pi\sigma_x\sigma_y} e^{-\frac{(UAV_x(i)-UAV_x)^2}{2\sigma_x^2} + \frac{(UAV_y(i)-UAV_y)^2}{2\sigma_y^2}} \quad (3.5)$$

being σ_x, σ_y the standard deviation of equal values to represent the omni-directional antenna propagation (Figure 3.2). The probability of success is formed between the intersection of the 2D Gaussian Function and current positions of the vehicles.

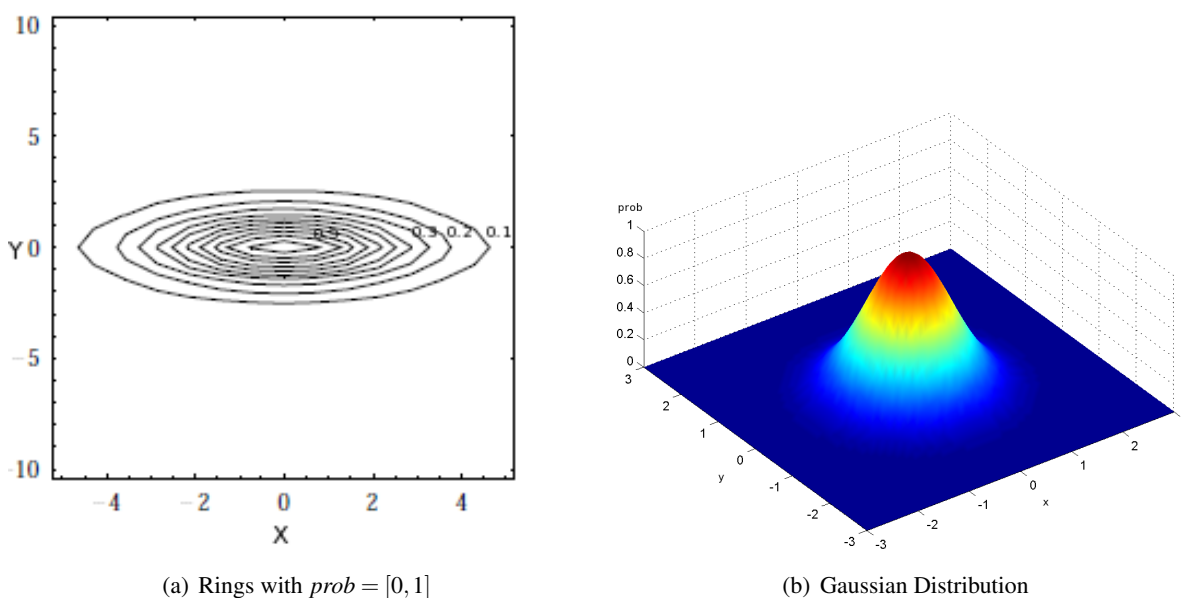


Figure 3.2: UAV to UAV communication range modeling

3.3 Problem Decomposition

In order to improve the development and address the problem in a proper manner, the system was divided in four major blocks. The four blocks can be identified (i) the Environment, responsible by the representation of the scenario of operation, targets movement and UAV sensory capabilities and assessment of the mission, (ii) Path Follow and Navigation, responsible by the vehicles path controller and waypoint generation as well, (iii) Tracking, responsible by the estimation of the target and management of targets position and velocities and (iv) Cooperation mechanism, responsible for the assignment and task scheduling of each vehicle towards the goal of the mission.

Chapter 4

Environment

In the simulation environment, there are several scenarios that need to be modeled and represented including the mobile objects traveling over the area. This chapter presents details of the modeling on the environment.

4.1 Introduction

The modeling of the real world simulated scenarios enables us to develop and analyze sophisticated control mechanisms without the need to carry on expensive real missions at the initial stages. The aerial vehicles and ground targets have a continuous models, with constraints such as limited sensor range capabilities, obstacles or enemies exposure, limited autonomy, limited fuel capacity and vehicle faults during mission. The simulated scenario enables these situations to be analyzed in detail and develop strategies to obtain the best solution and at the same time extract performance values for detailed analysis and further developments.

4.2 Ground Terrain Targets

Regarding the model of the ground targets traveling in the area, the objective is to make their movement completely random. This means that the targets have some kinematic restrictions that limits their degree of freedom. The initial model uses a simple differential robot, modeled by the following equations

$$\begin{bmatrix} x \\ y \\ \theta \end{bmatrix} = \begin{bmatrix} \frac{\Delta_{sr} + \Delta_{sl}}{2} \cos\left(\theta + \frac{\Delta_{sr} - \Delta_{sl}}{2b}\right) \\ \frac{\Delta_{sr} + \Delta_{sl}}{2} \sin\left(\theta + \frac{\Delta_{sr} - \Delta_{sl}}{b}\right) \\ \frac{\Delta_{sr} - \Delta_{sl}}{2b} \end{bmatrix} \quad (4.1)$$

where x, y, θ are the state variables, Δ_{sr}, Δ_{sl} is the traveled distance by each wheel and b is the axis length.

For motion and obstacle avoidance, the target/s posses two sensors that look forward for obstacles, Figure 4.1 on the path, according the sensor detection.

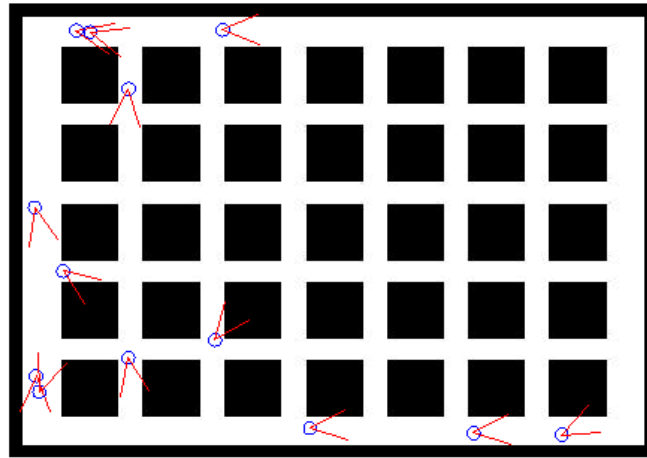


Figure 4.1: Several robots traveling over obstacles on the field.

On travel, robot shifts right, left to avoid obstacles or rotates over itself if blocked. On clear path, after some iterations the target starts to slide to one of the sides to avoid going in strait line continuously.

4.3 Obstacle Buildings

Using a black and white 2D image, any kind of pattern of city can be modeled. We use a random distribution bounded by user parameters, to modulate the height of the buildings over those black spots. The UAV must maintain a minimum altitude above the maximum height of the tallest building or to steer away from them if the collision avoidance is present. Given an obstacle pattern image in 2D, the building are constructed by the extrusion in 3D up to a maximum high value in a random fashion as show in Figure 4.2. This building intersected with the UAV field of view will result in the final observation of the ground terrain in order to search/pursuit targets.

The steps to develop the buildings is defined by the algorithm 4.1;

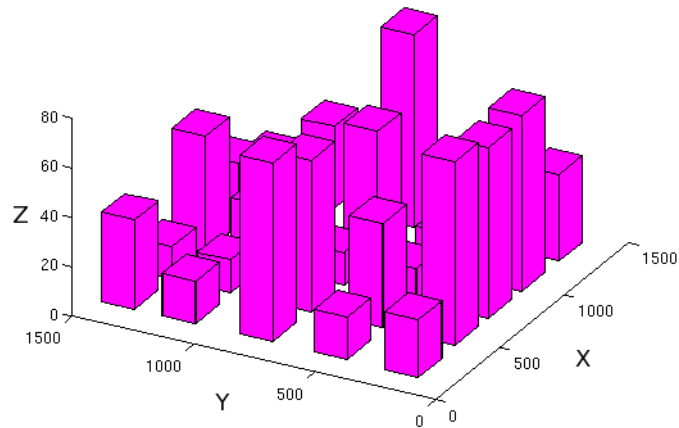


Figure 4.2: 3D world from a 2D gray image.

Algorithm 4.1 Extrude Buildings: $Obstacles = \text{extrudeBuildings}(RGBImage)$

Input: RGB image
 GRAY = convertImageToGray(RGB)
 threshold = determineThreshold(GRAY)
 BW = convertImageToBlackWhite(GRAY, threshold)
 BW = complementImage(BW)
 B,L = detectBoundaries(BW)
 STATS = determineRegionProperties(L)
for i=1 to length (STATS) **do**
 $C(i).data = STATS(i).ConvexHull$
end for
for j=1 to length (C) **do**
 $vert = \text{extractVertices}(C(i).data)$
 for j=1 to length (vert-1) **do**
 ComputeTopFace
 ComputeLateralallFace
 SaveinObstacles(j).face
 end for
end for
return Obstacles at final run

4.4 Constructing the Observations

Taking the presence of building shadow and occlusions to the projected footprint into account, the UAV camera does not see the ground target that are inside of the shadow of the building

regarding the cone of observation, enabling a more real world scenario regarding the tracking of terrain targets.

4.4.1 Constructing the Observation cone

In first hand is need to construct a observation cone such as shown in Figure 4.3.

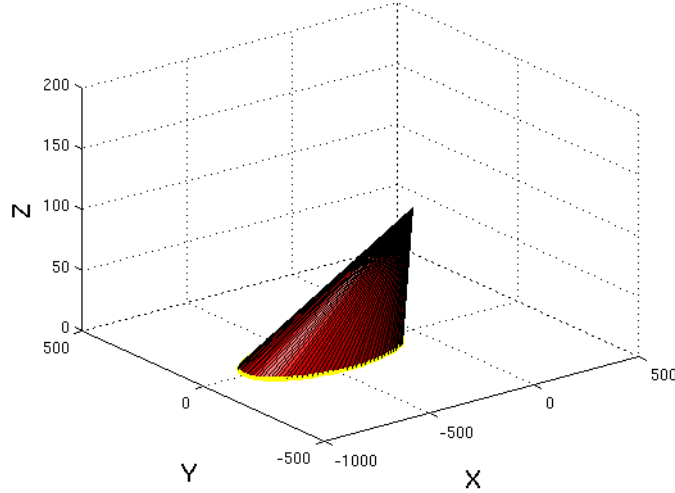


Figure 4.3: Projected cone view of the terrain.

The inputs parameters are defined by the cone origin $ConeOrg$ that is the UAV current position $UAV pos_{x,y,z}$, the desired cone direction regarding the UAV heading $ConeDir$ defined in section 4.2 and the cone aperture $ConeAngle$,

$$ConeDIR = R_x * ConeDir, \quad (4.2)$$

where R_x is the rotation matrix regarding the current UAV bearing. With the definition of the origin point and two directional unit vectors for the ground plane $Plane(origin, dir1, dir2)$ the computation of the Cone-Plane intersection is represented by a *Gaussian* PDF.

A point X on a cone is defined by the equation (in matrix form) $(X - coneOrg)^T * M * (X - coneOrg) = 0$ and matrix M is defined as

$$M = coneDir * coneDir^T - \cos(coneAngle)^2 * \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad (4.3)$$

and a $Conevertex - planeOrigin$ directional vector defined by

$$conePlaneDir = planeOrg - coneOrg. \quad (4.4)$$

A point X on a plane is defined by equation

$$X = \text{planeOrg} + x_1 * \text{planeDir1} + x_2 * \text{planeDir2}. \quad (4.5)$$

Combining the plane equation with the cone equation results in a quadratic equation

$$c_1 * x_1^2 + 2 * c_2 * x_1 * x_2 + c_3 * x_2^2 + 2 * c_4 * x_1 + 2 * c_5 * x_2 + c_6 = 0. \quad (4.6)$$

This equation can be represented as a matricial equation $X^T * C * X = 0$, with C the conic matrix. The elements of the conic matrix are arranged

$$\begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \end{bmatrix} = \begin{bmatrix} \text{planeDir1}^T * M * \text{planeDir1} \\ \text{planeDir1}^T * M * \text{planeDir2} \\ \text{planeDir2}^T * M * \text{planeDir2} \\ \text{conePlaneDir}^T * M * \text{planeDir1} \\ \text{conePlaneDir}^T * M * \text{planeDir2} \\ \text{conePlaneDir}^T * M * \text{conePlaneDir} \end{bmatrix}, \quad (4.7)$$

composing in the following conic matrix

$$C = \begin{bmatrix} c_1 & c_2 & c_4 \\ c_2 & c_3 & c_5 \\ c_4 & c_5 & c_6 \end{bmatrix}, \quad (4.8)$$

that can be decompose into different parts of the conic matrix in

$$\begin{bmatrix} C_R \\ C_T \\ C_{\text{delta}} \end{bmatrix}, \quad (4.9)$$

while taking the intersection of a cone into account and a plane can be formed either using a ellipse, parabola or hyperbola. A canonical form of the conic is determined by transforming the original conic through a rotation R and a translation t , i.e. by applying a homogeneous transformation H . To find R , diagonalize C_R by computing its eigenvalues, i.e

$$C_R = R * \lambda * R^T. \quad (4.10)$$

The translation T centering the ellipse to the canonical form is determined by

$$T = -R * \text{inv}(\lambda) * R^T * C_T. \quad (4.11)$$

With rotation R and translation T , the homogeneous transformation matrix H can be defined as

$$H = \begin{bmatrix} R & T & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad (4.12)$$

begin the canonical conic matrix C_c defined by

$$C_c = H^T * C * H. \quad (4.13)$$

C_c results in a diagonal matrix with elements (C_{c1}, C_{c2}, C_{c3}) , which defines the canonical quadratic equation

$$C_{c1} * x_{c1}^2 + C_{c2} * x_{c2}^2 + C_{c3} = 0. \quad (4.14)$$

This quadratic equation can be re-written as a ellipse equation

$$x_{c1}^2/a^2 + x_{c2}^2/b^2 = 1, \quad (4.15)$$

with the parameters

$$\begin{aligned} a &= \sqrt{-C_{c(3,3)}/C_{c(1,1)}}, \\ b &= \sqrt{-C_{c(3,3)}/C_{c(2,2)}}. \end{aligned} \quad (4.16)$$

Similarly, the ellipse is represented as a diagonal covariance matrix σ_c

$$\sigma_c = \begin{bmatrix} a^2 & 0 \\ 0 & b^2 \end{bmatrix}. \quad (4.17)$$

Finally, the 2D Gaussian distribution with median Mu and variance σ is computed using the translation vector T and the canonical covariance matrix σ_c transformed through rotation R .

$$\begin{aligned} Mu &= T, \\ \sigma &= R * \sigma_c * R^T. \end{aligned} \quad (4.18)$$

With this Gaussian distribution, the derivation of the intersection points with the plane can be obtained by defining a number of segments on the conical shape N_{seg} and the revolution angle in one semi-plane. The segments are equally spaced from $[-\pi, \pi]$, being each intersection point with

the plane computed using

$$\begin{aligned} X_i(x,y) &= \left[\cos(seg_i), \sin(seg_i) \right] * real(\sqrt{Sigma}) + Mu^T, \\ interSeg(i) &= planeOrg + X_{ix} * planeDir1 + X_{iy} * planeDir2. \end{aligned} \quad (4.19)$$

This results in a closed polygon over the intersected plane represented in Figure 4.4.

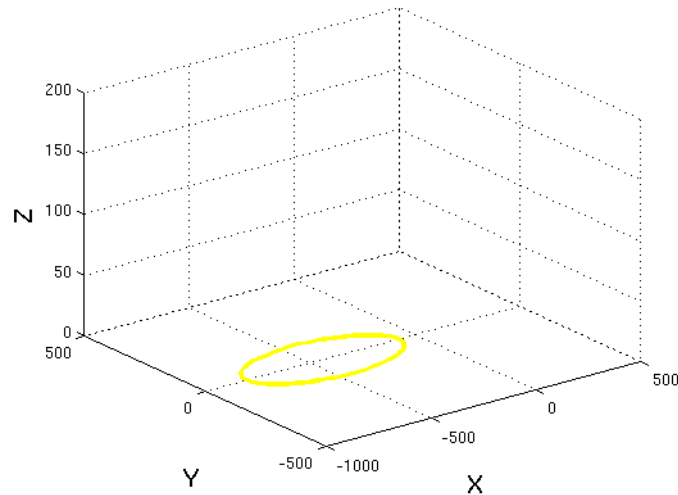


Figure 4.4: Intersected polygon of the cone with the normal plane.

Using the 2D Gaussian distribution Mu and σ values, the linear distribution of the points on plane x, y is possible to define a Gaussian distribution that models the distortion of the lens camera that fits inside the intersected area. Figure 4.5 presents the 2D Gaussian over a linear space of $[-1, 1]$ in both axes.

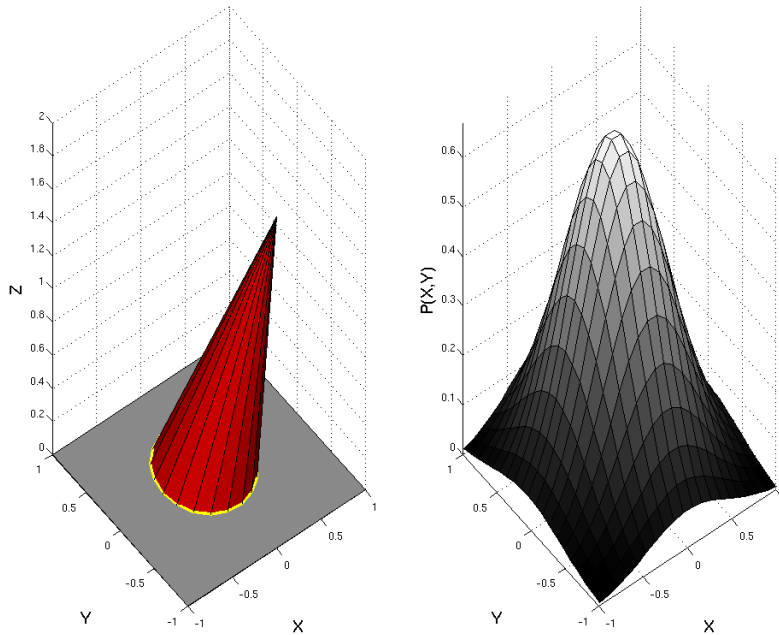


Figure 4.5: 2D Gaussian distribution over a plane $[-1, 1]$ with 20 points definition.

4.4.2 Obtain the targets observations

The valid observations of the targets are those that are inside the intersected polygon with the normal plane described in subsection 4.4.1. Using the calculated points, the observable targets are constructed. Checking if a point is inside a convex polygon, the most simple method to achieve this is the use of the Crossing Number Algorithm [86]. The algorithm is ray-casting to the right. In each iteration, the test point is checked against one of the polygon's edges. The first line of the if-test succeeds if the point's *y-coord* is within the edge's scope. The second line checks whether the test point is to the left of the line. If this is true, then a line is drawn rightwards from the test point crosses that edge. By repeatedly inverting the value of *c*, the algorithm thus counts the number of times that rightward line crosses the polygon. If it crosses a odd number of times, then the point is inside. If a even number, the point is outside.

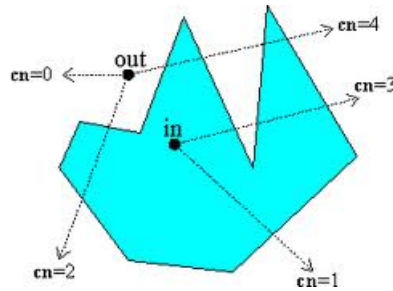


Figure 4.6: Representation of the Crossing Number Algorithm.

4.4.3 Check for occlusions

The algorithms described in subsection 4.4.2 does not take the obstacles that might cover the view of the UAV regarding the ground targets into account. To enable this using the observations and the obstacles computed earlier, the mechanism of checking if a target is in the shadow of the obstacle regarding the UAV point-of-view can be simply verifying if the formed line between the target and the UAV position intersect any face of the obstacles polygons. If true, this observation does not enter on the set of valid collected observations by the UAV sensor. Using the following algorithm 4.2, is possible to consider the occlusions made by obstacles.

Algorithm 4.2 Extrude Buildings: $Observations = \text{getObservations}(UAVPos, TargPos, Obstacles)$

Input: UAV Position, Targets Positions, Obstacles

for $i=1$ to $\text{lenght}(TargPos)$ **do**

$inHull = \text{checkInHull}(TargPos(i), ConePlaneInter)$

if $inHull = \text{True}$ **then**

$OBS(k) = TargPos(i)$

$k = k + 1$

end if

end for

for $j=1$ to $\text{lenght}(OBS)$ **do**

for $l=1$ to $\text{lenght}(Obstacles)$ **do**

$\text{computeLineTargUAV}(UAVPos, OBS(j))$

$\text{intersectLineTargObstacles}(line, Obstacles(l))$

if $\text{line intersect} == \text{True}$ **then**

Check next target

else

Add to the valid observations $ValidObs = OBS(j)$

end if

end for

end for

return $ValidObs$ at final run

The Figure 4.7 shows the intersection of the obstacle over the footprint cone path to a particular target.

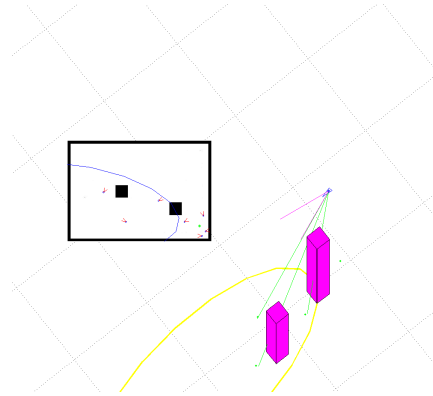


Figure 4.7: Example of the target inside the footprint but blocked by buildings.

4.4.4 Mechanization of the observations

In order to obtain the coordinates of an target on world frame, we need to determine the points P_x and P_y using the current vehicle state and the angle formed between body and ground, assuming that targets is on the zero z axis. Figure 4.8 uses equation 4.20 to perform this calculation.

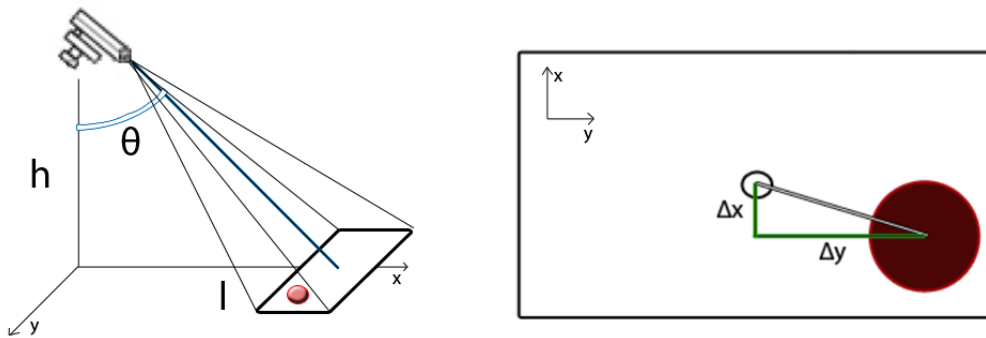


Figure 4.8: Mechanization of observation.

$$\begin{aligned}
 P_x &= \tan(\theta) \cdot h - \frac{2 \cdot r \cdot \Delta_x}{\max p}, \\
 P_y &= \frac{2 \cdot r \cdot \Delta_y}{\max p}.
 \end{aligned}
 \tag{4.20}$$

The r corresponds to the radius of the object and p to its perimeter. Δ_x and Δ_y correspond to the deployment of the object regarding base reference and h the height of the camera.

4.4.5 Obstacle Avoidance

The buildings for the UAV may present problems in term of collision while tracking a target. To detect the obstacle, the vehicle must be equipped with a laser range finder that follows the model represented in Figure 4.9. The observations gathered will allow to define a new feasible waypoint for the UAV that minimizes the distance from the current estimation of the assigned target while avoiding the obstacle. The relevant parameters of this sensor model are the maximum range, maximum angle aperture, angle step, and the standard deviation error of the returned measurements.

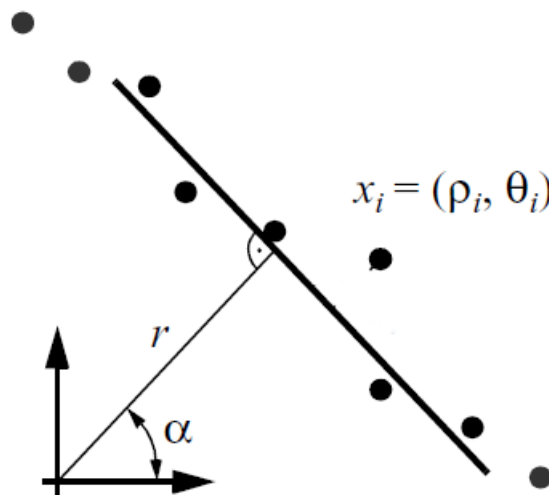


Figure 4.9: LIDAR observation model

The observations gatherer are the distance ρ_i and the angle θ_i forming the point χ_i in polar coordinates. The r, α are the straight-line parameters from the Least-Mean Square estimator.

The follow algorithm describes the steps to accomplish obstacle avoidance, if no obstacle found in the range, the feasible waypoint is the one previously supplied.

Algorithm 4.3 Obstacle Avoidance: $NewWayPoint = getCollisionAvoidance()$

Input: Obstacles, UAVState, UAVSensorParam, DesiredWayPoint

Waypoint = DesiredWayPoint

for $i=1$ to length (Obstacles) **do**

$Points = getTopFace(Obstacles(i).face)$

if $Points.z \geq$ UAV Height **then**

$[rho, theta] = GetLaserMeasurements(Points)$

if Laser Return valid observations **then**

$Computethe\ farthest\ point\ closeto\ the\ Target$

$Waypoint = newWaypoint$

end if

end if

end for

return $WayPoint$ at final run

4.5 Environment Final Result

The results are shown in Figure 4.10 and the complete interaction in Figure 4.11

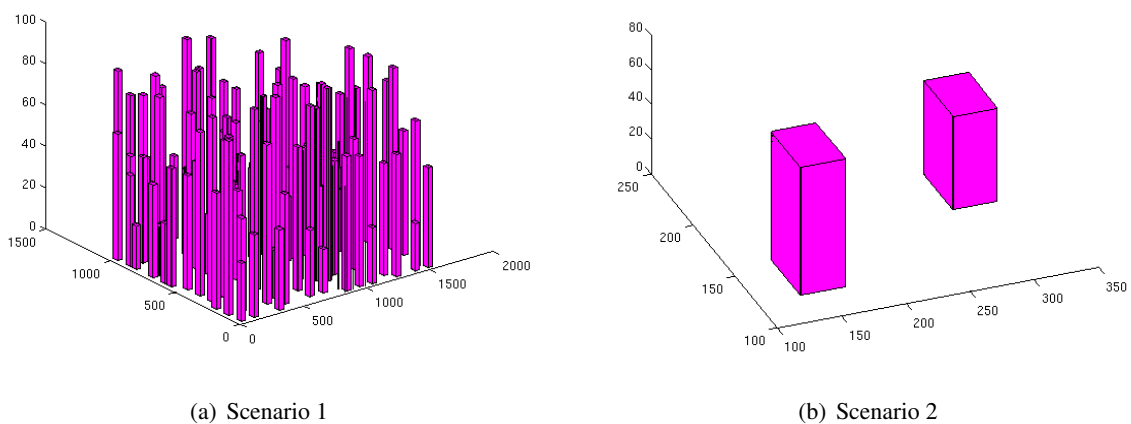


Figure 4.10: Two different scenarios.

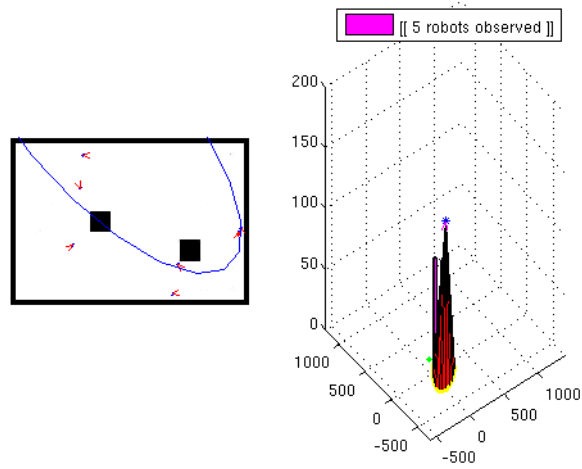
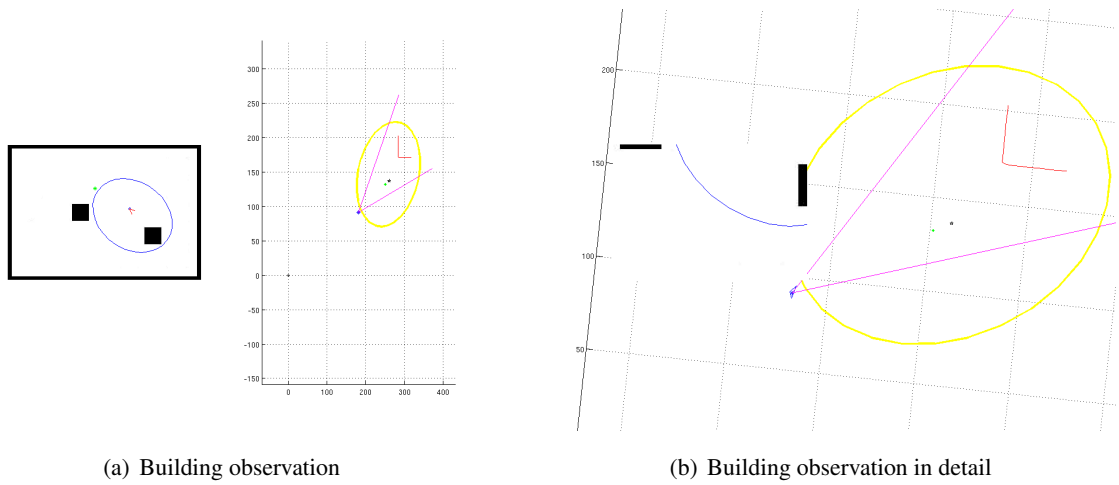


Figure 4.11: Example the complete interaction with buildings obstacles

Regarding the obstacle avoidance, Figure 4.12 shows a plot of the cloud points of the laser during course travel.



(a) Building observation

(b) Building observation in detail

Figure 4.12: Laser range finder observations

Taking in account that in small UAV's the payload is very constrained, the use of a small LiDAR can be impossible. One alternative is to use a simple laser distance and this returns valid observations, start a sinusoidal maneuver and collect the point cloud and form the obstacle observation.

Chapter 5

Path Following and Management

In this chapter, we present guidance laws for tracking straight-lines segments and constant altitude circular orbits (loiters). The combination of these supports the path management and the tracking of targets during the course of the mission.

5.1 Introduction to Path Following

In trajectory tracking the vehicle is commanded to be at a particular location at a specific time, causing the vehicle to move in a desired fashion. For a fixed-wing vehicle, the desired position is constantly moving (target following). In path following the time dependence of the problem is eliminated.

5.2 Straight-line Path Following

A straight line path is described by two vectors in \mathbb{R}^3 , namely,

$$\mathcal{P}_{line}(\mathbf{r}, \mathbf{q}) = \{\mathbf{x} \in \mathbb{R}^3 : \mathbf{x} = \mathbf{r} + \lambda \mathbf{q}, \lambda \in \mathbb{R}\} \quad (5.1)$$

where $\mathbf{r} \in \mathbb{R}^3$ is the origin of the path, and $\mathbf{q} \in \mathbb{R}^3$ is a unit vector whose direction indicates the desired direction, Figure 5.1.

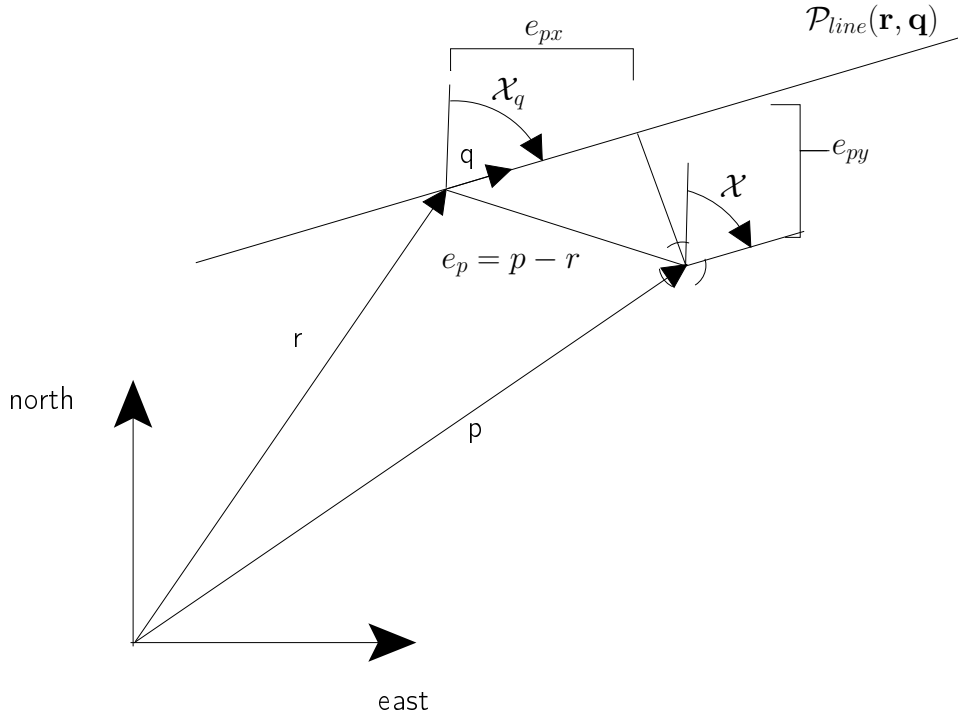


Figure 5.1: Configuration of the vehicle

The course angle of $\mathcal{P}_{line}(\mathbf{r}, \mathbf{q})$, is measured from north and its given by

$$\chi_q \triangleq \text{atan2} \frac{q_e}{q_n}, \quad (5.2)$$

where $\mathbf{q} = (q_n, q_e, q_d)^T$ express the north, east, and down components of the unit direction vector.

The path-following problem is solved in a frame relative to the straight-line path. Selecting \mathbf{r} as the center of the path frame, with the x -axis aligned with the projection of \mathbf{q} onto the local north-east plane, the z -axis aligned with the inertial z -axis, and the y -axis selected to create a right-handed coordinate system, then

$$\mathcal{R}_i^p \triangleq \begin{bmatrix} \cos \chi_q & \sin \chi_q & 0 \\ -\sin \chi_q & \cos \chi_q & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

is the transformation from the inertial frame to the path frame, and

$$\mathbf{e}_q = \begin{pmatrix} e_{px} \\ e_{py} \\ e_{pz} \end{pmatrix} \triangleq \mathcal{R}_i^p (\mathbf{p}^i - \mathbf{r}^i) \quad (5.3)$$

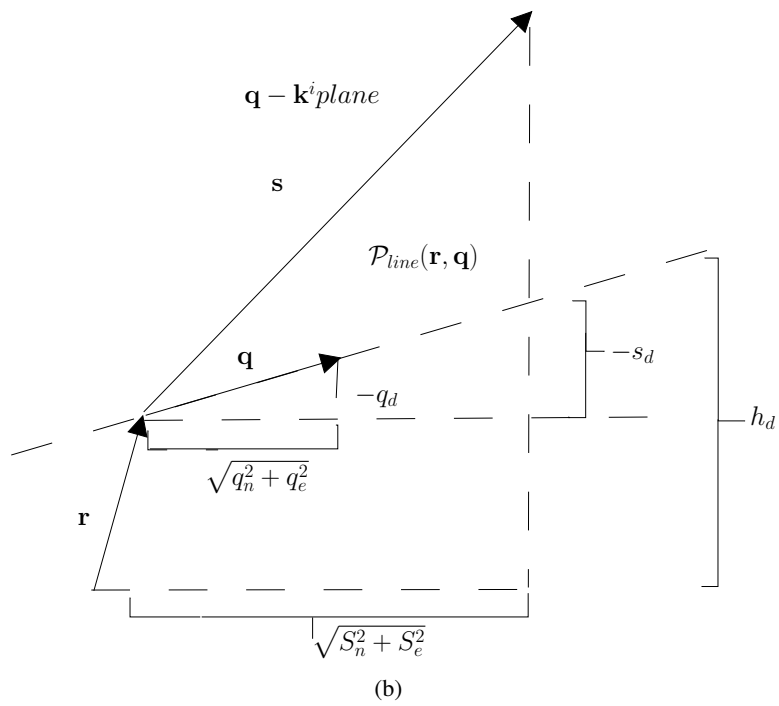
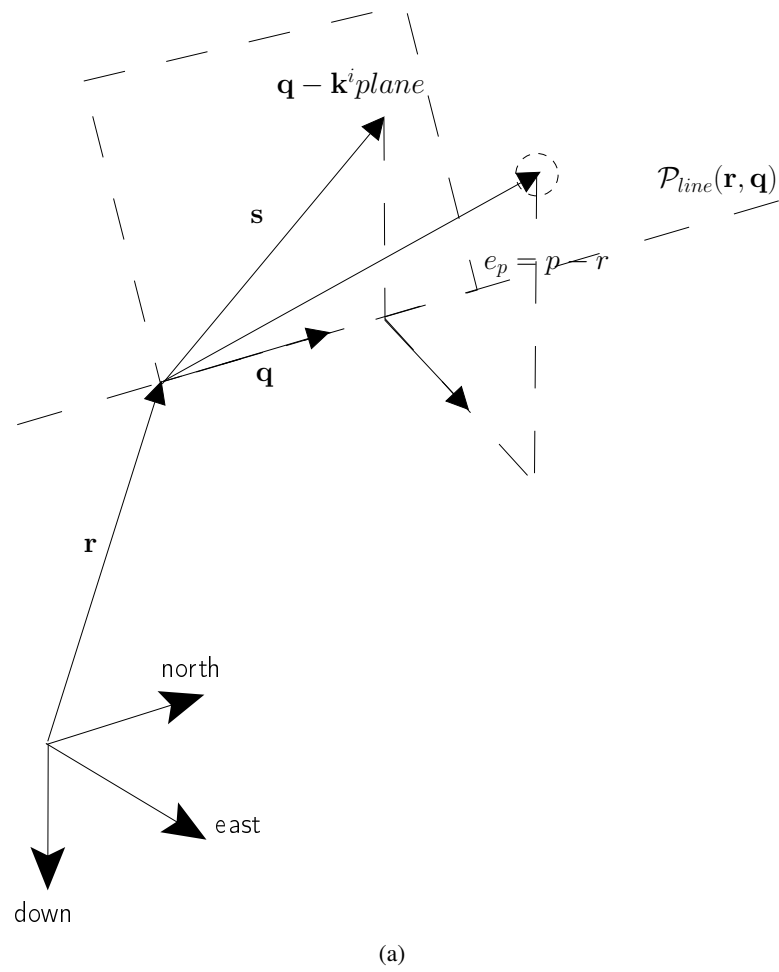


Figure 5.2: Desired altitude calculation for Straight-line path

is the relative path error expressed in the path frame. The relative error dynamics in the north-east inertial plane, expressed in the path frame considering the current vehicle course χ is given by

$$\begin{pmatrix} \dot{e}_{px} \\ \dot{e}_{py} \end{pmatrix} = \begin{pmatrix} \cos \chi_q & \sin \chi_q \\ -\sin \chi_q & \cos \chi_q \end{pmatrix} \begin{pmatrix} V_g \cos \chi \\ V_g \sin \chi \end{pmatrix} = V_g \begin{pmatrix} \cos(\chi - \chi_q) \\ \sin(\chi - \chi_q) \end{pmatrix}. \quad (5.4)$$

For path-following, we need to regulate the cross-track error e_{py} to zero by commanding the course angle. The relevant dynamics are therefore given by

$$\dot{e}_{py} = \sin(\chi - \chi_q), \quad (5.5)$$

being χ the current course angle, χ^c the commanded course and χ_q the desired course.

$$\dot{\chi} = b_\chi(\dot{\chi}^c - \dot{\chi}) + b_\chi(\chi^c - \chi), \quad (5.6)$$

with b being constants regarding controller gain for angle and its first derivative. The lateral straight-line path-following problem is to select χ^c so that $e_{py} \rightarrow 0$ when χ_q is known.

The geometry for straight-line path-following in the longitudinal direction is show in Figure 5.2. The calculation of the desired altitude takes the projection of the relative path onto the vertical plane containing the path direction vector \mathbf{q} as referred in Figure 5.2 into account. The projection of \mathbf{e}_p is referred as \mathbf{s} . Referring to the vertical plane that contains the path represented in Figure 5.2 using similar triangles their relation is given by

$$\frac{-S_d}{\sqrt{S_n^2 + S_e^2}} = \frac{-q_d}{\sqrt{q_n^2 + q_e^2}}, \quad (5.7)$$

where $\sqrt{S_n^2 + S_e^2}$ is the north-east distance formed by the desired segment S_d and the reference frame. Similar, $\sqrt{q_n^2 + q_e^2}$ forms the distance using the north-east unit vectors generated from the straight line to the desired final altitude and projected into the down unit vector q_d .

The projection \mathbf{s} of the relative error vector is defined as

$$\mathbf{s}^i = \begin{pmatrix} S_n \\ S_e \\ S_d \end{pmatrix} = \mathbf{e}_p^i - (\mathbf{e}_p^i \cdot \mathbf{n})\mathbf{n} \quad (5.8)$$

where

$$\mathbf{e}_p^i = \begin{pmatrix} S_n \\ S_e \\ S_d \end{pmatrix} \triangleq \mathbf{p}^i - \mathbf{r}^i = \begin{pmatrix} p_n - r_n \\ p_e - r_e \\ p_d - r_d \end{pmatrix} \quad (5.9)$$

and the unit vector normal to the $\mathbf{q} - \mathbf{k}^i$ plane is given by

$$\mathbf{n} = \frac{\mathbf{q} \times \mathbf{k}^i}{\|\mathbf{q} \times \mathbf{k}^i\|}. \quad (5.10)$$

The desired altitude for a vehicle at \mathbf{p} following the straight-line path $\mathcal{P}_{line}(\mathbf{r}, \mathbf{q})$ is given by

$$h_d(\mathbf{r}, \mathbf{p}, \mathbf{q}) = -r_d + \sqrt{S_n^2 + S_e^2} \left(\frac{q_d}{\sqrt{q_n^2 + q_e^2}} \right). \quad (5.11)$$

The altitude dynamics are given by

$$\ddot{h} = b_i(\dot{h}^c - \dot{h}) + b_h(h^c - h), \quad (5.12)$$

being b controller gains, h^c and \dot{h}^c the commanded altitude and commanded rate of climb respectively and h the current altitude. The longitudinal straight-line path following problem is to select h^c so that $h \rightarrow h_d(\mathbf{r}, \mathbf{p}, \mathbf{q})$.

5.2.0.1 Longitudinal Guidance Strategy

Taking in account the desired altitude specified in the equation (5.11) and the dynamics modeled by the equation (5.12), a good path-following can be constructed with $h^c = h_d(\mathbf{r}, \mathbf{p}, \mathbf{q})$, a zero steady-state error in altitude for straight line paths is possible. Taking in account that the control laws in the climb and descent zones will cause the vehicle to climb or descent in the altitude-hold zone, the pitch attitude control is used to control the altitude. Assuming all proper control loop, the dynamics follow transfer function

$$\frac{h}{h^c} = \frac{b_h s + b_h}{s^2 + b_h s + b_h}. \quad (5.13)$$

Defining the altitude

$$e_h \triangleq h - h_d(\mathbf{r}, \mathbf{p}, \mathbf{q}) \quad (5.14)$$

results that

$$\begin{aligned} \frac{e_h}{h^c} &= 1 - \frac{h}{h^c} \\ &= \frac{s^2}{s^2 + b_h s + b_h}. \end{aligned} \quad (5.15)$$

Applying the theorem of final value, results in

$$\begin{aligned} e_{h,ss} &= \lim_{s \rightarrow 0} s \frac{s^2}{s^2 + b_h s + b_h} h^c \\ &= 0, \text{ for } h^c = \frac{H_q}{s}, \frac{H_q}{s^2}. \end{aligned} \quad (5.16)$$

5.2.0.2 Lateral Guidance Strategy

The lateral Guidance Strategy has to select the commanded course angle χ^c in equation (5.6) so that e_{py} in equation (5.5) is driven to zero asymptotically. A desired course angle at every point over the straight-line results in the vehicle moving towards the path. The set of desired courses angles are named as vector fields, Figure 5.3.

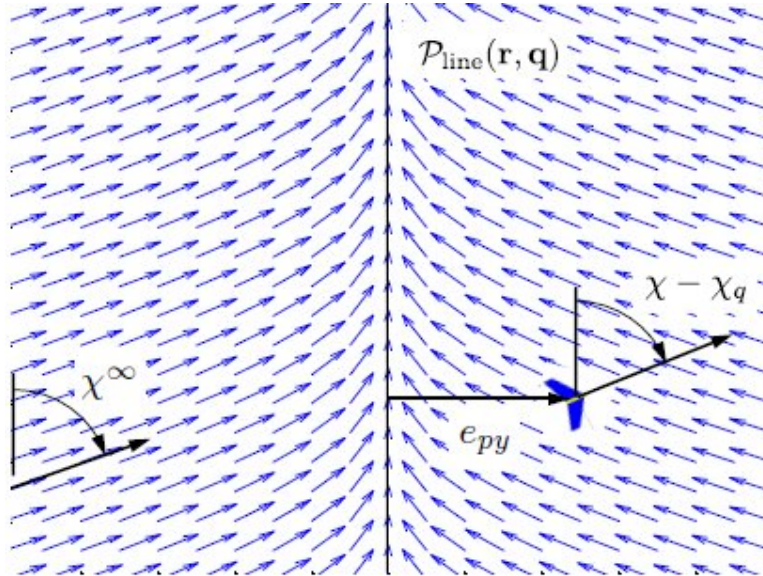


Figure 5.3: Vector field for straight-line path following.

With this the vector field is given by e_{py} . When this one is large, the vehicle is directed to the desired path with a course angle $\chi^\infty \in [0, \frac{\pi}{2}]$, and so that as e_{py} and the desired course approaches

zero. This is accomplished by the definition of desired course as

$$\chi(e_{py}) = -\chi^\infty \frac{2}{\pi} \tan^{-1}(K_{path}e_{py}) \quad (5.17)$$

where K_{path} is a positive constant that influences the rate of the transition from χ to zero. The value of K_{path} influences the abruptness or smooth transition over the desired course. Restricting χ^∞ to be contained between $\chi^\infty \in [0, \frac{\pi}{2}]$ results in

$$-\frac{\pi}{2} < \chi^\infty \frac{2}{\pi} \tan^{-1}(K_{path}e_{py}) < \frac{\pi}{2} \quad (5.18)$$

for all values of e_{py} . Since the \tan^{-1} and \sin over $[-\frac{\pi}{2}, \frac{\pi}{2}]$ are odd functions, then $e_{py} \rightarrow 0$ argued with the Lyapunov function $W(e_{py}) = \frac{1}{2}e_{py}^2$ with $\chi = \chi_q + \chi(e_{py})$ since

$$\dot{W} = -V_g e_{py} \sin\left(\chi^\infty \frac{2}{\pi} \tan^{-1}(K_{path}e_{py})\right) \quad (5.19)$$

is less than zero for $e_{py} \neq 0$. The commanded for lateral path following is then given by

$$\chi^c(t) = \chi_q - \chi^\infty \frac{2}{\pi} \tan^{-1}(K_{path}e_{py}(t)). \quad (5.20)$$

If χ_q is computed directly from equation (5.2) this can lead to a problem due to the fact that atan2 returns $\pm\pi$. To solve this problem, χ_q can be computed as

$$\chi_q = \text{atan2}(\mathbf{q}_e, \mathbf{q}_n) + 2\pi m \quad (5.21)$$

where $m \in \mathcal{N}$ is defined so that $-\pi \leq \chi_q - \chi \leq \pi$, and atan2 stays in the four-quadrant.

5.3 Orbit Following

A orbit path is described by a center $\mathbf{c} \in \mathbb{R}^3$, a radius $\rho \in \mathbb{R}$ and a direction $\lambda \in \{-1, 1\}$, as

$$\mathcal{P}_{orbit}(\mathbf{c}, \rho, \lambda) = \{\mathbf{r} \in \mathbb{R}^3 : \mathbf{r} = \mathbf{c} + \lambda\rho(\cos\varphi, \sin\varphi, 0)^T, \varphi \in [0, 2\pi]\} \quad (5.22)$$

where $\lambda = 1$ and $\lambda = -1$ signifies clockwise and counterclockwise orbit respectively. The center of the orbit is expressed in inertial coordinates so that $\mathbf{c} = (c_n, c_e, c_d)^T$, where $-c_d$ represents the desired altitude of the orbit and to maintain altitude $-h^c = -c_d$. The guidance strategy for orbit following is best derived in polar coordinates suggested in Figure 5.4.

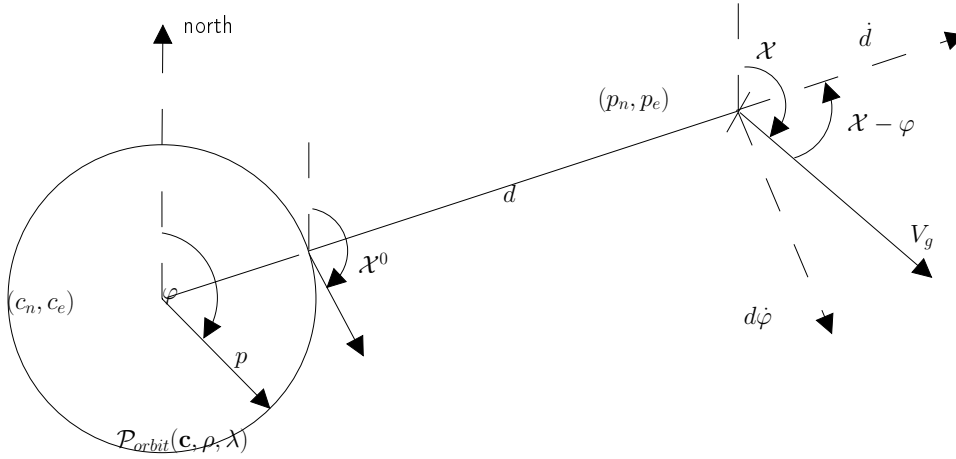


Figure 5.4: Orbital path with center (c_n, c_e) .

The constant altitude vehicle dynamics can be derived by rotating the differential equations that describes the motion in the north \dot{p}_n and east \dot{p}_e directions as

$$\begin{pmatrix} \dot{p}_n \\ \dot{p}_e \end{pmatrix} = \begin{pmatrix} V_g \cos \chi \\ V_g \sin \chi \end{pmatrix} \quad (5.23)$$

thought the phase angle φ so that equations of motion represents the vehicle motion in the normal tangential directions to the orbit as

$$\begin{pmatrix} \dot{d} \\ d\dot{\varphi} \end{pmatrix} = \begin{pmatrix} \cos \varphi & \sin \varphi \\ -\sin \varphi & \cos \varphi \end{pmatrix} \begin{pmatrix} \dot{p}_n \\ \dot{p}_e \end{pmatrix} = \begin{pmatrix} V_g \cos(\chi - \varphi) \\ V_g \sin(\chi - \varphi) \end{pmatrix} \quad (5.24)$$

Taking in account the Figure 5.4, the vehicle dynamics in polar coordinates are given by

$$\dot{d} = V_g \cos(\chi - \varphi) \quad (5.25)$$

$$\dot{\varphi} = \frac{V_g}{d} \sin(\chi - \varphi) \quad (5.26)$$

$$\dot{\chi} = -b_\chi \chi + b_\chi (\chi^c - \chi) \quad (5.27)$$

Figure 5.4, for a clockwise or counterclockwise orbit, the desired course angle is given by $\chi^0 = \varphi + \frac{\pi}{2}$ and $\chi^0 = \varphi - \frac{\pi}{2}$ respectively. Generalizing we have

$$\chi^0 = \varphi + \lambda \frac{\pi}{2} \quad (5.28)$$

When the distance between the vehicle and the orbit is large $d \gg \rho$, is desired that the vehicle flies into the center point of the orbit, resulting in a approximate desired course

$$\chi_d = \chi^0 + \lambda \frac{\pi}{2} \quad (5.29)$$

and when $d = \rho$ the desired course is $\chi_d = \chi^0$, resulting on a candidate course as

$$\chi_d(d - \rho, \lambda) = \chi^0 + \lambda \tan^{-1} \left(k_{orbit} \left(\frac{d - \rho}{\rho} \right) \right), \quad (5.30)$$

where $k_{orbit} > 0$ specifies the rate of transition from $\lambda \pi/2$ to zero. Arguing that if $\chi = \chi_d$ using the Lyapunov function $W = \frac{1}{2}(d - \rho)^2$ the tracking objective is satisfied. The differentiation W along the trajectory gives

$$\dot{W} = -V_g(d - \rho) \sin \left(\tan^{-1} \left(k_{orbit} \left(\frac{d - \rho}{\rho} \right) \right) \right) \quad (5.31)$$

being negative definite since the argument of sin is in the $[-\frac{\pi}{2}, \frac{\pi}{2}]$ for all $d > 0$, what implies that $d \rightarrow \rho$ asymptotically. The course command on orbit following results in

$$\chi^c(t) = \varphi + \lambda \left[\frac{\pi}{2} + \tan^{-1} \left(k_{orbit} \left(\frac{d - \rho}{\rho} \right) \right) \right]. \quad (5.32)$$

Similarities in the computation of path angle χ_q , if the angular position of the orbit φ is computed in between $\pm\pi$ there will be a sudden jump of 2π . To avoid, φ need to be computed as

$$\varphi = \text{atan2}(p_e - c_e, p_n - c_n) + 2\pi m \quad (5.33)$$

where $m \in \mathcal{N}$ is defined so that $-\pi \leq \varphi - \chi \leq \pi$.

5.4 Path Manager

The path manager has the objective to manage the waypoints sequence by the scaling of the straight-line and orbits/loiter maneuvers and drive the vehicle on the desired route.

5.4.1 Transition Between Waypoints

Consider the ordered sequence of waypoints

$$W = \{\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_N\} \quad (5.34)$$

in each $\mathbf{w}_i = (w_{n,i}, w_{e,i}, w_{d,i})^T \in \mathbb{R}^3$. Switching from waypoints leave from one segment to another. For that is need to know when the vehicle as reached \mathbf{w}_i .

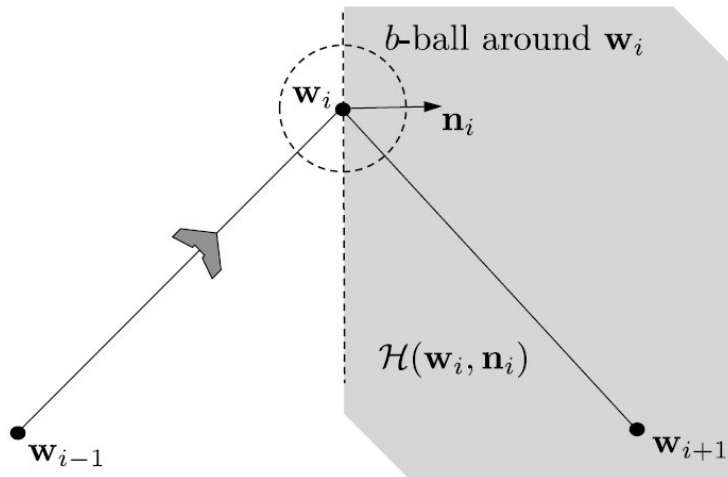


Figure 5.5: Ball region around \mathbf{w}_i for the transition point.

One strategy is to switch from one segment to another $\mathbf{w}_i \rightarrow \mathbf{w}_{i+1}$ when it reaches the circular area around the terminal point \mathbf{w}_i , as shown in Figure 5.5 and its defined by

$$\|\mathbf{p}(t) - \mathbf{w}_i\| \leq b, \quad (5.35)$$

being b the area of the circular and $\mathbf{p}(t)$ the current position of the vehicle. If the circular area is small then the vehicle never enter this area due to non-convergence problems derived from external forces or segments too short in distance and large angle between them. A refined approach that has robustness to tracking errors is the use of a half-plane switching criteria. Taking a point $\mathbf{r} \in \mathbb{R}^3$

and the nominal vector $\mathbf{n} \in \mathbb{R}^3$ define a half plane as

$$\mathcal{H}(\mathbf{r}, \mathbf{n}) \triangleq \{\mathbf{p} \in \mathbb{R}^3 : (\mathbf{p} - \mathbf{r})^T \mathbf{n} \geq 0\}. \quad (5.36)$$

Defining a unit vector pointing in the direction of the line $\overline{\mathbf{w}_i \mathbf{w}_{i+1}}$ as

$$\mathbf{q} \triangleq \frac{\mathbf{w}_{i+1} - \mathbf{w}_i}{\|\mathbf{w}_{i+1} - \mathbf{w}_i\|} \quad (5.37)$$

and the unit normal to 3-D half plane that separates the line $\overline{\mathbf{w}_{i-1} \mathbf{w}_i}$ from $\overline{\mathbf{w}_i \mathbf{w}_{i+1}}$ is given by

$$\mathbf{n}_i \triangleq \frac{\mathbf{q}_{i-1} - \mathbf{q}_i}{\|\mathbf{q}_{i-1} - \mathbf{q}_i\|}, \quad (5.38)$$

and the algorithm for following waypoints is described in 5.4.1

Algorithm 5.1 Follow Waypoints: $(\mathbf{r}, \mathbf{q}) = \text{followWpp}(W, \mathbf{p})$

Input: Waypoint path $W = \{\mathbf{w}_1, \dots, \mathbf{w}_n\}$ Vehicle position $\mathbf{p} = (p_n, p_e, p_d)^T$

Require: $N \geq 3$

if New waypoint path W is received **then**

 initialize waypoint index $i \leftarrow 2$

end if

$\mathbf{q}_{i-1} \leftarrow \frac{\mathbf{w}_i - \mathbf{w}_{i-1}}{\|\mathbf{w}_i - \mathbf{w}_{i-1}\|}$

$\mathbf{q}_i \leftarrow \frac{\mathbf{w}_{i+1} - \mathbf{w}_i}{\|\mathbf{w}_{i+1} - \mathbf{w}_i\|}$

$\mathbf{n}_i \leftarrow \frac{\mathbf{q}_{i-1} - \mathbf{q}_i}{\|\mathbf{q}_{i-1} - \mathbf{q}_i\|}$

if $\mathbf{p} \in \mathcal{H}(\mathbf{w}_i, \mathbf{n}_i)$ **then**

 increment $i \leftarrow (i + 1)$ until $i = N - 1$

end if

return $\mathbf{q}, \mathbf{q} = \mathbf{q}_{i-1}$ at each step time

and enables the vehicle to route from the prior point \mathbf{w}_{i-1} to posterior \mathbf{w}_i being i a counter that is incremented until reach the last waypoint and initialized to $i = 2$ to immediately execute the next waypoint.

The geometry near the transition is shown in Figure 5.6. Using the unit vector \mathbf{q}_i aligned with the line between waypoints \mathbf{w}_i and \mathbf{w}_{i+1} as defined in equation (5.37), the resulting angle between $\overline{\mathbf{w}_{i-1} \mathbf{w}_i}$ and $\overline{\mathbf{w}_i \mathbf{w}_{i+1}}$ is

$$\mathcal{Q} \triangleq \cos^{-1}(-\mathbf{q}_{i-1}^T \mathbf{q}_i). \quad (5.39)$$

If the radius of the fillet is R , as show in Figure 5.6, the the distance between the waypoint \mathbf{w}_i and the location where the fillet intersects the line $\overline{\mathbf{w}_i\mathbf{w}_{i+1}}$ is $R/\tan\frac{\mathcal{Q}}{2}$ and the distance between \mathbf{w}_i and the center of the circle is $R/\sin\frac{\mathcal{Q}}{2}$, resulting that the distance over \mathbf{w}_i and the edge of the circle along \mathcal{Q} is given by $R/\sin\frac{\mathcal{Q}}{2} - R$. The maneuver is taken over the straght-line $\overline{\mathbf{w}_{i-1}\mathbf{w}_i}$ until entering the half-plane \mathcal{H} represented in Figure 5.7.

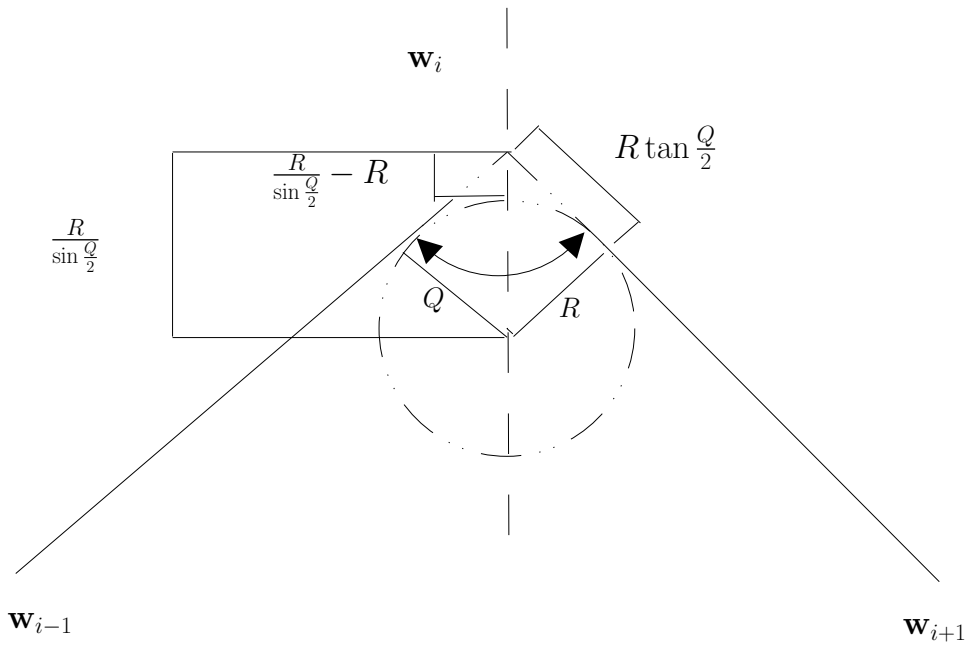


Figure 5.6: Geometry associated with the insertion of the fillet.

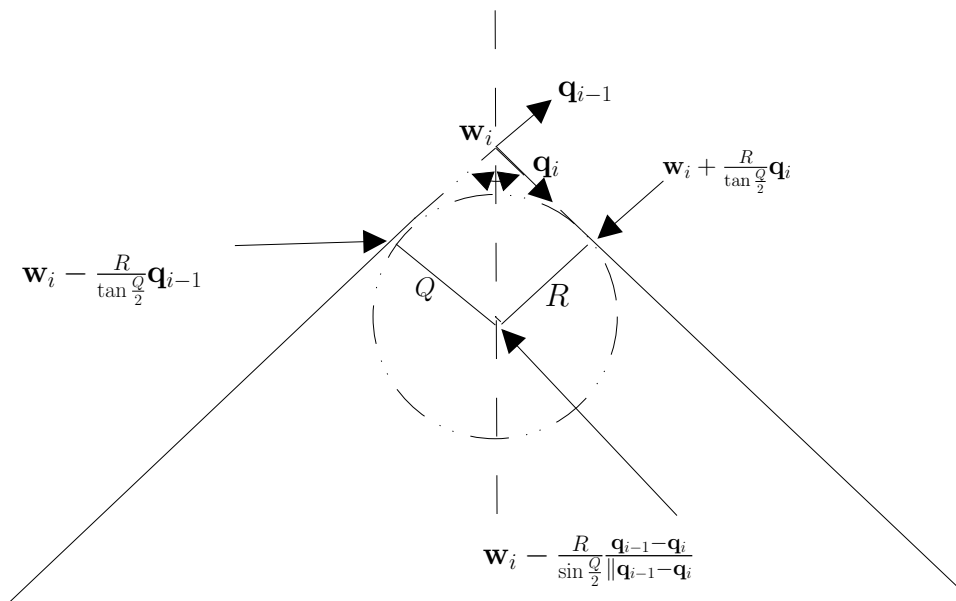


Figure 5.7: Definition of the half-planes between waypoints.

the center is given by

$$\mathbf{c} = \mathbf{w}_i - \left(\frac{R}{\sin \frac{\varrho}{2}} \right) \frac{\mathbf{q}_{i-1} - \mathbf{q}_i}{\|\mathbf{q}_{i-1} - \mathbf{q}_i\|}, \quad (5.40)$$

similarly, half-plane \mathcal{H}_1 location is

$$\mathbf{r}_1 = \mathbf{w}_i - \left(\frac{R}{\tan \frac{\varrho}{2}} \right) \mathbf{q}_{i-1}. \quad (5.41)$$

The other half-plane \mathcal{H}_2 location is respectively

$$\mathbf{r}_2 = \mathbf{w}_i - \left(\frac{R}{\tan \frac{\varrho}{2}} \right) \mathbf{q}_i. \quad (5.42)$$

With this formulation we can define two states. *State* = 1 to follow the path along $\overline{\mathbf{w}_{i-1}\mathbf{w}_i}$ with the parameters $r = \mathbf{w}_{i-1}$ and $q = \mathbf{q}_{i-1}$ and is tested to see if as reached first half-plane and need to switch to orbit mode defined by *State* = 2. Using algorithm described in 5.4.1, results in a much smooth curve inside the path, Figure 5.8, without relevant overshoot out of the area of path.

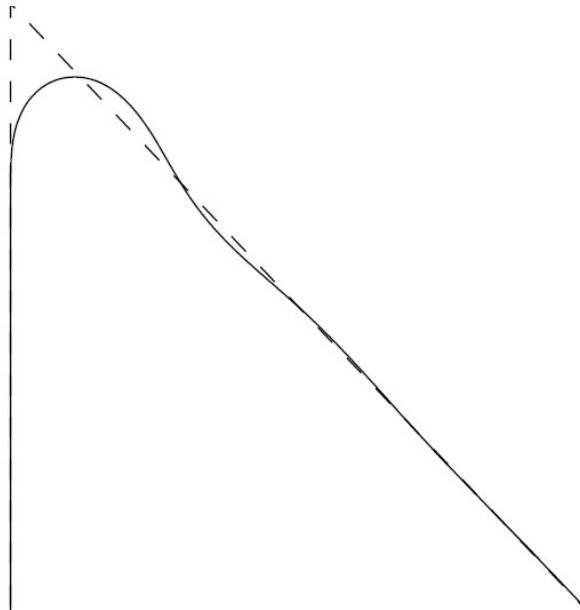


Figure 5.8: Smooth curve produced with fillets insertion.

Algorithm 5.2 Follow Waypoints With Fillets: $(flag, \mathbf{r}, \mathbf{q}, \rho, \lambda) = \text{followWppFillet}(W, \mathbf{p}, R)$

Input: Waypoint path $W = \{\mathbf{w}_1, \dots, \mathbf{w}_n\}$ Vehicle position $\mathbf{p} = (p_n, p_e, p_d)^T$, fillet radius R

Require: $N \geq 3$

if New waypoint path W is received **then**

 initialize waypoint index $i \leftarrow 2$ and state machine: state $\leftarrow 1$

end if

$\mathbf{q}_{i-1} \leftarrow \frac{\mathbf{w}_i - \mathbf{w}_{i-1}}{\|\mathbf{w}_i - \mathbf{w}_{i-1}\|}$

$\mathbf{q}_i \leftarrow \frac{\mathbf{w}_{i+1} - \mathbf{w}_i}{\|\mathbf{w}_{i+1} - \mathbf{w}_i\|}$

$Q \leftarrow \cos^{-1}(\mathbf{q}_{i-1}^T \mathbf{q}_i)$

if state = 1 **then**

 flag $\leftarrow 1$

$\mathbf{r} \leftarrow \mathbf{w}_{i-1}$

$\mathbf{q} \leftarrow \mathbf{q}_{i-1}$

$\mathbf{z} \leftarrow \mathbf{w}_i - \left(\frac{R}{\tan Q/2}\right) \mathbf{q}_{i-1}$

if $\mathbf{p} \in \mathcal{H}(\mathbf{z}_i, \mathbf{q}_{i-1})$ **then**

 state $\leftarrow 2$

end if

else

if state = 2 **then**

 flag $\leftarrow 2$

$\mathbf{c} \leftarrow \left(\frac{R}{\sin Q/2}\right) \frac{\mathbf{q}_{i-1} - \mathbf{q}_i}{\|\mathbf{q}_{i-1} - \mathbf{q}_i\|}$

$\rho \leftarrow R$

$\lambda \leftarrow \text{sign}(q_{i-1, n} \mathbf{q}_{i, e} - q_{i-1, e} \mathbf{q}_{i, n})$

$\mathbf{z} \leftarrow \mathbf{w}_i + \left(\frac{R}{\tan Q/2}\right) \mathbf{q}_i$

if $\mathbf{p} \in \mathcal{H}(\mathbf{z}_i, \mathbf{q}_i)$ **then**

 increment $i \leftarrow (i + 1)$ until $i = N - 1$

 state $\leftarrow 1$

end if

end if

end if

return $flag, \mathbf{r}, \mathbf{q}, \rho, \lambda$ at each step time

The \mathcal{W} is the path length estimate without considering the fillets

$$\mathcal{W} \triangleq \sum_{i=2}^N \|\mathbf{w}_i - \mathbf{w}_{i-1}\|, \quad (5.43)$$

while \mathcal{W}_F is the path length estimate considering the fillets resulting in the final length of

$$\mathcal{W}_F = \|\mathcal{W}\| + \sum_{i=2}^N \left(R_{\varrho_i} - \frac{2R}{\tan \frac{\varrho_i}{2}} \right). \quad (5.44)$$

5.4.2 Dubins Paths

The objective of the Dubins path is to connect two points in the two-dimensional Euclidean plane by the joining of circular arcs, encapsulating the constraints on the curvature ratio and straight lines and forming a feasible path. Using the formulation in [85] the vehicle follows kinematics laws

$$\begin{aligned} \dot{x} &= V \cos \nu, \\ \dot{y} &= V \sin \nu, \\ \dot{\nu} &= u, \end{aligned} \quad (5.45)$$

where V is the speed constant and control $u \in [-\bar{w}, \bar{w}]$ for the constraint on turn rate. The optimal path between two different configurations are always a arc followed by a straight line. The radius of the circular arc are given by V/\bar{u} . Simplifying the problem, restrictions are made, namely using a constant speed and altitude being user defined. Rewriting R , the circular arcs that define a Dubins path are assumed to be as large as the minimum turn radius of the vehicle. The configuration is defined as (\mathbf{p}, χ) being \mathbf{p} the inertial point and χ the course angle. Taking in account the start configuration (\mathbf{p}_s, χ_s) and the end final configuration (\mathbf{p}_e, χ_e) , the Dubins path is represented as a arc of radius R followed by a straight line, and ending with another arc of radius R . Figure 5.9, shows four possible paths for the vehicle depending on the circumstances of the current position and the desired end positions and angles.

5.4.3 Path Length Computation

The computation of the path is needed in order to determine the Dubins path taking in account the four possible configurations and the selecting the maneuver that presents the shortest path length. Assuming that \mathbf{p}_s and \mathbf{p}_e corresponds to the starting and end position respectively and the χ_s, χ_e to start and end directions with \mathbf{c}_s and \mathbf{c}_e defining the start and end circle locations. Figure 5.10 shows an R-S-R maneuver given by the position \mathbf{p} , the course χ and the radius R , the center of the right and left circles, \mathbf{c}_r and \mathbf{c}_l resulting in equations

$$\mathbf{c}_r = \mathbf{p} + R \left(\cos \left(\chi + \frac{\pi}{2} \right), \sin \left(\chi + \frac{\pi}{2} \right), 0 \right)^T \quad (5.46)$$

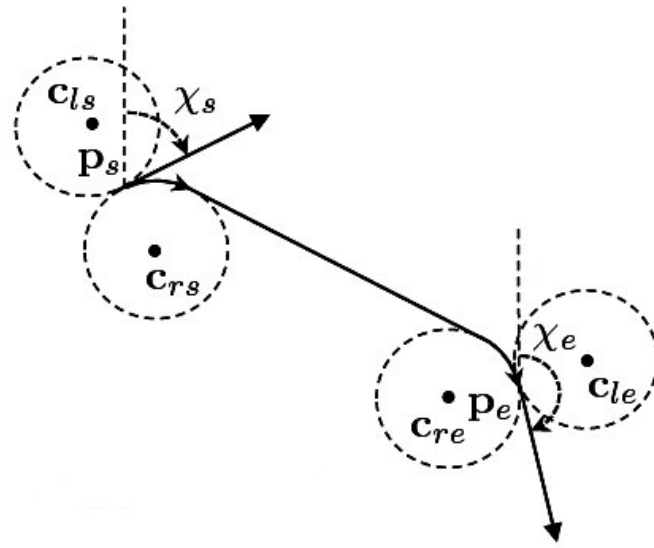
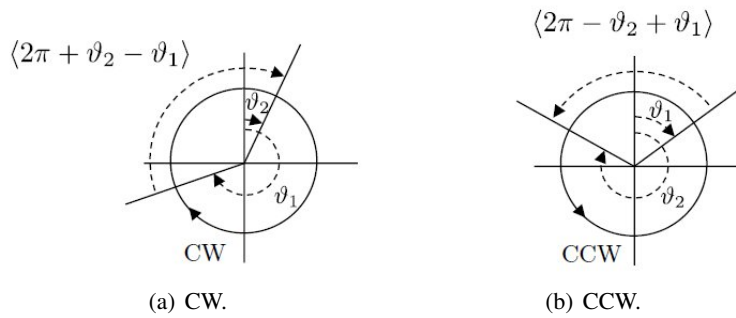


Figure 5.9: Example of one path scenario: R-S-R.

$$\mathbf{c}_l = \mathbf{p} + R(\cos(\chi - \frac{\pi}{2}), \sin(\chi - \frac{\pi}{2}), 0)^T \quad (5.47)$$

Figure 5.10: Angular distance between angles ϑ_1 and ϑ_2 for (CW) and (CCW).

Assuming that both ϑ_1 and ϑ_2 are inside 0 and 2π for clockwise circles, the angular distance between ϑ_1 and ϑ_2 is

$$\|\vartheta_2 - \vartheta_1\|_{CW} \triangleq \langle 2\pi + \vartheta_2 - \vartheta_1 \rangle \quad (5.48)$$

with

$$\langle \varphi \rangle \triangleq \varphi \bmod 2\pi \quad (5.49)$$

or CCW circles

$$\|\vartheta_2 - \vartheta_1\|_{CW} \triangleq \langle 2\pi - \vartheta_2 + \vartheta_1 \rangle \quad (5.50)$$

Case I: R-S-R The geometry for this case is show in Figure 5.11,

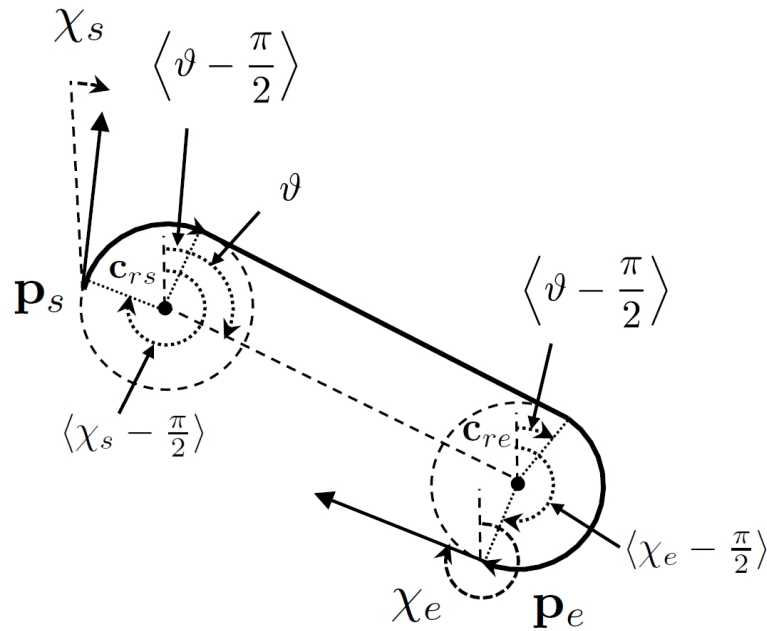


Figure 5.11: Dubins Case R-S-R.

where ϑ is the angle formed by the line between \mathbf{c}_{rs} and \mathbf{c}_{re} . Using the equation from (5.48), the distance traveled along \mathbf{c}_{rs} comes

$$R \langle 2\pi + \langle \vartheta - \frac{\pi}{2} \rangle - \langle \chi_s - \frac{\pi}{2} \rangle \rangle \quad (5.51)$$

similarity for \mathbf{c}_{re} is

$$R \langle 2\pi - \langle \vartheta - \frac{\pi}{2} \rangle + \langle \chi_e - \frac{\pi}{2} \rangle \rangle \quad (5.52)$$

and the total path length I is

$$L_1 = \|\mathbf{c}_{rs} - \mathbf{c}_{re}\| + R\langle 2\pi + \langle \vartheta - \frac{\pi}{2} \rangle - \langle \chi_s - \frac{\pi}{2} \rangle \rangle + R\langle 2\pi - \langle \vartheta - \frac{\pi}{2} \rangle + \langle \chi_e - \frac{\pi}{2} \rangle \rangle \quad (5.53)$$

Case II: R-S-L The geometry for this case is show in Figure 5.12,

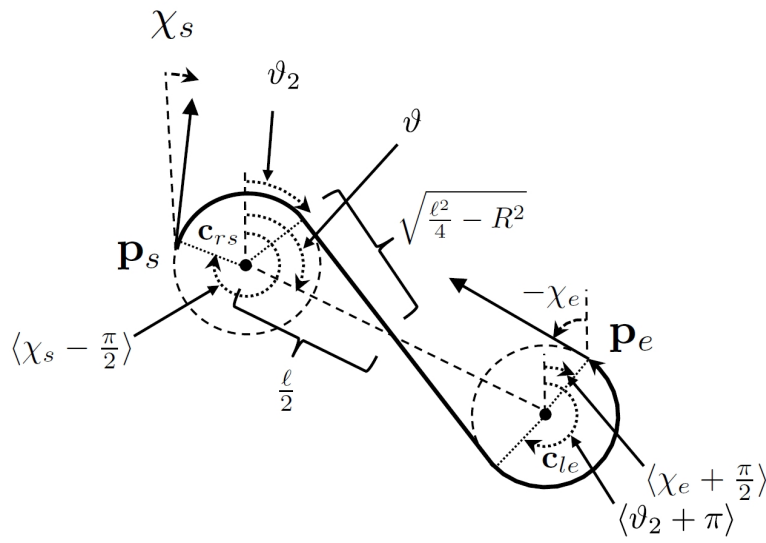


Figure 5.12: Dubins Case R-S-L.

where ν is the angle formed by the line between \mathbf{c}_{rs} and \mathbf{c}_{le} , $l = \|\mathbf{c}_{le} - \mathbf{c}_{rs}\|$ with

$$\vartheta_2 = \vartheta - \frac{\pi}{2} + \sin^{-1} \left(\frac{2R}{l} \right) \quad (5.54)$$

using the same base equation (5.48), the distance traveled along \mathbf{c}_{rs} is

$$R\langle 2\pi + \langle \vartheta_2 \rangle - \langle \chi_s - \frac{\pi}{2} \rangle \rangle \quad (5.55)$$

Using the symmetric equation (5.50), the distance traveled along \mathbf{c}_{ie} is

$$R\langle 2\pi + \langle \vartheta_2 + \pi \rangle - \langle \chi_e + \frac{\pi}{2} \rangle \rangle \quad (5.56)$$

and the total path for this maneuver is

$$L_3 = \sqrt{l^2 - 4R^2} + R\langle 2\pi + \langle \chi_s + \frac{\pi}{2} \rangle - \langle \vartheta + \vartheta_2 \rangle \rangle + R\langle 2\pi + \langle \chi_e - \frac{\pi}{2} \rangle - \langle \vartheta + \vartheta_2 - \pi \rangle \rangle \quad (5.61)$$

Case IV: L-S-L The geometry for this case is show in Figure 5.14,

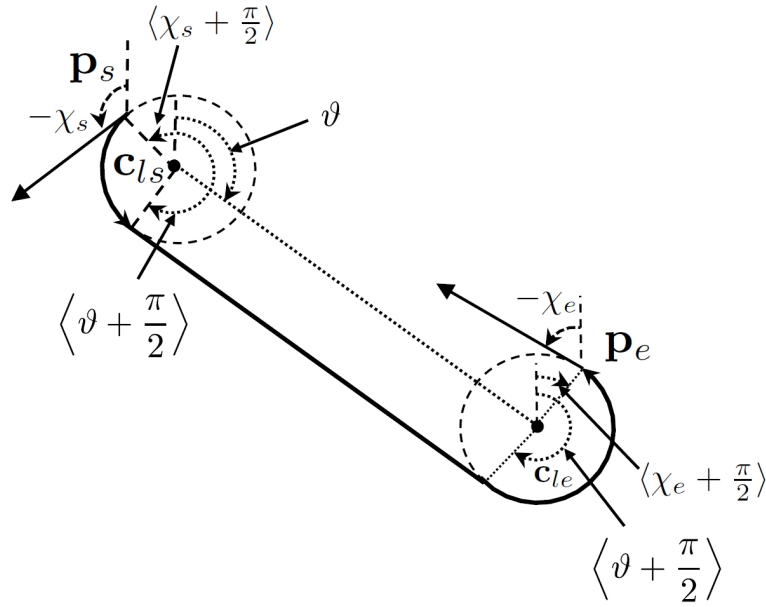


Figure 5.14: Dubins Case L-S-L.

where ϑ is the angle formed by the line between \mathbf{c}_{ls} . Using the base equation (5.50), the distance traveled along \mathbf{c}_{ls} is

$$R\langle 2\pi + \langle \chi_s + \frac{\pi}{2} \rangle - \langle \vartheta + \frac{\pi}{2} \rangle \rangle. \quad (5.62)$$

Using the symmetric equation (5.48), the distance traveled along \mathbf{c}_{le} is

$$R\langle 2\pi - \langle \chi_e + \frac{\pi}{2} \rangle + \langle \vartheta + \frac{\pi}{2} \rangle \rangle \quad (5.63)$$

and the total path for this maneuver is

$$L_4 = \|\mathbf{c}_{ls} - \mathbf{c}_{le}\| + R\langle 2\pi + \langle \chi_s + \frac{\pi}{2} \rangle - \langle \vartheta + \frac{\pi}{2} \rangle \rangle + R\langle 2\pi + \langle \vartheta + \frac{\pi}{2} \rangle - \langle \chi_e + \frac{\pi}{2} \rangle \rangle, \quad (5.64)$$

being χ_s, χ_e the course angle at start point and end point respectively.

5.5 Implementation

In order to drive the agents to the desired positions, we developed a simple controller that implements two basic maneuvers, straight-line and loiter. With these maneuvers, agents are drive to desired position to accomplishing the target following that was assigned or perform search maneuvers. When assigned, the agent is supplied with waypoint W_{i+i} given by the prediction of the tracking system for the instant $k + 3$. This prediction is used by the tracking controller while the tracking system only uses $k + 1$ to incorporate prediction. The carrot chasing algorithm uses an virtual target point (VTP) to direct the UAV to its chase.

1) *Straight line following*: Following a straight line involves two steps: (i) determining cross-track error (d) and (ii) updating the location of the virtual target. The VTP (s) in Figure 5.15 is distanced from q by δ . The desired heading (ψ_d) is updated based on the location of s . The process of finding new (ψ_d) continues until W_{i+1} is reached. Algorithm 5.3 defines a proportional controller with gain $k > 0$.

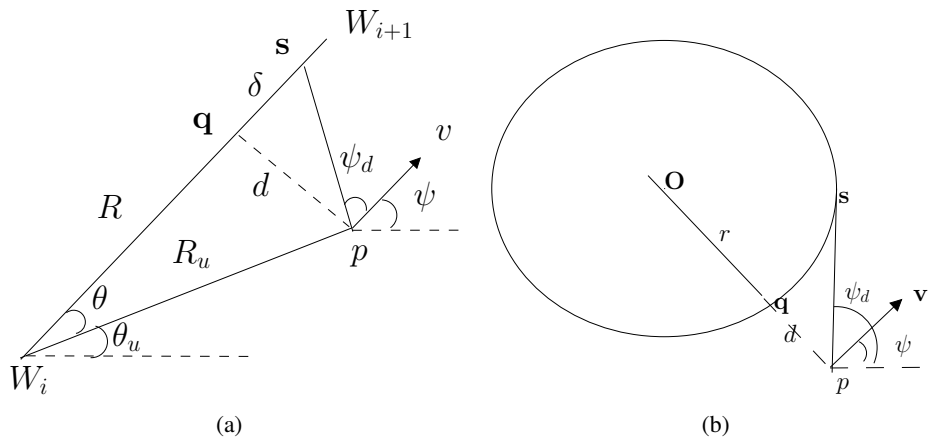


Figure 5.15: (a) The UAV follow a straight line path (b) The UAV follow a or orbit of radius r .

2) *Loiter*: On the loiter maneuver, the UAV is drive to circle around a point (O) with a radius of r . For loiter maneuver be accomplished, the UAV must be located on the circle and the heading direction must be orthogonal to the line Oq . Assuming that UAV is located at p and cross-track error $d = ||O - p|| - r$. With this, VTP (s in Figure) can be generated. The convergence is controlled by parameter λ .

Algorithm 5.3 Algorithms for carrot chasing algorithm**Straight Line following**

$$\text{Init : } W_i = (x_i, y_i), W_{i+1} = (x_{i+1}, y_{i+1}), p = (x, y), \psi$$

$$R_u = \|W_i - p\|, \theta = \text{atan2}(y_{i+1} - y_i, x_{y+1} - x_i)$$

$$\theta_u = \text{atan2}(y - y_i, x - x_i), \beta = \theta - \theta_u$$

$$R = \sqrt{R_u^2 - (R_u \sin(\beta))^2}$$

$$(\hat{x}_t, \hat{y}_t) \leftarrow ((R + \delta) \cos \theta, (R + \delta) \sin \theta) \% s = (\hat{x}_t, \hat{y}_t)$$

$$\psi_d = \text{atan2}(\hat{y}_t - y, \hat{x}_t - x)$$

$$u = k(\psi_d - \psi)$$

Loiter

$$\text{Init : } O = (x_l, y_l), r, p, \psi$$

$$d = \|O - p\| - r$$

$$\theta = \text{atan2}(\hat{y}_l - y, \hat{x}_l - x)$$

$$(\hat{x}_l, \hat{y}_l) \leftarrow (r \cos(\theta + \lambda), r \sin(\theta + \lambda)) \% s = (\hat{x}_l, \hat{y}_l)$$

$$\psi_d = \text{atan2}(\hat{y}_l - y, \hat{x}_l - x)$$

$$u = k(\psi_d - \psi)$$

The initial implementation, Figure 5.16 shows the correct convergence to the desired straight line path maneuver.

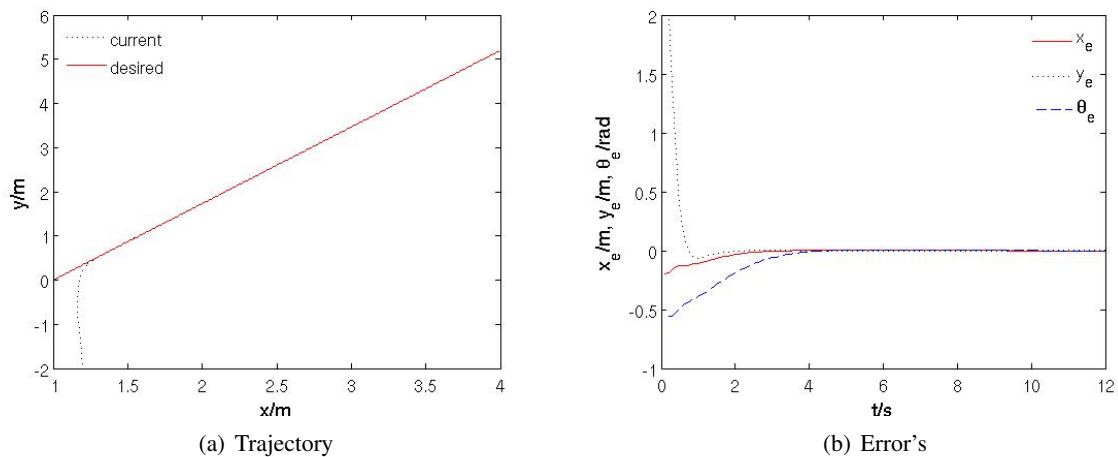


Figure 5.16: Straight line path and error components

Regarding loiter/orbit maneuvers, Figure 5.17 shows the correct convergence to the desired path maneuver with the correspondent convergence errors during the maneuver.

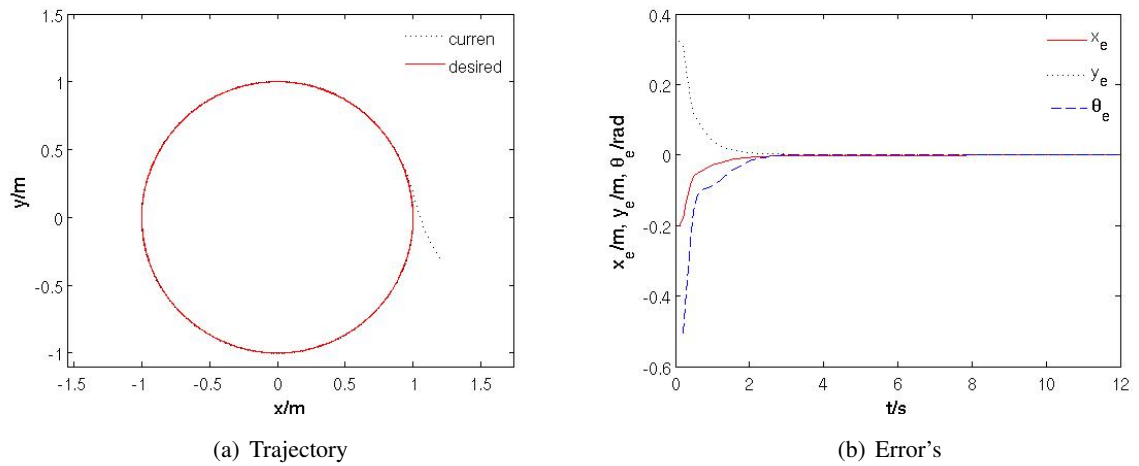


Figure 5.17: Orbit path and error components

The Figure 5.18 show the combination of the Dubin's path with the primitive maneuvers on the expansion trees with a A-Star search algorithm to find the feasible minimal path to the objective while avoiding the obstacles.

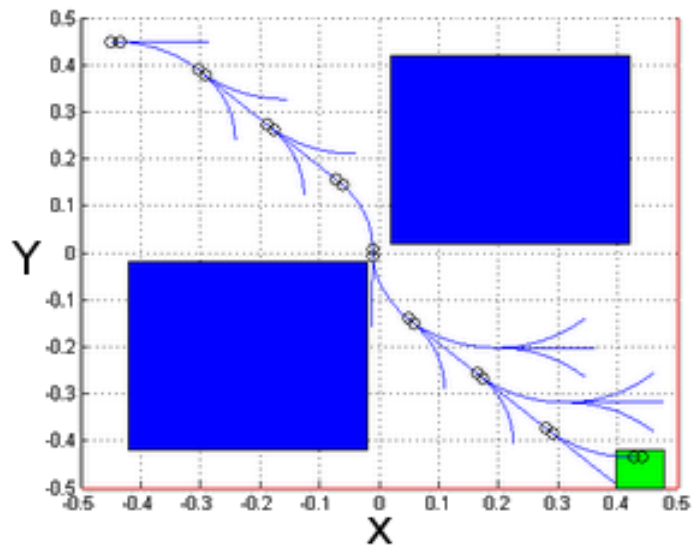


Figure 5.18: Optimal Dubins Path's with A-Star discovery.

A-Star is used to search for a solution path to the goal objective using the 2 basic maneuvers that encapsulate the kinematics constraints of the vehicle, namely turn radius. A-Star is responsible for finding the minimal path from the expanded nodes and their correspondent heuristics build during course with the awareness sensor. The value of turn radius affects directly the ability to find a feasible path to the goal in minimal time. Figure 5.19 shows the same obstacle scenario

were the turn radius is minor and the A-Star was not able to find a solution to the objective with the feasible maneuvers available.

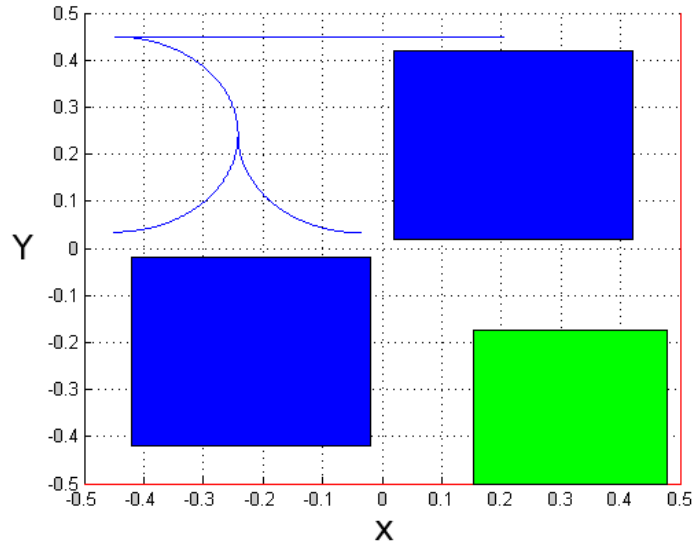
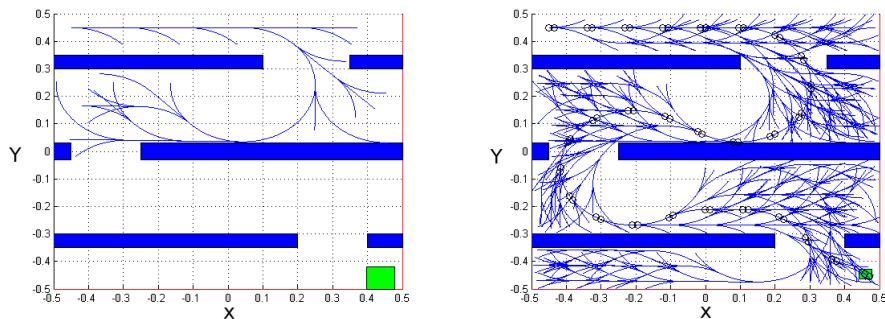


Figure 5.19: Dubins Path's with A-Star discovery with minor turn rate maneuver.

Figure 5.20 shows different scenario and the calculation of the feasible path to the goal.



(a) Dubins Path's with A-Star discovery with minor turn rate maneuver.

(b) Dubins Path's with A-Star discovery with bigger turn rate maneuver.

Figure 5.20: Dubins Path's with A-Star discovery.

Chapter 6

Tracking

In this chapter, we introduce the formulation of the particle for tracking and develop a multi-target tracking system. We also show the ability of the tracking system to track a ship in the ocean.

6.1 Introduction Tracking

The motion models used for robots and the sensor models are typically imperfect. Hence probabilistic techniques are used to take the stochastic nature of these models into account. The techniques integrate imperfect models and imperfect sensors through probabilistic laws, such as "Bayes" rule. Many recently fielded state-of-the-art robotics systems employ probabilistic techniques for perception [87, 88, 89]. Similarly, the problem of tracking a varying number of non-rigid faces is a similar problem regarding the model observation and target distributions that may present highly non-linear and non-Gaussian distributions. On the other hand, the presence of a large and varying number of objects generates complex interactions resulting in several ambiguities on the distribution. To address this problems Sequential Monte-Carlo techniques are used to increase the precision of the localization. We assume that the UAV is equipped with camera sensor to detect the target. Since, the target geo-location cannot be determined accurately, we use a particle filter to estimate the target geo-location in the global coordinate frame.

6.2 Particle filter

The filtering is formed by applying the $p(\mathbf{x}_k|\mathbf{Z}_k)$ to the distribution of the latent state \mathbf{x}_k conditional on the past data $\mathbf{Z}_k = (\mathbf{z}_i, i = 1, \dots, k)$ resulting in the transition probability. Similar to the Kalman filter, the particle filter uses two steps, prediction and update stages, which are based on recursive structure of optimal Bayesian filtering. These steps arise from Bayes rule, which express the filtering distribution by the likelihood, $p(\mathbf{z}_k|\mathbf{x}_k)$ and predicted density, $p(\mathbf{x}_k|\mathbf{Z}_{k-1})$:

$$p(\mathbf{x}_k|\mathbf{Z}_k) \approx p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{Z}_{k-1}) \quad (6.1)$$

While the likelihood function is typically known, the predictive distribution of the state is given as the integral against the previous period's filtering density

$$p(\mathbf{x}_k|\mathbf{Z}_{k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})d\mathbf{x}_{k-1}. \quad (6.2)$$

These densities define the prediction and updating stages, but computing these distributions is difficult due to complex analytic forms that the system may form. These analytical solutions are only available on fewer cases, namely those that include Gaussian models with finitely states. In the Gaussian case, $p(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1}) \approx N((\boldsymbol{\mu}, \boldsymbol{\sigma}_{k-1}^2))$ and the Kalman filter provides well defined update mechanisms on $(\boldsymbol{\mu}, \boldsymbol{\sigma}_{k-1}^2) \rightarrow (\boldsymbol{\mu}_k, \boldsymbol{\sigma}_k^2)$. In general, $p(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})$ is a complex function, potentially a high-dimensional vector, \mathbf{Z}_{k-1} , which prevents the application of standard Bayesian methods on update. Particle filter use a discrete approximation of the optimal filtering problem $p(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})$,

$$p^N(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1}) = \sum_{i=1}^n w_{k-1}^i \delta_{\mathbf{x}_{k-1}^{(i)}}, \quad (6.3)$$

where δ is the Dirac function and $\{w_{k-1}^{(i)}, \mathbf{x}_{k-1}^{(i)}\}_{i=1}^N$ denotes a set of weights and particles. Given an particle approximation, equation 6.1 becomes a sum instead of an integral. The filtering recursion is now replaced with a simulation step to generate the next set of weights and particles

$$\{w_k^{(i)}, \mathbf{x}_k^{(i)}\}_{i=1}^N \approx p^N(\mathbf{x}_k|\mathbf{Z}_k) \quad (6.4)$$

given the particles \mathbf{x}_{k-1}^i from $p^N(\mathbf{x}_{k-1}|\mathbf{z}_{k-1})$. This is key, as filtering is just an exercise on Bayesian updating. Expectations such as $E(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})$ are then easily computed as Monte Carlo averages $N^{-1} \sum_{i=1}^N \mathbf{x}_{k-1}^{(i)}$.

6.2.1 The Filtering Problem

As referred in 6.2, state space models consist of the following components: the observation equation, $p(\mathbf{z}_{k-1}|\mathbf{x}_{k-1})$, a state evolution or transition, $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ and an initial state distribution or population, $p(\mathbf{x}_0)$. The observation equation is a likelihood function providing the distribution of the observed data conditioned to the latent states. A simple example that can be solved using the Kalman filter is the linear Gaussian specification assuming that parameters are known.

$$\begin{aligned} \mathbf{z}_k &= \mathbf{x}_k + \boldsymbol{\sigma} \boldsymbol{\varepsilon}_k^z, \\ \mathbf{x}_k &= \mathbf{x}_{k-1} + \boldsymbol{\sigma}_x \boldsymbol{\varepsilon}_k^x. \end{aligned} \quad (6.5)$$

In general, assume the relation between the states and observations is nonlinear and described as

$$\begin{aligned}\mathbf{z}_{k-1} &= F(\mathbf{x}_{k-1}, \boldsymbol{\varepsilon}_{k-1}^z), \\ \mathbf{x}_k &= G(\mathbf{x}_{k-1}, \boldsymbol{\varepsilon}_k^x),\end{aligned}\tag{6.6}$$

for known F and G functions. Common non-linear models include discrete outcome models with stochastic volatility. Often, the function F is obtained from a known model, such as kinematic model, and $\boldsymbol{\varepsilon}_{k-1}^z$ can be an additive or multiplicative error defined by the uncertainties on the model.

A filtering algorithm has the objective to provide an approximation to the sequence of densities, $p(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})$ for $k = 1, \dots, N$. This is fundamentally different from the smoothing problem in which the inference on the latent state is based on the complete set of data samples, $p(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})$ which includes both past and future data. The distribution $p(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})$ as the solution to the filtering problem follows the well-known optimal properties of the posterior distribution. The filtering problem can be used for prediction of states or model estimation.

6.2.2 Particle Filter Method

The particle filter algorithm solves the fundamental problem of recursive Bayesian filtering by computing the integral in equation 6.2 using a discrete approximation to the filtering density. In general, a discrete approximation to a density consists of weights and state, given by $\{\mathbf{w}_{k-1}^{(i)}, \mathbf{x}_{k-1}^{(i)}\}_{i=1}^N$. The $\mathbf{x}_{k-1}^{(i)}$ is not deterministic, but rather is the stochastic outcome of the algorithm. A particle approximation to $p^N(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})$ is defined using equation 6.3. The particle approximation $\{\mathbf{w}_{k-1}^{(i)}, \mathbf{x}_{k-1}^{(i)}\}_{i=1}^N$ can be transformed into an equally weighted random sample from $p^N(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})$ by sampling from the discrete distribution $\{\mathbf{w}_{k-1}^{(i)}, \mathbf{x}_{k-1}^{(i)}\}_{i=1}^N$.

This procedure is commonly defined as *resampling*, producing a new sample with uniformly distributed weights so that $\mathbf{w}_{k-1}^{(i)} = N^{-1}$. In general, resample is done by drawing from a multinomial distribution. One important issue in the resampling step is the computational speed due to the large number of particles required for the correct representation of the distribution. Given the particle approximation at time $k-1$, the particle approximation at time k , the integral

$$p^N(\mathbf{x}_k|\mathbf{Z}_k) \approx \int p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{x}_{k-1})p^N(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})d\mathbf{x}_k,\tag{6.7}$$

can be transformed into an sum by

$$p^N(\mathbf{x}_k|\mathbf{Z}_k) = \frac{1}{N} \sum_{i=1}^n p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{x}_{k-1}^{(i)}),\tag{6.8}$$

showing that particle approximation for $p^N(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})$ is updated to a particle approximation for $p^N(\mathbf{x}_k|\mathbf{Z}_k)$. The problem in particle filtering is how to sample from $p^N(\mathbf{x}_k|\mathbf{Z}_k)$. The particle filtering algorithm essentially consists on different ways of sampling from $p^N(\mathbf{x}_k|\mathbf{Z}_k)$ taking the form of a mixture distribution.

6.2.3 SIR - Sequential Importance Resampling

The classic approach to sampling is a mixture distribution where, we first select a mixture component to simulate. The representation in equation 6.7 is not amenable to sampling using the classic approach because of the natural mixture weights, $p(\mathbf{Z}_k|\mathbf{x}_k)$, that are dependent on the values of the as-yet-to-be updated state variables. In most particle filtering algorithms, importance sampling is used to circumvent this problem. The classical particle filtering algorithm is known as the sampling/importance resampling (SIR) algorithm. The algorithm is simply by itself, relying only on two steps: given samples from $p^N(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})$,

$$\begin{aligned} \text{Step1. (Propagate)} \quad \mathbf{x}_k^{(i)} &\approx p(x_{t+1}|\mathbf{x}_{k-1}^{(i)}) \text{ for } i = 1, \dots, N, \\ \text{Step2. (Resample)} \quad \mathbf{x}_k^{(i)} &\approx \text{Mult}_N(\{\mathbf{w}_k^{(i)}\}_{i=1}^N), \end{aligned} \quad (6.9)$$

where the normalized importance weights are calculated as

$$\mathbf{w}_k^{(i)} = \frac{p(\mathbf{z}_k|\mathbf{x}_k^{(i)})}{\sum_{i=1}^N p(\mathbf{z}_k|\mathbf{x}_k^{(i)})}. \quad (6.10)$$

The particle filter has two requirements: the likelihood function to be evaluated and the states to be simulated. Virtually every model used in practice satisfies these assumptions. The justification is based on the weighted bootstrap algorithm or SIR algorithm, developed to simulate posterior distributions that take the form of $L(\mathbf{x})p(\mathbf{x})$, where L is the likelihood and p the prior. To sample from this distribution, first draw an independent sample $\mathbf{x}^{(i)} \approx p(\mathbf{x})$ and computed the normalized weights $\mathbf{w}^{(i)} = L(\mathbf{x}^{(i)}) / \sum L(\mathbf{x}^{(i)})$. The samples drawn from an discrete distribution quantified by their weights $\{\mathbf{x}^{(i)}, \mathbf{w}^{(i)}\}_{i=1}^N$ tends to form the product density, $L(\mathbf{x})p(\mathbf{x})$ as the number of particles N increases. The target density is

$$p(\mathbf{x}_k|\mathbf{Z}_k) \approx p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{Z}_{k-1}), \quad (6.11)$$

taking an individual sample from $p^N(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})$, sampled from

$$p^N(\mathbf{x}_k|\mathbf{Z}_{k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})p^N(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1}) dx_k, \quad (6.12)$$

by drawing $\mathbf{x}_k^{(i)} \approx p(\mathbf{x}_k|\mathbf{x}_{k-1}^{(i)})$ for $i = 1, \dots, N$. Since $p^N(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})$ is a discrete distribution, implies that $\{\mathbf{x}_k^{(i)}\}$ forms a independent sample from $p^N(\mathbf{x}_k|\mathbf{Z}_{k-1})$. Resampling with the appropriate weights provides the approximate sample from $p(\mathbf{x}_k|\mathbf{Z}_k)$. The SIR requires two distributions which are known

$$\begin{aligned} p(\mathbf{x}_k|\mathbf{x}_{k-1}) &\approx N(\mathbf{x}_{k-1}, \sigma_{\mathbf{x}}^2), \\ p(\mathbf{z}_k|\mathbf{x}_k) &\approx \exp\left(-\frac{1}{2} \frac{(\mathbf{z}_k - \mathbf{x}_k)^2}{\sigma^2}\right). \end{aligned} \quad (6.13)$$

The SIR is given by

$$\begin{aligned} \text{Step 1 : (Sample)} \quad \mathbf{x}_k &\approx N(\mathbf{x}_{k-1}^{(i)}, \sigma_{\mathbf{x}}^2) \\ \text{Step 2 : (Draw)} \quad \mathbf{x}_k &\approx \text{Multi}_N(\{\mathbf{w}(x_k^{(i)})\}_{i=1}^N), \end{aligned} \quad (6.14)$$

where

$$\mathbf{w}(\mathbf{x}_k^{(i)}) = \frac{\exp(-\frac{1}{2}(\mathbf{z}_k - \mathbf{x}_k^{(i)})/\sigma^2)}{\sum_{i=1}^N \exp(-\frac{1}{2}(\mathbf{z}_k - \mathbf{x}_k^{(i)})^2/\sigma^2)}. \quad (6.15)$$

The SIR algorithm cannot be used directly due to degeneracy of the sample weights [90]. This occurs when a vast majority of the weights are concentrated in a single particle. Due to this, the resampling step results in only a single or small set of particles being sampled multiple times and does not improve the weight degeneracy. Another identified problem is that the propagated states are drawn from prior distribution, $p(\mathbf{x}_k|\mathbf{x}_{k-1})$, without accounting for the next period of observations, \mathbf{z}_k . This imposes that the simulated states may not be important or posses an high likelihood, $p(\mathbf{z}_k|\mathbf{x}_k)$ regions. When a large \mathbf{z}_k which is generated by a large observation \mathbf{x}_k , the SIR algorithm draws samples from $p(\mathbf{x}_k|\mathbf{x}_{k-1}^{(i)})$ ignoring the fact that a large observation \mathbf{z}_k have occurred. To mitigate this problem, one approach is to chose an alternative importance density. Instead of drawing from the prior, $p(\mathbf{x}_k|\mathbf{x}_{k-1})$, a importance density that depends on the next observation is draw

$$\mathbf{x}_k^{(i)} \approx q(\mathbf{x}_k|\mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k). \quad (6.16)$$

The unnormalized weights are

$$\mathbf{w}_k^{(i)} \approx \frac{p(\mathbf{z}_k|\mathbf{x}_k^{(i)})p(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)})}{q(\mathbf{x}_k|\mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k)}. \quad (6.17)$$

The “optimal” importance sampling density, in terms of minimizing the variance of the weights importance, is $p(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{z}_k)$.

6.2.4 SIS - Sequential Importance Sampling

A general approach for filtering is known as sequential importance sampling based on an generalized SIR algorithm. For a given function $F(\mathbf{x}_{k-1})$ we have

$$E[F(\mathbf{x}_k|\mathbf{Z}_k)] = \frac{E_q[\mathbf{w}_k F(\mathbf{x}_k|\mathbf{Z}_k)]}{E_q[\mathbf{w}_k|\mathbf{Z}_k]}, \quad (6.18)$$

which is the measure formula, q is an importance density, used to estimate the expectations of the state variable function, and the change of probability measure given by $\mathbf{w}_{k-1} = p(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})/q(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})$. Given samples from the importance density, the Monte-Carlo sampling estimate comes

$$E^N[F(\mathbf{x}_k|\mathbf{Z}_k)] \approx \sum_{i=1}^N \mathbf{w}_k^{(i)} F(\mathbf{w}_k^{(i)}), \quad (6.19)$$

and the constant of proportionality is determined by $\sum_{i=1}^N \mathbf{w}_k^{(i)}$. The recursive specification for $p(\mathbf{x}_k|\mathbf{Z}_k)$ for SIS is

$$p(\mathbf{x}_k|\mathbf{Z}_k) \approx p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{x}_{k-1})P(\mathbf{X}_{k-1}|\mathbf{Z}_{k-1}), \quad (6.20)$$

and for $q(\mathbf{x}_k|\mathbf{Z}_k)$ is

$$q(\mathbf{X}_k|\mathbf{Z}_k) \approx q(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{z}_k)q(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1}). \quad (6.21)$$

Given the recursive specification of the target and importance density, the weights are computed recursively:

$$\begin{aligned} \mathbf{w}_k^{(i)} &= \frac{p((\mathbf{X}_k)^{(i)}|\mathbf{Z}_k)}{q((\mathbf{X}_k)^{(i)}|\mathbf{Z}_k)} \\ &\approx \frac{p(\mathbf{z}_k^{(i)}|\mathbf{x}_k^{(i)})P(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)})}{q(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k)} \mathbf{w}_{k-1}^{(i)}. \end{aligned} \quad (6.22)$$

Many models have a memory-less stochastic process based on Markov structure, which implies that sample proposals that are dependent only on the subset \mathbf{z}_k and not the whole vector \mathbf{Z}_k are reasonable. In this case, the importance weights are simplified and the optimal importance

function can be used. SIS importance sampling algorithms are similar to the SIR algorithm, but without continuous resampling in all steps. The performance of SIS methods depend on the accuracy of the level of approximation made in the importance sampling. In general, the variance of the importance weights increases over time, leading to the problem of weight degeneracy, whereby a vast majority of the weight are placed on a small set of particles. One way to fix to this problem is to resample as refereed. This process can be carry out every period or only when the weights become too imbalanced. If performed every period, $\mathbf{w}_{k-1}^{(i)} = N^{-1}$, them SIS algorithm collapses to the SIR algorithm.

6.2.5 Resampling

Its desirable that the resampling process occurs only when the level of degradation of the weight reaches a threshold to save computational power. Normally the degree of degeneration is quantified by measuring the effective number of particles that evolves in a proportional inverse way to the value of degeneration. Each time this value surpass threshold, the resampling algorithm is executed. An approximation for the number of effective particles involved can be obtained by the expression:

$$N_{eff} \approx \frac{1}{\sum_{i=1}^N (\mathbf{w}_k^{(i)})^2}. \quad (6.23)$$

Less the sum of the particles weight bigger is the effective number of particles representing the state. This is one of the main disadvantages of the use of SIS described before. The main disadvantage in SIS is in the implementation where we cannot control how many particles are following each target in multi target scenarios, and the low tolerance to false negatives. The SIS algorithm can be resumed by the fluxogram presented by Figure 6.1.

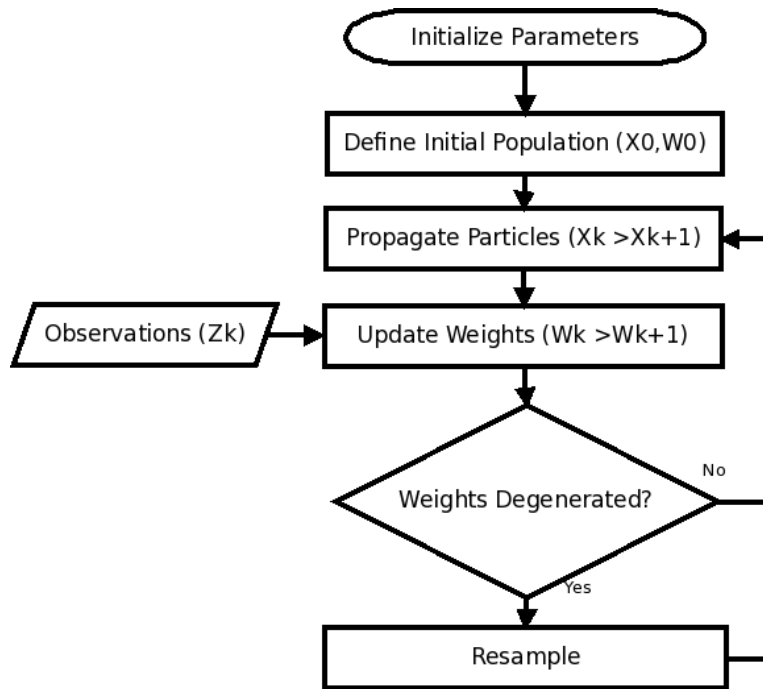


Figure 6.1: Particle Filter Fluxogram.

6.3 Multiple Target Implemented Mechanism

To address the problems identified before, we decide to implement a derivative mechanism to handle most of the problems described. The several steps of the algorithm are similar to the basic implementation, but can be divided in three main parts, Initialization, Detection of lost targets, Predict and Update stages, described in Figure 6.2.

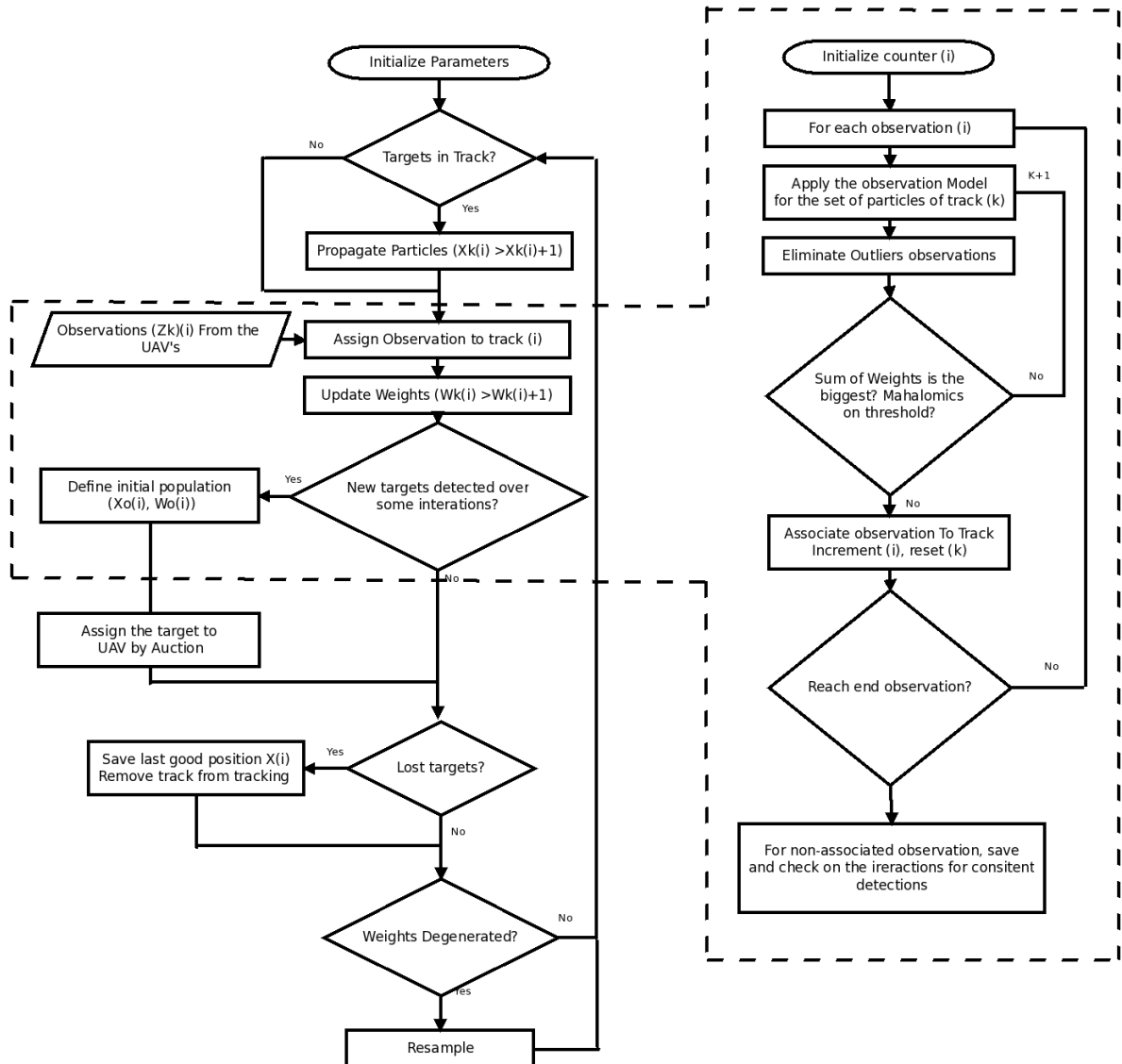


Figure 6.2: Implemented Particle Filter Mechanism

Particle Filter Initialization In our implementation, we used one filter with a set of particles for each target. However this approach requires a correct initialization and an adequate strategy for the data association. For the initialization of the targets, and using the field of view of the camera, we define a set of static particles that cover all the observable area from the UAV and the function is periodically called to see if a new target appears and needs to be initialized. To detect a new target its determined the local maximum of the weight regarding static particles.

This is made using a window of multiple value centered over the maximum points enabling the consideration of values in the neighbors points to the calculation of the target initial position. The choice of this value is directly related to the maximum speed of the targets may present in a period of time. Posterior to this, a filtering process is made to eliminate close targets that might being

already followed. When a static target is determined in a consistent way over tree iterations, a new set of mobile particles is launched to track that specific target. At the same time this mechanisms handles contaminated measurements because the implementation does not try to initialize a new filter immediately, however it waits for new observations to come and matching them near the site were it made its first appearance. If the level of match is above a certain value, a new filter is launch. Of course this technique does not grant the tracking of all targets in the first instants that don't have consistent observations. The initialization of the particles and propagation intend to model following the model the dubins model, or learning the movement from the observations made.

Detection Lost Targets To handle the situation of a target getting lost, caused by the lack of observations for a long time of that target, two assumptions can be made. The target has disappeared from our field of view, or the occlusion due to obstacles occurs for a long time. In both situations, the continuous analysis of the dispersion of particles for all targets permits to identify those that don't have observations over some iterations. Keep in mind that the dispersion value of the particles over the space is related to the quality of the estimation, the targets that have their particles dispersion above and threshold are eliminated from the track and last good known position is returned.

Prediction and Update Regarding the propagation of the particles, the use of random controls enables to handle any distribution that the target may present or with just a simple change of flag use the dynamic model learned from the movement of the target.

Regarding update and the data association strategy, for each target we apply the model of observation relatively to the set of particles of each target, and the target in which the total sum of weight of the particles is the maximum is associated with that specific observation. The algorithm 6.1 describes in detail the association of the observation to target and update of weights.

This mechanism has several advantages, one of them is that a observation is only associated to one target, and at the the same time, one target can be associated with several observations, accommodating observations with clutter around the target. In this case those are used forming a single composed observation intersecting a gate of acceptance with a 2D Gaussian distribution for weight calculation. The agglomeration model for the particles follow the equations:

$$\begin{aligned}
 alvos(1,i) &= \sum x(:,1,i) \cdot w(:,i) \\
 alvos(2,i) &= \sum x(:,2,i) \cdot w(:,i) \\
 alvos(3,i) &= \tan^{-1} \frac{alvos_k(2,i) - alvos_{k-1}(2,i)}{alvos_k(1,i) - alvos_{k-1}(1,i)} \\
 alvos(4,i) &= \sqrt{(alvos_k(2,i) - alvos_{k-1}(2,i))^2 + (alvos_k(1,i) - alvos_{k-1}(1,i))^2} \\
 alvos(5,i) &= alvos_k(3,i) - alvos_{k-1}(3,i)
 \end{aligned} \tag{6.24}$$

The two first are the target position x, y that are formed by the sum of the particle position \mathbf{x} pondered by their weights \mathbf{w} . The bearing is given by the third entry on equation, formed by the arc-tangent of the current position at i and the prior position $i - 1$. Entry four, computes the Δ_s (interval of space) to form the speed of the target. Final entry gives the delta bearing from the current bearing and the older one.

Algorithm 6.1 ObsModel: $W = \text{ObsModel}(x, w, obs)$

Input: X, W, Obs

$n\ obs = \text{getNumberObs}(obs)$

$n\ targets = \text{getNumberTargets}(X)$

if $n\ obs > 0$ **then**

for $i=1$ to $\text{lenght}(n\ obs)$ **do**

for $j=1$ to $\text{lenght}(n\ targets)$ **do**

$range = obs(range)$

$bearing = obs(bearing)$

$zxx = obs(x)$

$zyy = obs(y)$

$\Delta x = X_x(j) - zxx$

$\Delta y = X_y(j) - zyy$

$rangeerr = \sqrt{\Delta x^2 + \Delta y^2} - range$

$bearingerr = \tan^{-1} \frac{\Delta y}{\Delta x} - bearing$

$dist(j) = \exp^{rangeerr^2 + bearingerr^2}$

end for

$soma\ err = \sum(dist)$

$index = \text{find}(somaerr == \max(somaerr))$

$w(idx) = w(idx) + w(idx) * dist(idx)$

end for

end if

for $i=1$ to $\text{lenght}(n\ targets)$ **do**

$somat = \sum(w(i))$

if $somat \neq 0$ **then**

$w(i) = w(i) / somat$

end if

end for

return W at final run

6.4 Results Obtained

The images below show three different scenarios that might occur during the tracking of ground targets. The Figure 6.3 presents a scenario where the targets do not cross

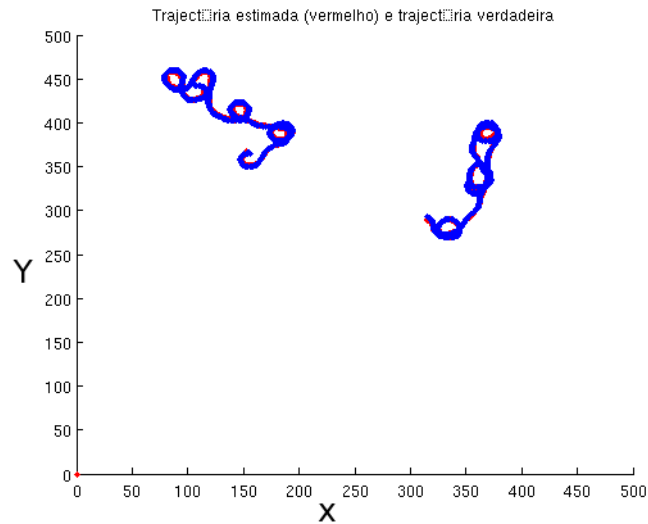


Figure 6.3: Tracking two target with no crossing paths.

between them and the range of the sensor cannot track due to imposed sensor range and bearing limitations.

In Figure 6.4 is denoted the correct data association mechanism in presence of targets

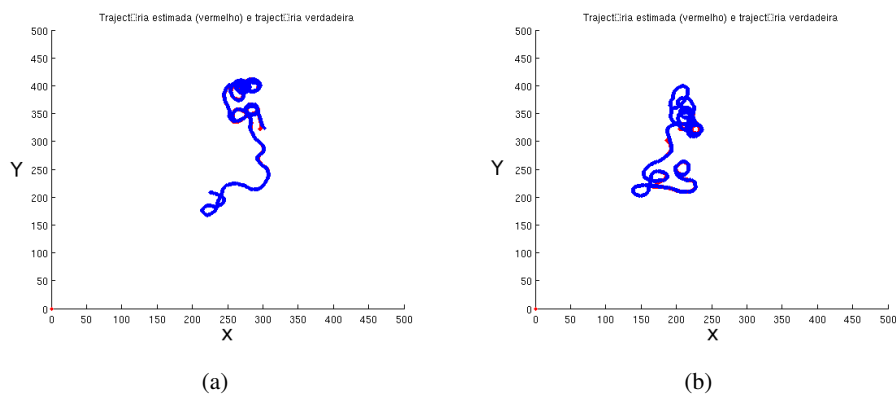


Figure 6.4: Tracking two target with crossing paths (a) and (b)

that cross between then showing that the tracking does not get disturbed this situation.

The Figure 6.5 shows the performance of the track with cluttered measures and false positives.

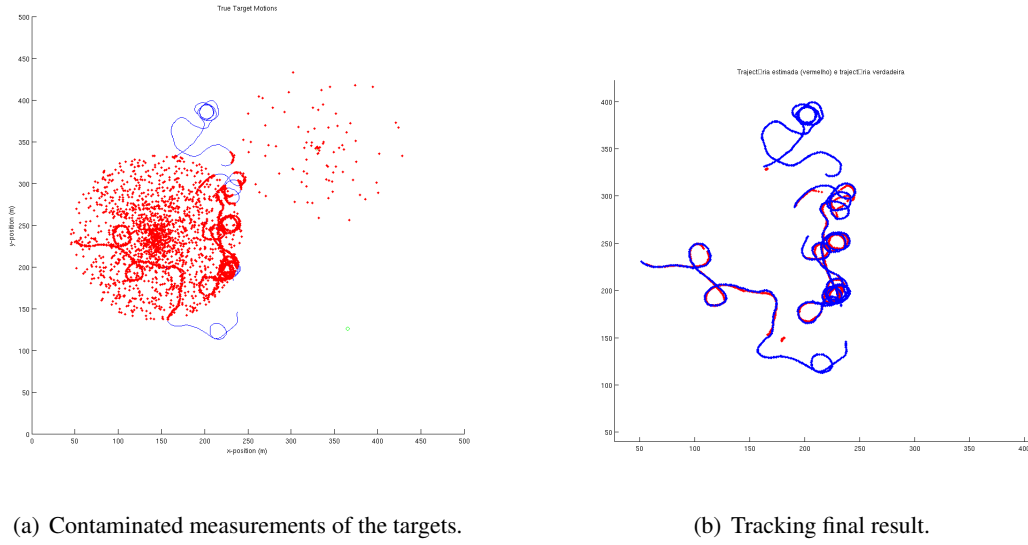


Figure 6.5: Tracking robustness to noise.

This shows the correct data association and rejection of misleading measurements regarding the current state of tracking. The cluttered measurements are aggregated as one observation and associated to a specific target in track. False positives measurements are rejected taking in account the current state of track avoiding the divergence of the tracking.

6.5 Single Tracking Implementation using OpenCV

In order to track objects using particle filters, an implementation in OpenCV has been developed enabling the tracking of designated objects selected by the user. To do this, the object to track needs to be supplied with a small sample while using the Condensation Algorithm [91] with systematic resampling for the contour of the object. In order to apply the Condensation Algorithm [91], which is general used to tracking curves in image-streams, specific probability densities must be established both for the dynamics of the object and for the observation process. Defining \mathbf{x} as a linear parameterisation of the curve and the allowed transformations of the curve are represented by linear transformations of \mathbf{x} . Objects are modelled as a curve (or set of curves) and represented at time t by a parameterised image curve $\mathbf{r}(s, t)$. The parameterisation is in terms of B-splines, so

$$\mathbf{r}(s, t) = (\mathbf{B}(s) \cdot \mathbf{Q}^x(t), \mathbf{B}(s) \cdot \mathbf{Q}^y(t)), \text{ for } 0 \leq s \leq L \quad (6.25)$$

where $\mathbf{B}(s)$ is a vector $.(B_1(s), \dots, B_{N_B}(s))^T$ of B-spline basis functions, Q^x and Q^y are vectors of B-spline control point coordinates and L is the number of spans. The spline was restricted to a

fixed shape-space vectors \mathbf{X} defined by

$$\begin{pmatrix} \mathbf{Q}^x \\ \mathbf{Q}^y \end{pmatrix} = \mathbf{W}\mathbf{X} + \begin{pmatrix} \bar{\mathbf{Q}}^x \\ \bar{\mathbf{Q}}^y \end{pmatrix} \quad (6.26)$$

where the matrix W is a matrix of rank N_X considerably lower than the $2N_B$ degrees of freedom of the unconstrained spline. The space is constructed by applying an appropriate combination of three methods to build the W matrix

- 1) determining analytically combination of contours derived from one or more views,
- 2) capturing sequences of key frames of the object in different poses,
- 3) and performing principal components analysis on a set of outlines of the deforming object.

To estimate the conditional probability considering the simple form of Bayes rule

$$p(x|z) = \alpha p(z|x)p(x) \quad (6.27)$$

the following steps are carried out:

Sample Select an set of N samples prof the prior distribution distribution using equation 6.9.

Weights Assuming that $p(\mathbf{Z}_k|\mathbf{X}_{k-1})$ is known, the probability weights can be evaluated form the measurements using equation 6.10. The procedure can be resumed in the Figure 6.6. The prior distribution is represented in red and the conditional probability of the observations, given the object state, $p(\mathbf{Z}_{k-1}|\mathbf{X}_{k-1})$, in black. The samples selected from are weighted by $p(\mathbf{Z}_k|\mathbf{X}_k)$ to form the sample set represented in blue. The height of the sample is proportional to $p(\mathbf{Z}_k|\mathbf{X}_k)$. The number of samples within an interval is proportional to $p(\mathbf{x})$.

The problem now is to estimate the mean position from the given samples. Using the weighted samples, the mean is given by

$$\bar{x} = \frac{\sum_{i=1}^{N-1} x p(\mathbf{Z}_{k-1}|\mathbf{X}_{k-1})}{\sum_{i=1}^{N-1} p(\mathbf{Z}_{k-1}|\mathbf{X}_{k-1})} \quad (6.28)$$

In the condensation algorithm all distributions, including the prior are represented as samples. To show how sampling is done from weighted samples, a quantity, c_k^i is defined, called the cumulative probability for sample i at instant step k . c_k^i is defined by

$$c_k^i = 0, c_k^i = c_k^{i-1} + \pi_k^i, i\{1, \dots, N-1\} \quad (6.29)$$

being c_k^i obtained from the accumulating samples probabilities. The samples set S_k is formed from prior set S'_{k-1} by generating a random number r between the interval $[0, 1]$ and finding the smallest index i such $c_k^i > r$. New samples are formed from previous samples by picking those with higher probability (Figure 6.7).

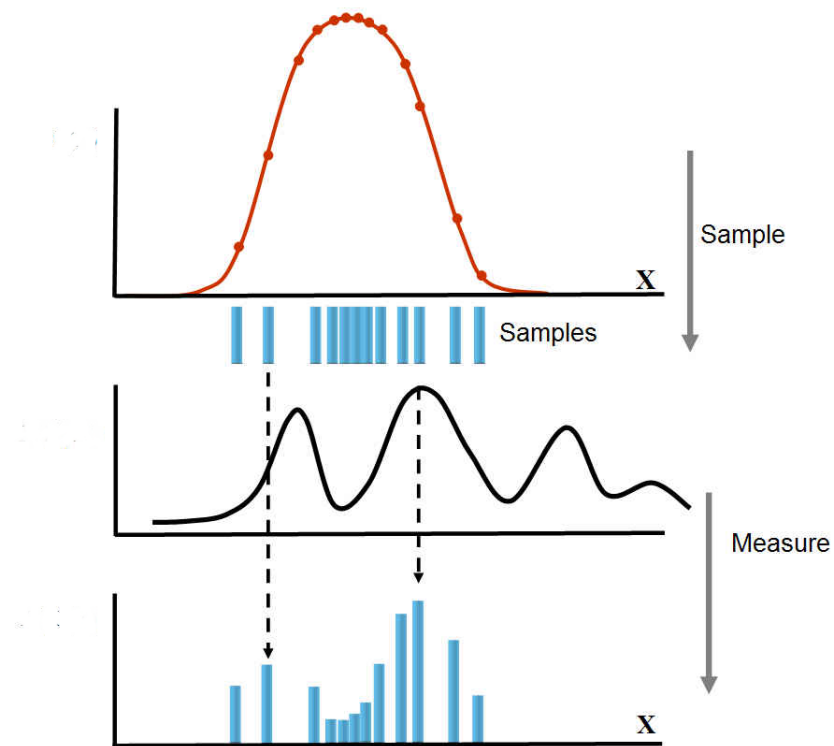


Figure 6.6: The use of sampling to estimate the a posteriori conditional probability (represented by the blue bars).

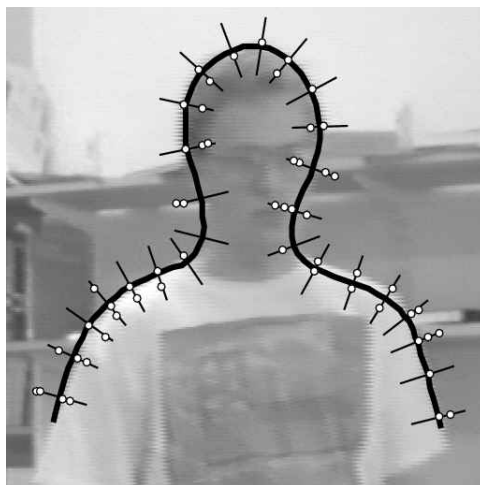


Figure 6.7: The object representation and measurement process for active contours. Copyright M. Isard, Oxford University.

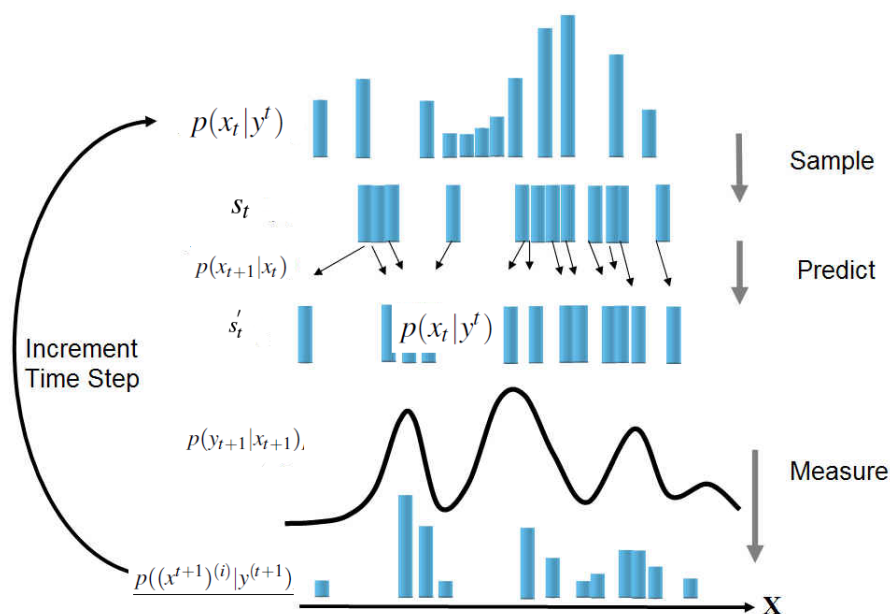


Figure 6.8: The full predict-correct cycle of the condensation algorithm.

Figure 6.8 resumes the full cycle of the condensation algorithm.

The complete description of the condensation algorithm can be found in Isard [92].

The likelihood is based on the Bhattacharya distance [93] between two RGB histograms defined as

$$\begin{aligned}
 Bhatt(w_1, w_2) = & \frac{1}{8} (\mathbf{m}_2 - \mathbf{m}_1)^T \left(\frac{\mathbf{C}_1 - \mathbf{C}_2}{2} \right)^{-1} (\mathbf{m}_2 - \mathbf{m}_1) \\
 & + \frac{1}{2} \log \left(\frac{\det \left(\frac{\mathbf{C}_1 - \mathbf{C}_2}{2} \right)}{\sqrt{\det(\mathbf{C}_1) \cdot \det(\mathbf{C}_2)}} \right)
 \end{aligned} \tag{6.30}$$

where w_1 and w_2 are the two-class problems normally distributed mean \mathbf{m}_i and covariance matrix \mathbf{C}_i .

The first term is equal to zero when both have same mean value and the second term is identical to zero when both have same covariance matrix. When the first term dominates, the classes are separable by their means, and separated by their covariance difference when the second term dominates.

6.5.1 Results

The video of a Navy vessel taken by an UAV during the REP exercise (Rapid Environmental Picture) annually organized by the Portuguese Navy and FEUP, was used to evaluate our technique. The Figure 6.9 shows the engaging process defined by the user and the state of the tracking.

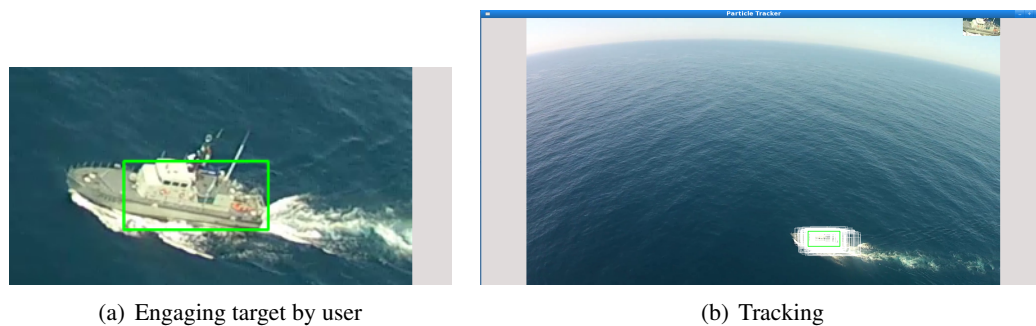


Figure 6.9: Engage of the object by user and correspondent tracking.

The partial occlusions are a problem that all particle filters must deal in a robust manner. In Figure 6.10 presents the occlusion of the object in a track due to the fact that is out of the view of the camera, and later recovery of the track even with a strong change in the heading of the UAV that has not been incorporated in the tracking.

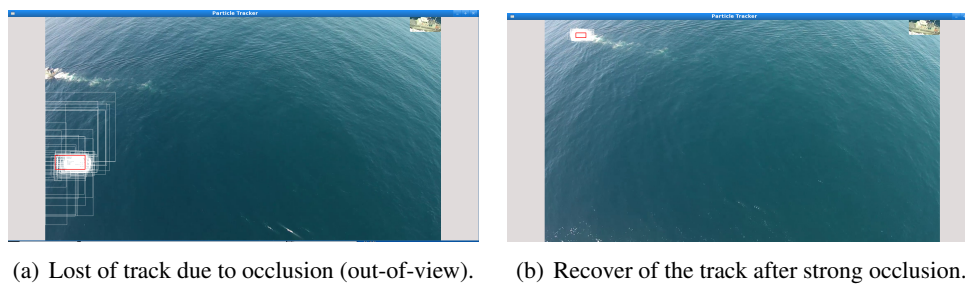


Figure 6.10: Tracking robustness to occlusion.

Further developments of this algorithm will provide the tracking of multiple targets at the same time. But initially is still need to address the engaging of the target (auto-detection) and refresh of the sample image. Several techniques are available, such as Back-Ground subtraction [94], Mean-Shift segmentation techniques [95], and the assignment of objects using the Mahalanobis distance [96] or the Hungarian method [97] avoiding use the current Bhattacharya distance [93] implementation that may perform badly if the objects to track are very similar on their RGB histograms. One of the challenges is to segment correctly facing the sun reflection on the water. In terms of the algorithm, this can be interpreted as a object of interest to be tracked or eventually dissimulate the object to be detected over the white spot. The engaging mechanism must be robust to the situations and extract the object in a very irregular background. State of the vehicle must be also included in the model of tracking to ensure that change of course does not affect the tracking performance.

Chapter 7

Cooperation

This chapter presents the cooperation techniques based on auctions and how multi-agents are controlled.

7.1 Introduction

Cooperative group behavior implies that participants in the group share a common objective and act according to a defined mutual interest. Effective cooperation often requires that individuals coordinate their actions. Coordination can be expressed in many forms ranging from staying out of each others way to directly assisting another individual in a shared task. In general, group cooperation is facilitated by coordinating the actions of individuals. However, each individual may not necessarily need to cooperate directly with every other participants in the present group to effect group behavior. For example, fish in schooling behavior react only to other fish that are nearby. This behavior can be denominated has *Local Coordination*. On the other extreme is the *Global Coordination*, that normally takes the configuration of centralized coordination or distributed with policies on the of information exchange and decisions made with the active participants to pursue a common goal.

7.2 Task Allocation

Simple approaches to handle partially or complete communication failures remove the faulty agent from the mission operation. These approaches solves the conflicts that may occur during communication failures. Other approach is to consider the agents as non-cooperative, applying a simple greedy strategy to allocate themselves to the nearest target. In these two approaches the overall performance of the mission can be severely affected by the lack of one agent or by auto allocation of several UAV's to the same target, resulting in interference among the agents tasks. To increase the overall performance of the mission, depending of the type of faults manifested, the agent becomes a non-cooperative agent. The non-cooperative faulty agent is allowed to perform

the task, but in a very restricted manner, giving priority to the healthier ones if they detect and assign the target in consideration.

7.2.1 Non Faulty Agents

For the task allocation, auction mechanism are well suited for cooperation among the agents. The main reason for the use of auctions, is their scalability and easy implementation ([98, 99, 100]). The behaviors carried out by faulty agents are completely independent from the healthier ones, and result from the exclusion in the coordination stage. The auction is triggered by the detecting agent, broadcasting the bid and receive the responses of bidding from other healthier agents in range, Figure 7.1. When the agent A_i detects a target T_k , it broadcasts the availability of the target as

$$P(i) = (A_i, T_k, J_k^i). \quad (7.1)$$

The cost of the assignment $J_k^i \in \{\gamma_i, \infty\}$. The value of J_k^i represent the initial minimum cost of the target that can be constructed with distance metrics and other relevant parameters. If agent A_i is free then, $J_k^i = \gamma_k^i$, where γ_k^i represents the distance between A_i and target T_k . When A_i is occupied with a target, then $J_k^i = \infty$. The agent that detects the target starts the auction. The agents receiving the broadcast information in the neighborhood of $A_i \in N(A_i)$ or by the centralized system will place a bid of the form Q_j , if they are not assigned to any other target and $J_k^j \leq J_k^i$

$$Q_j = (A_j, A_i, T_k, J_k^j). \quad (7.2)$$

The bid is broadcasted back to the auctioneer agent A_i as show in Figure 7.1 or the assignment result communicated to the agent. The auctioneer agent or the centralized system collects the bids and elects the agent with the lowest cost (considering the metrics defined) as the winner. The result of the decision is reported to the agents and that save the target id and the assigned agent on their table. The winner agent will assign to the target and move towards him while the non winner agents will continue to search (see Figure 7.1) or continue tracking if already assigned to a target.

Due to limited communication range, its assumed that the maximum communication range is $r_c > 4r_s$ and the probabilities of communication success following a Gaussian Distribution described in chapter 4. The rules for the auction be performed are:

- Agent $A_j \in N(A_i)$ will participate on the bidding if it is a free agent with full capabilities and is on the range.
- If the auctioneer has more than one agent as the lower cost bid, then, it will select the agent that has the higher id number.

The triggering of auctioning of a target and the assignment is instantaneous after detection, and the agents have limited communication ranges, therefore, different agents can detect the same target at various instants of time and trigger the auction mechanism. To avoid such auctioning, if agent A_i auctions the target at time t and target has been already assigned to other agent A_j at time $t' < t$, then A_j can revoke a invalid auction flag if the cost is greater, $J_{k(1)}^j > J_{k(2)}^i$ and if present

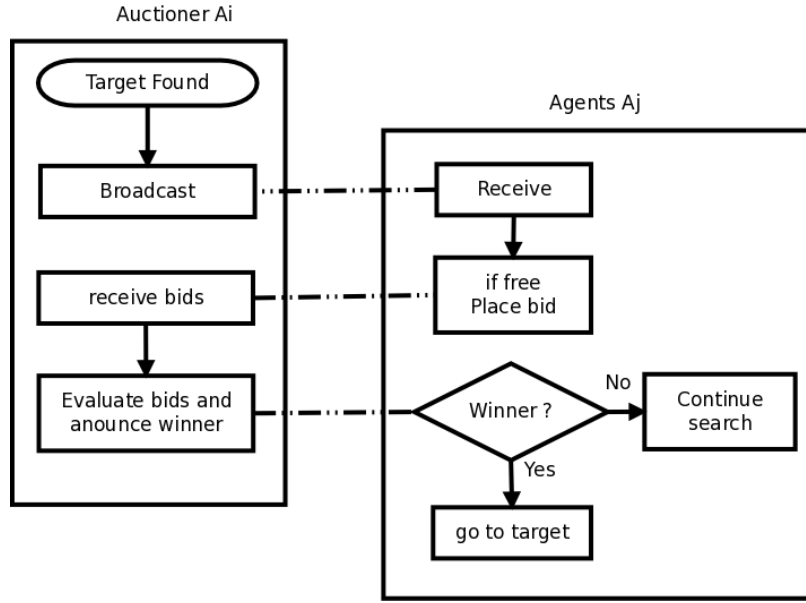


Figure 7.1: The decision process of agent A_i which is the auctioneer and the decision process of agent A_j who is a bidder

on the internal table as assigned, allowing the use of the combined observations gathered by both agents. This allows the exchange of agents in track of a target, considering always the one that has lower cost to the target may be the designated one. Using the above rules and the auction process the agents perform search and tracking missions of unknown number of targets. If a target is lost, the agent A_t reports that situation and becomes a free agent, performing search maneuvers over the last good known position and accepting bids. If detects again triggers the auction mechanism to assign a agent to that target.

7.2.1.1 Explicit Cooperation

When on track, a UAV can be affected by occlusions from obstacles during tracking. Taking limited sensing range into account, is not possible to pursue continuous tracking of the detected targets by one single UAV. Hence, by using the auction mechanism, the strategy comes up to explicit asking for cooperation among the free agents $A_j \in A_i \subset \{A_i \in \text{dist}\{A_t, A_i\} < \text{Max}_{\text{range}}\}$ that have the minimum distance to A_t in order to assist him. After finding a free agent, the strategy consist of having the recruited agent A_j performing circular patterns over the area of target that A_t has in track. To accomplish this, the center radius of the circular pattern is located on half the distance between the target and the UAV A_t position. When agent A_t cost $J_{k(1)}^j$ over T_k is higher than $J_{k(2)}^j$ in a certain predefined threshold, the recruited agent becomes the assigned A_t if the the cost $J_{k(2)}^j$ is the lower one to that target, being A_t de-committed from the track of T_k , and becoming a free agent. The collision avoidance controller can use this feature, triggering the explicit cooperation mechanism to minimize the possibility of occlusion to the target T_k and

avoiding losing of the track. This mechanism can be defined as a master-slave enabling the track of a high-valuable target if that behavior is enabled.

7.3 Implementation

The results of the implementation on shows the auction being made using three targets with three UAV's and the process of voting and assignment target to the UAV's.

```

The perceived value of targets (colluns) by each bidder (Agent) (rows):
3 3 6
2 5 3
8 2 7
Bets(in epsilons) : 0 0 1
Bidder           : 0 0 1
Left todo       : 0 2 3
-----
Bets(in epsilons) : 0 1 1
Bidder           : 0 2 1
Left todo       : 0 0 3
-----
Bets(in epsilons) : 1 1 1
Bidder           : 3 2 1
Left todo       : 0 0 0
-----
Assignment Cost  : 19.248
Happiness        : 1 1 1
Calc took       : 2.560 ms

ans =

     3     2     1

```

Figure 7.2: Capture from *MATLAB* command window showing the results of the auction assignment of UAV's to targets.

The computation time is cheap compared to the time involved in the collection and receiving information among all the agents that are involved directly. We should take in account that this mechanism is initiated when new targets are found or where a specific constrain situation raises. At present, the different communication and UAV failures were not modeled, but can be easily accommodated. More parameters can be used to build the target cost, namely those that take in account fuel consumption and sensing capabilities. For the moment the Dubins distance is considered as cost.

Chapter 8

Overview of results obtained and analysis

This section provides the results of the developed algorithms described in previous chapters and analysis of their performance. In order to achieve a comprehensive analysis, the results are described in an incremental fashion from simple isolated tasks to cooperative tracking.

8.1 Simulated experimental plan

Taking in account the need of analysis on the performance of the several mechanisms developed in this thesis, a set of experiments were defined in order to extract some valuable measures and establish a causal relation of the identified limitations in the overall performance of the system. To conduct this, Table 8.1 identifies these experiments as well the main objective and relevant parameters.

Table 8.1: List of experiments to be conduct.

Experiment Case	Objective	Number UAV's	Number targets
1	Evaluate Tracking and path follow	1	1
2	Master-Slave on Visibility time	2	1
3	Evaluate Building Density	1	1
4	Evaluate Speed Ratio	1	1
5	Evaluate impact of UAV number	N	1
6	Evaluate Cooperation different N_u/N_v ratios(1,2)	N	N

8.1.1 Simple Target Following

One initially attained results is the target following by one UAV. The Figure 8.1 presents the result of the target following with estimation using only random controls on the prediction of the

target. This situation enables to follow targets that present very extreme dynamics such as ball bouncing in a wall for example.

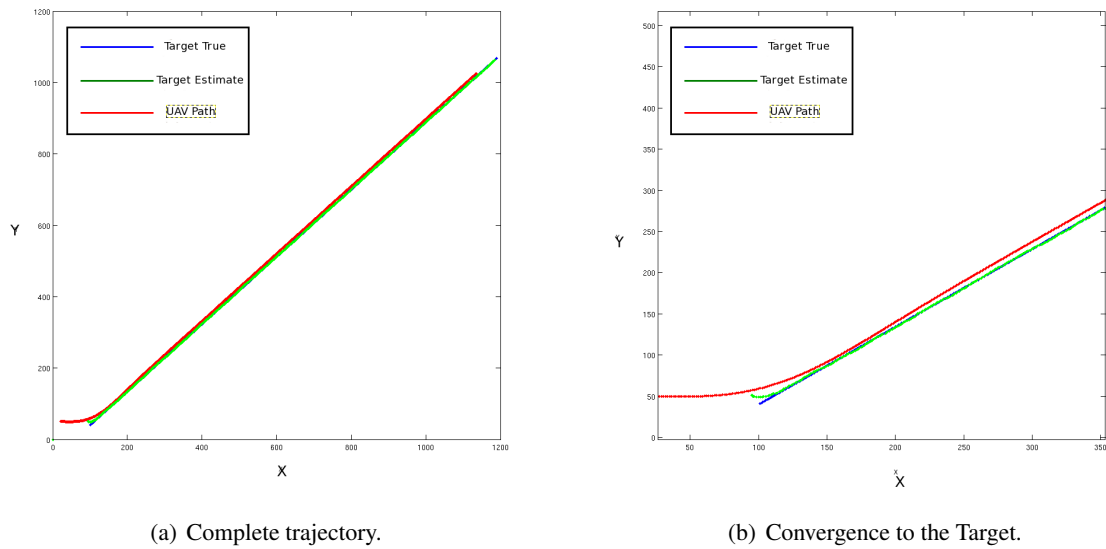


Figure 8.1: UAV Tracking one target with random controls on the prediction stage.

A simulation was made regarding the tracking and loitering over the target when this one is not seen for a substantial period of time. The objective was to measure the correct change of modes between following and loitering over the last good know position of the target. This maneuver enables the UAV to engage the lost target again. Figure 8.2 shows the paths taken by the UAV, the ground target and the estimation of the position of the target.

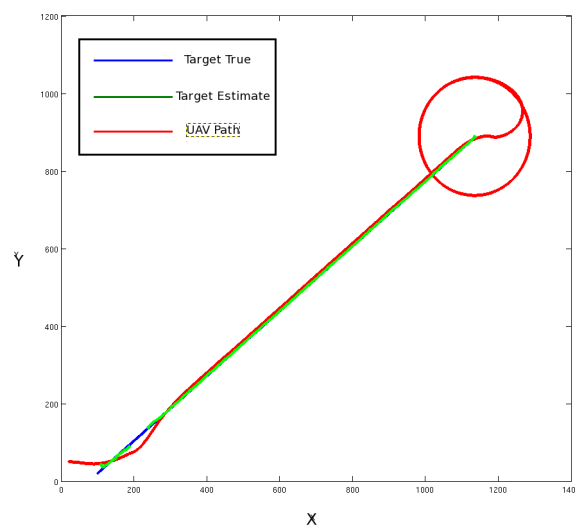


Figure 8.2: UAV trajectory (RED), Target true position (BLUE) and Target estimation (GREEN).

The other result is the target following with estimation taking the actual dynamics of the target into account. The estimation error was of 1.5 meters, taking in account Gaussian noise introduced on the observations. The result shows that for linear target trajectories, the UAV can predict the target location with reasonably accuracy. Figure 8.3 presents the results on the target following for this situation.

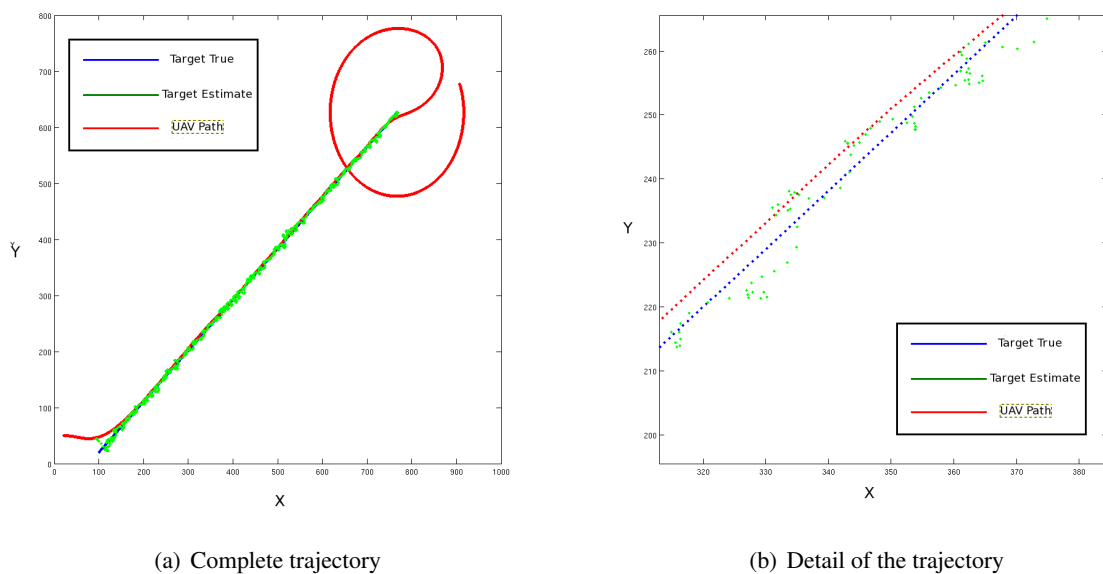


Figure 8.3: UAV Tracking one target with dynamic controls on the prediction stage.

The Figure 8.4 show the trajectory taken by the UAV, target and estimation joined with the high density building obstacles.

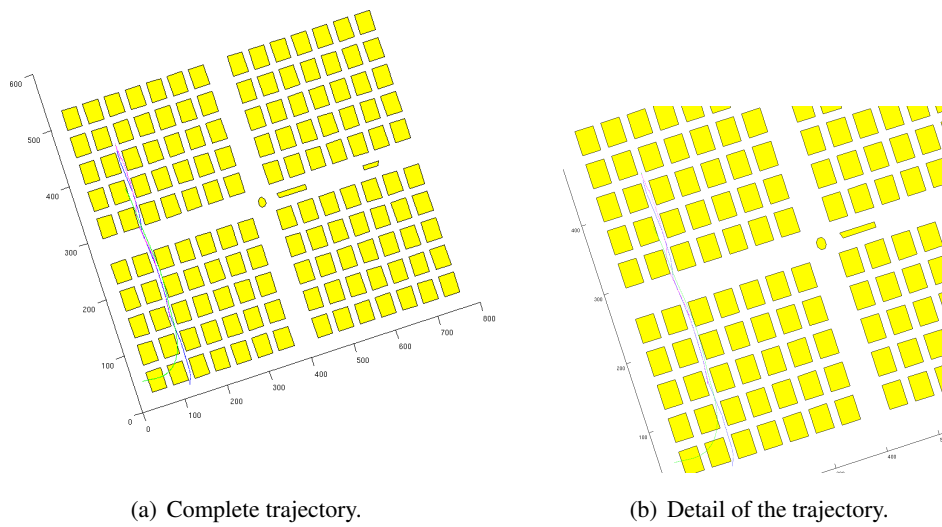


Figure 8.4: UAV Tracking one target over the buildings obstacles.

Long run for a overall view of the performance is show in Figure 8.5.

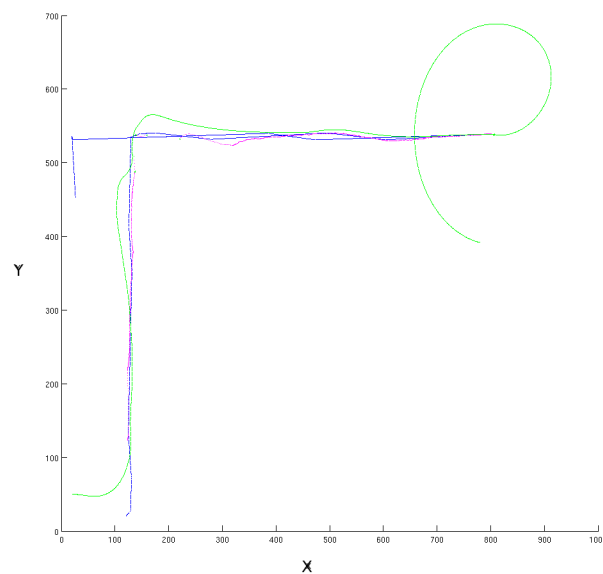


Figure 8.5: Long run test. UAV trajectory (GREEN), Target true position (BLUE) and Target estimation (RED).

8.1.1.1 Overall Performance of Simple Target Following

In order to measure the performance of the algorithm and obtain valuable results for the development of higher level controllers, we derive:

- **Maximum invisibility time:** The maximum invisible time will give an indication of the effectiveness of the tracking strategy.

- **Response:** Taking in account the worst case scenario $UAV_z / MediumHeight_{buildings} \approx 0.9$ and obstacle density 0.6, and the defined value of 17 time steps with target speed of 2m/s resulting in approx 35 meters in pure prediction with immediately recover by the filter. This parameter was determined by removing the target from observations after engaged and quantify the amount of time that target is predicted until being completely removed from the track with the defined parameters. With the defined obstacle density, we quantify the maximum invisible time created and this value was way up the 17 steps barrier, reinforcing the need of cooperation to accomplish continuous track without losses.

- **Variation of parameters:** What is the effect of speed ratio $\frac{V_t}{V_u} < 1$, sensing range, obstacle density on the tracking performance.

- **Response:** The effect of speed ratio is namely the need for the UAV to loiter time to time to cope with the speed of the target. Taking in account the non-gimbal camera, the radius must be adequate to perform an update of the target in less than 17 steps after the spreading of particles are very large. The sensing range affects directly the tracking namely in the change of dynamics of the target. We predict the turning rate, but this value needs to be carefully under-estimated due to the sensibility of the angle on trajectory vs UAV dynamics resulting in a long time to recover the view of the target. The obstacle density takes a big part on the invisible time, namely the ratio of height and density of the buildings vs UAV altitude. The angle of the projected footprint affects very much this, due to the fact that the UAV may take a shortcut to converge with the target, the building may present a strong occlusion over that period. After the 17 step barrier, the target is removed from tracking, but it can be reengaged due to the fact that the footprint over the last known position catches again. But we don't have 100% sure it's the same target due to the lack of extra features to describe it.

8.1.2 Explicit Cooperation

The explicit cooperation has the purpose to assign two UAV's to one target in a Master-Slave configuration. This enables the possibility of tracking a high valuable target, presenting some advantages when the number of vehicles available is high vs a target that presents strong dynamics, making full use of multiple observations that occur during tracking and minimize occlusion time. The strategy comes to assign the second UAV to make loiter maneuvers over the set UAV (Master) and target. Figure 8.6 shows the Master UAV tracking the target while being assisted by a second UAV executing loiter maneuver over the area.

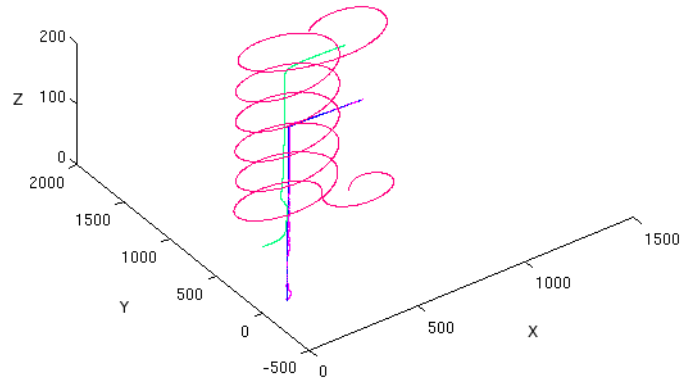


Figure 8.6: Master-Slave configuration tracking a high valuable target. UAV one (GREEN), UAV two (MAGENTA), Target True (BLUE), Target Estimation (PINK).

8.1.3 Impact of Obstacle Density on Simple Target Following

The objective of this simulation is to quantify the impact of the increase of the obstacle density on the observation time of the target by one UAV by the graph representation on Figure 8.7

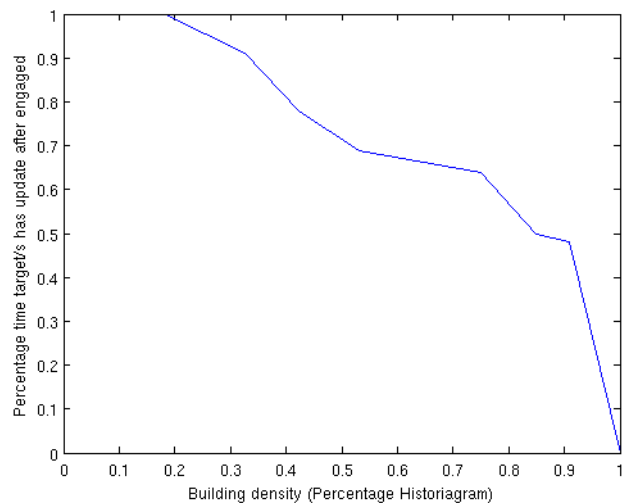


Figure 8.7: Impact of increasing the obstacle density on the observation time.

The graph shows the impact on the observation time of the target. This is due to the fact that buildings present occlusions and no observations can be made during the period that the UAV is traveling to render with the target. Large values of density result also in the lost of the track due to the fact that the 17 steps time boundary is achieved and the tracking system removes the target

from tracking. But at the same time it detects again, but with no correlation with the last target tracked. One other problem is that a high density makes that the target get stuck in one place, being almost a situation of the zero speed. This situation leads to the UAV loitering over the area increasing the visibility time of the target.

8.1.4 Impact of Speed Ratio on Simple Target Following

The speed ratio V_t/V_u has a direct impact on the time that a target is tracked, described in subsection 8.1.1.1. Figure 8.8 shows that a ratio above one, the UAV after some time is no able to follow the target due to the difference of speed among them. But for the ratio below one, the UAV executes loitering maneuvers that decrease the invisible time of the target.

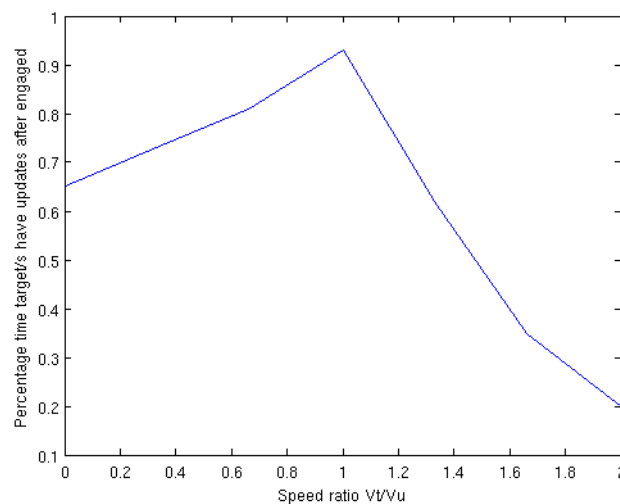


Figure 8.8: Impact of speed ratio with fixed obstacle density.

8.1.5 One Target Following by several UAVs

The objective of this simulation is to quantify the impact of using several UAV's in dense areas that present occlusions to the track a single target. As show before, the building density has a direct impact on the availability of the UAV to gather observation due to direct occlusion or by the need to execute evasive maneuvers. For a fixed obstacle density, is quantified the impact of using more that one UAV on the visible time of the target after engaged. Figure 8.9 show the result of that particular scenario where the number of UAV's present affects directly the number of observations that are made to the target.

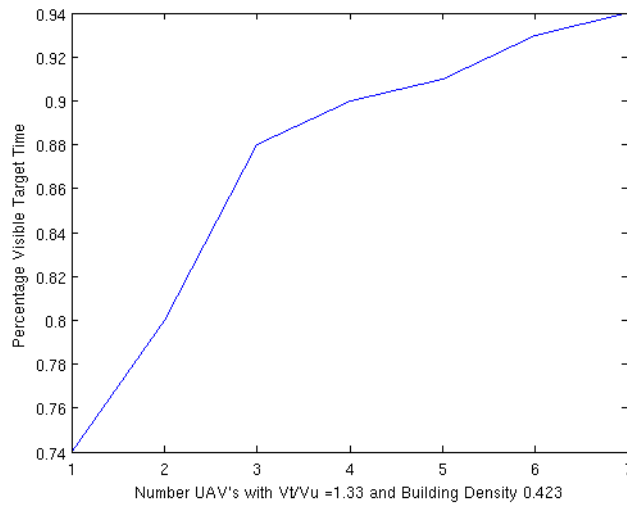


Figure 8.9: Effort of increasing the number of UAV's for a single target with explicit cooperation disabled.

Activating the explicit cooperation, a free UAV is recruited to help the assigned UAV to track the target. That result in a dramatically decrease of the invisible time of the target represented in Figure 8.10, while the recruited UAV's stays in a loiter maneuver over the area of the assigned UAV and target.

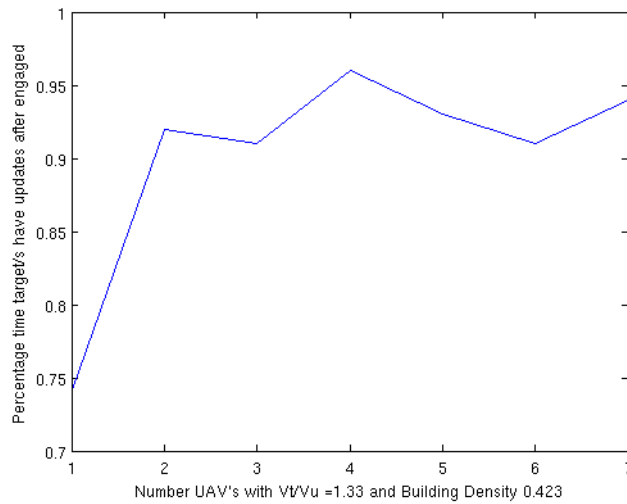
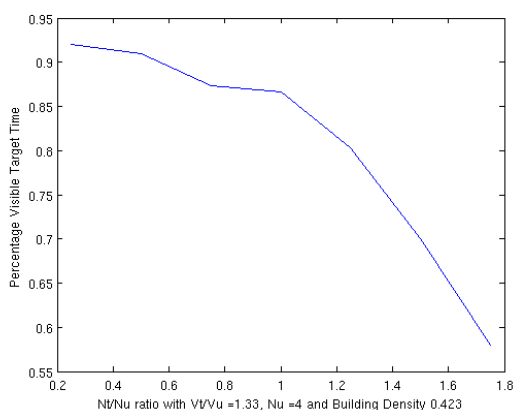


Figure 8.10: Effort of increasing the number of UAV's for a single target with explicit cooperation enabled.

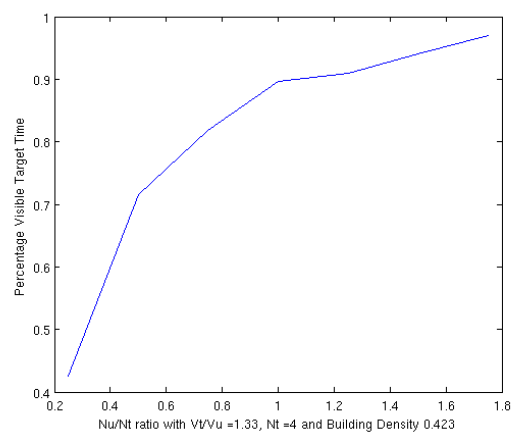
The increase of the UAV number affects in some manner due to the fact that some observations can be made by them also, but they become secondary because the two UAV configuration almost grant a good performance.

8.1.6 Multiple Targets Following by several UAVs

The objective of this simulation is to present the performance of the overall system in the track of unknown number of ground targets. With the generalization of the coordination controller, it is possible to find the best strategy to achieve the objectives functions described in chapter 3. Taking in account the impact of the use of multiple UAV's to follow one target, the ratio of Targets-UAV's N_u/N_t has a impact on the amount of time that targets are seen. Figure 8.11 shows different scenarios where the ratio between the UAV's and targets is different with a fixed terrain density of 0.423. In the ratio $N_t/N_u < 1$, any constrain on terrain can cause the assigned UAV to lose the track, but taking in account the occupancy of the terrain by other UAV's on tracking mission and the cooperation, the target can be seen by other UAV's and the tracking systems continues to have updates for that particular target what ensures that the UAV assigned to that target can continue its pursuit even without seen the target.



(a) Effort on the increase of target's for fixed UAV number (4).



(b) Effort on the increase of UAVs for fixed target number (4).

Figure 8.11: Effect of the increase of target for fixed UAV number *vs* increase of the UAV number for fixed target number.

Ratio $N_t/N_u > 1$, the assigned target can be tracked directly, but the left targets can be tracked in some period of time when the UAV's in track see that particular target. When a new target is detected, the cooperation mechanism is triggered and by auction see if any UAV is available to assign to the target, but all are already assigned. The best effort is to continue to the current assignment and track the target when possible. If the density of UAV's on terrain is high, this target can be tracked without being assigned to any UAV in particular increasing the time that a set of targets is observed.

Ratio $N_u/N_t > 1$ the most relevant impact is the initial dispersion of the targets *vs* vehicles initial position. This can increase the time of initial detection of a target due to the fact that the

UAV has to take more time to travel to pass in that point in the search maneuver and is dependent of the targets being assigned meanwhile, but after all targets engaged they are tracked.

Chapter 9

Other application areas

The applications of the track and cooperation can arise in numerous situations. One scenario is the forest fire patrol [4], (Figure 9.1), in order to have a continuous information of the evolution of the fire front. Another application is the singular target pursuit over dense areas where the target is considered high valuable to the mission.

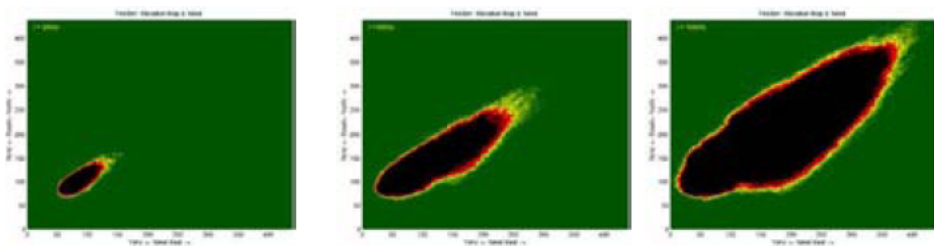


Figure 9.1: Forest fire spreading example. Image from [4].

Several surveillance task demands a continuous surveillance [101] with high frequency passing over the points of coverage enabling to patrol sea operations [102], track of dolphins and tuna from fishing boats, (Figure 9.2), ensuring that the fish you buy in supermarkets is dolphin-safe. Also collection of evidences of illegal activities occurring over the specified area of operation or Whale-ship Collision Avoidance advise for traveling boats



Figure 9.2: ScanEagle's UAV platform in sea operation.

Oceanography sciences can benefit from the use of Multi-UAV systems, such as Pollution, (oil spills, noise, harmful algal blooms, alien species, and disease organisms). Ocean Acidification,

enabling its early detection while supplies vital information to conduct other monitoring tasks with more precise sensor using other genre of vehicles such as AUV's.

Terrain mapping can be also addressed with the use of Multi-UAV's systems. With the use of LiDAR equipped UAV mapping terrain topography for digital soil mapping, or even using simple low cost onboard cameras is possible to create a high detailed terrain map from a area of interest, merging the images collected, Figure 9.3 using computer vision algorithms such as image mosaicing [5] and increase the definition of the maps available to public use.

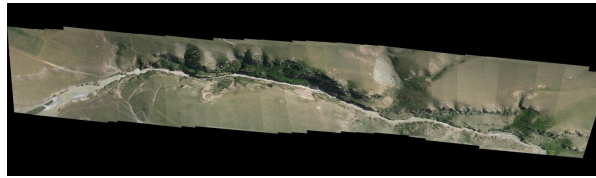
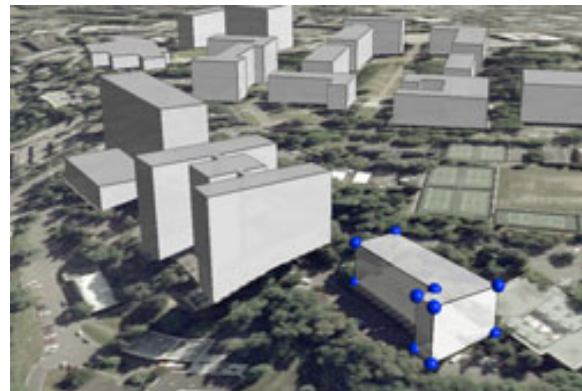


Figure 9.3: Image mosaicing of terrain view. Image from [5].

If elevation is available and georeferences, the collected data can be integrated into open source frameworks such as NASA World Wind [103]. Figure 9.4.



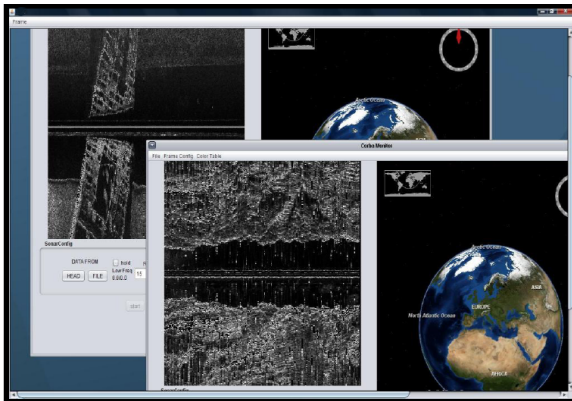
(a) Nasa World Wind Features



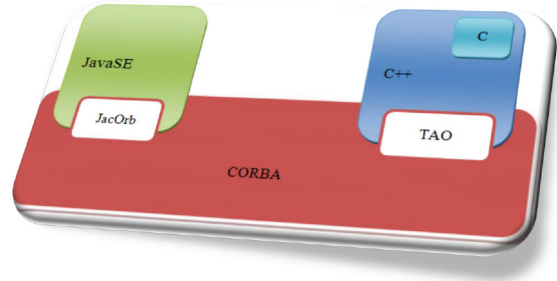
(b) Runtime construction of 3D shapes using the World Wind Java controlled airspace toolkit

Figure 9.4: Nasa World Wind examples.

This NASA World Wind framework enables the representation the data in a easy, friendly user interface, allowing the use of annotations on the map such as balloons attached to the position on the Globe with the description of the feature of interest. The author of this thesis as already worked with this framework, namely the integration in real time of images/data points collected from several Side-Scan sensor deployed in Autonomous Surface Vehicles (ASV), using distributed mechanism for the communications between the several vehicles and the framework on the station base, enabling the display of the under-water collected data on the fly or offline for detailed analysis. Figure 9.5 shows the results of the initial framework developed.



(a) Framework GUI developed.



(b) Code Architecture used.

Figure 9.5: Nasa World Wind Framework.

This framework shows the full potential to the final user of the data that might be gathered using multi-vehicle systems, heterogeneous or not. This kind of frameworks reduce the distance between the research and development of vehicles and their use by non-trained people to conduct data analysis over the information gathered.

Chapter 10

Conclusions and Further Work

In the last chapter, a summary of the main contributions of this work are made, and some directions for further work are provided.

10.1 Main Contributions

The work in this thesis can be described as the design, implementation and analysis of a cooperative persistent target tracking system for surveillance missions. The contributions of this thesis can be divided into four main contributions.

Simulation enabling the representation of the real constraints that urban scenarios may raise to the vehicles during mission. Also enables the development and test many other kinds of configurations such as the mixing of aerial agents with terrain agents, providing support for the developed algorithms.

Tracking mechanism enables singular or cooperative multi-tracking of non-Gaussian distributions among the set of the agents deployed in the field in a easy manner, making full use of the available observations from different perspectives to improve estimation quality and at same time handling with erroneous measurements in a robust manner. The flexibility enables to be used on centralized or distributed approaches without any considerable changes on main implementation.

Path Following and Planning provides the basic control laws for the vehicles and plan paths to the goal.

Auction mechanisms enables dynamic assignment of vehicles to targets in a dynamical way, avoiding conflicts. It has the flexibility to incorporate several parameters relevant to the mission and obtain the best possible solution/configuration on the assignment process.

10.2 Further Work

We hope that the work resulting from this thesis opens a new paradigm for persistent surveillance. Currently, there are some niche areas that can be improved. Together with those points, we have included some suggestions towards implementation and deployment in real platforms.

Improve collision avoidance Using the laser range sensor available, implement more robust avoidance algorithms that minimize the possibility of lost of track due to occlusions on situations where only the assigned UAV is available to gather observations on the target, and explore the system proposed by Saunders in [104].

Increase auction parameters consideration The cooperation mechanism taking variables, such as fuel consumption, communication fails, range limitations and non-heterogeneity among vehicles. An higher level mechanism should be developed in order to ensure that the mission objectives are accomplish and the UAV's configuration is the optimal to maximize target visibility and detection.

Tracking implementation in OpenCV and C++ Extrapolate the work of multi-target and multi-observation to this framework. Future considerations on the development need to be proper addressed, namely on the mechanization of the observations such has the rotation matrix $R_B^W R_T^B$ from the frame of the camera to the world using the UAV state information. Detailed analysis has to be made on the mechanization and determination of these parameters to avoid large errors on the observations and consequentially degradation of quality of the estimation.

OpenCV Segmentation Further developments need to be done in the detection (segmentation) of the object. To achieve this, some considerations need to be made regarding the scenario of operation (water background, terrain background and others) to select and develop the best segmentation algorithm or flexible one supplying the tracking algorithm with this information. It's suggested that for the initial steps construct a steady multi-camera setup that replicates the UAV state conditions refereed and test several items after flying.

10.3 Final Remark

A persistent surveillance system was modeled and implemented in *MATLAB*. Some initial implementation were made on OpenCV to create the terrain for the complete implementation on real platforms. The implementation of the tracking mechanism reveals a great potential namely in the use of multi-observations gathered by other vehicles. This amplifies the potentiality of the cooperation/assignment algorithms, while relaxing the complexity of its implementation. The result shows that with a limited number of vehicles is possible to achieve a good performance regarding the visibility time of the ground targets.

References

- [1] David Rathbun, Ph. D, Sean Kragelund, and Anawat Pongpunwattana. An evolution based path planning algorithm for autonomous motion of a uav through uncertain environments. In *Algebra Universalis*, pages 551–607, 2002.
- [2] Steven M. Lavalle. Rapidly-exploring random trees: A new tool for path planning. Technical report, 1998.
- [3] Durant White. Multiple sensor tracking. *Robotics*, 2004.
- [4] D.W. Casbeer, R.W. Beard, T.W. McLain, Sai-Ming Li, and R.K. Mehra. Forest fire monitoring with multiple small uavs. In *American Control Conference, 2005. Proceedings of the 2005*, pages 3530–3535 vol. 5, 2005. doi:10.1109/ACC.2005.1470520.
- [5] Q. Liu, W. Liu, L. Zou, J. Wang, and Y. Liu. A new approach to fast mosaic uav images. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XXXVIII-1/C22:271–276, 2011. URL: <http://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XXXVIII-1-C22/271/2011/>, <http://dx.doi.org/10.5194/isprsarchives-XXXVIII-1-C22-271-2011> doi:10.5194/isprsarchives-XXXVIII-1-C22-271-2011.
- [6] Joel G. Manathara, P. B. Sujit, and Randal W. Beard. Multiple uav coalitions for a search and prosecute mission. *J. Intell. Robotics Syst.*, 62(1):125–158, April 2011. URL: <http://dx.doi.org/10.1007/s10846-010-9439-2>, <http://dx.doi.org/10.1007/s10846-010-9439-2> doi:10.1007/s10846-010-9439-2.
- [7] P. B. Sujit and Randy Beard. Multiple uav exploration of an unknown region. *Annals of Mathematics and Artificial Intelligence*, 52(2-4):335–366, April 2008. URL: <http://dx.doi.org/10.1007/s10472-009-9128-7>, <http://dx.doi.org/10.1007/s10472-009-9128-7> doi:10.1007/s10472-009-9128-7.
- [8] Joel George, P. B. Sujit, and João B. Sousa. Search strategies for multiple uav search and destroy missions. *Journal of Intelligent and Robotic Systems*, 61(1-4):355–367, 2011.
- [9] P. B. Sujit and S. Saripalli. An empirical evaluation of co-ordination strategies for an auv and uav. *J. Intell. Robotics Syst.*, 70(1-4):373–384, April 2013. URL: <http://dx.doi.org/10.1007/s10846-012-9728-z>, <http://dx.doi.org/10.1007/s10846-012-9728-z> doi:10.1007/s10846-012-9728-z.
- [10] C. Hue, J.-P. Le Cadre, and P. Perez. Tracking multiple objects with particle filtering. *Aerospace and Electronic Systems, IEEE Transactions on*, 38(3):791–812, 2002.

- [11] Stuart M Adams and Carol J Friedland. A survey of unmanned aerial vehicle (uav) usage for imagery collection in disaster research and management. In *Proceedings of the Ninth International Workshop on Remote Sensing for Disaster Response*, 2012.
- [12] P. B. Sujit and R. Beard. Distributed sequential auctions for multiple uav task allocation. In *American Control Conference, 2007. ACC '07*, pages 3955–3960, 2007. doi:10.1109/ACC.2007.4282558.
- [13] U. Zengin and A. Dogan. Real-time target tracking for autonomous uavs in adversarial environments: A gradient search algorithm. *Robotics, IEEE Transactions on*, 23(2):294–307, 2007. doi:10.1109/TRO.2006.889490.
- [14] JamesF. Smith III and ThanhVuH. Nguyen. Fuzzy logic based uav allocation and coordination. In JuanAndrade Cetto, Jean-Louis Ferrier, José Miguel Costa dias Pereira, and Joaquim Filipe, editors, *Informatics in Control Automation and Robotics*, volume 15 of *Lecture Notes Electrical Engineering*, pages 81–94. Springer Berlin Heidelberg, 2008. URL: http://dx.doi.org/10.1007/978-3-540-79142-3_8, http://dx.doi.org/10.1007/978-3-540-79142-3_8 doi:10.1007/978-3-540-79142-3_8.
- [15] J. Sousa, T. Simsek, and P. Varaiya. Task planning and execution for uav teams. In *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, volume 4, pages 3804–3810 Vol.4, 2004. doi:10.1109/CDC.2004.1429331.
- [16] Michael Negnevitsky. *Artificial Intelligence: A Guide To Intelligent Systems*, pages 24–. Pearson Education, 2005 - Computers, 2005.
- [17] Axel Niehaus and R.F. Stengel. An expert system for automated highway driving. *Control Systems, IEEE*, 11(3):53–61, 1991. doi:10.1109/37.75579.
- [18] J.C. Latombe. *Robot motion planning*. London: Kluwer Academic, 1991.
- [19] A. Dogan. Probabilistic approach in path planning for uavs. In *Intelligent Control. 2003 IEEE International Symposium on*, pages 608–613, 2003. doi:10.1109/ISIC.2003.1254706.
- [20] O. Takahashi and R.J. Schilling. Motion planning in a plane using generalized voronoi diagrams. *Robotics and Automation, IEEE Transactions on*, 5(2):143–150, 1989. doi:10.1109/70.88035.
- [21] Ellips Masehian and M. R. Amin-Naseri. A voronoi diagram-visibility graph-potential field compound algorithm for robot path planning. *J. Robot. Syst.*, 21(6):275–300, June 2004. URL: <http://dx.doi.org/10.1002/rob.v21:6>, <http://dx.doi.org/10.1002/rob.v21:6> doi:10.1002/rob.v21:6.
- [22] S. Martinez J. Cortes and F. Bullo. Robust rendezvous for mobile autonomous agents via proximity graphs in arbitrary dimensions. *IEEE Transactions on Automatic Control*, 2006.
- [23] Y. Saab and M. VanPutte. Shortest path planning on topographical maps. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 29(1):139–150, 1999. doi:10.1109/3468.736370.
- [24] H. I. Yang and Y. J. Zhao. Trajectory planning for autonomous aerospace vehicles amid known obstacles and conflicts. *Journal of Guidance, Control and Dynamics*, 2004.

- [25] Maria A. Carravilla Jose F. Oliveira. Investigação operacional exercicios. *Lectures Operational Research, chapter 8*, 2011.
- [26] Dave Ferguson and Anthony Stentz. Field d*: An interpolation-based path planner and replanner. In Sebastian Thrun, Rodney Brooks, and Hugh Durrant-Whyte, editors, *Robotics Research*, volume 28 of *Springer Tracts in Advanced Robotics*, chapter 22, pages 239–253. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007. URL: http://www.ri.cmu.edu/publication_view.html?pub_id=5122, doi:10.1007/978-3-540-48113-3_22.
- [27] M.A. Baumann, D.C. Dupuis, S. Leonard, E.A. Croft, and J.J. Little. Occlusion-free path planning with a probabilistic roadmap. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 2151–2156, 2008. doi:10.1109/IROS.2008.4651035.
- [28] Zheng Sun, David Hsu, Tingting Jiang, Hanna Kurniawati, and John H Reif. Narrow passage sampling for probabilistic roadmap planning. *Robotics, IEEE Transactions on*, 21(6):1105–1115, 2005.
- [29] Zhiye Lee and Xiong Chen. Path planning approach based on probabilistic roadmap for sensor based car-like robot in unknown environments. In *Systems, Man and Cybernetics, 2004 IEEE International Conference on*, volume 3, pages 2907–2912 vol.3, 2004. doi:10.1109/ICSMC.2004.1400774.
- [30] C. Lanzoni, A. Sanchez, and R. Zapata. Sensor-based motion planning for car-like mobile robots in unknown environments. In *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, volume 3, pages 4258–4263 vol.3, 2003. doi:10.1109/ROBOT.2003.1242258.
- [31] F. Lingelbach. Path planning using probabilistic cell decomposition. In *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, volume 1, pages 467–472 Vol.1, 2004. doi:10.1109/ROBOT.2004.1307193.
- [32] E. Horiuchi and K. Tani. A probabilistic approach to path planning with object boundary uncertainties. In *Advanced Robotics, 1991. 'Robots in Unstructured Environments', 91 ICAR., Fifth International Conference on*, pages 1698–1701 vol.2, 1991. doi:10.1109/ICAR.1991.240362.
- [33] R. Murphey S. Butenko and P. M. Pardalos. Cooperative control: Models, applications and algorithms. *Cooperative Control: Models, Applications and Algorithms, Kluwer Academic Publishers*, 2003.
- [34] R. Sharma. Locally efficient path planning in an uncertain, dynamic environment using a probabilistic model. *Robotics and Automation, IEEE Transactions on*, 8(1):105–110, 1992. doi:10.1109/70.127244.
- [35] I.K. Nikolos, K.P. Valavanis, N.C. Tsourveloudis, and A.N. Kostaras. Evolutionary algorithm based offline/online path planner for uav navigation. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 33(6):898–912, 2003. doi:10.1109/TSMCB.2002.804370.
- [36] C. Hocaoglu and A.C. Sanderson. Planning multiple paths with evolutionary speciation. *Evolutionary Computation, IEEE Transactions on*, 5(3):169–191, 2001. doi:10.1109/4235.930309.

- [37] R. Vaidyanathan, C. Hocaoglu, T.S. Prince, and R.D. Quinn. Evolutionary path planning for autonomous air vehicles using multi-resolution path representation. In *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, volume 1, pages 69–76 vol.1, 2001. doi:10.1109/IROS.2001.973338.
- [38] Dong Jia. Parallel evolutionary algorithms for uav path planning. In *in Proceedings of AIAA 1st Intelligent Systems Technical Conference*, 2004.
- [39] A. Alvarez, A. Caiti, and R. Onken. Evolutionary path planning for autonomous underwater vehicles in a variable ocean. *Oceanic Engineering, IEEE Journal of*, 29(2):418–429, 2004. doi:10.1109/JOE.2004.827837.
- [40] B.J. Capozzi and J. Vagners. Navigating annoying environments through evolution. In *Decision and Control, 2001. Proceedings of the 40th IEEE Conference on*, volume 1, pages 646–651 vol.1, 2001. doi:10.1109/.2001.980177.
- [41] Dave Ferguson, N. Kalra, and A. Stentz. Replanning with rrts. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 1243–1248, 2006. doi:10.1109/ROBOT.2006.1641879.
- [42] Anna Yershova, Léonard Jaillet, Thierry Siméon, and Steven M. LaValle. Dynamic-domain rrts: Efficient exploration by controlling the sampling domain. In *IN PROCEEDINGS IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION*, pages 3867–3872, 2005.
- [43] Randal W. Beard, Timothy W. McLain, and Michael Goodrich. Coordinated target assignment and intercept for unmanned air vehicles, 2002.
- [44] Feng-Li Lian and Richard Murray. Real-time trajectory generation for the cooperative path planning of multi-vehicle systems. In *Decision and Control, 2002, Proceedings of the 41st IEEE Conference on*, volume 4, pages 3766–3769. IEEE, 2002.
- [45] Henrique Ribeiro Delgado da Silva. Controlo de formacoes de veiculos aéreos não tripulados. *Dissertacao Mestrado, Instituto Superior Tecnico*, 2012.
- [46] J.S. Bellingham, M. Tillerson, M. Alighanbari, and J.P. How. Cooperative path planning for multiple uavs in dynamic and uncertain environments. In *Decision and Control, 2002, Proceedings of the 41st IEEE Conference on*, volume 3, pages 2816–2822 vol.3, 2002. doi:10.1109/CDC.2002.1184270.
- [47] M Bernadine Dias and Anthony Stentz. A comparative study between centralized, market-based, and behavioral multirobot coordination approaches. In *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, volume 3, pages 2279–2284. IEEE, 2003.
- [48] Tom Schouwenaars, Jonathan How, and Eric Feron. Decentralized cooperative trajectory planning of multiple aircraft with hard safety guarantees. In *Proceedings of the AIAA guidance, navigation and control conference*, pages 6820–6825, 2004.
- [49] Timothy W McLain and Randal W Beard. Cooperative path planning for timing-critical missions. In *American Control Conference, 2003. Proceedings of the 2003*, volume 1, pages 296–301. IEEE, 2003.

- [50] E. W.Frew and D. A. Lawrance. Cooperative stand-off tracking of moving target by a team of autonomous aircraft. in *Proceedings of the AIAA Guidance, Navigation and Control Conference and Exhibit*, 2005.
- [51] P.J. Shea, T. Zadra, D. Klamer, E. Frangione, R. Brouillard, and K. Kastella. Precision tracking of ground targets. In *Aerospace Conference Proceedings, 2000 IEEE*, volume 3, pages 473–482 vol.3, 2000. doi:10.1109/AERO.2000.879873.
- [52] J. K. Hedrick R. Sengupta. Strategies of path planning for a uav to track a ground vehicle. in *Proceedings of the Second Annual Symposium on Autonomous Intelligent Networks and Systems*, 2003.
- [53] A. Vaughn and X. Xiao. A vehicle following methodology for uav formations. in *Proceedings of the Fourth International Conference on Cooperative Control and Optimization*, 2003.
- [54] P. Theodorakopoulos and S. Lacroix. A strategy for tracking a ground target with a uav. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 1254–1259, 2008. doi:10.1109/IROS.2008.4650939.
- [55] T. Bhattacharya, A. Premji, T.J. Nohara, and Peter Weber. Evaluation of fast mht algorithms. In *Radar Conference, 1998. RADARCON 98. Proceedings of the 1998 IEEE*, pages 213–218, 1998. doi:10.1109/NRC.1998.678003.
- [56] Chee-Yee Chong, D. Garren, and T.P. Grayson. Ground target tracking—a historical perspective. In *Aerospace Conference Proceedings, 2000 IEEE*, volume 3, pages 433–448 vol.3, 2000. doi:10.1109/AERO.2000.879870.
- [57] M. Betke, D. E. Hirsh, A. Bagchi, N.I. Hristov, N.C. Makris, and T.H. Kunz. Tracking large variable numbers of objects in clutter. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1–8, 2007.
- [58] Michael R Anderberg. Cluster analysis for applications. Technical report, DTIC Document, 1973.
- [59] Samuel Blackman and Artech House. Design and analysis of modern tracking systems. *Boston, MA: Artech House*, 1999.
- [60] M. Sanjeev Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *Signal Processing, IEEE Transactions on*, 50(2):174–188, 2002. doi:10.1109/78.978374.
- [61] Michael Johannes and Nicholas Polson. Particle filtering. in *Proceedings of IEEE Aerospace Conference*, 2006.
- [62] M. Mallick, S. Maskell, T. Kirubarajan, and N. Gordon. Littoral tracking using particle filter. In *Information Fusion, 2002. Proceedings of the Fifth International Conference on*, volume 2, pages 935–942 vol.2, 2002. doi:10.1109/ICIF.2002.1020912.
- [63] M. Bozorg, E.M. Nebot, and H.F. Durrant-Whyte. A decentralised navigation architecture. In *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, volume 4, pages 3413–3418 vol.4, 1998.

- [64] Lee-Ling Ong, B. Upcroft, Tim Bailey, M. Ridley, S. Sukkarieh, and H. Durrant-Whyte. A decentralised particle filtering algorithm for multi-target tracking across multiple flight vehicles. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 4539–4544, 2006. doi:10.1109/IROS.2006.282155.
- [65] J. F. G. N. J. Gordon Doucet, A. de Freitas. Sequential monte carlo methods in practice. *Springer-Verlag*, 2001.
- [66] Hue. C Vermaak J. Gangnet M. Perez, P. Color-based probabilistic tracking. *European Conference on Computer Vision*, 2002.
- [67] Blake A. Isard, M. Condensation conditional density propagation for visual tracking. *International Journal on Computer Vision*, pages 5–28, 1998.
- [68] Le Cadre J.P. Perez P. Hue, C. Tracking multiple objects with particle filtering. *IEEE Transactions on Aerospace and Electronic Systems*, pages 791–812, 2002.
- [69] G.A. Rodriguez-Yam, R.A. Davis, and L.L. Scharf. A bayesian model and gibbs sampler for hyperspectral imaging. In *Sensor Array and Multichannel Signal Processing Workshop Proceedings, 2002*, pages 105–109, 2002. doi:10.1109/SAM.2002.1191009.
- [70] S Grime and Hugh F Durrant-Whyte. Data fusion in decentralized sensor networks. *Control engineering practice*, 2(5):849–863, 1994.
- [71] T. Parisini E. Franco and M. M. Polycarpou. Cooperative control of discrete time agents with delayed information exchange. *Proceedings of the 43rd IEEE Conference on Decision and Control*, 2004.
- [72] W. B. Dunbar R. O. Saber and R. M. Murray. Cooperative control of multivehicle systems using cost graphs and optimization. *Proceedings of the American Control Conference*, 2003.
- [73] Daniel Mellinger, Michael Shomin, Nathan Michael, and Vijay Kumar. Cooperative grasping and transport using multiple quadrotors. In *Distributed autonomous robotic systems*, pages 545–558. Springer, 2013.
- [74] M. Ding C. Zheng and C. Zhou. Cooperative path planning for multiple air vehicles using a coevolutionary algorithm. in *Proceedings of the 1st International Conference on Machine Learning and Cybernetics*,, 2002.
- [75] R. Beard and T. McLain. Multiple uav cooperative search under collision avoidance and limited range communication constraints. in *Proceedings of the 42nd IEEE Conference on Decision and Control*, 2003.
- [76] D. Sun R. Zhu and Z. Zhou. Cooperation strategy of unmanned air vehicles for multitarget interception. *Journal of Guidance, Control and Dynamics*, 2005.
- [77] M. Wheeler, B. Schrick, W. Whitacre, M. Campbell, R. Rysdyk, and R. Wise. Cooperative tracking of moving targets by a team of autonomous uavs. In *25th Digital Avionics Systems Conference, 2006 IEEE/AIAA*, pages 1–9, 2006. doi:10.1109/DASC.2006.313769.
- [78] C. Schumacher. Ground moving target engagement by cooperative uavs. in *Proceedings of American Control Conference*, 2005.

- [79] Steve Rasmussen Tal Shima and Dave Cross. Assigning micro uavs to task tours in an urban terrain. *IEEE Transactions on Control Systems Technology*, 2007.
- [80] P. Chandler T. McLain and M. Pachter. A decomposition strategy for optimal coordination of unmanned air vehicles. in *Proceedings of the American Control Conference*, 2000.
- [81] L. Pollini, F. Giulietti, and M. Innocenti. Robustness to communication failures within formation flight. In *American Control Conference, 2002. Proceedings of the 2002*, volume 4, pages 2860–2866 vol.4, 2002. doi:10.1109/ACC.2002.1025223.
- [82] Jongho Shin, H Jin Kim, Seungkeun Kim, and Yongsoon Yoon. Formation flight control under communication failure. In *Proceedings of the 1st international conference on Robot communication and coordination*, page 47. IEEE Press, 2007.
- [83] R.K. Mehra, J.D. Boskovic, and Sai-Ming Li. Autonomous formation flying of multiple ucavs under communication failure. In *Position Location and Navigation Symposium, IEEE 2000*, pages 371–378, 2000. doi:10.1109/PLANS.2000.838327.
- [84] D. Dionne and C.-A. Rabbath. Multi-uav decentralized task allocation with intermittent communications: the dtc algorithm. In *American Control Conference, 2007. ACC '07*, pages 5406–5411, 2007. doi:10.1109/ACC.2007.4282637.
- [85] L. E. Dubins. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *American Journal of Mathematics*, 79:497–516, 1957.
- [86] Helena Kocurova. Analysis of algorithms for computing the crossing number. Master's thesis, Comenius University, Bratislava, 2003.
- [87] H.F. Durrant-Whyte. Autonomous guided vehicle for cargo handling applications. *International Journal of Robotics Research*, page 15(5), 1996.
- [88] C. Thorpe and H. Durrant-Whyte. Field robots. *ISRR*, 2001.
- [89] G. Dissanayake S. Williams and H.F. Durrant-Whyte. Towards terrainaided navigation for underwater robotics. *Advanced Robotics*, pages 15–5, 2001.
- [90] M. Sanjeev Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *Signal Processing, IEEE Transactions on*, 50(2):174–188, 2002. doi:10.1109/78.978374.
- [91] J. Garcia, A. Gardel, I. Bravo, J.L. Lazaro, and M. Martinez. Tracking people motion based on extended condensation algorithm. *Systems, Man, and Cybernetics: Systems, IEEE Transactions on*, 43(3):606–618, 2013. doi:10.1109/TSMCA.2012.2220540.
- [92] Michael Isard and Andrew Blake. Condensation—conditional density propagation for visual tracking. *International journal of computer vision*, 29(1):5–28, 1998.
- [93] S. Dubuisson. The computation of the bhattacharyya distance between histograms without histograms. In *Image Processing Theory Tools and Applications (IPTA), 2010 2nd International Conference on*, pages 373–378, 2010. doi:10.1109/IPTA.2010.5586745.
- [94] M. Seki, H. Fujiwara, and K. Sumi. A robust background subtraction method for changing background. In *Applications of Computer Vision, 2000, Fifth IEEE Workshop on.*, pages 207–213, 2000. doi:10.1109/WACV.2000.895424.

- [95] Dorin Comaniciu and Peter Meer. Mean shift analysis and applications. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 2, pages 1197–1203. IEEE, 1999.
- [96] R. De Maesschalck, D. Jouan-Rimbaud, and D.L. Massart. The mahalanobis distance. *Chemometrics and Intelligent Laboratory Systems*, 50(1):1 – 18, 2000. URL: <http://www.sciencedirect.com/science/article/pii/S0169743999000477>, doi:[http://dx.doi.org/10.1016/S0169-7439\(99\)00047-7](http://dx.doi.org/10.1016/S0169-7439(99)00047-7).
- [97] Felix Luetteke, Xu Zhang, and Joerg Franke. Implementation of the hungarian method for object tracking on a camera monitored transportation system. In *Robotics; Proceedings of ROBOTIK 2012; 7th German Conference on*, pages 1–6, 2012.
- [98] M.B. Dias, Robert Zlot, N. Kalra, and A. Stentz. Market-based multirobot coordination: A survey and analysis. *Proceedings of the IEEE*, 94(7):1257–1270, 2006. doi:[10.1109/JPROC.2006.876939](https://doi.org/10.1109/JPROC.2006.876939).
- [99] B.P. Gerkey and M.J. Mataric. Sold!: auction methods for multirobot coordination. *Robotics and Automation, IEEE Transactions on*, 18(5):758–768, 2002. doi:[10.1109/TRA.2002.803462](https://doi.org/10.1109/TRA.2002.803462).
- [100] Maitreyi Nanjanath and Maria Gini. Repeated auctions for robust task execution by a robot team. *Robot. Auton. Syst.*, 58(7):900–909, July 2010. URL: <http://dx.doi.org/10.1016/j.robot.2010.03.011>, <http://dx.doi.org/10.1016/j.robot.2010.03.011> doi:10.1016/j.robot.2010.03.011.
- [101] Morgan Quigley, Michael A Goodrich, Stephen Griffiths, Andrew Eldredge, and Randal W Beard. Target acquisition, localization, and surveillance using a fixed-wing mini-uav and gimbaled camera. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 2600–2605. IEEE, 2005.
- [102] A.M. Goncalves-Coelho, L.C. Veloso, and V.J.A.S. Lobo. Tests of a light uav for naval surveillance. In *OCEANS 2007 - Europe*, pages 1–4, 2007. doi:[10.1109/OCEANSE.2007.4302314](https://doi.org/10.1109/OCEANSE.2007.4302314).
- [103] D.G. Bell, F. Kuehnel, C. Maxwell, R. Kim, K. Kasraie, T. Gaskins, P. Hogan, and J. Coughlan. Nasa world wind: Opensource gis for mission operations. In *Aerospace Conference, 2007 IEEE*, pages 1–9, 2007. doi:[10.1109/AERO.2007.352954](https://doi.org/10.1109/AERO.2007.352954).
- [104] Jeffery B. Saunders, On Call, Andrew Curtis, Al W. Beard, and Timothy W. Mclain. Static and dynamic obstacle avoidance in miniature air vehicles,” in aiaa infotech at aerospace, 2005, paper no. In *in Proceedings of the Infotech@Aerospace Conference*, pages 2005–6950, 2005.