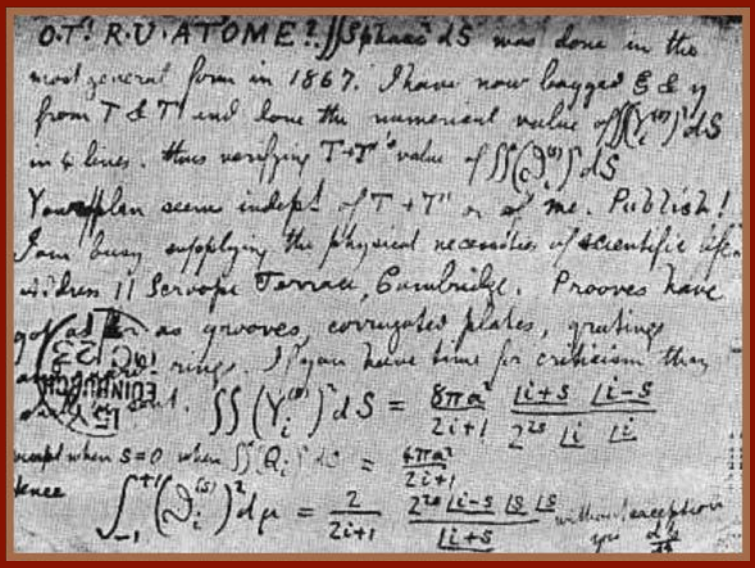
Tesis Doctoral

Trajectory planning based on collocation methods for multiple aerial and ground autonomous vehicles



Autor: Santiago Vera Rendón

Directores: Guillermo Heredia Benot y Aníbal Ollero Baturone

Escuela Técnica Superior de Ingeniería

Universidad de Sevilla

2015

Propuesta de tesis doctoral para la obtención del título de
Doctoral Internacional por la Universidad de Sevilla

Trajectory planning based on collocation methods for multiple aerial and ground autonomous vehicles

Autor:

Santiago Vera Rendón

Directores:

Guillermo Heredia Benot
Profesor titular

Aníbal Ollero Baturone
Profesor Catedrático

Dpto. Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2015

Tesis doctoral: Trajectory planning based on collocation methods for multiple aerial and ground autonomous vehicles.

Programa de doctorado: Robótica, Sistemas y Vehículos Autónomos. Telerrobótica

Autor: Santiago Vera Rendón.

Directores: Guillermo Heredia Benot y Aníbal Ollero Baturone.

El tribunal nombrado para juzgar el proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2015

El Secretario del Tribunal

A mis padres

Agradecimientos

Este documento de tesis que aquí comienza, culmina una era de trabajo, esfuerzo, retos y superaciones personales tanto en mi vida laboral como en la personal. Justo ahora miro hacia atrás y veo una gran evolución que se ha intensificado en estos últimos años. Muchos cambios condicionados por las personas que, de una u otra manera, han estado a mi lado. Padres, hermanos, amigos... todos son responsables de ser quien soy y estar donde estoy.

Es por eso que quiero dedicar sólo unas líneas a algunas personas, dejándome detrás a muchas otras que también han sido igual de importante. Empezando por la familia, agradecer a mis padre Antonio e Isabel por dármelo todo, y especialmente, por su cariño. Y a mis hermanos Juan Antonio, Paco e Isabel que siempre han sido referentes a seguir. Todos ellos me han apoyado siempre en mis decisiones y me han arropado con su cariño.

También me siento afortunado por tener grandes amigos con los que he compartido mis buenos y malos momentos. Me gusta definir a los amigos como "esa parte de la familia que tú eliges a lo largo de la vida". Por eso quiero dedicarles estas líneas a todos.

Agradecer a mis tutores Guillermo Heredia y Aníbal Ollero el apoyo que siempre han mostrado y por ofrecerme todos los medios necesarios para realizar este trabajo. Especialmente, agradecer a Guillermo su paciencia y constancia. También dar las gracias a mi compañero José Antonio Cobano, ya que nada de esto hubiera sido posible sin su apoyo y trabajo. Finalmente agradecer la compañía del resto de compañeros y amigos del laboratorio los cuales han hecho más llevadero el trabajo durante todos estos años.

No quiero olvidarme de expresar mi agradecimiento a los miembros del grupo de investigación del LARICS de la Universidad de Zagreb. Especialmente quiero agradecer el apoyo recibido por el Profesor Zdenko que me hizo sentir como uno más de la familia. Y agradecer también la compañía y ayuda recibida de mis amigos y compañeros de laboratorio Tomislav y Frano.

Por último quiero mostrar mi gratitud a la consejería de Innovación Ciencia y Empresa de la Junta de Andalucía y a la Comisión Europea por el apoyo económico dado a mi trabajo a través de diversos proyectos.

Cornelia Wachter, Angela Schaefer, David Alejo, Julia Vera García, Francisco Vera, Juan Antonio Vera... sólo ellos saben por qué están aquí.

Santiago Vera Rendón

Sevilla, 2015

Acknowledgements

This thesis paper that begins here, ends an era of work, effort, challenges and personal exceedances both in my work and personal life. Right now I look back and see a great evolution that has intensified in recent years. Many changes caused by people who, in one way or another, have been by my side. Parents, siblings, friends... all of them are responsible for who I am and for where I am.

For this reason I want to dedicate some special words to some of them, leaving behind many others who have also been equally important. Starting with the family, I want to thank to my parents Antonio and Isabel who give me everything, and especially for their affection. And to my siblings Juan Antonio, Paco and Isabel who always have been a reference to follow. All of them always have supported me in my decisions and have been close with their love.

I am also fortunate for having very good friends with whom I have shared my ups and downs. I like to define my friends as "that part of the family that you choose throughout life" So I want to dedicate these words to all of them.

I want to say thank you to my tutors Guillermo Heredia and Aníbal Ollero for the support that they always have shown and for offering me all necessary means to carry out this work. Especially thanks to Guillermo for the patience and perseverance. I also thank my colleague José Antonio Cobano, because none of this would have been possible without his support and help. Finally a notable mention to thank my friends and lab mates who have made the job more bearable for all these years.

I do not want to forget to thank the all members of the LARICS research group from University of Zagreb. I especially want to thanks the support received by Professor Zdenko who made me feel like one of the family. Also a thanks to my friends and lab mates Frano and Tomislav for the support and help received.

Finally I want to show my gratitude to the Ministry of Innovation, Science and Enterprise of the Andalusian Government and the European Commission for the financial support given to my work through different projects.

Cornelia Wachter, Angela Schaefer, David Alejo, Julia Vera García, Francisco Vera, Juan Antonio Vera... only they know why they are here.

Santiago Vera Rendón

Seville, 2015

Abstract

This PhD Thesis focuses on the optimal trajectory generation for multiple autonomous vehicles. Nowadays, there are many outdoor and indoor applications where teams of robots and unmanned vehicles are very useful. In these cases, the ability of autonomous navigation is given by the trajectory planning methods. Between trajectory planning methods, there are a subset of them which focuses on finding the optimal solution given by a specific criterion. Furthermore, trajectory planning methods are very important in cooperative applications. A constrained problem has to be solved when two or more vehicles share the same space in a multiple vehicle application.

This thesis presents a new optimal trajectory planning method based on collocation methods. Direct collocation and Pseudospectral collocation are a set of methods widely used in optimal control problems. These methods, based on solving numerically optimal control problems by effective discretization and converting it to an algebraic nonlinear programming (NLP) problem, can be efficiently solved by applying optimization tools.

The work presented in this dissertation covers specific aspects like accuracy, scalability (number of vehicles in a scenario) and computation time of the different collocation algorithms. A new algorithm based on collocation method which is named S-Adaptive Pseudospectral collocation is also presented.

In addition, this dissertation considers multiple aerial and ground vehicles, and several kinds of applications and scenarios (with and without fixed). Finally, the thesis presents extensive experimental results with teams of UAV (Unmanned Aerial Vehicles) and UGV (Unmanned Ground Vehicles). The indoor experiments are shown in order to validate the multi-vehicle trajectories obtained by the methods.

Contents

LIST OF FIGURES

**

LIST OF TABLES

1. INTRODUCTION

Imagination is more important than knowledge. For knowledge is limited to all we now know and understand, while imagination embraces the entire world.

Albert Einstein

This PhD Thesis presents contributions in the field of trajectory planning and coordination within vehicle teams, and more precisely, within multiple unmanned vehicles. Collocation methods have been increasingly used in previous years to obtain optimal solutions for different problems in engineering. The following thesis presents a new point of view in the application of collocation methods to trajectory planning for multiple vehicles.

This chapter begins with first presenting the motivation of the research carried out. Then, the main objectives which are covered in this thesis are presented in detail. Afterwards, the outline and main contributions of the thesis are introduced. Finally, the framework in which the research has been developed is described.

1.1. Motivation

Interest in unmanned vehicles has been greatly increased in the last decade. Given the ability to plan a trajectory based on sensor input, the unmanned vehicles gain the capability to avoid obstacles or other vehicles, track targets, optimize certain performance characteristics such as endurance, and generally adapt to a dynamic situation.

Initially this interest has been focused on military uses, mainly in the utilization of Unmanned Aerial Vehicles (UAVs) for different applications, but over time the number of civil applications has increased strongly, both with UAVs and Ground Unmanned Vehicles (UGVs). The higher mobility and maneuverability of a UAV with respect to ground vehicles makes them very useful in performing complex tasks such as data and image acquisition of areas otherwise inaccessible using ground means, localization of targets, tracking, map building and others. In recent years the technologies for autonomous aerial vehicles have experienced an important development, making the research in aerial autonomous systems affordable for universities and research centers.

On the other hand, the complexity of some applications requires the cooperation between several robots. Moreover, even if cooperation is not required, collision

avoidance is one of the more important functions that has to be carried out in multi-vehicle scenarios. This concept is directly related to the safety and robustness of any application with autonomous vehicles.

This thesis presents a centralized method of optimal trajectory planning for multiple unmanned vehicles. The method can be used both with aerial or ground vehicles, as well as in mixed scenarios with aerial and ground vehicles. This method is based on Pseudospectral (PS) collocation. PS collocation is a technique that solves numerically optimal control problems and which has gained popularity in recent years in several applications [1] [2].

Vehicle trajectory planning is formulated as an optimal control problem, including the differential equations of the vehicle dynamics and physical constraints, such as actuator saturations, avoiding collisions with the environment or other vehicles, etc., and an optimization index that should be minimized, as for example the total distance travelled or the energy consumption. The PS collocation methods try to solve these optimal control problems approximating the states and control inputs by polynomials, and by enforcing the differential equations and constraints on specific instants of time called collocation points. Thus PS collocation methods effectively discretize the optimal control problem and convert it to an algebraic nonlinear programming problem, which can be efficiently solved by applying optimization tools like SNOPT solver [3].

This formulation of the trajectory planning problem allows for a good representation of the real vehicle behavior in real applications, and thus the solution trajectories will be realistic and easily tracked by the autonomous vehicles in real-time.

The study presented in this thesis covers behavior analysis of different PS collocation methods when they are used in vehicle trajectory planning. Although PS collocation methods have been formulated in the general case in the literature, it is important to analyze how certain aspects like discretization or approximation of the solution, for example, may affect their performance for trajectory planning problems. The work presented in this dissertation covers specific aspects like accuracy, scalability (number of vehicles in a scenario) and computation time for implementation in real-time. Finally, a new collocation algorithm designed especially for trajectory planning of multiple vehicles moving simultaneously in the same area is developed and presented in this thesis, and comparative results with the previous methods are presented in multiple scenarios, validated in simulation and real experiments.

1.2. Objectives

The main objective in this dissertation consists of an extensive behavior analysis of the different collocation methods, when they are used in vehicle trajectory planning. In order to do so, this thesis strives to accomplish the following objectives:

- Full classifications of multiple collocation methods. This point tries to distinguish between classical Direct collocation methods and the recent Pseudospectral collocation method, paying more attention to the multiple algorithms of Pseudospectral methods.
- Application of collocation methods to trajectory planning. Collocation methods are general optimal control solver. Advantages and drawbacks will be analyzed from the results obtained of trajectory planning scenarios.
- Performance study of the collocation method. This point tries to study very important concepts like accuracy of the solution, scalability of the method and the global system, the possibility of use in real-time system, uses of the method in centralized or distributed scenario, etc.
- Search configurations that improve the performances of collocation method (issues like computation time or accuracy).
- Development of the S-Adaptive PS collocation method. This is a specific algorithm that focuses on solving the problems that regular collocation methods incur during trajectory planning problems.
- Ground vehicles and consideration of fixed obstacles. Collocation methods in aerospace applications have been extensively studied in the literature. However there has been much less results in the application of collocation methods for ground robots and ground autonomous vehicles. An objective of this thesis is the application of collocation methods to ground vehicles, including scenarios with fixed obstacles.
- Validation of different algorithms. This point focuses on the validation of the algorithms through a large number of simulations and real experiments. A large number of different scenarios are considered with various types of vehicles.

1.3. Outline and main contributions

This PhD Thesis makes contributions in the field of trajectory planning when multiple vehicles are considered. Theoretical results, oriented to the optimal trajectory planning, are derived from the studies of collocation and from the contribution of a new algorithm based on a Legendre-Gauss-Radau PS collocation method. A summary of the contents of each chapter in presented in the following paragraphs:

Chapter 1 introduces the objectives and scope of the thesis, which is placed within the framework of the European project EC-SAFEMOBIL [1] "Estimation and Control for SAFE wireless high MOBILity cooperative industrial systems".

Chapter 2 presents a review of similar related works in the literature. Trajectory

[1] http://www.ec-safemobil-project.eu/

planning and collision avoidance methods have been studied extensively, and this chapter presents the latest contributions to this field.

An overview of collocation methods is introduced in **Chapter 3**. This chapter presents the basic concept of these methods, and their evolution. A small introduction to the numerical analysis and spectral methods is given. Then, a global classification of different collocation algorithms is presented. This section is completed by a detailed description of PS collocation methods.

In **Chapter 4,** the problem of trajectory planning is analyzed from the point of view of collocation methods. Their advantages and drawbacks are studied, and a new method witch improves the traditional colocation method is presented. This is the S-Adaptive PS method and a detailed explanation is presented. This algorithm has been published in the Journal of Intelligent and Robotic Systems [4], and has been accepted to the International Conference on Robotics and Automation (ICRA 2015) [5].

In **Chapter 5** several real applications related to three different scenarios are presented: one with multiple aerial vehicles, another with multiple autonomous ground vehicles and the last one includes an aerial and a ground autonomous vehicle. Several collocation algorithms are used for trajectory generation with different vehicles (fixed-wing and rotary-wing UAVs, Ackermann steering and differential drive UGVs). The following contributions have been published: An Air Traffic Management (ATM) application has been presented in the paper published in the 11th Portuguese Conference on Automatic Control [6]. The application presented in this paper considers the trajectory generation for converging air traffic problem. A collision avoidance application for multi-UAVs based on the hp-Adaptive PS method has been published in the International Conference on Robotics and Automation (ICRA 2014) [7]. In this case, real trajectories of five UAVs are flown. A comparative study between two optimal trajectory planning method (hp-Adaptive algorithm and Heuristic Velocity Planning) has been submitted to the Journal of Aerospace Information Systems [8].

Chapter 6 presents the validation of most of the applications presented in Chapter 5 by experiments carried out in two different testbeds. Aerial experiments have been carried out in the Center for Advanced Aerospace Technologies (CATEC) in Seville (Spain). Ground experiments have been carried out in the Laboratory for Robotics and Intelligence Control System (LARICS) at the University of Zagreb (Croatia). The experiments performed in the University of Zagreb were carried out in the framework of a three-month international stay in this university.

The results from these experiments with real robots have also been published. A real-time distributed algorithm base on PS Legendre-Gauss-Radau method has been published in the International Conference on Unmanned Aircraft Systems (ICUAS 2014) [9]. This method implements a rolling horizon policy which allows execution of the algorithm in low computation time. An UAV collision avoidance application based on S-Adaptive PS method will be published in the proceeding of the ICRA 2015 [5]. And finally, the advances obtained in the scenario with ground vehicles are being written to be submitting to the Journal of Intelligent and Robotic Systems. This scenario

considers four UGVs and multiple obstacles.

The Thesis is completed with **Chapter 7** that summarizes the conclusions and proposes some guidelines for further research.

The main publications associated with this thesis are summarized in the following table, along with other publications in the field of planning methods which have been produced during the thesis elaboration period:

- S. Vera, F. Petric, G. Heredia, Z. Kovacic, and A. Ollero, "Multirobot trajectory planning based on S-Adaptive Pseudospectral collocation". Journal of Intelligent and Robotic Systems. Pending submitting.
- S. Vera, J. A. Cobano, G. Heredia and A. Ollero, "Safe Trajectory Planning for Multiple Aerial Vehicles with Segmentation-Adaptive Pseudospectral Collocation", in *IEEE International Conference on Robotics and Automation (ICRA'15).* Accepted in January 2015, pending publication.
- S. Vera, J. A. Cobano, G. Heredia and A. Ollero, "Collision Avoidance for Multiple UAVs using Rolling-horizon Policy", *Journal of Intelligent and Robotic Systems*. Accepted in December 2014, pending publication.
- S. Vera, J. A. Cobano, G. Heredia and A. Ollero, "An Hp-Adaptive Pseudospectral Method for Conflict Resolution in Converging Air Traffic", in *Proceeding of the 11th Portuguese Conference on Automatic Control CONTROLO'2014. Springer Lecture Notes in Electrical Engineering* Vol. 321, pp. 333-343. 2014.
- S. Vera, J. A. Cobano, G. Heredia and A. Ollero, "An hp-adaptive pseudospectral method for collision avoidance with multiple UAVs in real-time applications", in *Proceeding of the IEEE International Conference on Robotics and Automation (ICRA'14)* pp. 4717- 4722. 2014.
- S. Vera, J. A. Cobano, D. Alejo, G. Heredia and A. Ollero, "Optimal conflict resolution for multiple UAVs using pseudospectral collocation", *Journal of Aerospace Information Systems.* Submitted in June 2014, pending acceptance.
- S. Vera, J. A. Cobano, G. Heredia and A. Ollero, "Rolling-horizon trajectory planning for multiple UAVs based on pseudospectral collocation", in *Proceeding of the International Conference on Unmanned Aircraft Systems (ICUAS'14)* pp. 516-523. 2014.
- J. A. Cobano, D. Alejo, S. Vera, G. Heredia and A. Ollero, "Multiple gliding UAV coordination for static soaring in real-time applications", in *Proceeding of the IEEE International Conference on Robotics and Automation (ICRA'13)* pp. 782-787. 2013.
- S. Vera, G. Heredia, I. Maza, and A. Ollero, "Sistema de planificación para el ensamblado de estructuras mediante múltiples robots basado en los algoritmos IHS y RRT*", in *Proceeding of Robot 2013: First Iberian robotics conference* pp. 839 – 846. 2013.

- S. Vera, I. Maza, and A. Ollero, "Algoritmo para la secuenciación en el ensamblaje de estructuras mediante robots", in *Proceeding of the XXXIII Jornadas de Automática* pp. 839-846. 2012.
- S. Vera, A. Jiménez-González, G. Heredia, and A. Ollero, "Algoritmo Minimax aplicado a vigilancia con robots móviles", in *Proceeding of Robot 2011: (Robótica Experimental)* pp. 468-474. 2011.
- J. A. Cobano, D. Alejo, S. Vera, G. Heredia, S. Sukkarieh, A. Ollero, "Human Behavior Understanding in Networked Sensing". *Springer International Publishing*. Switzerland, 2014, ch. Distributed Thermal Identification and Exploitation for Multiple Soaring UAVs, pp. 359–378. 2014.
- S. Vera, "Path planning with Pseudospectral Collocation Methods", *Lecture organized by IEEE Croatia Section Robotics and Automation*, at University of Zagreb, Croatia. 2014.

1.4. Thesis framework

This dissertation has been mainly developed in the Robotics, Vision and Control Research Group (GRVC)[2] at the University of Seville (Spain), within the framework of several international and national research projects related to UAVs. The support of these projects was fundamental, since they have provided the funding and the equipment needed for this research.

The core of the work presented in this thesis has been performed in the framework of the European Project EC-SAFEMOBIL "Estimation and Control for SAFE wireless high MOBILity cooperative industrial systems" (FP7-ICT-2011.3.3-288082). EC-SAFEMOBIL focuses on the development of autonomous systems and vehicles that play an important role in many applications, including disaster management or the monitoring and measurement of events. Currently many missions cannot be accomplished or imply a high level of risk for the people involved e.g. pilots and drivers, as unmanned vehicles are not available or not permitted. These missions could be performed or facilitated by using autonomous helicopters with accurate positioning and the ability to land on mobile platforms such as ship decks. These applications strongly depend on the UAV reliability to react in a predictable and controllable manner in spite of perturbations such as wind gusts. On the other hand, the cooperation, coordination and traffic control of many mobile entities are relevant issues for applications such as automation of industrial warehousing, surveillance by using aerial and ground vehicles, and transportation systems.

EC-SAFEMOBIL is devoted to the development of sufficiently accurate common motion estimation and control methods and technologies in order to reach levels of reliability and safety to facilitate unmanned vehicle deployment in a broad range of

[2] http://grvc.us.es/

applications. It also includes the development of a secure architecture and the middleware to support the implementation. More detailed scientific and technical objectives of EC-SAFEMOBIL are the following:

- Development of new robust distributed probabilistic state estimation/prediction and event detection/tracking methods for complex high mobility systems.
- Development of new distributed methods for safe real-time networked cooperation, coordination, and control.
- Architectural paradigms for safe and secure industrial networked estimation and cooperative control.
- Very accurate coupled motion control of two mobile entities.
- Distributed safe reliable cooperation and coordination of many high mobility entities.

Additionally, this work has been supported by Spanish project RURBAN "Robótica Ubícua en Entornos Urbanos" (contract P09-TIC 5121), funded by the local government of Andalucía (Spain).

This thesis was also partially supported by the projects CLEAR "Misiones cooperativas de larga duración empleando robots aéreos" 2011/725 (orgánica 18.28.08.30.30) funded by the Spanish government and the project RANCOM[3] "Aplicación de Sensores de Rango a la Navegación y Coordinación de Múltiples Vehículos Aéreos no Tripulados" P11-TIC 7066 MO (2012/838) funded by the local government of Andalucía (Spain). Both projects are closely related to the topic of this thesis. The CLEAR project is based on the cooperation between aerial and ground robots to perform missions of long endurance due to the integration of autonomous subsystems for battery recharging or refueling. The RANCOM project is based on the development and experiment validation of navigation and coordination for multiple unmanned aerial vehicles based on range sensors.

Several parts of this dissertation have been developed in cooperation with the University of Zagreb (Croatia), the German Aerospace Research Center (DLR), the SELEX aerospace company (UK) and the Euroimpianti UGV manufacturer (Italy). All these centers are currently involved in the EC-SAFEMOBIL project.

It is also worthwhile to mention the stay at the LARICS laboratory of University of Zagreb, in which theoretical and experimental developments on teams of multiple ground vehicles were performed.

[3] http: //grvc.us.es/RANCOM/

2. RELATED WORK

He didn't know that it was impossible and that's exactly why he did.

Jean Cocteau

Vehicle motion planning is a special case of the general motion planning problem, which in general is very difficult to solve. Aspects such as vehicle degree of freedom, dimension and discretization of the scenario or the consideration of fixed and mobile objects in the scenario, all strongly affect the solution of the motion planning problem.

A large number of trajectory planning algorithms can be found in literature, depending on the desired results, some are more useful or relevant than others. In this Chapter, an overview and a classification of these methods are presented.

2.1. Introduction

For a robot to accomplish any given high-level task, this task should be translated into low-level commands understandable by the robot controller. This whole process, known as *motion planning*, can usually be broken down into numerous steps, beginning with high level task allocation then follow by trajectory planning, robot control and generation of actuator commands.

Trajectory planning consists of finding a time series of successive states that allows the movement of a robot from a starting configuration towards a goal configuration in order to achieve a task, such as tracking a ground vehicle from an aerial robot or landing on a platform or on a mobile ground vehicle. This trajectory should respect given *constraints*. These constraints can be geometric: for instance, the robot should not collide with the environment; they can also be kinodynamic constraints: for instance, the robot velocities, accelerations, or torques should be within specified limits, etc. Next, if several trajectories are possible, one should select the option that *optimizes* a certain objective, such as the trajectory execution time or energy consumption.

At the path level, most convenient to find the shortest path between the start and goal states. At the trajectory level, other optimization objectives such as minimum time or minimum energy should be considered. Note that because of kinodynamic constraints, the shortest path might not correspond to a minimum time or minimum-energy trajectory.

Vehicle motion planning is a special case of the general robot motion planning problem,

which is generally very difficult to solve, especially as the number of degrees of freedom increases. For example, in a typical UAV application, the vehicle operates in three-dimensional space, may have two to four degrees of freedom, and has differential constraints, including limited speed and maximum acceleration. The resulting problem space has from five to twelve dimensions, all of which are associated with the equations of motion and involve constraints on states and input variables. There is no existing algorithm that provides an exact analytic solution to such a problem in the general case. Indeed, even state of the art approximation algorithms operating on a three-dimensional subspace of this problem space are difficult to compute in real-time. Furthermore, several simplifications and sub-cases of the general problem have been proven to be unsolvable in polynomial time [10]. Approximation algorithms are however possible, and often rely on exact solutions to simplified sub-problems.

In this section a review of the main trajectory planning methods is presented. These algorithms solve the problem of trajectory planning regarding dynamically-constrained vehicles through an environment with obstacles. In addition, the problem of finding a trajectory that minimizes some cost functional is of interest. A general guidance problem is typically characterized by a two- or three-dimensional problem space, limited information about the environment, on-board sensors with limited range, speed and acceleration constraints, and uncertainty in vehicle state and sensor data.

Some metrics to measure the performance of these algorithms have been proposed including completeness, optimality, precision and computational complexity. A motion planning algorithm is considered to be *complete* if and only if it finds a path when one exists, and returns a variable stating no path exists when none actually does. It is considered to be *optimal* when it returns the ideal path with respect to some criterion. Note that any optimal planner is also complete. The completeness/optimality is also related to the discretization of the solution space, meaning that as the resolution of the discretization increases, an exact solution is achieved as the discretization approaches the continuum limit. *Precision* signifies the error of approximating the solutions with a family of curves. Finally, the *computational complexity* concerns the computational time needed to find a solution path. This metric is also related to the *efficiency* of the method, but efficiency is a global metric and it usually concerns various metrics at the same time.

2.2. Problem formulation

In this section, the basic concepts needed to formulate any general trajectory planning problem are presented. A complete description of the problem formulation can be found in [11]. Let $T \subset R$ denote the *time interval*, which may be bounded or unbounded. If T is bounded, then $T = [0, t_f]$, in which 0 is the *initial time* and t_f is the *final time*. If T is unbounded, then $T = [0, \infty)$. An initial time other than 0 could alternatively be defined without difficulty, but this will not be done here.

A *configuration* is a vector of variables that define the shape of the vehicle. Most vehicles

can be considered to be rigid bodies in three-dimensional space, and thus defined uniquely by six numbers: three position coordinates and three orientation coordinates. For example, a robot with a manipulator will generally have a much larger number of variables because each degree of freedom of the manipulator adds a variable to the configuration space. The set of all possible configurations of a vehicle is called the *configuration space* or *C-space*. Let the *state space* X be defined as $X = C \times T$, in which C is the usual C-space of the robot. A state x is represented as $x = (q, t)$, to indicate the configuration q and time t components of the state vector. Note that paths in X are forced to move forward in time.

The problem is conceptually illustrated in Figure 1. The main difficulty being that it is neither straightforward to construct an explicit boundary nor is there solid representation of either C_{free} or C_{obs} in the time.

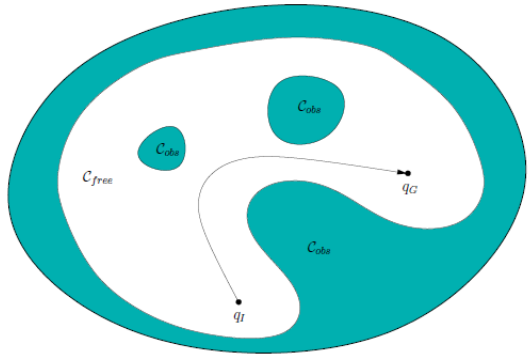


Figure 1: General graphic description of the Configuration Space.

The components are as follows:

1. A *world* W in which either $W = R^2$ or $W = R^3$.
2. A semi-algebraic *obstacle region* $O(t) \subset W$ in the world. It is assumed that the obstacle region is a finite collection of rigid bodies that undergoes continuous, time-dependent rigid-body transformations.
3. A semi-algebraic *robot* is defined in W. It may be a rigid robot A or a collection of m links: $A_1, \ldots, A_m$.
4. The configuration space, which is determined by specifying the set of all possible transformations that may be applied to the robot. From this, C_{free} and C_{obs} are derived.
5. A configuration $q_I \in C_{free}$ designated as the *initial configuration*. And a configuration $q_G \in C_{free}$ designated as the goal configuration. The initial and goal configurations together are often called a *query pair* and designated as (q_I, q_G).

6. The *state space* X is the Cartesian product $X = C \times T$ and a state $x \in X$ is denoted as $x = (q, t)$, to denote the configuration q and time t components. The obstacle region, X_{obs}, in the state space is defined as

$$X_{obs} = \{x \in X | A(q) \cap O(t) \neq \emptyset\} \quad (2.1)$$

$$X_{free} = X \setminus X_{obs}$$

7. In the same way, the state $x_I \in X_{free}$ is designated as the *initial state*, with the constraint that $x_I = (q_I, 0)$ for some $q_I \in C_{free}(0)$. In other words, at the initial time the robot cannot be in collision. And a subset $X_G \subset X_{free}$ is designated as the *goal region*.
8. A complete algorithm must compute a continuous, time-monotonic path, $\tau: [0, 1] \rightarrow X_{free}$, such that $\tau(0) = x_I$ and $\tau(1) \in X_G$.

Note that this general formulation does not express any constraints about robot motion. This means the model allows for infinite acceleration and unbounded speed. The robot velocity may change instantaneously, but the path through C must always be continuous.

It is shown in [12] that the general trajectory planning is NP-hard. Meaning that the complexity of the path planning problem increases exponentially with the dimension of the configuration space C. So the main problem occurs when the dimension of C is unbounded.

Next, an overview of the main planning methods is presented. These algorithms are classified into six main groups: graph search algorithms, exact algorithms, probabilistic algorithms, optimal probabilistic algorithms, heuristic search algorithms and optimal control algorithms.

2.3. Graph search method

Historically, the first path planning algorithms consisted of algorithms that found a path to connect two nodes in a graph. In these early algorithms, it was assumed that a graph $G =< G, V >$ that unites points in the configuration space in such a way that $G \in C_{free}$was either introduced by the designer or obtained by discretization. That is, dividing the configuration space into cells that could and could not be collision free, as shown in Figure 2.

The first developed algorithm that finds whether two vertices in a graph $G =< V, E >$ are connected or is the Depth First Search algorithm. It was introduced in 1889 by French mathematician Charles Pierre Trémaux as a strategy for solving mazes. The main principle was to explore one of the paths in the graph until the solution was found, an already visited node is found or no further paths exist. In such case, a backtracking process until the first node in which unexplored paths is carried out in order to explore another path.

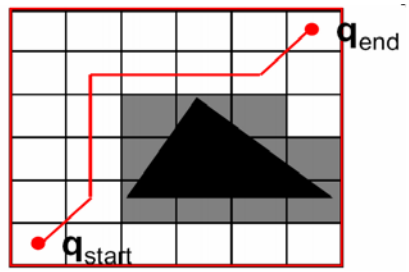


Figure 2: Cell mapping discretization of a 2D space. One solution is in red.

Another popular algorithm to find connectivity between graphs is the Breadth First Search algorithm (BFS). This algorithm instead of greedily following a path like DFS, explores first all the nodes connected to the starting node, then reproduces this procedure in each of the children nodes until the desired node is found or all nodes connected to the starting node are found. The BFS algorithm has two main advantages over the DFS: it can obtain the goal node in infinite graphs and in addition it returns the path with lowest number of edges. On the other hand, it is much more memory consuming than DFS. Note that in path planning problems, BFS algorithm returns the path with minimum length if all the edges have equal length.

The Djikstra algorithm [13] is the first algorithm that efficiently solved the problem of translating from one cell to another, with the lowest cost in a graph with weighted edges. Thus it can be used for obtaining the lowest cost path that connects the starting and goal nodes. It can be considered an extension of the BFS algorithm to weighted graphs (i.e. the edges of the graph have associated a cost). This algorithm will explore first the nodes with lowest distance to the starting node, updating the cost of the successor of the current node.

One of the main problems of the Djikstra algorithm is that it does not take into account the position of the goal in order to explore the graph. This can yield to unnecessary node exploring in most situations. In order to overcome this, it was found that some heuristics pertaining to the goal node distance could be introduced in the algorithm in order to direct the exploration of the graph more toward the goal node generating the greedy algorithms, exploring first the nodes which are closer to the goal node. These considerations would be mixed in the A* algorithm [14] in which the nodes are first explored not only taking into account the distance to the starting node but also the estimated distance to the goal node. It has been demonstrated that if the heuristic function fulfils some requirements, the A* algorithm returns the optimal path between the starting and goal nodes.

2.4. Obtaining the graph representation

The previous section, assumed a priori that the space is discretized in cells and each

cell can be free or have the presence of obstacles. However, obtaining and storing these cell maps in 3D environments and in large areas can be time and memory consuming. In this section, other methods for generating the graph that describes the configuration space are described.

2.4.1. Exact graph generation methods

Alongside the grid discretization method there are some methods for obtaining a graph in configuration spaces with polygonal obstacles. These methods are listed below:

- **Exact Cell Decomposition Methods**. In these methods the common idea is to divide the free space into areas which are called cells. The centroid of each cell, as well as the middle point of each division, are inserted as nodes in the graph and then nodes belonging to neighbouring nodes are connected. These methods include vertical and horizontal cell decomposition, see Figure 3a.
- **Visibility Graph (VG)**. This method can be used for finding shortest path in a configuration space with polygonal obstacles. In this case, each vertex of each obstacle is added to the graph and connected with other nodes if they have line of sight as shown in Figure 3b.
- **Voronoi Diagrams (VD)**. The visibility graph method can find the shortest path but its main drawback is that can make the robot move close to the obstacles. One technique to avoid this is to artificially expand the obstacles to a desired clearance (minimum distance to the obstacles) taking into account the shape of the robot or a safety area in which the robot is wrapped, as shown in Figure 3c.

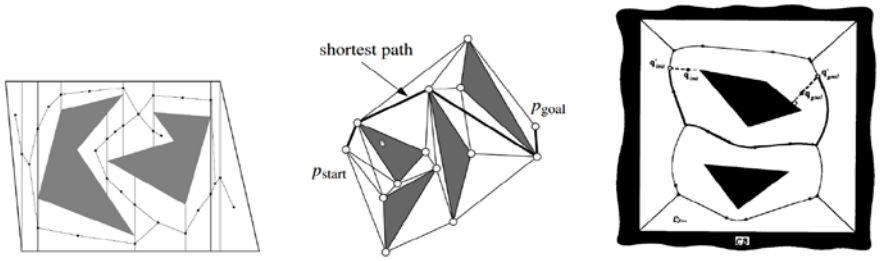


Figure 3: (a) Exact cell decomposition in 2D environment with polygonal obstacles. (b) Visibility graph. (c) Voronoi graph.

Note that the above algorithms are formulated in the Cartesian space and can be solved easily in a 2D Cartesian space. However, the 3-dimensional version of the Euclidean shortest path problem is much harder. In this case, the graph shortest path may not transverse the edges of the polyhedral obstacles but rather any point in the edges of the obstacles. This problem has been demonstrated to be NP-hard [10].

2.4.2. Probabilistic roadmaps

In order to overcome the limitations of exact methods in systems with high dimensional configuration spaces, probabilistic methods for generating the graph that describes C_{free} are introduced in [15]. The basic method is named as *Probabilistic Roadmaps (PRM)* and is divided into two steps: the graph generation step and the query step.

The first stage is the graph generation step, in which the graph *G* is computed by randomly generating nodes in C_{free} and connecting them with close enough existing nodes in *G*. Note that in this case, the nodes are not generated taking into account the geometry of the problem with the exception that the nodes and edges must be collision free.

The second stage is the query step. In this step, the initial and goal configurations (q_{init} and q_{goal}) of the problem to be solved are added and connected to *G*. Then, the shortest path is obtaining by applying one of the methods described in Section 2.3 to *G*. This graph *G* is generated once and then each time a path planning problem needs to be solved.

2.4.3. Rapidly-exploring Random Trees

The aforementioned methods, Probabilistic Roadmaps, perform an exhaustive exploration of the configuration space in order to generate a graph *G* that represents this space. Then, this graph can be used for multiple queries. In contrast, simple query algorithms generate the graph starting from the starting configuration of the problem and the exploration is usually stopped when the goal configuration is reached or is sufficiently close.

Rapidly-exploring Random Trees (*RRT*) [16] is a probabilistic algorithm that generates a tree shaped graph ($G \in C_{free}$). This tree will rapidly cover the configuration space until the goal configuration is sufficiently close from the tree. The basic RRT algorithm starts a tree by creating the root in q_{init} and extends the tree by generating random samples (q_{rand}) of the configuration space and by making the tree extend towards q_{rand}. This extended procedure is usually completed by interpolation in the basic RRT algorithm: the new vertex (q_{new}) will be located at a distance Δ_q from the closest node to $q_{rand}(q_{near})$ in the direction towards q_{rand}. Mathematically:

$$q_{new} = \Delta_q \frac{q_{new} - q_{near}}{q_{new} - q_{near}} \tag{2.2}$$

If the path between q_{near} and q_{new} is collision-free, this node is added to the tree. This procedure is repeated until the distance between the new node and final state q_{goal} goes below d_{min} .

Like PRM algorithms, RRT are claimed to be probabilistically complete: the

probability that the generated tree will be closer than a minimum distance of the goal node with probability that converges to zero with increasing time [17]. RRT is also capable of generating paths that meet kinodynamic constraints: in this case the tree will be extended by integrating a model of the vehicle for a determinate amount of time Δt.

2.4.4. Optimal probabilistic methods

The main problem of the basic probabilistic methods such as PRM and RRT is that even though they generate paths that unite q_{init} and q_{goal} without collisions, no considerations regarding the quality of the path are introduced. Thus, they are proven useful when generating paths in problems with high dimensionality and in cluttered environments. But their lack of consideration regarding to the quality of the path was evident when they were applied to mobile robot path planning. Their generated paths yielded to random like motions that were not properly optimized and were difficult to forecast.

Transition based RRT (*t-RRT*) [18] were proposed in order to generate paths with better quality when planning on Configuration-space with an associated *Cost map*. Its strategy is to sample first in the zones of C_{free} that have lowest cost (i.e. valleys of the cost function) and gradually broaden the sampling region if until a solution is found. This approach has been tested to obtain paths with more quality than the original RRT procedure. However, once the path is obtained, no further improvements on it are performed, so optimal convergence is not assured.

Optimal convergence of a probabilistic planner was first proven in [19], where *RRT** was proposed. This planner is found to be both probabilistic complete and asymptotically optimal. RRT* planning algorithm makes two main modifications to the original algorithm.

First, when a new sample is generated, the algorithm attempts to connect it not only to the nearest neighbour but also to a set of neighbours that are close enough. Only the connection that optimizes the path between the new sample and the starting configuration is added to the tree.

The other modification is called the rewiring step. In this phase, the current cost of the neighbours of the new sample is compared to the cost that would be obtained by traveling through the new sample. If this new cost is less than the current cost, the graph is rewired.

2.5. Evolutionary and Swarm Optimization Applied to Path Planning

As seen in Section 2.1 the problem of robot path planning for multiple vehicles is NP-Hard. This implies that no exact algorithm with polynomial complexity can be found to solve the problem. Some heuristic search algorithms such as evolutionary algorithms and swarm intelligence based algorithms can return a quasi-optimal

solution to these problems.

Evolutionary algorithms comprehend a huge set of general purpose optimization algorithms that are based on the evolution theory of the species that was first proposed in [20]. This set includes Genetic Algorithms, Genetic Programming (GP), Evolutionary Programming and many more. On the other hand, Swarm intelligence algorithms are inspired in the behaviour of natural swarms such as ants, birds, etc. Some of these algorithms are Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Cuckoo Search Algorithm (CSA) or Artificial Bee Colony (ABC) amongst others.

One of the first approaches to the genetic path planning can be found in [21]. In [22], the convenience of the genetic algorithm optimization to solve the path planning problem is deeply discussed. However, only simple results on discretized 2D space with no experimental results are presented.

Yet another planner is presented in [23] with the addition of a post processing step that smooths the path with Bezier curves. However, no extension to 3D trajectories is shown in the paper and neither relevant simulation studies nor experiments are given. In [24] fuzzy logic is used in the representation of the genome. Also, some indirect approaches can be found. For example, in [22] it is proposed to use the GA in order to tune the parameters of a planner based on Artificial Fields.

Methods based on Ant Colony Optimization (ACO) algorithms have also been proposed [25]. In [26], the application of a game theory approach to aerial conflict resolution is presented. These techniques present a disadvantage: they are not well suited for applications that require a high level of scalability for their application in systems of many UAVs.

On the other hand, PSO application to path planning algorithm is still underdeveloped. An application for space vehicles path planning is presented in [27]. In addition, some efforts a planner with efficient rearranging capabilities is presented in [28] and applied to mobile robots in dynamic environments.

2.6. Optimal control methods

The goal of optimal control theory is to determine the control input that will cause a system to achieve the control objectives, satisfying the constraints, and at the same time optimize some performance criteria. The trajectory planning problem is in general solved following an open loop terminal control problem. This strategy allows all the constraints acting on the dynamical system, including the dynamic constraints, to be taken into account in such a way that the resulting trajectory is admissible. However this problem has an infinite number of solutions. To eliminate this redundancy optimal control techniques can be used to select only one of them, the trajectory that optimize a given criterion. Once an admissible trajectory or the optimal one has been found, a closed loop tracking control strategy is in general used to follow it. It is very difficult to solve analytically optimal control problems even for the simplest cases. For this,

numerical methods must be employed.

There are three main approaches to numerically solve continuous time optimal control problems such as problem Optimal Control Problem (OCP):

1. **Dynamic Programming (DP) methods**: The optimality criterion in continuous time is based on the Hamilton-Jacobi-Bellman partial differential equation [29].
2. **Indirect methods**: The fundamental characteristic is that they explicitly rely on the necessary conditions of optimality that can be derived from the Pontryagin's Maximum Principle [30]. [31] provides a thorough and comprehensive overview of necessary conditions for different types of unconstrained and constrained optimal control problems.
3. **Direct methods**: They can be applied without deriving the necessary condition of optimality. Direct methods are based on a finite dimensional parameterization of the infinite dimensional problem. The finite dimensional problem is typically solved using an optimization method, such as nonlinear programming (NLP) techniques. NLP problems can be solved to local optimality relying on the so called Karush-Kuhn-Tucker (KKT) conditions, which give first-order conditions of optimality. These conditions were first derived by Karush in 1939 [32], and some years later, in 1951, independently by Kuhn and Tucker [33].

2.6.1. Indirect Methods

The indirect method, also named indirect shooting, relies on Pontryagin's Maximum Principle [30]. Typically, the optimal control problem is turned into a two point boundary value problem containing the same mathematical information as the original one by means of necessary conditions of optimality. This method iteratively solves the initial value problem and then evaluates constraints to adjust initial conditions. It suffers from a high sensitivity in the final results from small changes in the initial conditions. Betts, in [34] notes that indirect shooting is best used when the dynamics are benign due to this high initial condition sensitivity. An example of benign dynamics is a low-thrust orbit trajectory where the states evolve slowly over a long time period. Finally, the indirect shooting method requires a good initial guess which can be difficult to obtain.

2.6.2. Direct Methods

Unlike indirect methods, Direct methods can be used to solve the optimal control problem without derivation of the necessary conditions for optimality. Direct methods operate by parameterizing the optimal control problem into a nonlinear programming problem (NLP). Direct shooting methods integrate the state equations directly between

the nodes, while Direct collocation methods use a polynomial approximation to the integrated state equations between the nodes. The following sections give more detail:

- **Direct Shooting**. The Direct shooting method integrates the trajectory during the optimization. The controls are piecewise between each point and can be piecewise constant, piecewise linear, etc. The integration is performed by using piecewise control and the constraints are then evaluated. Based on some function of the constraints, the initial conditions are adjusted and the process iterates until convergence [34]. A problem with Direct shooting is the sensitivity of the final state to minute changes in the initial state. In order to overcome this, the integration can be restarted at intermediate points, thus breaking the trajectory into smaller segments to which the Direct shooting method can be more easily applied successfully. Direct shooting has been widely used and was originally developed for military space applications [34], and general trajectory optimization [35].
- **Direct Collocation**. Direct collocation differs slightly. It similarly discretizes the state trajectory into a series of points and approximates the segments between the points with polynomials. However, the difference between the first derivative of the interpolating polynomial at the midpoint of a segment and the first derivative calculated from the equations of motion at the segment midpoint is used as the defect. If this defect approaches zero, the interpolating polynomials are ensured to be a good approximation of the actual states.

  Direct collocation was introduced by Dickmanns [36] as a general method for solving optimal control problems. The Direct collocation method has seen wide use in spacecraft and satellite research. One of the first was [37] in which it was applied into a low earth orbit booster problem and a supersonic aircraft time-to-climb problem. The method has also been used in determining finite-thrust spacecraft trajectories [38] and optimal trajectories for multi-stage rockets in [39]. The problem of low-thrust spacecraft trajectories is investigated in [40] [41] [42] [43]. In particular, Reference [40] uses higher order Gauss-Lobatto quadrature rules instead of the original trapezoidal and Simpson rules. Later papers focus on trajectory planning for unmanned vehicles. In [44] is presented a problem of multi-UAVs with different time to arrival. In [45] a path planning method for camouflage application is presented. And in [46] an optimal trajectory planning for guided projectile is presented considering 4 phases of flight.
- **Pseudospectral Methods.** Pseudospectral methods are a class of Direct methods that discretize the states and controls of a trajectory optimization problem with unevenly spaced nodes. High-order (order equal to the number of nodes) polynomials of the Lagrange interpolating form are used to approximate the states and controls over the interval of interest. These methods offer increased accuracy with fewer nodes compared to Direct methods due to the uneven discretization scheme.

Razzaghi and Elnagar [47] were among the first to apply these methods to control of dynamic systems. Later papers present optimal trajectory planning applications like in [48] in which a method based on a Legendre-Gauss-Lobatto (LGL) distribution of points is presented using UAVs and UGVs. As well as in [49] it is proposed another LGL implementation focus on optimal trajectories planning for an Eco-Driving System for automated vehicle. In [2] is presented a trajectory planning for autonomous landing for multiple UAVs.

2.7. Conclusions

A review of the main trajectory planning algorithms is presented. These algorithms solve the trajectory planning problem of a dynamically-constrained vehicle through an environment with obstacles.

Four significant groups of algorithms are described in this chapter: Graph search methods, probabilistic methods, bio-inspired methods and optimal control methods. Between them, some algorithms have been classified as optimal algorithms. There is a focus on optimal control methods because this thesis pays special attention to optimal trajectory planning.

It has been shown that optimal control methods have been widely used in aerospace application for trajectory planning. Specifically, Direct and Pseudospectral collocation method have become the main numerical optimization methods. In the next chapter, a detailed description of collocation methods will be presented.

3. COLLOCATION METHODS

The fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another point.

Claude Shannon

This chapter presents an overview of different colocation methods. An outline of the history and the origin of the collocation methods theory are presented. The following sections will demonstrate why collocation method has come to be one of mainly numerical optimization methods. First, an introduction to numerical methods is presented. Then the optimal control problem is introduced as well as its definition as nonlinear problem. Finally, a global classification of the different collocation methods and a full description of the theory of the most important algorithms are described.

3.1. Numerical analysis and spectral methods

Differential equations appear in many areas of engineering and science as a fundamental way of modeling the physical behavior of systems. Although analytic methods for the solution of differential equation exist, they are only applicable to specific and simple cases. In most applications, differential equations have to be solved using numerical approximation methods that simplify or discretize the problem so that it can be executed by any computer.

Numerical analysis for solving partial and ordinary differential equations constitutes a scientific discipline with well-known methods of finite differences and, finite elements, among others. There is a family of methods known as *spectral methods*, which are well suited for the numerical solution of a broad class of differential equations.

Given a differential equation with boundary conditions, the idea of spectral methods is to try to approximate a solution v of the differential equation by means of truncated series of basic functions:

$$v_N = \sum_{i=0}^{N} c_i \varphi_i \tag{3.1}$$

where φ_i are the basis functions, N+1 is the number of terms of the approximation and c_i are the real coefficients, which are unknown. Thus two main questions arise: from which function class should φ_i be chosen; and how should the expansion coefficients c_i be determined.

The φ_i function class should meet several criteria: the approximations $\sum_{i=0}^{N} c_i \varphi_i$ of v must converge rapidly with increasing N; it should be easy to determine the coefficients of derivatives; and it should be fast to convert between the c_i coefficients and the value of v_N at some set of nodes.

For periodic problems the choice usually is: $\varphi_i(x) = e^{ikx}$, which is the base of the well-known Fourier spectral method. If the function v is infinitely smooth and periodic together with all its derivatives, then the k-th coefficient of the Fourier expansion decays faster than any inverse power of k. This property is known as *spectral accuracy*. In practice this never happens but spectral accuracy is attainable also for non-periodic but smooth functions. Another classical result of the approximation theory is that for analytical functions exponential (or spectral) decay of the coefficients can be obtained for basic functions of the Jacobi polynomials type, which are mutually orthogonal over the interval (-1,1). Chebyshev and polynomials are the most important special cases of Jacobi polynomials.

Chebyshev polynomials are defined by the recurrent relations:

$$\begin{aligned} T_0(x) &= 1 \\ T_1(x) &= x \\ T_{n+1}(x) &= 2xT_n(x) - T_{n-1}(x) \end{aligned} \tag{3.2}$$

On the other hand, *Legendre polynomials* are defined by:

$$\begin{aligned} L_0(x) &= 1 \\ L_1(x) &= x \\ L_{n+1}(x) &= \frac{2n+1}{n+1} x L_n(x) - \frac{n}{n+1} L_{n-1}(x) \end{aligned} \tag{3.3}$$

An example of Chebyshev and Legendre polynomials for different values of n are shown in Figure 4.

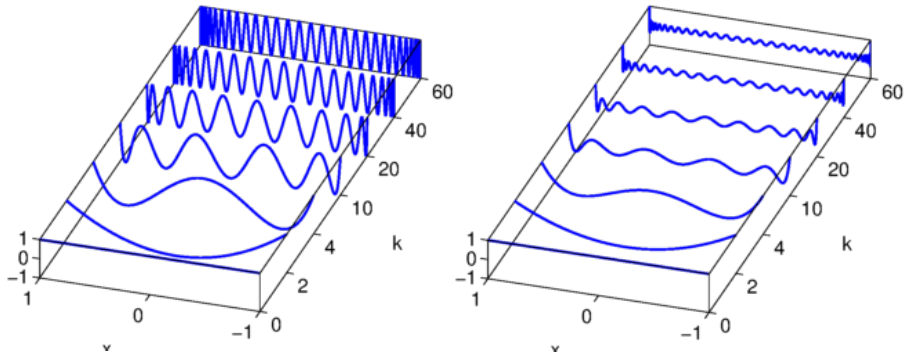


Figure 4: Chebyshev and Legendre polynomials for different values of *n*.

3.1.1. Pseudospectral approximation

Actually the spectral approximation defines a transformation from the physical space to the spectral space (like the Fourier coefficients in a Fourier transform). The coefficients c_i in the spectral approximation depend on all the values of the function v in the physical space and can only be computed by numerical integration. Since this cannot be performed exactly for arbitrary functions v, a set of approximate coefficients $\hat{c}_i$ can be obtained using an interpolating polynomial v_P of the function v defined by a finite set of interpolation points:

$$v_P = \sum_{i=0}^{N} \hat{c}_i \varphi_i \tag{3.4}$$

The interpolating polynomial satisfies the condition that it equals exactly the function in the interpolation points:

$$v_P(x_k) = \sum_{i=0}^{N} \hat{c}_i \varphi_i(x_k) = v(x_k) \tag{3.5}$$

These points are usually called *collocation points*, and are the points at which the differential equations are enforced; the approximation is equal to the exact solution in these points.

Now the question that arises is how to choose these collocation points x_k in a way that the residual (differences between the real function value and the approximation) in the remaining points is minimum. The answer is that the best election is what is known as *Gauss quadrature points*. A *quadrature rule* is just a method to approximate the integral of a function by a weighted sum of the values of the function in some specific points, called quadrature points. The Gauss quadrature rule generates an optimal approximation.

$$\int_{-1}^{1} f(x)dx \approx \sum_{i=0}^{N} w_i f(x_i) \tag{3.6}$$

where w_i are the *quadrature weights* and N is the number of points of the quadrature. The domain of integration for such a rule is conventionally taken as $[-1, 1]$.

Three of the more extended Gaussian quadrature's are known as Gauss, Gauss-Radau and Gauss-Lobatto respectively. The *Gauss quadrature points* are distributed on the interior of the interval but do not include the end points. On the other hand, *Gauss-Radau quadrature points* include one end point, and *Gauss-Lobatto quadrature points* include both end points of the interval. Figure 5 shows an example of the point's distribution in the interval $[-1, 1]$. Although the three types of quadrature points are similar, the effects in the numerical approximation are relevant. A detailed explanation can be found in [50].

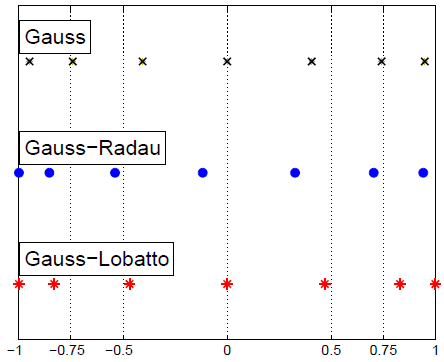


Figure 5: Distribution of the Gaussian quadrature nodes in normalized time.

Note that all three Gauss-type quadrature points are more concentrated on both end parts of the interval. This reduces the *Runge phenomenon*, the fact that the approximation error of a function by a polynomial is larger at the edges of the domain than in the interior. Then, although intuitively it may seem more adequate for a given problem to choose the collocation points evenly distributed in the domain, this is not possible, because it would introduce substantial numerical errors, and the number of collocation points would also be increased significantly, much slowing down the computations. This is a very important fact that will be used in the derivation of the new S-Adaptive Pseudospectral collocation method in Chapter 5.

It can be shown that spectral convergence is retained in replacing the continuous transform Equation (3.1) by the interpolating polynomial Equation (3.4) if the interpolation (or collocation) points are the corresponding Gauss-type quadrature points. Still, the coefficients $\hat{c}_i$ have to be computed. In practice however, the interpolation polynomials are sometimes written as a linear combination of *Lagrange interpolation polynomials* ϕ_i through the Gauss-type quadrature points:

$$v_P = \sum_{i=0}^{N} v_i \phi_i \tag{3.7}$$

The Lagrange polynomials are mathematically equivalent to the interpolating polynomials, and have the property that the coefficients are just given by the value of the function in the interpolation $v_i = v(x_k)$. Furthermore the derivative of the function can be computed just by a matrix multiplication, which makes numerical approximation faster and is the reason for choosing Lagrange polynomials.

From a general perspective, the fundamental interpretation of spectral methods was originally that the differential equation is transformed with the polynomial expansion in Equation (3.1). So, that we constantly work with the coefficients c_i as a representation of the function (the solution to the differential equation) instead of the usual explicit function values. But this leads to limitations when it comes to differential equations with variable coefficients or nonlinear problems. Instead we shall use the spectral representation for only a part of the solution procedure while we work directly with the

function in the physical space for the other part. This leads to what are known as *Pseudospectral methods*.

Different software implementations of Pseudospectral (PS) collocation methods have been published. All of which focus on the resolution of optimal control problems. The most recent are: software package *DIDO* [51], a Matlab implementation of Legendre PS method developed by the company Elissar Global. This implementation can be used by buying an academic license. Then GPOPS [52] is a Matlab implementation of a multi-interval Radau PS method developed by RP Optimization Research. There are different versions of this software, the most current being *GPOPS II*. A free academic license of *GPOPS* is available. *PSOPT* [53] is an open source software package written in C++. This implementation uses both Legendre and Chebyshev polynomials. Full software is distributed under the terms of the GNU Lesser General Public License. Another software implementation *OTIS* [54] was developed by Boeing for NASA, *SOCS* [34], *DIRCOL* [55], *GESOP* [56], *ICLOCS* [57], and *ACADO* [58].

3.2. Optimal control

Optimal control is the process of finding control and state law for a dynamic system over a period of time, so that the performance of the system is optimally oriented. The index which is used to quantify the performance of the system might include, for example, a measure of the control effort, a measure of the tracking error, a measure of energy consumption, a measure of the amount of time taken to reach a target, or any other quantity of importance to the operation of the system.

There are various types of optimal control problems, depending on the performance index, the type of time domain (continuous, discrete), the presence of different types of constraints, and what variables are free to be chosen. The formulation of an optimal control problem usually requires:

- A mathematical model of the system to be controlled.
- A specification of all boundary conditions on states, and constraints to be satisfied by states and controls.
- A specification of the performance index.

Optimal control problems arise in a wide variety of fields including virtually all branches of engineering, economics, and medicine. Due to the increasing complexity of optimal control applications, over the past two decades the subject of optimal control has transitioned from theory to computation.

In particular, computational optimal control has become a science in and of itself, resulting in a variety of numerical methods and its corresponding software implementations. The roots of these methods can be traced back to the works of Bernoulli and Euler [59]. The simplicity in this approach is based on a wide range of

deep theoretical issues that lie at the intersection of approximation theory, control theory and optimization. An example of this progress in large-scale computation and the robustness of the approach can be seen in [37], [60] , [61], [62], [63] just to name a few.

Numerical methods for solving optimal control problems are divided into two major classes: Indirect methods and Direct methods. In an *Indirect method*, the calculus of variations is used to determine the first-order optimality conditions of the original optimal control problem. The indirect approach leads to a multiple-point boundary-value problem that is solved to determine candidate optimal trajectories called *extremals*. Each of the computed extremals is then examined to see if it is a local minimum, maximum, or saddle point. Of the locally optimizing solutions, the particular extremal with the lowest cost is chosen.

On the other hand, in a *Direct method*, the state and/or control of the optimal control problem is discretized in some manner and the problem is transcribed to a nonlinear optimization problem or nonlinear programming problem (NLP). The NLP is then solved by using well known optimization techniques. Since a numerical solution of a NLP problem is much faster and efficient than solving the Indirect method of differential equations that satisfies endpoint and/or interior point conditions, Direct methods have been widely adopted for solving optimal control problems in real applications.

3.2.1. General formulation of the optimal control problem

The objective of an optimal control problem in continuous time is to determine the state and control variables that optimize an objective function, subject to dynamic constraints and boundary conditions. In collocation methods the optimal control problem is given by the following the standard form (known as *Bolza form*). This is a classical problem in calculus of variations on the conditions for an extremum, formulated in 1913 by O. Bolza. It can be formulated as:

Determine the state (equivalently, the *trajectory* or path), $x(t) \in \mathbb{R}^n$, the control input $u(t) \in \mathbb{R}^m$, the initial time, $t_0 \in \mathbb{R}$, and the terminal time, $t_f \in \mathbb{R}$, (where $t \in [t_0, t_f]$ is the independent variable) that optimizes the performance index:

$$J = \varphi\big(x(t_0), t_0, x\big(t_f\big), t_f\big) + \int_{t_0}^{t_f} g(x(t), u(t))dt \tag{3.8}$$

subject to the *dynamic constraints* (differential equations that describe the system dynamics)

$$\dot{x}(t) = \frac{dx}{dt} = f(x(t), u(t)) \tag{3.9}$$

inequality *path constraints* (continuous constraints of the state and control input during the whole trajectory $t \in [t_0, t_f]$):

$$C\big(x(t), u(t)\big) \leq 0 \tag{3.10}$$

and the *boundary conditions*

$$E\big(x(t_0), t_0, x\big(t_f\big), t_f\big) = 0 \tag{3.11}$$

where $f: \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$, $C: \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^s$, and $E: \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^q$. Furthermore, the function $\varphi: \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ is the *Mayer cost* and $g: \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ is the *Lagrangian cost*. Mayer cost function focuses on the boundaries of the problem (essentially focusing on the optimization in the initial and final states of the problem). On the other hand, Lagrangian cost focuses on optimization throughout the entire process.

3.3. Nonlinear optimization

A key ingredient to solving optimal control problems is the ability to solve nonlinear optimization or *nonlinear programming problems* (NLPs). An NLP takes the following general mathematical form:

Determine the vector of decision variables $z \in \mathbb{R}^n$ that minimize the cost function $f(z)$ subject to the algebraic constraints: (3.12)

$$g(z) = 0$$
$$h(z) \leq 0$$

where $g(z) \in \mathbb{R}^m$ and $h(z) \in \mathbb{R}^p$.

The NLP may either be *dense* (i.e., a large percentage of the derivatives of the objective function and the constraint functions with respect to the components of z are nonzero) or may be *sparse* (i.e., a large percentage of the derivatives of the objective function and the constraint functions with respect to the components of z are zero). Dense NLPs typically are small (at most consisting of a few hundred variables and constraints), while sparse NLPs are often extremely large (ranging anywhere from thousands to millions of variables and constraints).

NLP problems are usually solved using gradient-based methods. In a gradient-based method, an initial guess is made of the unknown decision vector z. At the k^{th} iteration, a search direction, p_k, and a step length, α_k, are determined. The search direction provides a direction in $\mathbb{R}^n$ along which to change the current value z_k while the step length provides the magnitude of the change to z_k. The update from z_k to z_{k+1} then has the form:

$$z_{k+1} = z_k + \alpha_k p_k \tag{3.13}$$

In the case of minimization, the search direction is chosen to "sufficiently decrease" the objective function in the form:

$$f(z_{k+1}) \leq f(z_k) + K\alpha_k \nabla f^T(z_k) p_k \tag{3.14}$$

and K is a parameter between 0 and 1. The most commonly used gradient-based NLP solution methods are *sequential quadratic programming* (SQP) and *interior-point* (IP) or *barrier* methods.

Extensive research has been done over the past two decades in SQP and IP methods. This research has led to extremely efficient and robust software programs for the numerical solution of NLPs. Examples of well-known software that use SQP methods include the dense NLP solver NPSOL [64] and the sparse NLP solvers SNOPT [3] and SPRNLP. Well known sparse interior point NLP solvers include BARNLP, LOQO, KNITRO, and IPOPT.

3.4. Collocation methods

Collocation methods are a set of methods focused on solving optimal control problems. These methods are based on transcription of a continuous-time problem into a finite-dimensional NLP problem. Collocation methods enforce a suitable interpolating function through the state values, and maintain the state derivatives at the collocation points spanning one time interval.

Collocation methods are characterized by the numerical integration method and the discretization method. In the case of discretization, there are three important concepts that have to be differentiated: *intervals* and *sub-intervals*, *nodes* and *collocation points*. The problem time line is defined by one interval $[t_0, t_f]$, where t_0 is the initial time in the problem and t_f being the final time. This interval could be split into sub-intervals $[t_{k-1}, t_k]$, where k is the number of sub-intervals. Every interval or sub-interval is splinted by nodes. Nodes definition is very close to the discretization concept of the problem. Distribution of the nodes throughout the segment is different in every collocation method. For example, Figure 6 shows the distribution of nodes in the case of Direct collocation method as well as in the case of PS collocation method. It can be seen that in Direct collocation, that distribution of nodes is equally spaced in the time interval $[t_0, t_f]$, while it does not happen in PS method.

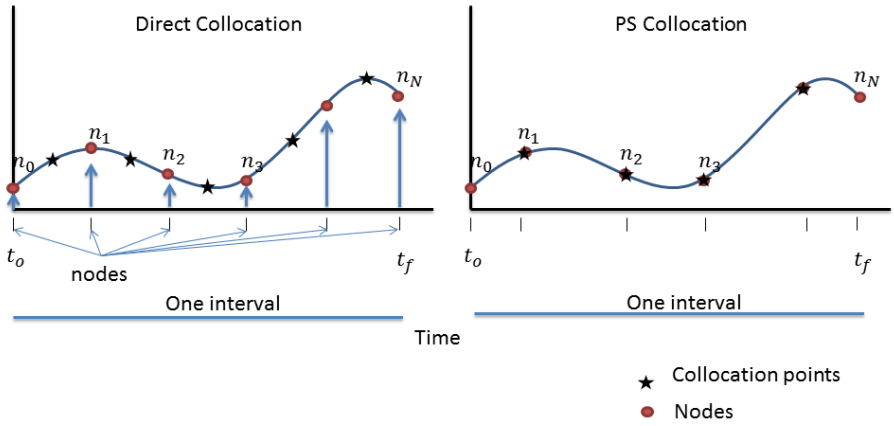


Figure 6: General distribution of nodes and collocation points in collocation methods.

Collocation points are the points in which dynamic equations of motion are enforced through quadrature rules or interpolation. Depending on the numerical integration method, the collocation point is either set at the midpoint between two nodes, which happens in Direct collocation method, or the collocation points are set at the same nodes, as seen in PS collocation methods. Specifically in PS method, collocation points are considered the sub-set of nodes $[n_1,\ n_2,\ \ldots, n_{N-1}]$ where N is the number of nodes. Figure 7 shows the distribution of nodes and collocation points in PS collocation methods. Shown above is the case of one interval, and below, two intervals or sub-intervals are presented.

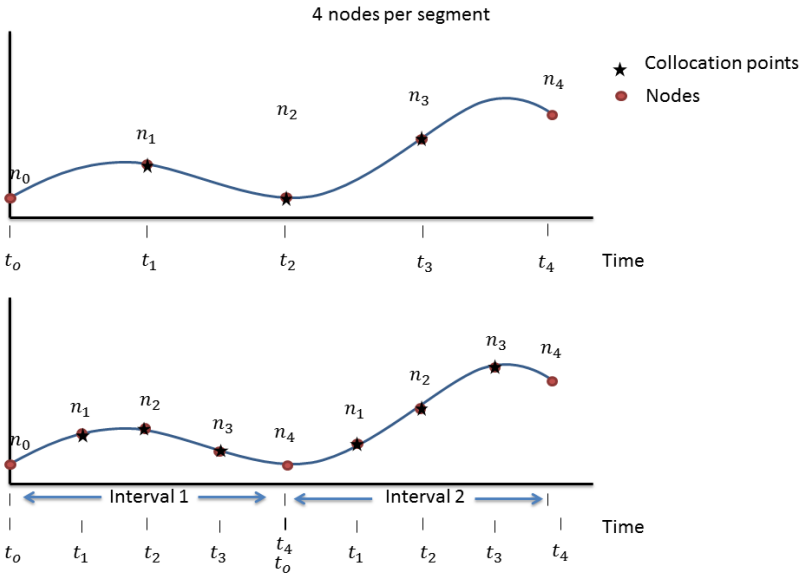


Figure 7: Distribution of nodes and collocation points in multi-interval PS methods.

In this example, four nodes are used, and it can be seen that the limit nodes n_0 and n_4 are never used as collocation point, in any case.

Considering this specification, collocation methods can be summarized into three groups of technics: *Direct collocation*, *PS collocation* and *sub-interval PS collocation* methods. An overview of them is presented next.

3.4.1. Direct collocation method

Direct collocation method is one of the simplest collocation methods. It is also known as *Hermite-Simpson collocation method* [65] [66] because of the integration method, or simply *Direct Collocation with NonLinear Programming* (DCNLP). This method is characterized by its use of equally spaced nodes in time to discretize the optimization problem given in Equations (3.8) - (3.11). Only one interval with the entire problem is considered. A third-order Hermite interpolating polynomial is used to interpolate the variables in the collocation point which is addressed in the midpoint of every two nodes.

An overview of the method is presented based on Hargraves and Paris [37]. The problem is discretized by nodes and every piece between two nodes is approximated with a Hermite cubic interpolating polynomials. Hermite interpolating polynomials are defined in terms of the values at nodes $x_i = x(t_i)$ and their first derivatives $\dot{x}_i = \dot{x}(t_i)$.

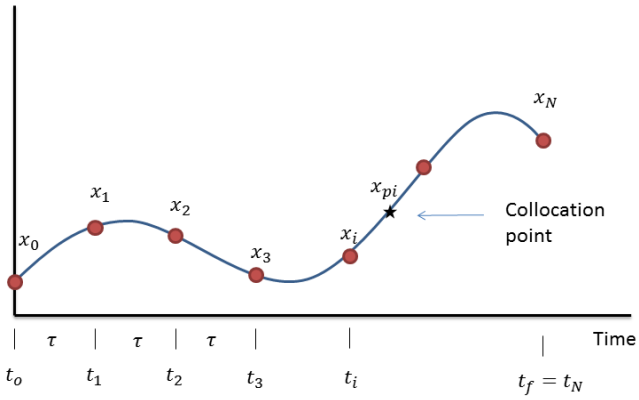


Figure 8: Direct collocation method scheme.

To ensure the approximating polynomials accurately represent the equations of motion, the derivative at the midpoint of each polynomial segment, $\dot{x}_{pi}(0.5)$, is compared to the equations of motion evaluated using the states at the interpolated segment midpoint, $f(x_{pi}(0.5), u_{pi}(0.5))$. The name of this midpoint is known as collocation point, x_c and u_c respectably.

$$x_{ci} = x_{pi}(0.5) = \frac{1}{2}(x_i + x_{i+1}) + \frac{\tau}{8}(\dot{x}_i - \dot{x}_{i+1}) \tag{3.15}$$
$$u_{ci} = u_{pi}(0.5) = (u_i + u_{i+1})/2$$

Additionally, the slope of the interpolated states at the collocation points is:

$$\dot{x}_{ci} = -\frac{3}{2\tau}(x_i - x_{i+1}) - \frac{1}{4}(\dot{x}_i + \dot{x}_{i+1}) \tag{3.16}$$

Then, defect is defined as:

$$\Delta = d_i = f(x_{ci}, u_{ci}) - \dot{x}_{ci} \tag{3.17}$$

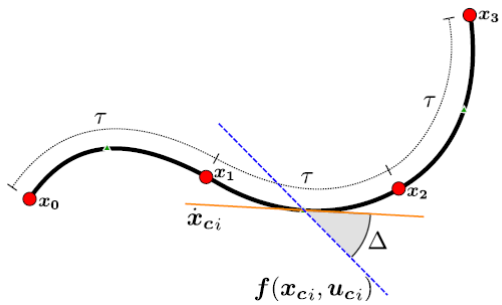


Figure 9: Interpolation diagram of Direct collocation.

The accuracy in the approximating polynomial of the equations of motion is achieved when Δ is driven toward zero. In Direct collocation, the convergence of the solution is given by the selection of τ, and consequently, increasing the number of nodes.

Now the NLP problem is parameterized by the state and control vectors at each node:

$$X^T = [x_1^T\ u_1^T\ x_2^T\ u_2^T\ x_3^T\ u_3^T\ \ldots\ x_n^T\ u_n^T] \tag{3.18}$$

and the constraint functions of problem consist of the defect vectors and the problem constraints evaluated at each node:

$$\begin{aligned} d_1(x,y) &= 0 \quad c(x_1, y_1) \leq 0 \\ d_2(x,y) &= 0 \quad c(x_2, y_2) \leq 0 \\ \cdots &\qquad \ldots \\ \cdots &\qquad \ldots \\ d_n(x,y) &= 0 \quad c(x_n, y_n) \leq 0 \end{aligned} \tag{3.19}$$

Now, through a solver method, the NLP problem is solved. The goal is to find the

parameter vector Equation (3.18) that minimizes the objective function J, subject to the constraints in (3.19). The objective function can typically be evaluated using some type of numerical integration along the state and control vectors.

This method is also named h-method between collocation methods [52]. Convergence in Direct collocation method is achieved by increasing the number of nodes placement [60] [67] [68].

There are other versions of this method. The new methods use a bigger interpolating polynomial instead of 3th order polynomial. For example, one method uses an *N-th order Hermite* interpolating polynomial. In this case, the distribution of the collocation points between two nodes is set by Legendre-Gauss-Lobatto distribution of points. This method is named *Hermite-Legendre-Gauss-Lobatto* (HLGL) collocation method [69]. Other similar methods are based, for instance, on Gauss or Radau distribution of point scheme's [70] [71].

3.4.2. Pseudospectral collocation method

Recently, a great deal of research has been done in the class of Gaussian quadrature orthogonal collocation methods, commonly known as Pseudospectral (PS) collocation method [62] [72] [73]. Some improvements and advances are introduced in comparison to Direct collocation method. PS collocation focuses on a global perspective of the problem, while Direct collocation solves the problem piece by piece. In PS methods, the entire problem is approximated by a quadrature rule as Gauss, Gauss-Lobatto, etc. while in Direct collocation, the problem is solved by $(N-1)$ Hermite polynomials, where N is the number of nodes. This change from local to global happens because it has been proven in numerical analysis that more sophisticated node selection methods are able to achieve significantly improved accuracy with much less amount of nodes. PS collocation method has a simple structure and it converges at an exponential rate [74] [75] [76], so it meets the inequality:

$$||\mathrm{f(t)} - I_N \mathrm{f(t)}||_{L^2} \leq \frac{C}{N^m} \tag{3.20}$$

where $f(t)$ is a function, I_N is the polynomial interpolation of the function with N points. m is the smoothness of $f(t)$ and C is a constant independent of N. As $N \to \infty$, the polynomial interpolation at the LGL nodes for example, converges to $f(t)$ under the L^2 norm at a rate of $1/N^m$. Note if $f(t)$ is C^∞, then $m = \infty$. This implies that the polynomial interpolation at these nodes converges at a spectral rate (faster than any given polynomial rate).

PS collocation method is characterized because variables of the entire time interval are typically approximated by a Lagrange polynomial, where the support points of this polynomial are chosen based on their association with a Gaussian quadrature. These methods are fully conditioned by the quadrature rule used in the approximation of the solution. This affects to the distribution of nodes as well as numeric integration of the

cost function. Definitions of nodes, also named quadrature nodes, are given by the zeros of the derivative of an orthogonal Jacobi polynomial (meanly Legendre or Chebyshev polynomial). Thus, between nodes and interpolants, a very large family of PS methods can be used to solve optimal control problems. The most popular are:

- Legendre-Gauss (LG) PS collocation method [73].
- Legendre-Gauss-Radau (LGR) PS collocation method [77] [78] [79].
- Legendre-Gauss-Lobatto (LGL) PS collocation method [62].
- Chebyshev-Gauss-Lobatto (CGL) PS collocation method [80].

A complete description of PS collocation method is presented next. This outline describes in detail a LGL PS collocation method [62] [81] [82] [83] [84], although this demonstration may be applicable to any PS methods.

The problem defined in Equations (3.8) - (3.11) is translated from time domain $t \in [t_0, t_f]$ to normalized domain $\tau \in [-1, 1]$ by related transformation:

$$\tau = \frac{2t - (t_0 + t_f)}{t_f - t_0} \tag{3.21}$$

The inverse is given by

$$t = \frac{t_f - t_0}{2}\tau + \frac{t_0 - t_f}{2} \tag{3.22}$$

and it is noted that

$$\frac{dt}{d\tau} = \frac{t_f - t_0}{2} \tag{3.23}$$

Figure 10 shows an example of how time domain is changed to $\tau \in [-1, 1]$, and time is discretized into N nodes.

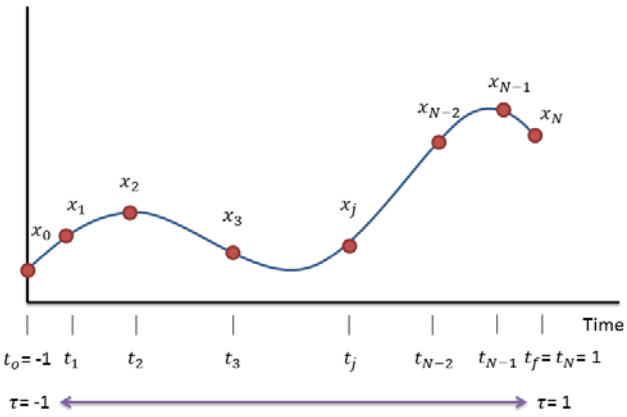


Figure 10: Pseudospectral collocation method scheme.

Then, the problem is formulated by finding the control vector function $u(\tau)$ and the corresponding state vector $x(\tau)$ that minimize the cost function

$$J(x,u) = \varphi\big(x(-1), x(1)\big) + \int_{-1}^{1} g(x(\tau), u(\tau))d\tau \tag{3.24}$$

subject to the dynamic constraints

$$\dot{x}(\tau) = \frac{dx}{d\tau} = f(x(\tau), u(\tau)) \tag{3.25}$$

inequality path constraints

$$C\big(x(\tau), u(\tau)\big) \leq 0 \tag{3.26}$$

and the boundary conditions

$$E\big(x(-1), x(1)\big) = 0 \tag{3.27}$$

Due to the use of the Legendre-Gauss-Lobatto quadrature, the problem is discretized by LGL distribution of nodes. Figure 11 shows an example of the distribution of 16 nodes. LGL nodes are defined by domain $\tau \in [-1, 1]$ as $\tau_0 = -1 < \tau_1 < \tau_2 < \cdots < \tau_N = 1$ where τ_1, τ_2, ... τ_{N-1} are the roots of the derivate of the N-th order Legendre polynomial.

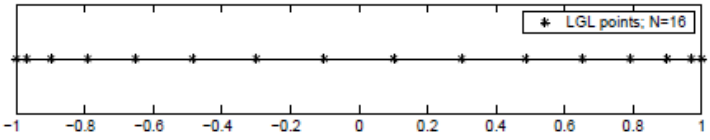


Figure 11: Diagram example of the distribution of 16 LGL nodes.

With $x(\tau)$ and $u(\tau)$ being the state and control vector of the problem, they will be approximated by $\boldsymbol{X}(\tau) \in \mathbb{R}^r$ and $\boldsymbol{U}(\tau) \in \mathbb{R}^r$. In the discretization, the state variables at the nodes are approximated by column vectors:

$$x(\tau_j) \approx \boldsymbol{X}^j = [\boldsymbol{X}_1^j, \boldsymbol{X}_2^j, \dots, \boldsymbol{X}_r^j]^T \tag{3.28}$$

where r is the number or variable that composes the state vector, and $\tau_j = [\tau_0, \tau_1, \dots, \tau_N]$ are the nodes. Similarly, $u(\tau_j) \approx \boldsymbol{U}^j$ is the approximation of the control inputs. Thus, a discrete approximation of the function $x_i(\tau)$ across all nodes is the row vector:

$$\boldsymbol{X}_i = [\boldsymbol{X}_i^0, \boldsymbol{X}_i^1, \dots, \boldsymbol{X}_i^N] \tag{3.29}$$

In the notations, the discrete variables are denoted by bold capital letters. Subscript i is used to discriminate the variable in the state vector $i = 1, 2, \dots, r$ and superscript j is used to discriminate the nodes $j = 0, 1, \dots, N$. Note that $\boldsymbol{X}_i^0$ and $\boldsymbol{X}_i^N$ are the variables i of state vector at initial and final instant.

Then the entire state vector can be represented this way:

$$\boldsymbol{X} = \begin{bmatrix} X_1^1 & \cdots & X_1^N \\ \vdots & \ddots & \vdots \\ X_r^1 & \cdots & X_r^N \end{bmatrix} \tag{3.30}$$

A continuous approximation of $x(\tau)$ is defined by its polynomial in the Lagrange form, denoted $x_i^N(\tau)$:

$$\begin{aligned} x_i(\tau) \approx x_i^N(\tau) &= \sum_{j=0}^{N} \boldsymbol{X}_i^j \phi_j(\tau) \\ u_i(\tau) \approx u_i^N(\tau) &= \sum_{j=0}^{N} \boldsymbol{U}_i^j \phi_j(\tau) \end{aligned} \tag{3.31}$$

where ϕ_j are the Lagrange basis polynomials:

$$\phi_j = \frac{1}{N(N+1)L_N(\tau_j)} \frac{(\tau^2 - 1)\dot{L}_N(\tau)}{\tau - \tau_j} \tag{3.32}$$

and L_N is the N-th order Legendre polynomial function, because we are using Legendre-Gauss-Lobatto method.

The derivative of $x_i^N(\tau)$ at the LGL node τ_j is easily computed by differentiating the approximation of Equation (3.31) with respect to τ.

$$\frac{dx_i^N(\tau)}{d\tau} = \sum_{j=0}^{N} \boldsymbol{X}_i^j \dot{\phi}_j(\tau) \tag{3.33}$$

where $\dot{\phi}_j(\tau_l)$ is an (N+1) X (N+1) matrix also named differentiating matrix, D, given by:

$$D = D_{j,l} = \begin{cases} \dfrac{L_N(\tau_j)}{L_N(\tau_l)} \dfrac{1}{(\tau_j - \tau_l)} & j \neq l \\ -\dfrac{N(N+1)}{4} & j = l = 0 \\ \dfrac{N(N+1)}{4} & j = l = N \\ 0, & otherwise \end{cases} \tag{3.34}$$

Finally, the discretization of the cost function of Equation (3.24) is given by the LGL quadrature

$$J(x,u) \approx \boldsymbol{J}(\boldsymbol{X},\boldsymbol{U}) = \varphi(\boldsymbol{X}^0,\boldsymbol{X}^N) + \sum_{j=0}^{N} g(\boldsymbol{X}^j,\boldsymbol{U}^j)w_j \tag{3.35}$$

where w_j are the LGL quadrature weights, defined by:

$$w_j = \frac{2}{N(N+1)} \frac{1}{[L_N(\tau_j)]^2} \tag{3.36}$$

Then, in summary, the entire problem defined in Equations (3.8) - (3.11) is translated to a discrete problem where the goal is to find a state vector $\boldsymbol{X}^j \in \mathbb{X}$ and a control vector $\boldsymbol{U}^j \in \mathbb{U}$, with $j = 0, 1, \dots, N$, that minimize the cost function:

$$J(\boldsymbol{X},\boldsymbol{U}) = \varphi(\boldsymbol{X}^0,\boldsymbol{X}^N) + \sum_{j=0}^{N} g(\boldsymbol{X}^j,\boldsymbol{U}^j)w_j \tag{3.37}$$

subject to the discrete dynamic

$$\sum_{j=0}^{N} D_{j,l}\boldsymbol{X}^j - f(\boldsymbol{X}^j,\boldsymbol{U}^j) = 0 \tag{3.38}$$

path constraints

$$C(\boldsymbol{X}^j,\boldsymbol{U}^j) \leq 0 \tag{3.39}$$

and the end points:

$$E(\boldsymbol{X}^1,\boldsymbol{X}^N) = 0 \tag{3.40}$$

PS methods are also named as p-method in some literature[z]. Convergence of the p-method is achieved by increasing the degree of the Lagrange polynomial approximation (increasing the number of nodes).

3.4.3. Sub-interval Pseudospectral collocation method

Sub-interval PS method introduces a new concept, which is a hybrid between Direct collocation and PS method. In Direct method the convergence is achieved by increasing the number of peace or sub-problem. A similar concept is applied here. In this case, the time interval $[t_0, t_f]$ is split into sub-intervals $[t_{k-1}, t_k]$, and every sub-interval is solved by classic PS method. Now, border equality's of sub-interval are added to PS methods constraints because continuity must be maintained. Differences between various sub-interval PS methods are given by how the problem is split, and by what PS-method is used to solved every sub-interval.

Note that in literature sub-intervals are also called segments. This concept could be confusing because the meaning of segment changes depends upon the author. In Direct collocation method a segment is defined by the time interval between two nodes. Due to the emergence of multi-interval PS method, segmentation definition has redefined by the sub-interval (of the initial interval$[t_0, t_f]$).

Sub-interval PS methods are very recent, only a few implementations of this technic can be found in literature. One example of this method is *hp-Adaptive collocation* [85]. This method was developed in order to significantly reduce the size of the finite-dimensional approximation problem, and thus improve computational efficiency of solving the NLP. This method increases the order of the polynomial when the solution is smooth and the convergence rate is small, and also splits the problem into sub-interval when solution changes rapidly. Then, this method merges h methods with p method, where the segmentation rate and the order of the polynomial interpolation could be different in every sub-interval. hp-Adaptive collocation is an iterative method and the stop condition is given when a suitable error or a bounded iteration is achieved.

These features ensure this method has a desirable adaptation to any optimization problem (this is a general purpose method), and the convergence rate is better than the classic PS method in most of problems. On the other hand, this is an iterative method and depending on the problem, the convergence of the solution may be not achieved in a suitable limit of iteration.

Following this method's philosophy, a new method named *S-Adaptive collocation method* has been developed in this Thesis. This method focuses on an advanced method of segmentation of the problem, and then, every sub-interval is solved by a fixed N-th Lagrange polynomial. Using this way, the convergence of the solution is achieved by a low number of nodes, and consequently, a significant improvement in computation time compared with hp-Adaptive is achieved. S-Adaptive has been especially designed for solving trajectory planning problems. This method will be fully detailed in Chapter 4.

A complete description of sub-interval PS collocation method is presented next. This outline uses the method presented by Christopher Darby in [86]. Sub-interval methods are considered to reformulate the problem defined in Equations (3.8) - (3.11) by dividing it into K intervals. An interval k is defined to begin at t_{k-1} and end at t_k,

where $t_0 < t_1 < t_2 < \cdots < t_k = t_f$. The time domain in each sub-interval k, $[t_{k-1}, t_k]$, is changed to normalized domain $\tau \in [-1, 1]$ by the transformation:

$$\tau = \frac{2t - (t_{k-1} + t_k)}{t_k - t_{k-1}} \tag{3.41}$$

An illustration of this concept description is given by Figure 12 in which the entire problem has been split into two sub-intervals.

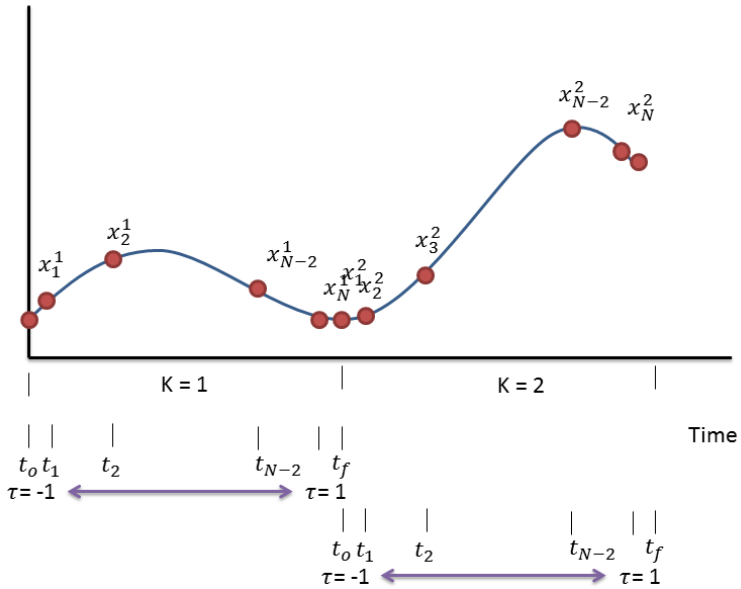


Figure 12: Multi-interval Pseudospectral collocation method scheme.

Follows the notation given in the PS method (previous Section 3.4.2) now, the problem is described as: Let $x^{(k)}(\tau)$ and $u^{(k)}(\tau)$ be the state and control in the k^{th} interval, and it is noted that the continuity in the state is enforced between intervals, therefore, $x^{(k)}(+1) = x^{(k+1)}(-1)$ with $(k = 1, \dots, K-1)$. Therefore, the optimal control problem is expressed in K intervales is defined by:

The minimization of the cost function

$$J = \varphi\left(x^{(1)}(-1), t_{k-1}, x^{(K)}(1), t_K\right) + \sum_{k=1}^{K} \frac{t_k - t_{k-1}}{2} \int_{-1}^{1} g(x^{(k)}(\tau), u^{(k)}(\tau)) d\tau \tag{3.42}$$

subject to the dynamic constraints

$$\dot{x}(\tau) = \frac{dx^{(k)}(\tau)}{d\tau} = \frac{t_k - t_{k-1}}{2} f(x^{(k)}(\tau), u^{(k)}(\tau)), \quad (k = 1, \dots, K) \tag{3.43}$$

inequality path constraints

$$\frac{t_k - t_{k-1}}{2} C\left(x^{(k)}(\tau), u^{(k)}(\tau)\right) \leq 0, \quad (k = 1, \dots, K) \tag{3.44}$$

the boundary conditions

$$E\left(x^{(1)}(-1), t_0, x^{(K)}(1), t_K\right) = 0 \tag{3.45}$$

and the interior point constraints

$$x^{(k)}(+1) - x^{(k+1)}(-1) = 0, (k = 1, \dots, K-1). \tag{3.46}$$

With this every k^{th}segment is solved as per classic PS collocation problem. The estate and control vector are then approximated by:

$$\begin{aligned} x^{(k)}(\tau) &\approx \boldsymbol{X}^{(k)}(\tau) = \sum_{j=0}^{N} \boldsymbol{X}_j^{(k)} \phi_j^{(k)}(\tau) \\ u^{(k)}(\tau) &\approx \boldsymbol{U}^{(k)}(\tau) = \sum_{j=0}^{N} \boldsymbol{U}_j^{(k)} \phi_j^{(k)}(\tau) \end{aligned} \tag{3.47}$$

Where $\boldsymbol{X}_j^{(k)} = \boldsymbol{X}^{(k)}(\tau_j^{(k)})$ and $\tau_j^{(k)}$is $(\tau_0^{(k)}, \tau_1^{(k)}, \tau_2^{(k)}, \dots, \tau_N^{(k)})$ and they are the LGL nodes in domain $\tau \in [-1, 1]$. It is noted that all vector functions of time are row vectors, $\boldsymbol{X}^{(k)}(\tau) = [X_1^{(k)}, X_2^{(k)}, \dots, X_r^{(k)}]$ where r is the number of variables of state vector; subscript j is used to discriminate the nodes $j = 0, 1, \dots, N$ and ϕ_j is the Lagrange basis polynomials:

$$\phi_j = \frac{1}{N(N+1)L_N(\tau_j)} \frac{(\tau^2 - 1)\dot{L}_N(\tau)}{\tau - \tau_j} \tag{3.48}$$

where L_N is the N-th order Legendre polynomial function. The aforementioned being completed using Legendre-Gauss-Lobatto method.

Next, an approximation to the derivative of the state is given by differentiating the approximation of Equation (3.47) with respect to τ.

$$\frac{d\boldsymbol{X}^{(k)}(\tau)}{d\tau} = \sum_{j=0}^{N} \boldsymbol{X}_j^{(k)} \dot{\phi}_j^{(k)}(\tau) \tag{3.49}$$

where $\dot{\phi}_j^{(k)}(\tau)$ is a (N+1) X (N+1) Gauss pseudospectral differentiation matrix D in the k^{th} interval, given by:

$$D = D_{j,l} = \begin{cases} \dfrac{L_N(\tau_j)}{L_N(\tau_l)} \dfrac{1}{(\tau_j - \tau_l)} & j \neq l \\ -\dfrac{N(N+1)}{4} & j = l = 0 \\ \dfrac{N(N+1)}{4} & j = l = N \\ 0, & otherwise \end{cases} \tag{3.50}$$

Then as happens in the previous section, the entire problem defined in Equations (3.8) - (3.11) is defined as NLP (considering all the *K* intervals). Now the goal is defined we find that the state vector $\boldsymbol{X}_j^{(k)} \in \mathbb{X}$ and a control vector $\boldsymbol{U}_j^{(k)} \in \mathbb{U}$ minimize the cost function:

$$J \approx \varphi\left(\boldsymbol{X}_0^{(1)}, t_0, \boldsymbol{X}_N^{(K)}, t_K\right) + \sum_{k=1}^{K} \sum_{j=1}^{N} \left(\frac{t_k - t_{k-1}}{2}\right) w_j^{(k)} g(\boldsymbol{X}_j^{(k)}, \boldsymbol{U}_j^{(k)}) \tag{3.51}$$

where $w_j^{(k)}$ are the Legendre-Gauss-Lobatto weights in interval k:

$$w_j = \frac{2}{N(N+1)} \frac{1}{[L_N(t_j)]^2} \tag{3.52}$$

subject to the discrete dynamic

$$\sum_{j=0}^{N} \boldsymbol{X}_j^{(k)} D_{ij}^{(k)} - \left(\frac{t_k - t_{k-1}}{2}\right) f(\boldsymbol{X}_j^{(k)}, \boldsymbol{U}_j^{(k)}) = 0 \tag{3.53}$$

the inequality path constraints:

$$\frac{t_k - t_{k-1}}{2} \boldsymbol{C}^{(k)}\left(\boldsymbol{X}_j^{(k)}, \boldsymbol{U}_j^{(k)}\right) \leq 0 \tag{3.54}$$

and the end points:

$$E(\boldsymbol{X}_1^1, \boldsymbol{X}_N^K) = 0 \tag{3.55}$$

It should be noted that the formulation presented here allows an arbitrary number of intervals K with arbitrary placement, and with an arbitrary number N of nodes.

3.4.4. Summary of methods

Figure 13 shows an overview of the main numerical optimal control methods used for trajectory optimization. In the figure the intermediate boxes show several methods. In case of collocation methods, it can be seen that there are mainly two large groups: Direct collocation and PS collocation methods. Then there is a third group, which

progresses on the sub-interval method.

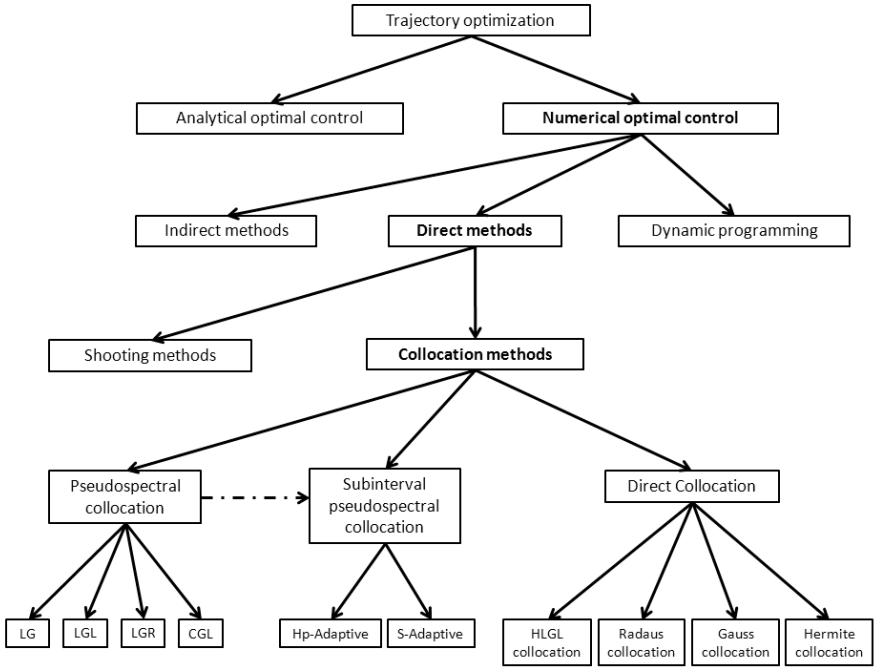


Figure 13: Taxonomy of trajectory optimization methods using optimal control.

3.5. Conclusions

Numerical analysis for solving partial and ordinary differential equations constitutes a scientific discipline in which interest is rapidly increasing. The Spectral and Pseudospectral methods compose a set of methods that approximate a solution of a differential equation by means of truncated series of basic functions.

Numerical methods for solving optimal control problems are mainly divided into Direct and Indirect methods. With the case of Direct methods, the state and/or control vectors of the optimal control problem are discretized in some manner and the problem is then transcribed to a nonlinear programming problem (NLP). Finally the NLP is solved by use of optimization techniques. Thus numerical solutions to NLP problems are faster and more effective than solving a system of differential equations that satisfies endpoint and/or interior point conditions (such as Indirect methods).

Collocation methods are a set of methods for solving optimal control problems that follow the Direct method's philosophy. Collocations methods discretize the state and control vectors into a set of collocation points, they are then approximated by a suitable polynomial. The dynamic equations of the system are enforced through quadrature rules at the collocation points and the objective function is typically evaluated using

some type of numerical integration along the state and control vectors. Collocation methods can be summarized into three groups of techniques: Direct collocation, PS collocation and sub-interval PS collocation methods. However, between PS methods, can be distinguished different depending of the distribution of points, interpolating polynomial or integrations rules. Some examples are Legendre-Gauss-Radau, Legendre-Gauss-Lobatto and Chebyshev-Gauss-Lobatto, all depict this well.

In the next chapter, the optimal trajectory planning problem will be introduced and a new algorithm (based on collocation) specialized in solving trajectory planning method will be.

4. MULTI-VEHICLE OPTIMAL TRAJECTORY PLANNING

Everything you can imagine is real.

Pablo Picasso

4.1. Introduction

The problem of safe trajectory generation for autonomous vehicles can be described by starting from the differential equations that describe the dynamical behavior of the vehicle, obtained from first principles. As has been seen in previous chapters, trajectory generation can be posed as an optimal control problem: the problem of finding a control law for the system described by the differential equations subject to some constraints (operational and related to collision avoidance), such that a certain optimality criterion is achieved.

This chapter focuses on solving the trajectory planning problem using collocation methods and its advantages and drawbacks will be analyzed. Collocation methods have been largely used for trajectory planning of single aerospace vehicles. An example of multi-aircraft trajectory planning has been presented in [2] which deals with the problem of multiple aircraft landings in order to reduce the waiting time, improve airport efficiency, and minimize fuel consumption while maintaining safety. Then, a new algorithm called S-Adaptive PS method will be presented, which proposes an advanced segmentation process in the PS collocation method.

4.1.1. Problem description

The kinematic equations of vehicles are used as white box models. Furthermore, dynamic aspects can be modeled like inertias, consumption, etc. in order to produce more realistic models. Any variable of the model may be used as optimal criterion in the optimal problem.

State vector $x(t)$ usually is defined by the position (x, y, z) and the orientation (roll, pitch, yaw) of the vehicle in time. Sometimes the speed variable is added in order to define a full waypoint. The input or control vector $u(t)$ is defined by the controls of the vehicle (forces, acceleration, angles of heading, etc.).

In the model, vehicles are considered as a point of mass, that all forces are applied in the center of mass of the vehicle. The trajectory is defined by a set of waypoints in time.

Figure 14 shows a diagram in which the trajectory is discretized by eight waypoints.

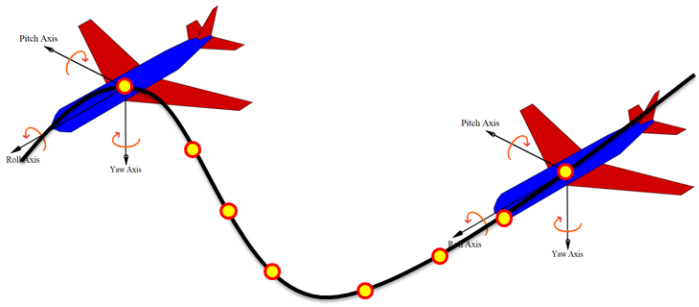


Figure 14: Diagram of trajectory defined by waypoints.

In case of collocation methods, the trajectory planning problem is translated into NLP problem by defining optimal criterion, constraints and boundary conditions.

As different optimal criteria can be considered:

- **Time:** Minimum arrival time.
- **Distance:** Minimum distance traveled.
- **Consumption:** Minimum consumption of vehicle.
- **Smooth trajectories:** smoothing control inputs.

Also a mixed criterion can be used when two of more criteria are weighted. Constraints are focused on limits in variables of the problem: boundaries of the scenario, speed, angles of heading of the vehicle, etc. State and control vectors have to be fully bounded with an upper and lower limit. Boundary conditions are determined by the initial and final state of the problem. Typically the initial state is fully defined, while a final state is only at times fully defined, and sometimes some variables of the state are free.

4.1.2. Multi-vehicles problem

When two or more vehicles are moving in the same scenario, new constraints must be added to the trajectory planning problem regarding collision avoidance.

In trajectory planning with collision avoidance, a safety distance has to be stabilized between every two vehicles. When this safety distance is violated, a collision resolution method is executed. Mainly, there are three collision avoidance techniques:

- **Velocity planning:** This consists in the generation of a new speed profile with the involved vehicles. Changes in the (orientation or altitude) are not allowed. In most cases of trajectory planning, the trajectories are straight, and collision is avoided by changing the speed of vehicles. This technique usually is applied to ground and aerial vehicles.

- **Changes of altitude:** In this case, when a collision is detected, the involved vehicles change the pitch angle and solve the problem passing one over the other. Speed and orientation could change, but is not always necessary. This technique is very usual in aerial vehicles.
- **Changes of orientation:** In this last case, collision is avoided by changing the yaw angle. In changing the orientation, a new trajectory is generated with the involved vehicles. This technique is applied in ground and aerial vehicles.

Combinations of some of these techniques can be done in a mixed policy when a 3 Degree of Freedom (DOF) vehicle is used, but the computation time of the collision detection methods increase with the use of different avoidance techniques.

When the multi-vehicle trajectory planning problem is translated into a NLP problem, in addition to the constraints shown in the previous section, new constraints are taken into account. In this case, path constraints are added to the safety distance constraints. Safety distance is modeled by the straight line distance between the center of mass of two vehicles (see Figure 15). All combinations of two vehicles are considered.

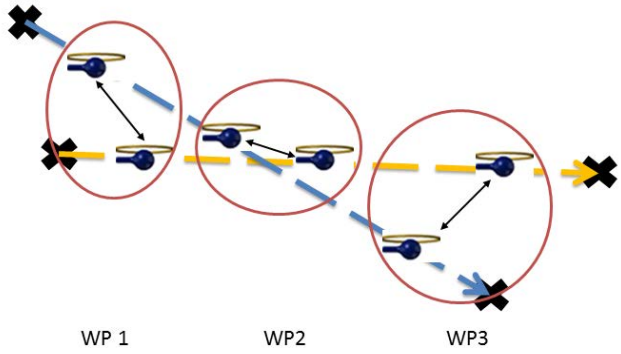


Figure 15: Diagram of safety distance between vehicles.

In case of a multi-vehicle problem, the optimal criterion is extended to all vehicles. Also a weight policy can be established in order to set up a priority order between vehicles.

4.1.3. Advantages and drawbacks of collocation methods

As considered in Chapter 2, there is a wide variety of trajectory planning or path planning methods in literature. Some of which focus on optimal trajectory planning. This is a NP-hard problem, and the execution time of these methods, in general, is usually high. Furthermore, when the number of involved vehicles increases, the computation time usually grows at an exponential rate, i.e. scalability[4] of optimal trajectory method is usually bad or very bad.

[4] Scalability is defined in this Thesis as computation time dependence with the increment of vehicles. i.e. how the computational time increases as well as the number of vehicles in the same scenario.

Collocation methods have recently received significant attention pertaining to complete space and orbit trajectories [40] [41]. The success of this method is based on the discretization of the dynamics problem and constraints in some points, instead of solving the continuous problem. One drawback of collocation methods is that the dependence between computation time and number of collocation points is very strong. Computation time as well as the number of collocation points increase.

Scalability of optimal trajectory planning methods has always been the weak point. Part of the study of this thesis focuses on this concept, because scalability is very important for the robustness of a trajectory planning system. Regarding collocation methods, there are some papers in literature that solve the problem of multi-vehicles successfully with Direct collocation or PS collocation [44], but in general there are no such studies with four or more vehicles. This Thesis presents a full analysis of scalability and the results in general are good (see Chapter 5).

One main advantage of collocation methods is that they use the kinematic models of vehicles in order to deploy the trajectories. the trajectories closer resemble reality. On the other hand, the discretization process of this method produces some drawbacks which must be under control. Collocation methods only solve the problem into the collocation points (waypoints), and then trajectories are generated by some interpolation method. According to the number of collocation points used and the difficulty of the problem, the results could be insufficient to maintain the safety distance throughout the entire trajectory. That is, the collocation method ensures that the trajectory constraints will be met in the collocation points, but there is the possibility of collision when a vehicle is moving from one waypoint to another. Due to this uncertainty, the safety distances between vehicles have to be checked after obtaining a solution and the number of collocation points must be readjusted if there is any collision.

The selection of correct number of collocation point is not easy. This number of collocation points need to be enough to generate good solutions (i.e. trajectories are safety after the interpolation), but not many that the computation time would be very high (the computation time increases with the number of collocation points).

The advantages and drawbacks of collocation methods can be summarized as:

Advantages:

- Trajectories are very close to reality.
- Multiple optimal criteria can be used.
- Priorities between vehicles can be established.
- Low computation time (related to number of points).
- Good scalability.

Drawbacks:

- Safety distance is not completely ensured: Trajectories are discretized and the safety distance between two points (waypoints) in the trajectory is not insured.
- Quality of solution depends on the number of collocation points.
- Selection of correct amount of collocation points is difficult balance, in order to find a good solution and a good execution time.

4.2. S-Adaptive Pseudospectral collocation method

The segmentation-Adaptive (S-Adaptive) method arises as an alternative to the drawbacks of using collocation methods as trajectory planning methods. This is an optimal control method, but it focuses on optimal trajectory planning. This method is listed in the sub-interval PS collocation method, and presents many improvements in comparison with other PS collocation methods.

As seen in previous sections, convergence in collocation methods is achieved by increasing the number of nodes. Computation time depends on the number of nodes as well. Increasing of number of nodes should be avoided as far as possible. Sub-interval PS methods demonstrate a better use of nodes, obtaining better results with less nodes. For example, the hp-Adaptive Pseudospectral method presents some improvements in the convergence of the solution more than the basic PS collocation method. On the other hand, the hp-Adaptive method usually uses a large number of nodes, and computation time is very high for a real-time applications.

S-Adaptive method performs an advanced method of splitting the problem, and only increases the number of nodes when it is absolutely necessary. In this way, the minimum number of used nodes is ensured.

4.2.1. Description

S-Adaptive PS method focuses on optimal trajectory planning and presents some improvements with respect to other collocation methods. Firstly the state vector and control vector are based on the input and output of the control of one or more vehicles. Second is that multiple vehicles considered and distance between every two vehicles is checked at the nodes. Third and most importantly, the safety distance is checked in the entire continuous trajectory and not only in the nodes. Fourthly the dynamic model of vehicles is considered in the trajectory generation. And fifth is that the solution uses low number of nodes, and computation time is fast.

S-Adaptive method considers splitting the time interval into segments and increasing the number of nodes within the segments only if it is necessary (when safety distance constraints are not met). For this, a *verification process* which checks if a minimum distance is maintained between every two vehicles (in continuous time) is used. The instants of time in which safety distance is violated are included into segments. In this way, after the segmentation process, we have temporal segments in which all vehicles

are separated at least by safety distance and temporal segments in which not all vehicles respect the safety distance.

The segmentation process also considers the of the vehicle. Sometimes the trajectory proposed solution in each segment is varies greatly, and although the trajectory continuity is ensured between two segments, an abrupt transition could be occur that may be unrealizable for some vehicles. Owing of this part of the segmentation process, every segment is increased by the horizon time t_h in the limits of the segment. This parameter models the inertia of changing the trajectory from one segment to another. For example, the value of t_h in a fixed-wing UAV will be greater than in a rotary-wing because the curvatures of the trajectories are bigger. Also, a rotary-wing can decrease speed and stop but a fixed-wing in only able to decrease its speed until a minimum stall speed.

S-Adaptive is an iterative method with a bounded number of iterations. In the first iteration, the entire problem is defined by the initial segment which is $[t_0, t_f]$. Then, the safety distance between vehicles is checked in the entire segment by the verification process. If the solution is rejected, the segmentation process splits the problem into segments. One or more segments can be generated in this step. After this, a new solution (trajectory) will be searched for the segments which do not meet the safety constraints. New collocation points are only added in these segments. Collocation points of the segments that meet the constraints remain without change. At the end of the iteration, the number and the distribution of collocation points has changed in the entire problem $[t_0, t_f]$. Then, this new solution returns to the verification process (in the second iteration). This method iterates until a solution is obtained or the limit of iteration is reached. Note that in the verification process, a small tolerance error (which is defined by the user) is allowed in this process.

The following parameters are considered in S-Adaptive SP collocation method:

- **Horizon time (t_h):** dynamic aspects of the vehicle, like the inertia effect, are modeling by this parameter of the vehicle. This parameter affects the segmentation process by adding a little period of time at the beginning and the end of every segment. Therefore the time interval will be increased by this parameter. The value of this parameter has to be set by some knowledge of the dynamic of the vehicle.
- **Number of collocation points per segment:** this parameter defines the number of collocation point which is used in every segment. i.e., it defines the degree of the polynomial interpolant of the PS method. It is always using the same polynomial.
- **Limit of iteration:** This parameter defines the limit of iterations that are allowed in the method. Due to the robustness of method, this is a bounded algorithm. Then, the algorithm is not complete.
- **Error:** This parameter defines a small margin of error in the verification process (safety distance between vehicles). The magnitude of this parameter is the same like the distance (meter).

- **Resolution step:** This parameter defines the step of interpolation used in verification process which simulates continuous time. This parameter will be explained in the next Section 4.2.2.

Figure 16 shows an example of the behavior of the S-Adaptive Pseudospectral method. In this example, eight collocation points have been used in the setting of the method. The initial segment is defined by t_i and t_f and eight collocation points are placed along the segment (black points), in the first iteration (iter1). The verification process checks the safety distance between vehicles and returns an interval defined by the time t_{ci} and t_{cf}, where the constraints are not met. The horizon time, t_h is added in order to define the final segment (red ellipse). In the second iteration (iter2), points of the segment are replaced for eight collocation points following the PS method solution (red points). The remaining of collocation points stay as the first iteration (black points). In the next iteration (iter3), after the verification process, two segments which do not meet the constraints are detected (red ellipses). Again eight collocation points replace the collocation points in each segment (red points) and the rest of collocation points stay as the previous iteration (iter2). Now, the verification process set that the safety distance constraints are met in the entire problem. Thus, so the algorithm finishes solution of the problem.

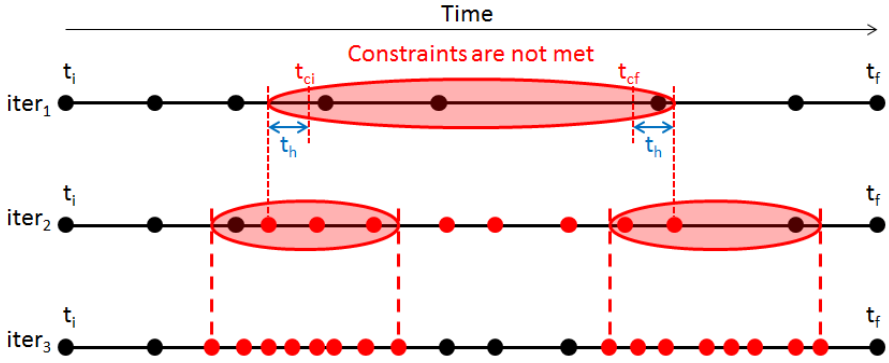


Figure 16: Diagram example of application of S-Adaptive Pseudospectral method.

Every subinterval is solved by PS collocation method. Specifically, using the algorithm Legendre-Gauss-Lobatto (LGL) PS collocation. As seen in Section 3.4.2, LGL distribution of points considers the boundary limits of interval time problem $[t_0, t_f]$ (or $[-1, 1]$ in normalized time). Note that this fact is very important to ensure the continuity of the trajectory. Using this method there is an overlap in the boundaries of the sub-segments, and the trajectories are continuous throughout problem.

4.2.2. Verification process

Verification process is a completely new step which has been introduced into the collocation methods. This method consists of the interpolation of the coordinates from the collocation points (waypoint position) taking into account the time; thus the speed

in the trajectory is interpolated as well. Then, when the trajectory is in continuous time, it is discretized by a small fixed step (defined as resolution step). After this, all distances between all vehicles in every discrete instant are checked.

Two parameters must be set the verification process:

- **Interpolation function:** This parameter is related with the type of trajectory that vehicle realizes. i.e., how the navigation module of the vehicle interpolates the waypoints. Most commonly used are: straight line, 3rd order spline and 3rd order polynomial.
- **Resolution step:** The fixed step depends on the problem. It can be used at any range from 1 second to 0.01 second.

4.2.3. Example

Figure 17 shows an example of the S-Adaptive PS collocation method. In this example, we have two vehicles with two initial trajectories (blue and red) and four collocation points each. In the analysis of initial trajectory, the four waypoints are interpolated by a 3rd order polynomial and the safety distance between both vehicles is checked using a 0.01 second step. After the verification process, it has been detected that safety distance is violated inside the black circle. So the time line problem is split and one segment is generated. Four new collocation points replace the collocation points inside the segment. By changing the trajectory (orientation and speed profile) of this segment only, the collision that was in the center of the trajectory has been solved.

In the second iteration, the verification process checks the entire trajectory, and substantiates that all safety distances are well. Then, the method finishes and returns these trajectories.

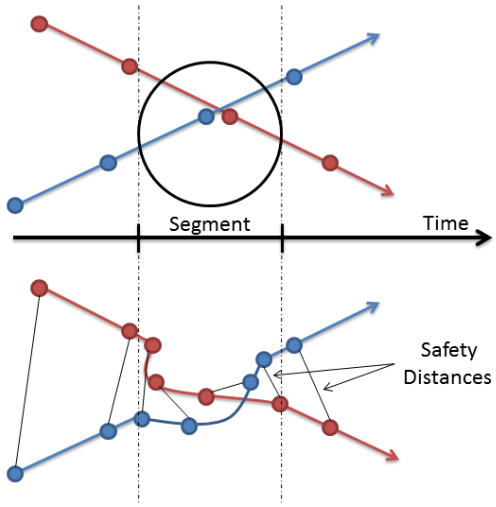


Figure 17: Diagram example of trajectory generation with the S-Adaptive method.

4.3. Conclusions

Multi-vehicle trajectory planning problem can be described as an optimal control problem in which some optimal criterion are achieved subject to a set of constraints. In this area, collocation methods have been largely used. Some advantages of collocation methods are that the resulting trajectories closely resemble reality, the computation time is low and the scalability of the method is favorable. On the other hand, the discretization process of collocation method constructs the constraints of the problem (safety distance in this case) only to have met in the collocations points (waypoints). A crash is then a possibility when two vehicles are traveling from one to another waypoint.

Selecting the appropriate number of collocation points in each problem is not easy. The computation time and quality of the solution are strongly tied to the number of collocation points. In order to solve this problem, S-Adaptive PS collocation method is developed.

S-Adaptive PS uses an advanced method of segmentation, which is focuses on safety trajectory constraints. This method analyzes the entire timeline checking if the safety distance between all the vehicles is met, and generates sub-segments for which the safety distance is violated. Then, a new solution is explored for said segments, using a Legendre-Gauss-Lobatto (LGL) PS collocation method. This is an iterative method that generates new segments in every iteration and only finishes when all safety constraints are met.

In the next chapter, the application of collocation methods will be presented. PS collocation, hp-Adaptive PS collocation and S-Adaptive PS collocation methods will be used in different kinds of scenarios, and aspects like computation time or scalability will be analyzed and compared.

5. APPLICATIONS OF COLLOCATION METHODS IN MULTI-VEHICLE TRAJECTORY PLANNING

To begin a big project, you need courage. To finish a big project, you need perseverance.

Carlos Cuauhtémoc

The aim of this chapter is to show the collocation methods performance as optimal trajectory planning when is used different kind of vehicles and scenarios. In the following sections, three implementations of PS collocation methods are used in order to test different optimal criteria, models of vehicles and scenarios. Furthermore, some of these scenarios presented here will be simulated and tested in the next chapter of validation.

5.1. Introduction

This chapter aims to show the versatility of collocation methods when they are used as trajectory planning. Four cases of application are shown in which different species of vehicles are used. This chapter is composed of 3 sections. The first one is focused on fixed-wing UAVs, the second on rotary-wing UAVs and finally UGVs are explored. A mixed scenario is also presented in which the trajectories of an UAV and UGV are planned in coordination at same time.

In these sections, four kinds of applications are distinguished: collision avoidance, landing of UAV, air traffic management and trajectory planning for UGV. A total of twelve scenarios have been used. With aerial vehicles, most of them are developed in 2D because velocity planning technique has been used for collision avoidance. This Thesis presents a comparative between different PS collocation implementations, simple models and scenarios have been selected for this purpose. Although a complex scenario is presented as well, in which two different vehicles, an UAV and an UGV are used in a 3D scenario.

Different collocation methods have been used. Among the three global groups of collocation methods presented in Section 3.4, the study focuses on PS collocation method introduced in Section 3.4.2 and sub-interval PS collocation method which was

introduced in Section 3.4.3. These are the most appropriate methods for trajectory planning. Furthermore, a comparison between sub-interval PS and Heuristic Velocity Planning with Optimization Phase (VP) method is presented.

The chapter is organized into four sections. Section 1 introduces all models of vehicles which being used in the subsequent section. Section 2 focuses on fixed-wing UAVs. In these instances an hp-Adaptive PS method is used. Three different applications will be tested. In particular, the scalability and the computation time of hp-Adaptive method will also be analyzed. Section 3 focuses on rotary-wing UAVs. This section presents a study of real-time configuration of collocation method when a collision avoidance problem is present. In which case, S-Adaptive, hp-Adaptive and LGL PS method will be compared. Finally section 4 presents ground vehicles applications. In this section trajectory planning application will be presented, but with the novelty that fixed obstacles are considered. In this case two scenarios will be presented, with different numbers of obstacles. Due to the high level of constraints with this problem and the high computation time needed to solve it, only S-Adaptive method is tested.

5.1.1. Vehicle mathematical models

Four simplified models are introduced in the following sections. Two of them are aerial vehicles, and the other two are ground vehicles.

Fixed-wing aircraft

A general model of a rigid-body aircraft that has been used extensively for performance analysis is presented here. The UAV dynamics is modeled by a 3 degrees-of-freedom (DOF) point mass model [87]. This model describes the motion of the UAV center of gravity, considering the thrust and aerodynamic forces that act on the UAV.

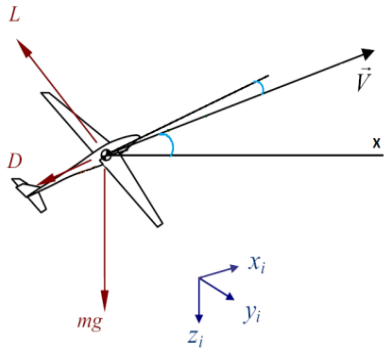


Figure 18: Fixed-wing model.

This model considers the following inputs and outputs:

- Inputs: Thrust force (T), angle of attack (α) and the bank angle (roll) (ϕ).

- Outputs: Position (x, y, h), airspeed (v), flight path angle (γ) and the heading angle (yaw) (Ψ).

The of the aircraft has been simplified by considering that flight is symmetric, and thus there is not any lateral force and sideslip $\beta = 0$.

The kinematic equations of the model are given by:

$$\begin{aligned} \dot{x} &= v\cos(\Psi)\cos(\gamma) \\ \dot{y} &= v\sin(\Psi)\cos(\gamma) \\ \dot{h} &= v\sin(\gamma) \end{aligned} \tag{5.1}$$

and the dynamic equations are:

$$\begin{aligned} \dot{v} &= \frac{1}{m}[-D + T\cos(\alpha) - mg\sin(\gamma)] \\ \dot{\Psi} &= \frac{1}{mv\cos(\gamma)}[L\sin(\phi) + T\sin(\alpha)\sin(\phi)] \\ \dot{\gamma} &= \frac{1}{mv}[L\cos(\phi) + T\sin(\alpha)\cos(\phi) - mg\cos(\gamma)] \end{aligned} \tag{5.2}$$

where m is the total UAV mass and g is the gravity. L and D are the lift and drag aerodynamic forces, respectively. They are defined as functions of the nondimensional lift and drag coefficients C_L and C_D.

$$\begin{aligned} D &= \frac{1}{2}\rho v^2 S C_D \\ L &= \frac{1}{2}\rho v^2 S C_L \\ C_D &= C_{D0} + (K C_L{}^2) \\ C_L &= C_{La}(\alpha - \alpha_1) \\ K &= \frac{1}{\pi A e} \end{aligned} \tag{5.3}$$

where S is the wing surface area, A is the aircraft wingspan, e is the Oswald efficiency factor, ρ is the air density and π is a mathematical constant approximately equal to 3.14159.

In order to compute the accumulated fuel consumption, it is considered the accumulated use of thrust, as is modeled by:

$$\dot{c} = P_0 + PT \tag{5.4}$$

where P is a constant related to the fuel consumption and P_0 is the minimum

consumption needed to keep the engine running of the aerial vehicle.

A 2D version of this model can be useful sometimes. For example in a landing problem we can focus on x-h plane. Then, considering bank angle $\phi = 0$, a heading $\Psi = 0$ is resulted and equation of the model can be resumed into 2D problem.

$$\dot{x} = v\cos(\gamma) \tag{5.5}$$

$$\dot{h} = v\sin(\gamma)$$

$$\dot{v} = \frac{1}{m}[-D + T\cos(\alpha) - mg\sin(\gamma)]$$

$$\dot{\gamma} = \frac{1}{mv}[L + T\sin(\alpha) - mg\cos(\gamma)]$$

Rotary-wing aircraft

A model of a quadrotor is considered for rotary-wing vehicles. This outline uses the dynamic analysis presented in [88] A six DOF simplified model, neglecting the Coriolis terms is considered.

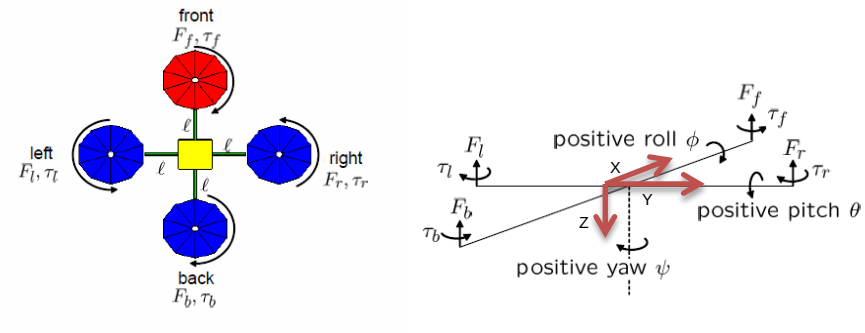


Figure 19: Rotary-wing model (quadrotor).

This model considers the following inputs and outputs:

- Inputs: Thrust force (T), torques in x-axe τ_ϕ, y-axe τ_θ and z-axe τ_ψ
- Outputs: Inertial position (p_n, p_e, p_d), and orientation roll pitch and yaw angle (ϕ, θ, Ψ).

The dynamic equations of the model are given by:

$$\ddot{p_n} = (-\cos(\phi)\sin(\theta)\cos(\psi) - \sin(\phi)\sin(\psi))\frac{T}{m} \tag{5.6}$$

$$\ddot{p_e} = (-\cos(\phi)\sin(\theta)\sin(\psi) + \sin(\phi)\cos(\psi))\frac{T}{m}$$

$$\ddot{p_d} = g - \cos(\phi)\cos(\theta)\frac{T}{m}$$

$$\ddot{\phi} = \frac{\tau_\phi}{I_x}$$

$$\ddot{\theta} = \frac{\tau_\theta}{I_y}$$

$$\ddot{\psi} = \frac{\tau_\psi}{I_z}$$

where

$$T = F_f + F_r + F_b + F_l \tag{5.7}$$

$$\tau_\phi = l(F_l - F_r)$$

$$\tau_\theta = l(F_f - F_b)$$

$$\tau_\psi = \tau_r + \tau_l - \tau_f - \tau_b$$

and F_f, F_r, F_b, F_l are the forces in each motor, l is the distance between two opposite motors, $m$is the total mass of vehicle

A second simplified version of the dynamic model considers that the quadrotor is moving on a plane, and thus the thrust T is needed to maintain the quadrotor in the same flight level, that the roll and pitch angles are small and there is no yaw rotation of the quadrotor.

$$\ddot{p_x} = -\cos(\phi)\sin(\theta)\frac{T}{m} \tag{5.8}$$

$$\ddot{p_y} = \sin(\phi)\frac{T}{m}$$

$$\ddot{p_z} = g - \cos(\phi)\cos(\theta)\frac{T}{m}$$

$$\ddot{\phi} = \frac{\tau_\phi}{I_x}$$

$$\ddot{\theta} = \frac{\tau_\theta}{I_y}$$

$$\ddot{\psi} = \frac{\tau_\psi}{I_z}$$

In this case, a 2D version of this model is achieved as well, by considering only a x-y plane, by setting the hovering thrust as $T_{hovering} = g = \frac{T}{m}$.

$$\ddot{x} = -\cos(\phi)\sin(\theta)\, T_{hovering} \tag{5.9}$$

$$\ddot{y} = \sin(\phi) T_{hovering}$$

$$\ddot{\phi} = \frac{\tau_\phi}{I_x}$$

$$\ddot{\theta} = \frac{\tau_\theta}{I_y}$$

Ackerman steering ground vehicle

Ackerman steering is the most common vehicle configuration in which wheels rolls in the direction they are pointing. The model presented here is based on the non-holonomic vehicle's definition [89]. This allows a simplification of the equations by considering that the wheel rolls without slipping.

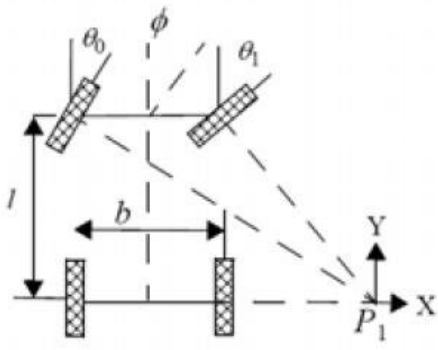


Figure 20: Ackerman steering model.

This model considers the following inputs and outputs:

- Inputs: front wheel orientation angle (α) and speed command (v_{in}).
- Outputs: Position (x, y), heading angle (yaw) (ψ) and the vehicle speed (v)

The kinematic and dynamic equations of the model are given by:

$$\dot{x} = v\cos(\psi) \tag{5.10}$$

$$\dot{y} = v\sin(\psi)$$

$$\dot{\psi} = v\left(\frac{\tan(\alpha)}{l}\right)$$

$$\dot{v} = \frac{1}{\tau}(v - v_{in})$$

where τ is the time constant of the drive motor model and l is the distance between front wheels and back wheels.

Differential-drive ground vehicle

Differential-drive is an example of the most popular way to drive indoor mobile robots [89]. There are two main wheels, each being attached to its own motor. A third wheel (not visible in Figure 21) is placed in the rear to passively roll along while preventing the robot from falling over.

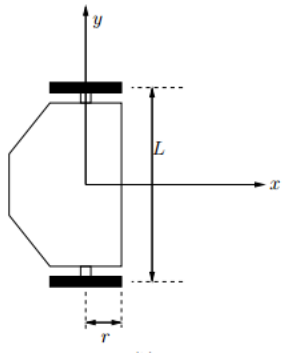


Figure 21: Differential-drive model.

This model considers the following inputs and outputs:

- Inputs: front right and left wheel speed (w_R, w_L).
- Outputs: Position (x, y) and heading angle (yaw) (ψ)

The kinematic equations of the model are given by:

$$\begin{aligned} \dot{x} &= \frac{r(w_R + w_L)}{2}\cos(\psi) \\ \dot{y} &= \frac{r(w_R + w_L)}{2}\sin(\psi) \\ \dot{\psi} &= \frac{r(w_R - w_L)}{L} \end{aligned} \tag{5.11}$$

where L is the distance between wheels and r is the radius of the wheel.

5.1.2. Optimal criteria

Multiple optimal criteria have been tested in the different sections. Depending on the application and the scenario, one criterion is more useful than the other. An overview of all criteria is presented here.

- **Distance travelled**. The minimization of the distance travelled by all the vehicles is a very common objective in trajectory generation. This criterion tries to minimize the distance between the different collocation points of the trajectory:

$$J = \sum_{i=1}^{M} \sum_{j=1}^{N} \sqrt{(x_{i,j} - x_{i,j-1})^2 + (y_{i,j} - y_{i,j-1})^2} \tag{5.12}$$

  where M is the number of vehicles, and N is the number of collocation points.

- **Fuel consumption**. For fixed-wing aircraft, in many cases the main objective is to minimize the fuel consumption of all aircraft. This is clearly the case in Air Traffic Management (ATM), where fuel savings of a few percent translate into significant cost reductions. Although for simple aircraft models it can be equivalent to minimizing the distance travelled. Aircraft models used in ATM fuel consumption are closely related to the thrust force generated by the aircraft engines, as can be seen in Equation (5.4), which is also related to changes of velocity and altitude. The optimization criterion can be defined as:

$$J = \sum_{i=1}^{M} C_{i,N} \tag{5.13}$$

  where $C_{i,N}$ is the accumulated fuel consumption for vehicle *i* at the last collocation point N, and M is the total number of vehicles.

- **Smooth trajectories.** It is important that the trajectories be smooth, to ensure there is no sudden curvature change. This can be achieved in aerial vehicles by minimizing the pitch and roll angles. Then, the optimization index can be defined as:

$$J = \sum_{i=1}^{M} \left(\frac{\sum_{j=1}^{N} |\theta_{i,j}|}{N} + \frac{\sum_{j=1}^{N} |\phi_{i,j}|}{N} \right) \tag{5.14}$$

  where M is the number of vehicles, N is the number of collocation points and $|\theta_{i,j}|$ and $|\phi_{i,j}|$ are the absolute values of the pitch and roll angles of vehicle i at collocation point *j*, respectively.

- **Comfort**. In commercial aviation and ATM, in some cases passenger comfort can be achieved by minimizing the change in speed with respect to the cruise speed. This criterion is also useful for UAVs when possible conflicts between UAVs are solved by velocity planning (changes of speed).

$$J = \sum_{i=1}^{M} \sum_{j=1}^{N} \left| v_{i,j} - v_{i,cruise} \right| \tag{5.15}$$

where M is the number of vehicles, N is the number of collocation points, $v_{i,j}$ is the speed of vehicle i at collocation point j, and $v_{i,cruise}$ is the cruise speed of aircraft i.

- **Time**. In many scenarios the goal is to complete the task quickly, time efficiency, so the criterion is based on minimum final time of the problem.

$$J = t_f \tag{5.16}$$

5.1.3. Implementation

The study presented below has been implemented in Matlab. Mainly, three implementations have been tested. Two of which are commercial implementation: The DIDO software can be found in the official site[5]. That implements a Legendre-Gauss-Lobatto SP method and its NLP solver is based on Sequential Quadratic Programming (SQP) and named SNOPT. The second one is GPOPS. The later version of this software, named GPOPS2, can be found in the official site[6]. That implementation is a sub-interval PS collocation method based on Legendre-Gauss-Radau PS method and uses either the NLP solver IPOPT or SNOPT. Finally, the third implementation has been developed in this Thesis, Its name is S-Adaptive SP method, and is based on a Legendre-Gauss-Lobatto SP method and its NLP solver is based on SNOPT.

Matlab is an interpreted language that is executed into a virtual machine. Therefore the computation time of algorithms is higher than other implementation such as C++. The choice of Matlab in this thesis is used as the common developed framework and is vital to the comparison of different collocation methods.

All algorithms have been run in a PC with a CPU Intel Core i7-3770 @ 3.4 Ghz and 16 GB of RAM. The operating system used in the simulations was Kubuntu Linux 12.10 OS.

5.2. Applications with fixed-wing aircraft

Collision-free trajectory generation is a relevant function for fixed-wing aircraft flying in a common airspace, either manned or UAVs. This section presents three different cases, all solved using the collocation methods. The first one is the general collision avoidance problem for UAVs that flies at the same flight level. The second one is an Air Traffic Management (ATM) application. This is the converging air traffic problem, in which several aircraft are approaching to the terminal area. The third case deals a fixed-wing

[5] http://www.elissarglobal.com/academic/get-dido/try-dido/
[6] http://www.gpops2.com/Purchase/Purchase.html

UAV that has to land on a UGV that is moving on the runway.

Scalability is an important concept in the trajectory planning methods. The reliability of this kind of method focuses on the integrity of the system when the number of vehicles increases. In this section, scalability will be studied in different applications, and a comparison with other optimal trajectory planning will be presented as well.

Two of the three applications presented in this section have been performed with the model in Equations (5.1) - (5.4). This model has been completed with parameter of UAV shown in Figure 22. This aircraft is an UAV of 3 meter wingspan and 20 Kg mass, developed in the University of Seville. The entire parameters necessary for this model are given by Table 1. The case of the ATM application has been performed with a simple model of the aircraft and a different setting presented in Table 6.

Figure 22: UGAV developed by the GRVC research group based on the Mugin UAV platform.

The collocation method used in this section is hp-Adaptive PS collocation. This is a sub-interval PS method (Section 3.4.3) based on Legendre-Gauss-Radau quadrature. This method is a Matlab implementation which solves the NLP problem using the SNOPT solver.

Table 1: Parameters of the UAV: UGAV.

Characteristics	
Stall Speed	11 m/s
Maximum Speed	42 m/s
Cruise Speed	25 m/s
Maximum Thrust	132 N
Mass	20 Kg
Consumption coefficient	0.1

5.2.1. Collision avoidance and scalability

This section analyzes the general problem of collision-free trajectories generation for multiple UAVs which are flying in the same flight level. The problem is solved by using the hp-Adaptive collocation method, described in [52].

Two representative scenarios have been used. In the first one, scenario S1 which is shown in Figure 23, several UAVs fly in parallel and cross the paths of several other UAVs. Scenario S2 is a typical "star" configuration (see Figure 24), where the paths of several UAVs cross at the same point, which is considered as one of the worst case scenarios, because the possibility of that happens is very small. The conflict resolution used in this scenario is velocity planning, so no changes of heading or altitude are allowed. The characteristics of both scenarios are given in Table 2.

The initial trajectories of all the UAVs are known, but they do not maintain the safety distance and there will be conflicts between them. The objective is to generate new collision-free trajectories that maintain the initial and final points and the initial and final time instant (also known as Expected Time of Arrival, ETA), while minimizing the optimization index, which in this case, is the minimum consumption.

A comparison of computation time and scalability between hp-Adaptive method and the Heuristic Velocity Planning with Optimization Phase (VP) is presented.

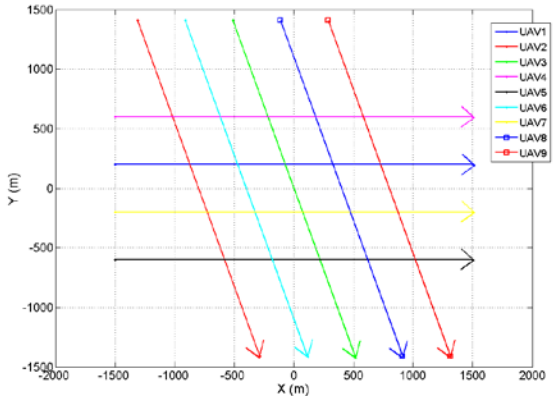


Figure 23: Scenario S1, transversal cross (9-UAVs).

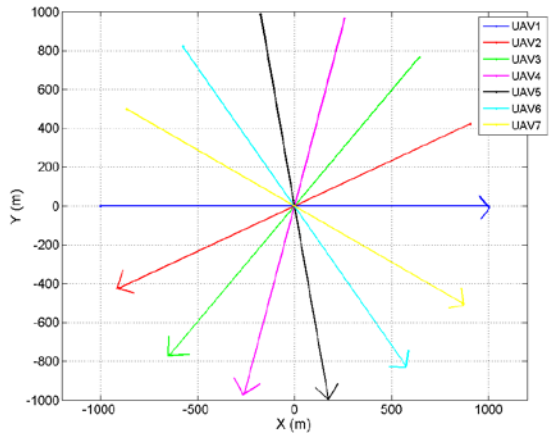


Figure 24: Scenario S2, star configuration (7-UAVs).

Scalability test considers S1 and S2. In case of S1, up to 9 UAVs are considered. In case of S2, up to 7 UAVs are considered. Scenario S2 is more difficult than S1 because of the conflict point is the same for every UAV. However, the coordination process of 9 UAVs in scenario S1 has to be considered as difficult as well.

Table 2: Specifications of the scenarios S1 and S2.

Characteristics	
Area dimensions S1	3000 x 3000 m
Flight distance S1	3000 m
ETA[1] S1	120 s
Area dimensions S2	2000 x 2000 m
Flight distance S2	2000 m
ETA[1] S2	80 s
Safety distance	200 m
CDR[2]	Velocity planning

[1]ETA: Expected Time of Arrival.
[2]CDR: Conflict Detection and Resolution.

Results of hp-Adaptive method

The hp-Adaptive collocation method has been used to generate new collision-free trajectories in both scenarios for different number of UAVs (up to nine in scenario S1 and up to seven in scenario S2). The results are summarized in the next subsection dealing with scalability. In the following the results of two example cases, one for scenario S1 and the other for scenario S2, are presented.

Figure 25 and Figure 26 show the solution of the worst case of scenario S1. In this case, nine UAVs are considered in the computation of the new speed profile which solves the conflicts between all UAVs. Figure 25 shows the trajectories (waypoints generated), no changes of heading are allowed in this scenario. Figure 26 shows the speed profiles computed, that is, the speed from that waypoint to the next one. It can be seen that all the speed variations are inside the allowed speed interval. In the picture, UAV4 is the only vehicle which changes strongly its speed to avoid the other UAVs. The rest of vehicles change smoothly their speed. This is because the optimal criterion was minimum consumption, so the variations of speed are minimal.

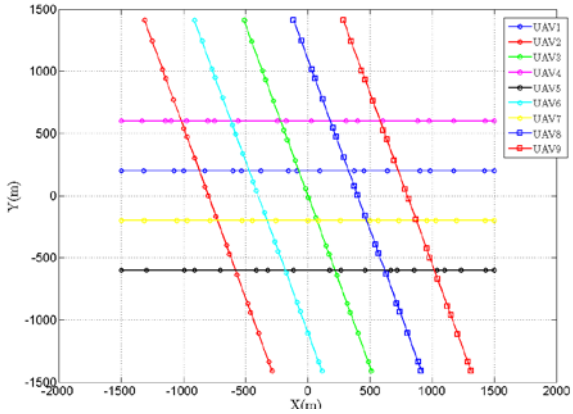


Figure 25: Solution of the scenario S1 with 9 UAVs. Trajectories.

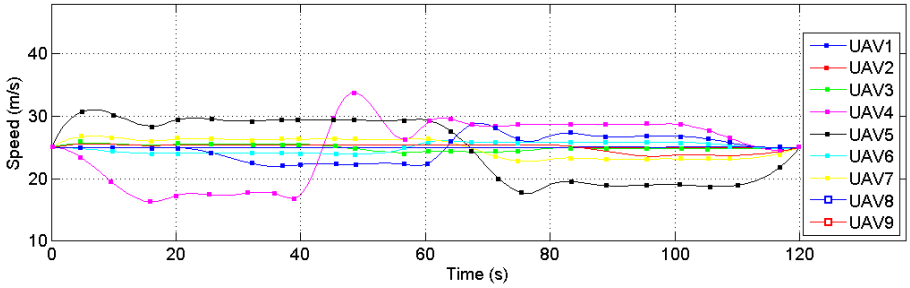


Figure 26: Solution of the scenario S1 with 9 UAVs. Speed profiles.

The second illustration presented is based on scenario S2. This case presents a simple example of three UAVs with the same collision point. This scenario presents the worst case because the probability of three UAVs colliding at the same point in an area of one million square meters is very low.

Figure 27 and Figure 28 show the results obtained. The first one presents the straight line trajectories generated by the hp-Adaptive method. The second one presents the speed profile of each UAV. In Figure 28 it can be seen that a conflict is detected around the position (0, 0). To avoid it, UAV3 increases its speed to pass through the conflict before. Then UAV2 passes through the conflict without changing its speed. Finally, UAV1 decreases its speed and passes through the conflict later. After all UAVs have been passed through the conflict, each UAV increases or decreases its speed to meet the ETA. UAV2 keeps its cruise speed so the ETA is met.

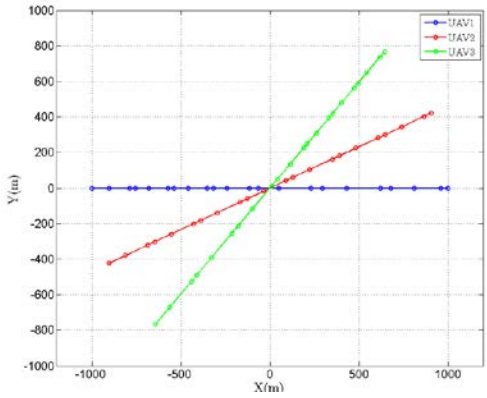


Figure 27: Solution of the scenario S2 with 3 UAVs. Trajectories.

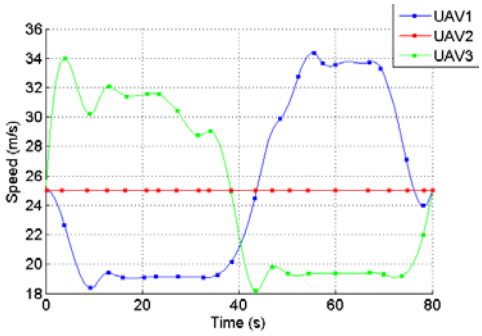


Figure 28: Solution of the scenario S2 with 3 UAVs. Speed profiles.

Scalability of hp-Adaptive method

The scalability study is very important to ensure the robustness of the method. In this thesis, scalability is defined as the dependence between the computation time and the number of vehicles considered in the same scenario. A linear dependence between both is considered as a good scalability but in a NP-hard method (like trajectory planning methods) the increase of computation time follows an exponential increase rate with the number of vehicles.

In this section two studies are presented (scenario S1 and S2 respectively). Seven scenarios (from 3 to 9 UAVs) have been tested in the first study, and six scenarios (from 2 to 7 UAVs) have been tested in the second one. One hundred executions have been performed in each number of UAVs in order to obtain an average computation time. Table 3 and Table 4 show the execution time[7] and its standard deviation in scenario S1 and S2, respectively. In case of Table 3, the execution time which has been obtained up to seven UAVs (less than 1.5 seconds), allows the use of this method in real-time applications. Then, the cases of eight or nine UAVs are very dense and many conflicts

[7] Computation time of the algorithm in MATLAB

have been detected. Therefore, the execution time increases in these cases. It is very interesting that the standard deviation of the time is zero or in case of nine UAVs, very closed to zero.

Table 3: Scalability results of the hp-Adaptive method in the scenario S1.

Number of UAVs	UAVs considered	Mean time (s)	σ_{Time} (s)	Cost (N)
3	3-7-9	0.258	0.009	41.936
4	1-5-6-8	0.797	0.008	65.793
5	1-2-3-4-5	0.816	0.009	72.575
6	1-2-3-5-7-9	1.079	0.009	86.722
7	1-2-3-4-5-7-9	1.481	0.005	98.722
8	1-2-3-4-5-6-7-9	8,244	0.022	134.498
9	1-2-3-4-5-6-7-8-9	37.892	0.260	176.483

Obtained results in Table 4 prove that scenario S2 is very difficult to solve. It can be seen that execution time increases with the number of UAVs, but the rate of increases is not exponential as usually happens in case of NP-hard problems.

Table 4: Scalability results of the hp-Adaptive method in the scenario S2.

Number of UAVs	UAVs considered	Mean time (s)	σ_{Time} (s)
2	1-2	0.768	0.006
3	1-2-3	2.200	0.003
4	1-2-3-4	4.600	0.013
5	1-2-3-4-5	9.226	0.023
6	1-2-3-4-5-6	14.459	0.031
7	1-2-3-4-5-6-7	20.638	0.025

Figure 29 shows a comparative between computation times of scenario S1 and S2. Note that computation time of scenario S2 grow up faster than in scenario S1. This is because the solution of scenario S2 is quite extreme. From this picture it can be concluded that the dependence of computation time is closer to the type of scenario (how difficult is the resolution of the conflict) than the number of UAVs.

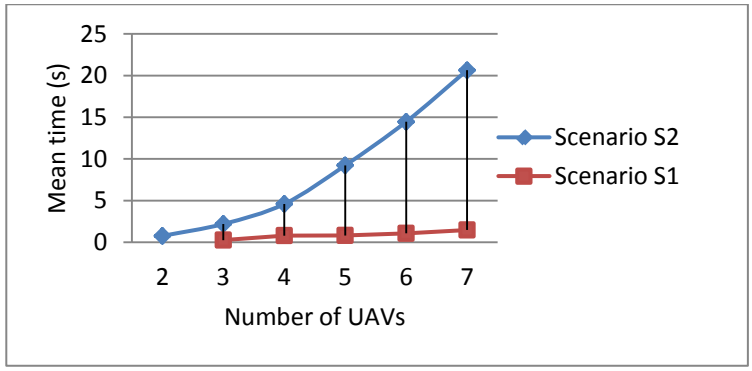


Figure 29: Results of the scalability test of the scenarios S1 and S2.

Comparative hp-Adaptive vs VP method

A comparison between the proposed method and the method called Heuristic Velocity Planning with Optimization Phase (VP) presented in [90] will be shown in this section. VP algorithm is an optimal trajectory planning method based on search tree method and it considers an optimization step which reduces the computation time.

Table 5 shows the average of execution times[8] obtained from each method considering the scenario S1. It can be seen that the main advantage of hp-Adaptive is that the computation of solution is significantly lower than VP. Specifically in the cases of 7, 8 and 9 UAVs, the computation time of hp-Adaptive is very low, in comparison with VP. Figure 30 shows a comparison of both methods and it can be seen that computation time of VP method increases exponentially while hp-Adaptive method does not.

[8] Computation time of the algorithm in MATLAB

Table 5: Comparison between the hp-Adaptive and the Heuristic Velocity Planning with Optimization Phase methods considering the scenario S1.

Number of UAVs	hp-Adaptive (s) Mean time	VP (s) Mean time
3	0.258	0.200
4	0.797	1.65
5	0.816	4.82
6	1.079	17.6
7	1.481	59.7
8	8,244	206.4
9	37.892	764.6

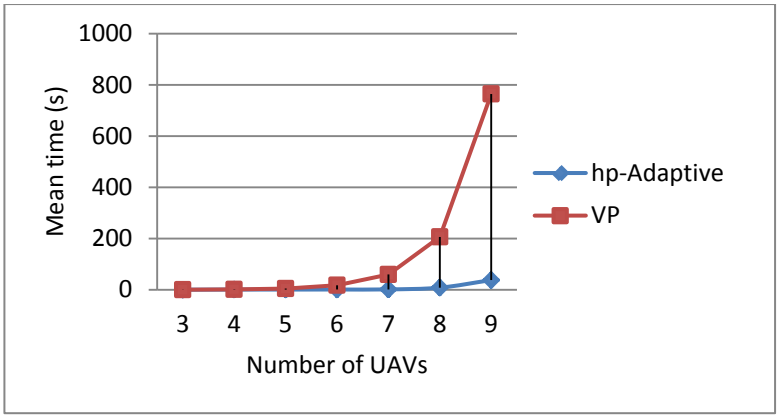


Figure 30: Computation time comparison of the hp-Adaptive PS collocation and the Heuristic Velocity Planning with Optimization Phase.

Other comparison is performed by considering the scenario S2. The conclusions obtained are similar that in scenario S1, but with the difference that in scenario S2 the VP method does not find a solution when six or seven UAVs are considered. Scenario S2 is very difficult to solve and the VP method is not able to solve it. On the other hand, hp-Adaptive method finds the optimal solution but it needs enough time.

5.2.2. Converging air traffic problem

The application considered in this section concerns conflict resolution among multiple aircraft in converging air traffic. This is a typical application in Air Traffic Management (ATM) that not only concerns a collision avoidance problem. Converging air traffic is also a sequencing problem, because in this problem there is a number of aircraft that have to share a common space and they have to fly in a queue following a suitable order of arrival. This strategy usually is applied when the aircraft is approaching to a runway. In a sequencing problem, sometime there are some aircraft which have priority, for example because the level of fuel is low. These priorities are represented by locked and unlocked aircraft. A locked aircraft is an aircraft which cannot change its speed profile, in case that a conflict is detected, and the other aircraft have to change their speed in order to avoid the collision.

In this section, the behavior of hp-Adaptive PS collocation method in converging air traffic problem is analyzed in two scenarios: first without any locked aircraft and second with one locked aircraft. Figure 31 shows the considered scenario. The scenario S3 presents some initial trajectories which converge into a TMA entry point, and after this point, all the aircraft follow the same trajectory in a queue. The characteristics of the scenario are given by Table 6. The model of aircraft in this application simulates the 2D kinematic of a commercial aircraft and it only considers the dynamic of accelerations. Table 6 shows as well the speed limits which have been utilized for this aircraft. The waypoints are defined by 2D coordinates (x, y) and the speed from that waypoint (v). It is assumed that all aircraft trajectories are known and the safety distances must be maintained all the time. The used optimal criterion is minimum changes of speed Equation (5.15). The goal is to find the conflict-free trajectories that minimize the optimal criteria and meets the problem constraints. The problem is finished when all the aircraft go through the entry point and all of them travel in order with cruise speed. The collision avoidance is solved by velocity planning at the same flight level.

Finally, two cases are considered in scenario S3:

- **All aircraft are cooperative**, only unlocked aircraft are considered. A first configuration considers up to 7 unlocked aircraft. This scenario is used in the scalability test and the study of behavior of the hp-Adaptive algorithm in this scenario. And the second configuration considers only four aircraft (A1, A2, A3 and A4). This scenario is used to test the algorithm when different safety distances are used.
- **One aircraft is non-cooperative** and the others are cooperative. This configuration considers three unlocked aircraft (A1, A2 and A4) and one locked aircraft (A3). This scenario is used to test the algorithm when locked aircraft are used.

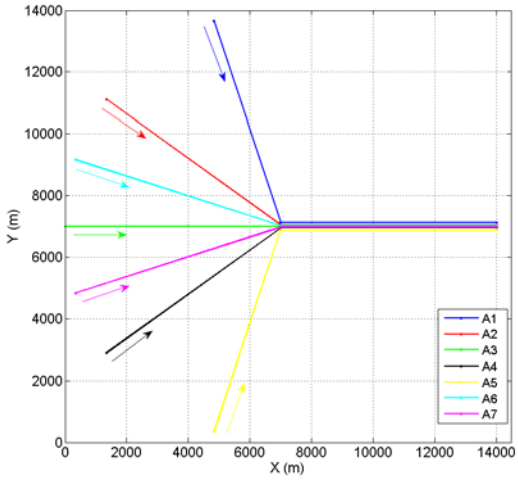


Figure 31: Scenario S3, converging air traffic problem.

Table 6: Specifications of the scenario S3.

Characteristics	
Area dimensions S3	14000 x 14000 m
Flight distance	7000 m
ETA[1] to entry point	120 s
Safety distance	500 m
Maximum Speed	200 m/s
Minimum Speed	50 m/s
Cruise Speed	100 m/s
CDR[2]	Velocity planning

[1]ETA: Expected Time of Arrival.
[2]CDR: Conflict Detection and Resolution.

Results of hp-Adaptive method

Multiple cases with different numbers of aircraft have been tested in scenario S3. Figure 32 shows an example of seven aircraft. To illustrate the problem, four instants of time are shown. In this picture, aircraft are represented by color and the square represents the area of safety distance (500 x 500 m. in this case). If some contact between two squares occurs, it means that the safety distance between two aircraft is violated. Analyzing the picture, it can be seen how all the aircraft move to the entry point and after this, all of them move in a queue with cruise speed. All trajectories are straight lines and velocity planning solves the collision avoidance. There are no priorities between the aircraft in this case. The optimal criterion in this application consists of

finding the sequence of aircraft in the minimum time as possible, Equation (5.16).

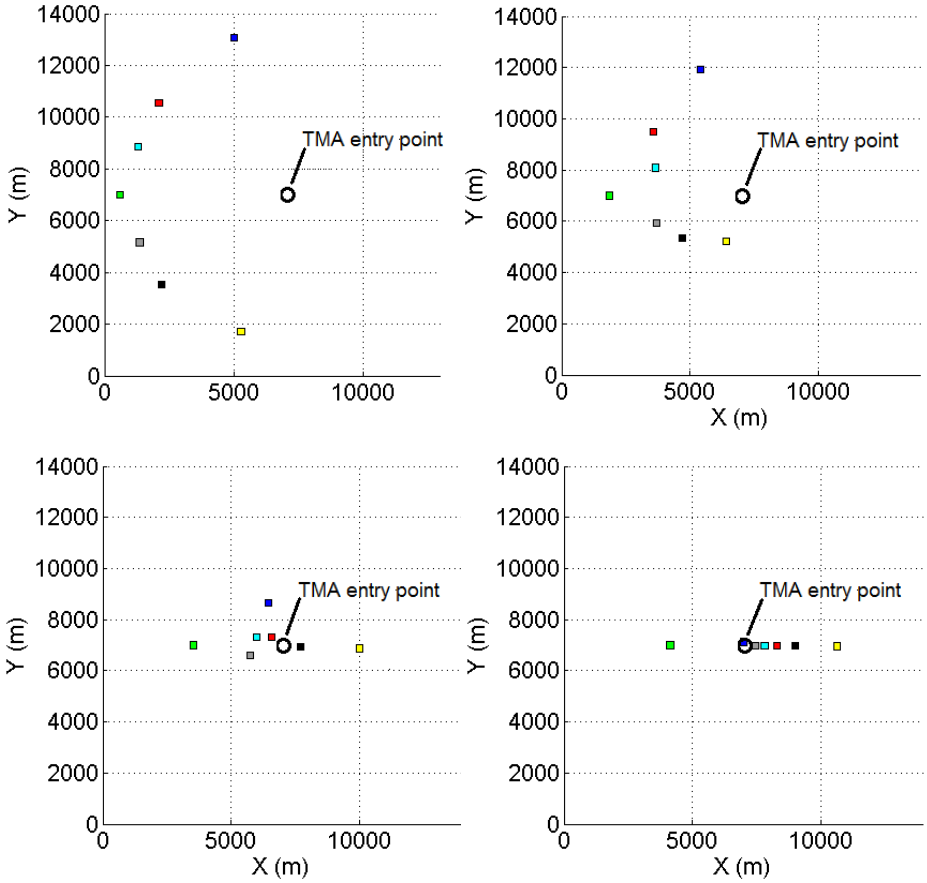


Figure 32: Solution of the scenario S3 with 7 aircraft. Ilustration of trajectories.

The first study in the following, considers four aircraft in scenario S3 – unlocked. This scenario presents a simplification of the converging air traffic problem. There are no priorities between aircraft. The sequencing is conditioned by optimal criterion, that is, minimum time arriving to the final state of the problem.

Figure 33 shows the trajectories of the solution by using the hp-Adaptive method. Waypoints are represented in this case by circles. Figure 34 depicts the computed speed profile. It can be seen that a lot of changes of speed are needed. This is because of the selected optimal criterion. In order to solve the problem as fast as possible, then, many changes of speed are needed. Note that all aircraft are unlocked, so each one presents a different speed profile with the initial and final speed being set at cruise speed.

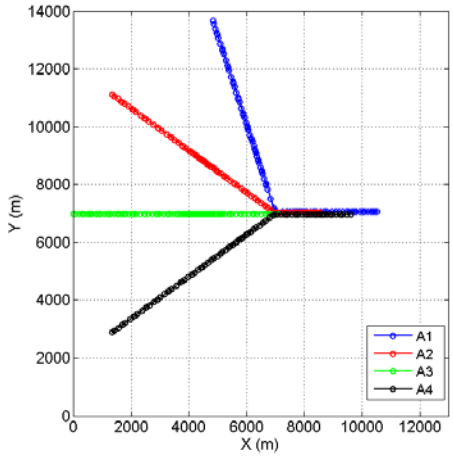


Figure 33: Solution of the scenario S3 with 4 cooperative aircraft. Trajectories.

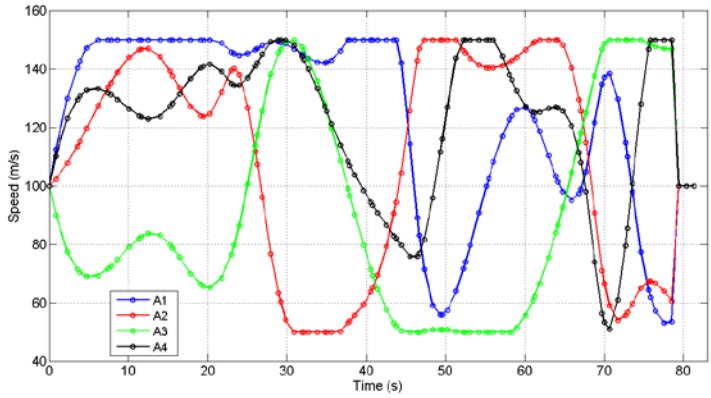


Figure 34: Solution of the scenario S3 with 4 aircraft cooperatives. Speed profiles.

Analyzing both pictures, it is very interesting to see the high number of collocation points (waypoints) which have been used by the hp-Adaptive method. In the following sections, it can be seen that it is not necessary to use so many collocation points when other paradigms, such as the S-Adaptive PS collocation method, are used.

Finally, in order to check obtained solutions, the waypoints of the trajectories have been interpolated in time by a fixed step (0.5 seconds). Then safety distances between every two aircraft have been measured. Figure 35 shows the result obtained in this validation process. Note that the safety distance (500 m) is met during the whole flight, so the trajectories are safe. During the phase and the access to the entry point, it can be seen that distances are close to the minimum allowed. Such is, the case of aircraft A2 and A4, but it can be concluded that the hp-Adaptive method solves the problem.

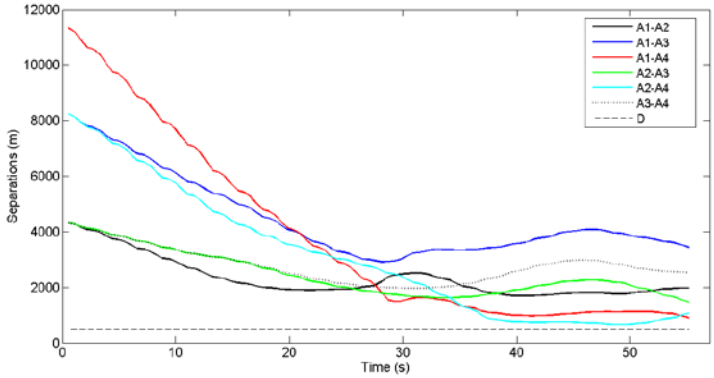


Figure 35: Solution of the scenario S3 with 4 aircraft cooperatives. Safety distances.

Second study considers four aircraft in scenario S3 – one-locked. In this case, the aircraft A1, A2 and A4 are unlocked and A3 is locked. This means that aircraft A3 will continuously fly at cruise speed while the rest of aircraft speeds vary in order to avoid any collision.

Results of the study are presented in Figure 36 and Figure 37. The trajectories are similar to the previous case, but the speed profile in Figure 37, is considerably different. It can be seen that aircraft A3 (in green) does not change speed (it moves at cruise speed the entire time). Aircraft A1 (in blue) increases speed in the beginning and is the first to arrive to the entry point. Aircraft A4 (in black) also increases its speed and is the second one that arrives to the entry point. Aircraft A2 (in red) initially decreases speed, in order to establish coordination with A1 and A4, and then it increases its speed. A2 is the third one to arrive. Finally, aircraft A3 arrives to the entry point, so it can be seen that the strategy performed by the unlocked aircraft was to reach before the locked aircraft.

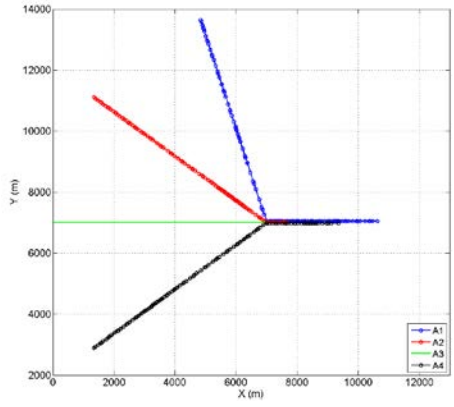


Figure 36: Solution of the scenario S3 with 4 aircraft (1 aircraft no-cooperatives). Trajectories.

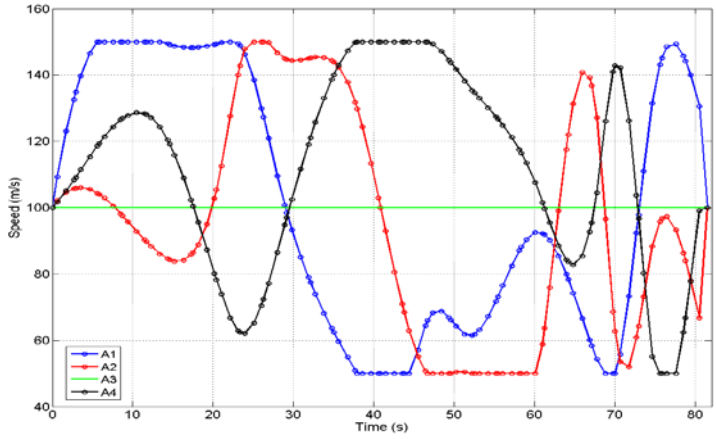


Figure 37: Solution of the scenario S3 with 4 aircraft (1 aircraft no-cooperatives). Speed profiles.

In order to check the solution, the safety distances are presented in Figure 38. It can be seen that all distances between the aircraft are greater than the established minimum safety distance (500 meter). Only the distance between the aircraft A2 and A3 (in green) is approaching the limit. These two aircraft are the last ones to arrive at the entry point and since one immediately follows the other, it is normal that the safety distance is close to the limit.

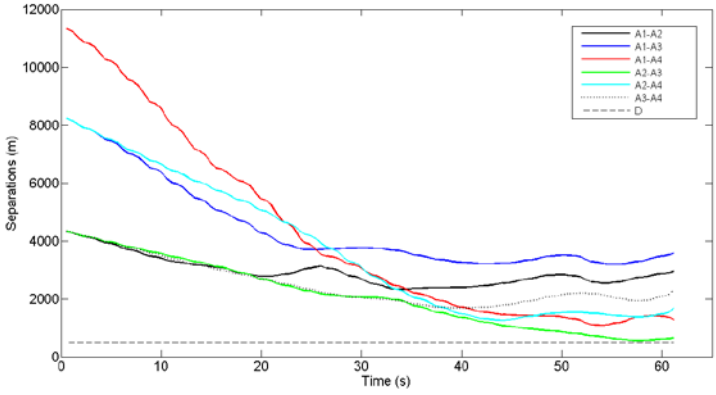


Figure 38: Solution of the scenario S3 with 4 aircraft (1 non-cooperative aircraft). Safety distances.

Scalability of hp-Adaptive method

It is very interesting analyzing the scalability of the hp-Adaptive method again. The aim is to consolidate the scalability of hp-Adaptive other type of application. In this case, the scenario S3 is used and six cases, from two to seven aircraft, are considered in the scalability test. Twenty simulations are performed for each number of aircraft in order to obtain the average of computation time.

Table 7 shows the mean computation time[9], its standard deviation, and the aircraft considered in scenario S3.

Table 7: Scalability results of the hp-Adaptive method in the scenario S3.

UAVs	UAVs considered	Mean time (s)	σ_{Time} (s)
2	A1,A2	3.017	0.013
3	A1,A2,A6	3.683	0.004
4	A1,A2,A3,A6	4.009	0.016
5	A1,A2,A3,A6,A7	9.968	0.038
6	A1,A2,A3,A4,A6,A7	12. 641	0.065
7	A1,A2,A3,A4,A5,A6,A7	25.082	0.236

Obviously, the computation time of the conflict-free trajectories depends on the number of aircraft. It can be seen in Figure 39, that the case of seven aircraft does not follow the growth trend of the other cases. That is because the limit of aircraft that can be used in the scenario S3 is close to seven (taken into account the space available and the safety distance constraints).

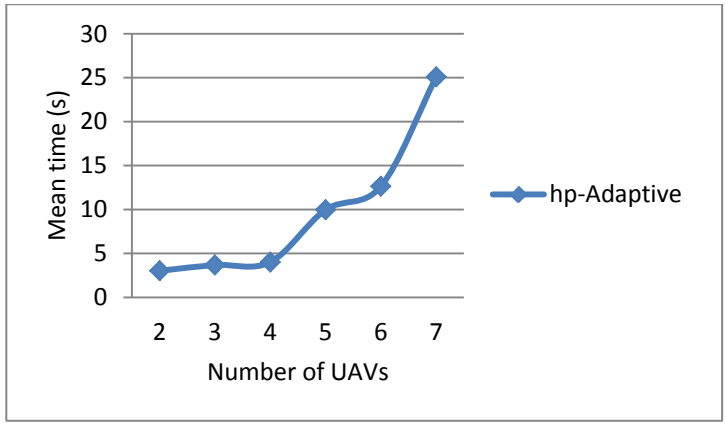


Figure 39: Results of the scalability test of scenario S3.

[9] Computation time of the algorithm in MATLAB

5.2.3. Landing of a UAV on a UGV

The application considered in this section is based on the automatic and safe landing of a fixed-wing UAV on top of a mobile ground platform (UGV). This application has been deployed in one of the five scenarios composing the EC-SAFEMOBIL FP7 project. This scenario is based on the development of safe and reliable landing technologies for very light long endurance platforms in UGV. The aircraft used is a very light solar UAV without landing gear because it obtains a significant increase of the payload capacity, range, endurance and/or performance of the aircraft. In this scenario, UGV are supposed to move along a runway while the aircraft performs its final approach maneuver just before touchdown. Now, the rendezvous operation between two vehicles is produced.

In this scenario, it is required that the trajectories described by vehicles have to be smooth and it is very important to take into account the kinematic and dynamic constraints of the vehicles.

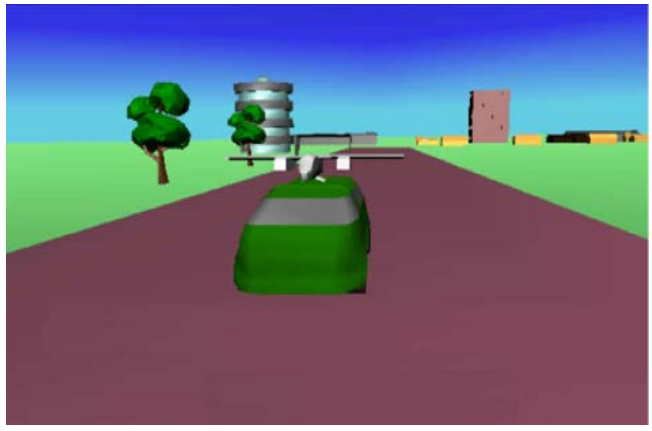


Figure 40: Scenario S4, landing on a UGV.

The system describes in Scenario S4 consist on a 3D system trajectory coordination between 2 different vehicles. The UGV moves on a runway of an airport and describes a trajectory on the ground plane (x, y) while the UAV descends altitude and describes a trajectory in the plane (x, h). Figure 40 shows a picture of the interface used in the simulation. This problem is characterized by the initial and final conditions: The initial condition is that the UAV is flying on an altitude of 60 meter with cruise speed while UGV is waiting at the runway. Final condition is that both vehicles are moving with the same speed and orientation, and the altitude of UAV is the top of UGV. Full characteristics of scenario are given in Table 8. The goal is to find the coordinated trajectories for both vehicles in the minimal amount of time possible.

Two models of vehicles have been considered in this scenario. The fixed-wing UAV model presented in Equation (5.5) and the Ackerman steering UGV model defined in Equation (5.10). The UGV model considers 180º orientations on the runway (never

reverse). Then, the initial orientation of the UGV influences the time in the rendezvous operation. The best case is defined when the UAV and UGV are parallel and the same orientation. The worst case is defined when UAV and UGV are perpendicular.

The hp-Adaptive PS method is going to be used to analyze this third scenario. This implementation of Legendre-Gauss-Radau PS collocation method has the advantage that the problem can be modeled in different and independent phases. This software implementation is able to solve the entire problem, taking into account several phases in which the dynamics, parameters and constraints are different. This propriety is interesting for the modeling process of this scenario.

The optimal criterion considered in this case is the minimum time in the rendezvous operation, Equation (5.16).

Table 8: Specifications of the scenario S4.

Characteristics	
Runway size	200 x 800 m
UGV Minimum Speed	0 m/s
UGV Maximum Speed	30 m/s
UAV Minimum Speed	11 m/s
UAV Maximum Speed	42 m/s
UAV Cruise Speed	35 m/s

Results of hp-Adaptive method

Different configurations or models of the same scenario have been analyzed. Based on the multi-phase setting of the hp-Adaptive method, the opportunity of modeling the entire problem in one or two phases has been studied. Since the performances of the method was better, the decision to split the problem into two phases was taken: the approach phase, in which only the UAV is moving until it reaches a predefined zone, and the landing phase, in which both the UAV and the UGV move coordinately until both vehicles reach a zero relative velocity and the final landing can be performed.

This section solves the worst case scenario of S4. That is, the UAV and UGV initial orientation is perpendicular. Figure 41 and Figure 42 show the trajectories obtained by hp-Adaptive method for both vehicles. In blue is represented the approach phase, while the landing phase is in red. In case of the UGV, it can be seen that the entire trajectory is in red because during the approach phase the UGV is stopped. The initial orientation of UGV is perpendicular to the trajectory of the UAV. Obviously, this initial orientation affects the time needed to perform the operation. In the case of UAV, it can be seen (in blue) that the UAV lost altitude as fast it can, during 6 seconds, and then the

UAV stabilizes the speed and altitude (in 10 meter) to start the second phase.

After 6 seconds of the approaching phase, the UAV is near to the runway and the landing phase starts (in red). In this phase, it can be seen how the UGV meets the speed and orientation of the UAV, while the UAV is approaching to the UGV smoothly. Then in the final state, both vehicles travel in coordination with the same direction and speed, and the UAV altitude is 2 m., so the final state of the problem is achieved and consequently, touchdown phase can be accomplished.

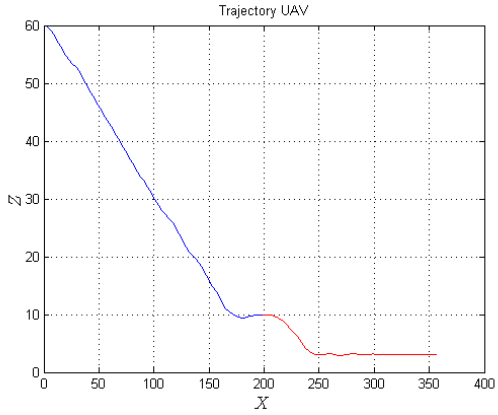


Figure 41: Solution of the scenario S4. Trajectory of the UAV.

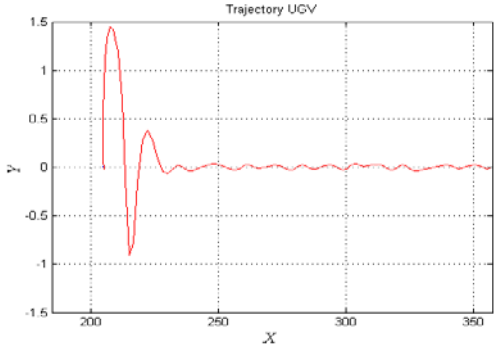


Figure 42: Solution of the scenario S4. Trajectory of the UGV.

Figure 43 and Figure 44 show the speed profile of both vehicles. In blue, The UAV decreases its speed in order to lose altitude. When the UAV reach the altitude of 10 meter, the speed is stabilized in 15 m/s. In red, the UGV start to move and during 6 seconds both vehicles try to match their speeds. It can be seen that the UAV increases its speed at the beginning of this phase. That is because of the change of the altitude. The effects of gravity cause acceleration in the UAV. Finally, the goal is achieved when the speed of both vehicles is around 15 m/s and the altitude of UAV is 2 m.

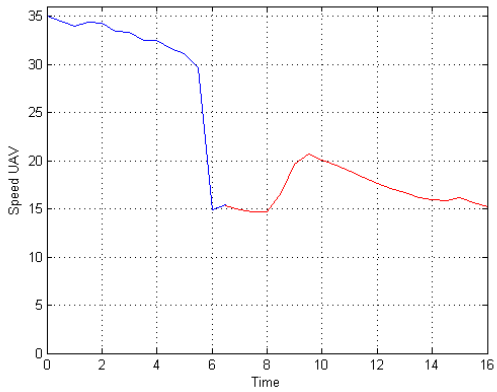


Figure 43: Solution of the scenario S4. Speed profile of the UAV.

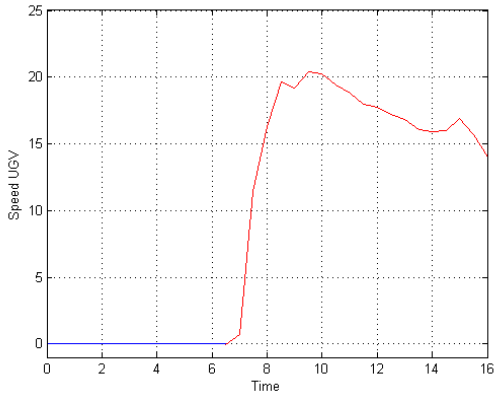


Figure 44: Solution of the scenario S4. Speed profile of the UGV.

5.3. Real-time application with multiple rotary-wing UAVs

The interest of rotary-wing vehicles has been increased in recent years. Particularly, the interest regarding quadrotors has been strongly increased because of its simplicity and low cost. This makes the study of collision avoidance application in new and different scenarios. This section is focused on the analysis of three different collocation methods in the same scenario. Additionally, the novelty presented in this section shows a real-time application of trajectory planning.

The real-time systems of vehicles guidance can be considered in a two-level module: a high level module which generates the waypoints by trajectory planning (following some optimal criterion), and a low level module that focuses on reaching every way point on a singular goal. In this low level module, a minimum computation time is required while in the high level module, a computation time of one or two seconds

(depending of the problem) is enough.

In this section, the design of a high level trajectory planning method is presented. In order to satisfy the constraints in computation time, two specific setting for the hp-Adaptive and the LGL PS collocation algorithms are presented comparing both algorithms with the S-Adaptive PS collocation.

As it has been seen in Section 4.2, S-Adaptive PS method has been developed especially for trajectory planning in low computation time. So in this first section, results of collision avoidance and scalability of this method are studied. Then, a specific configuration of hp-Adaptive algorithms is presented. The hp-Adaptive is a general-purpose method for solving optimal control problem. This algorithm has a few parameters that must be tuned in order to obtain optimal results in trajectory planning problems. Finally the last section focuses on the LGL PS collocation method. In this case, a rolling horizon configuration is presented. This technique consists in planning the trajectory piece by piece. This method is executed periodically in time, and generates a sub-trajectory in each iteration, thus a distributed method is resulted.

In this section, the model in Equation (5.9) has been used. This model has been completed with the parameters of quadrotor shown in Figure 45. This UAV is an AscTec Hummingbird of 200g of mass. The entire parameters necessary for the modeling are given by Table 9.

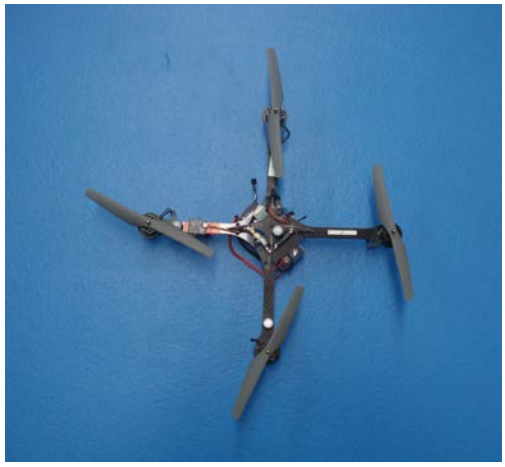


Figure 45: Quadrotor used in the rotary-wing simulations.

The collocation methods implemented in this section are three: an hp-Adaptive PS collocation method and S-Adaptive SP collocation method. Both algorithms are sub-interval PS method presented in Section 3.4.3. The first one is based on Legendre-Gauss-Radau quadrature and the second one is based on Legendre-Gauss-Lobatto quadrature. And finally, the third implementation is a Legendre-Gauss-Lobatto PS method (without segmentation), shown in Section 3.4.2.

Table 9: Parameters of the UAV: Hummingbird.

Characteristics	
Minimum Speed	0.1 m/s
Maximum Speed	2 m/s
Cruise Speed	0.65 m/s
Mass	200 g

The same scenario is used in the three configurations. Teams of 4 or 5 UAVs flying in the same flight level (without fixed obstacles) are presented. Collision avoidance is solved changing the speed and heading as well as velocity planning. Different optimal criteria are also analyzed.

5.3.1. Collision avoidance and scalability

The goal in this section is to test the performances of the S-Adaptive method when it is used as optimal trajectory planning method. This method has been developed to solve optimal trajectory planning in low computation time (see Section 4.2). Next, two studies will be presented: the first one analyzes the results of one scenario in which two different optimal criteria are used, and the second one focuses on the scalability and computation time tested. Furthermore, a comparison between the S-Adaptive and the basic LGL PS method is presented, in which important advantages of S-Adaptive method will be seen.

Figure 46 shows the scenario considered in this section. In this study, two different combination of trajectories in the same scenario S5 are used (scenario (a) and (b)) and two different optimal criteria Equation (5.12), minimum distance traveled and Equation (5.14), minimum changes of input controls are compared in each case. Flight distances are 5 meter. A full description of scenario is given by Table 10. The waypoints are defined by coordinates (x, y) and the speed from that waypoint (v). It is assumed that all UAVs trajectories are known and the ETA and the safety distances must be maintained.

The goal is to find the conflict-free trajectories that minimize the optimal criteria J_d and J_a while the safety distance constraints are met in the entire problem.

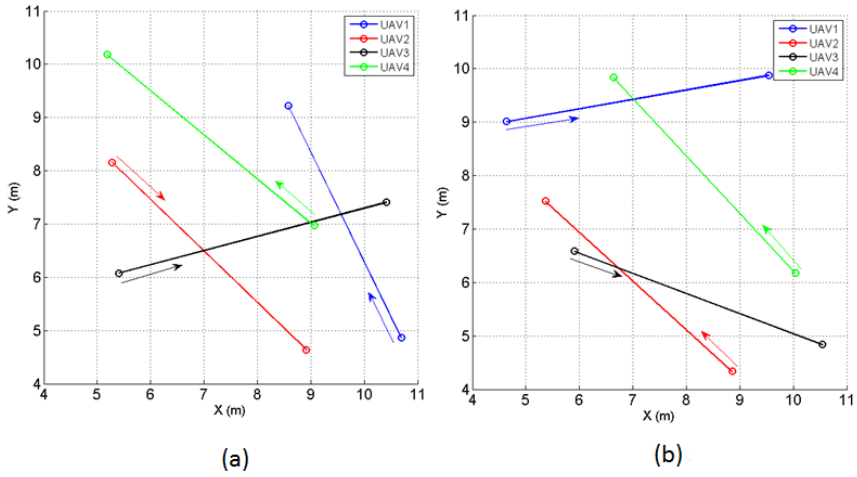


Figure 46: Scenario S5, comparison of two optimal criteria (J_d and J_a). Collision avoidance in the same flight level.

Table 10: Specifications of the scenario S5.

Characteristics	
Area dimensions S5	15 x 15 m
Flight distance	5 m
ETA[1]	9 s
Safety distance	1 m
Optimal criterion J_a	$\sum_{i=1}^{M}\left(\frac{\sum_{j=1}^{N}\lvert\theta_{i,j}\rvert}{N}+\frac{\sum_{j=1}^{N}\lvert\phi_{i,j}\rvert}{N}\right)$
Optimal criterion J_d	$\sum_{i=1}^{M}\sum_{j=1}^{N}\sqrt{(x_{i,j}-x_{i,j-1})^2+(y_{i,j}-y_{i,j-1})^2}$
CDR[2]	Changes of speed & changes of heading

[1]ETA: Expected Time of Arrival.
[2]CDR: Conflict Detection and Resolution.

Results of S-Adaptive PS method

Figure 47 shows the trajectories of the solution of scenario S5 (a) when function cost J_d is used. It can be seen that trajectories obtained are straight trajectories because of the optimal criterion. This means that the collision avoidance is solved by changing the speed profile, also known as velocity planning. The number of collocation point used in the solution is very low, only 15 points have been necessary to solve the problem. The reduction of collocation points is quite important in comparison with hp-Adaptive

method.

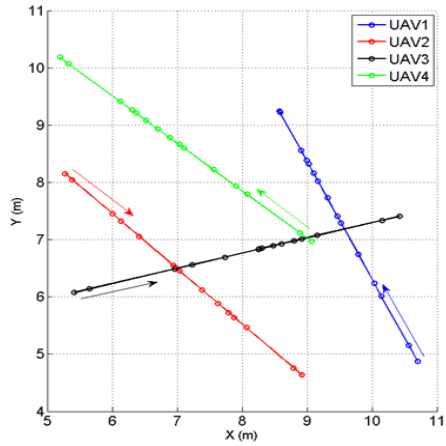


Figure 47: Solution of the scenario S5(a) with J_d optimal criteria. Trajectories.

Figure 48 shows the speed profile of the solution. Analyzing the profile, it can be seen that the range of values of speed are very large (from 0.3 to 1.3 m/s) and the changes of speed are abrupt. One more time, the optimal criterion used affects strongly to the solution.

It can be seen that mainly collisions problems are between UAVs 1, 2 and 3 because the variation of the speeds is important, while UAV 4 maintains it speed more or less without changes. Moreover, coordination between UAV 3 and UAVs 1 and 2 occurred in order to avoid the collision: UAV 3 decreases the speed and UAVs 1 and 2 increase it.

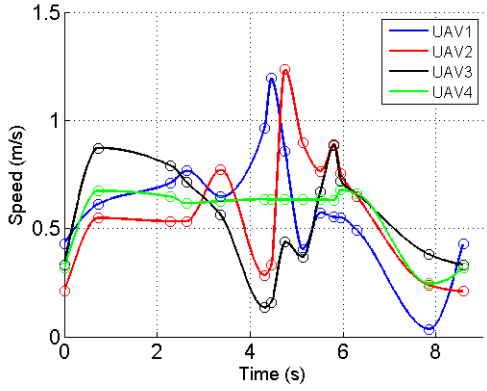


Figure 48: Solution of the scenario S5(a) with J_d optimal criteria. Speed profiles.

In order to perform a comparison, the resulted trajectories of the same scenario S5 (a) when the optimal criterion J_a is used, are presented in Figure 49. Watching the picture, it can be seen that now there are some changes of heading in the trajectories (not straight trajectories). Results obtained consider smooth curvatures because of optimal criterion rewards changing the speed and heading of vehicles at the same. Additionally,

the solution uses less collocation points than in the cases of J_d, then faster is the algorithm.

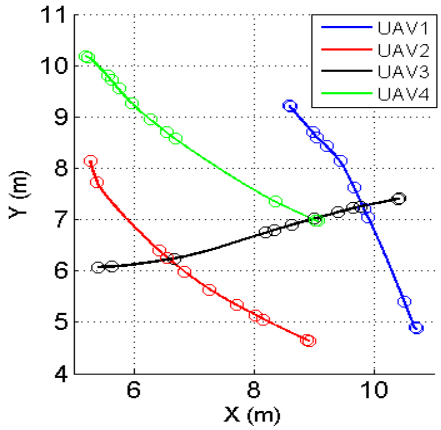


Figure 49: Solution of the scenario S5(a) with J_a optimal criteria. Trajectories.

Figure 50 shows the speed profile of the solution. It can be seen that the speed profile obtained by the J_a and J_d are very different. In the case of J_a, the optimization focuses on the changes of the control inputs, then, smooth curvature in the trajectories and in the speed profiles are obtained. Additionally, this criterion rewards that the range of speed used was very small. Then the conclusion is that this criterion performs better solution (because the abrupt changes of speed generated by the criterion J_d could make unstable the flight of small UAVs).

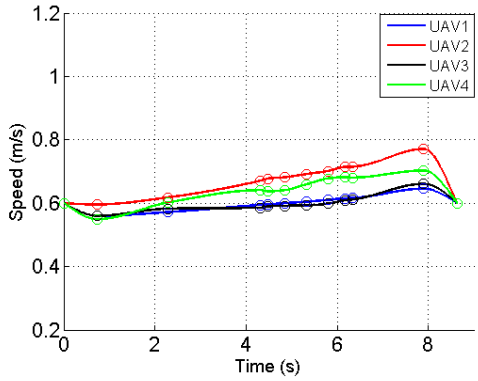


Figure 50: Solution of the scenario S5(a) with J_a optimal criteria. Speed profiles.

Scalability of S-Adaptive method

In this section, a scalability test of the S-Adaptive method is performed. The aim is to test the performance of this new method when the number of vehicles increases in the same scenario. For this study, nine tests have been performed: from 2 to 10 UAVs have been considered in scenario S5. Figure 51 shows the scenario when 10 UAVs are

considered Figure 51. Fifty randomly scenarios have been generated in each of the nine cases, in order to obtain an average computation time.

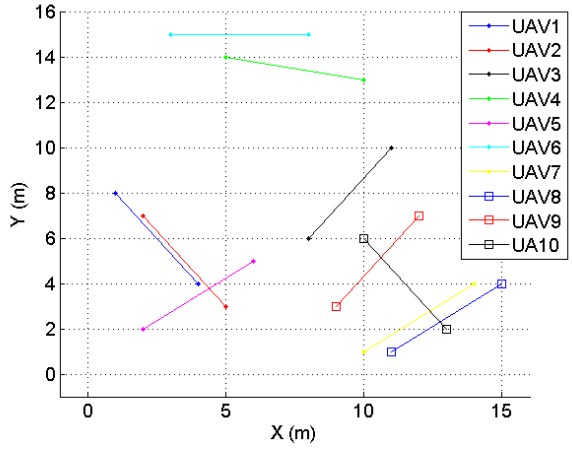


Figure 51: Scalability test of the S-Adaptive method in the scenario S5.

Table 11 and Figure 52 show the execution time[10] and its standard deviation of the test results. Like in previous studies, computation time depends on number of UAVs. As the number of UAVs considered in the same size of scenario increases, the density of collisions increases, and more difficult is to solve the problem. In particular, cases with 9 or 10 UAVs are very difficult to solve and the computation time is high. However, it can be seen in Figure 52 that the rate of increment is not exponential as usually happens in NP-hard problems. Then it can be concluded that the scalability of the S-Adaptive method is good.

Table 11: Scalability results of the S-Adaptive method in the scenario S5.

UAVs	Mean Time (s)	Standard deviation (s)
2	0.4067	0.0733
3	0.7171	0.1621
4	1.2464	0.6382
5	1.7913	0.7876
6	2.6251	1.3515
7	3.0576	1.1229
8	5.0438	1.8169
9	8.7274	3.9441
10	10.2359	3.9312

[10] Computation time of the algorithm in MATLAB

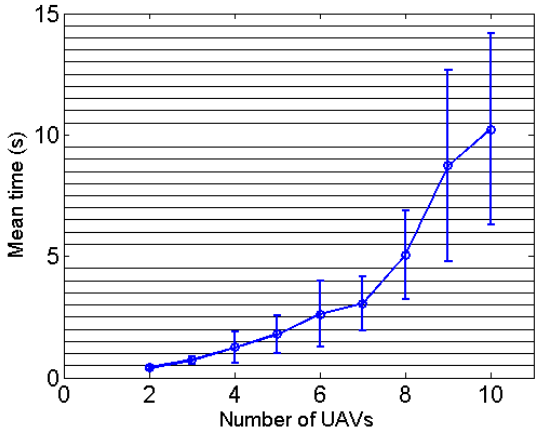


Figure 52: Results of the scalability test of the scenario S5.

Comparative S-Adaptive vs LGL PS method

Many comparisons can be performed in order to point out the advantages of the S-Adaptive method. S-Adaptive PS method is a sub-interval collocation method which performs a specific segmentation process and uses a PS collocation method. In particular, the kernel of S-Adaptive method is based on a Legendre-Gauss-Lobatto (LGL) PS method, so this comparison provides some improvements comparing S-Adaptive and standard LGL PS method.

This study focuses on the quality of the solution and the computation time needed to obtain the solution. In this case, scenario S5(b) has been considered for comparison(see Figure 46). This is a simple scenario composed of four UAVs which have two conflicts (two by two). J_d has been the optimal criterion considered in both cases. This criterion rewards the straight-line trajectories.

Finally, it is important to remark that the LGL standard implementation has been set with seven collocation points. The number of collocation points affects the quality of the solution and the computation time. It has been tested a different number of points and seven collocation points providing a relation between quality and computation time.

Figure 53(a) shows the solution given by the LGL PS method. It can be observed that the trajectories obtained are straight lines, so the conflicts have been solved by velocity planning. At first, the trajectories obtained seem correct, but Figure 53(b) shows the verification process of the solution. The verification process consists in the temporal interpolation of the collocation points and the safety distances checked between every two UAVs using a small fixed step. In the picture (b), it can be seen that safety distances between UAV 2 and UAV 3 has been violated during the period of time [2.5, 4] (green line), so the solution is not correct. The computation time needed for this solution was 1.326 seconds.

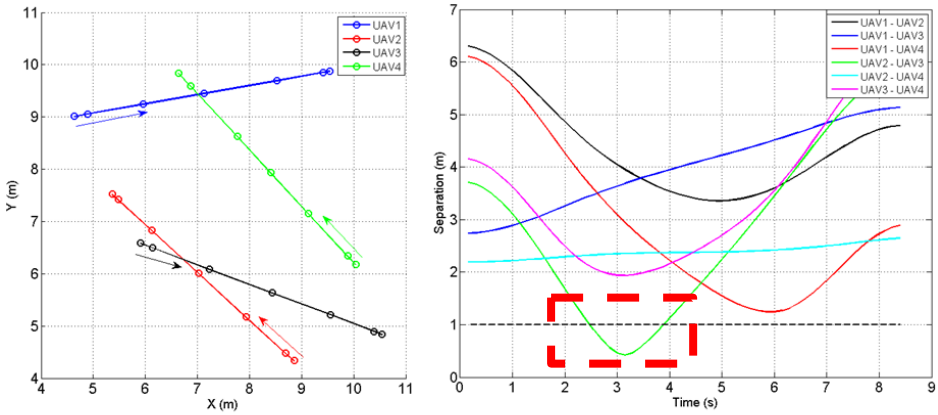


Figure 53: Solution of the scenario S5(b) using LGL PS method. In the left: trajectories. In the right: safety distances.

Several tests have been carried out increasing the number of collocation points. A valid solution has been obtained by computing thirty collocation points and needing several hours to obtain the solution.

The same problem has been solved by the S-Adaptive showing the obtained trajectories in Figure 54. In this case, one of the conflicts (between UAV 1 and UAV 4) has been solved by velocity planning while the other conflict has been solved by changes in the heading. In this case, S-Adaptive only needed 13 collocation points to obtain a good solution (if the S-Adaptive algorithm finds a solution it is because the verification process has been passed and there is no collision in continuous time).

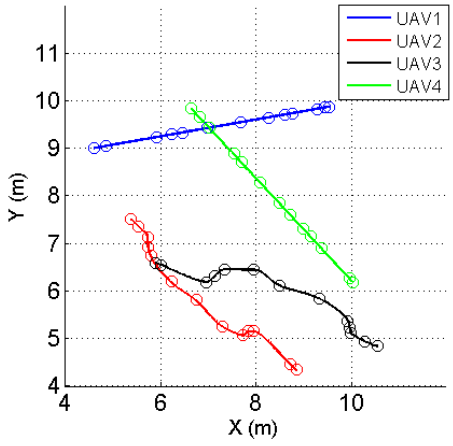


Figure 54: Solution of the scenario S5(b) with J_d optimal criteria. Trajectories.

The computation time needed by S-Adaptive was 1.30 seconds. Considering that LGL

method need hours for solving the same problem with success, it can be concluded that S-Adaptive performs suitable improvements in compare with the LGL PS method.

5.3.2. Real-time configuration of hp-Adaptive PS method

The hp-Adaptive PS method is a general purpose method for solving optimal control problems. This method has a set of parameters which can be tuned in order to improve the performances of the method.

The goal in this section is to study of the best configuration of parameters which performs the hp-Adaptive method as real-time[11] trajectory planning method. As it can be seen in [85] , the hp-Adaptive PS method has the following parameters:

- **Initial mesh**: Is the initial distribution of segments and collocation points. By default, the initial mesh considered is only a segment with twenty nodes.
- **Tolerance error**: Is the allowed approximation error in each collocation point.
- **SplitMult**: Is the index of segmentation of the mesh and it is related with the capability to increase the number of segments in every iteration. It also influences the convergence time of the method.
- **Number of nodes per segment**: Is also called collocation points. This number defines the degree of the polynomial of interpolation used in each segment. The number is defined between four and twelve nodes per segment. Initially, each segment has four nodes. This number can increase in each segment in order to meet the tolerance error.
- **Number of iterations**: Is the number of times that the method can iterate to obtain a solution. In this study, this number is limited to five.

Two scenarios are considered in this study (see scenarios S6 and S7 in Figure 55). Scenario S6 is a typical "star" configuration where the paths of five UAVs cross at the same point, which is considered as one of the worst case scenarios. Scenario S7 is a realistic scenario of 5 UAVs, in which two conflicts are detected and a coordination of the five UAVs is necessary to solve the problem. It is assumed that all UAVs trajectories are known and they are 10 meter long. In both scenarios, the ETA and the safety distances should be maintained. The characteristics of the scenarios are given in Table 12.

As in previous sections, a verification process is considered. Then the waypoints of the solution are interpolated in time, and the safety distance is checked between every two UAV. In this case, safety distance considered is 1 meter.

[11] In this study, the real-time configuration considers a computation time of 1 or 2 seconds.

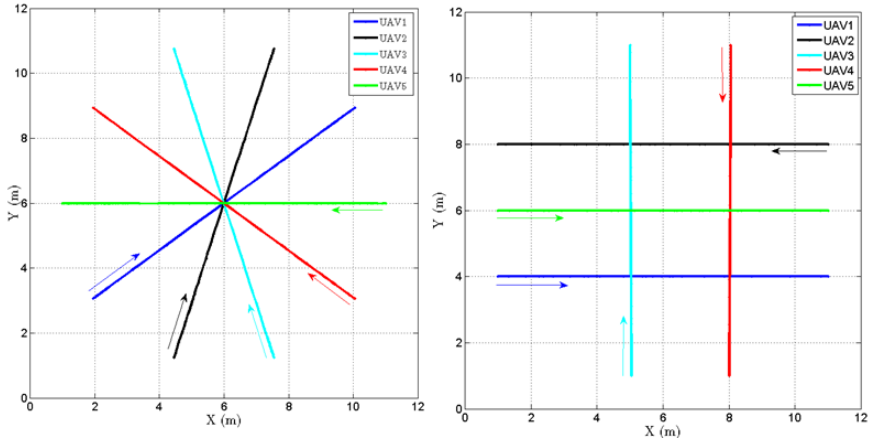


Figure 55: Scenario S6 and S7, collision avoidances with velocity planning. In the left: scenario S6 five-UAVs-star. In the right: scenario S7 lateral cross.

Table 12: Specifications of scenarios the S6 and S7.

Characteristics	
Area dimensions scenario	12 x 12 m
Flight distance	10 m
ETA[1]	15 s
Safety distance	1 m
CDR[2]	Velocity planning

[1]ETA: Expected Time of Arrival.
[2]CDR: Conflict Detection and Resolution.

Description of the study

The goal of the following study consists in finding the best configuration of parameters that provide less computation time of the hp-Adaptive algorithm. Furthermore, this specific setting has to adapt to the number of UAVs.

Three parameters are studied for this purpose: the tolerance error (E_t), the SplitMult (S) and the safety distances (D). The safety distance is a parameter of the problem but sometimes it is a good strategy to set up a bigger safety distance than considered in the verification process. In this way, this extra-margin balances the drawbacks of the discretization process of collocation methods.

The following values of these parameters are studied. In the case of SplitMult, S, the values (1.0, 1.1, 1.2, and 1.3) have been considered. In the case of tolerance error, E_t, the values (0.001, 0.005, 0.01, 0.05 and 0.1) have been considered. Finally taking into account

that the safety distance considered in the scenario is 1.0 meter, the following safety distances have been tested (1.0, 1.1, 1.2, 1.5, 2.0 and 2.5). A hundred and twenty possible combinations of values results in the study. Moreover, this study has been performed in both scenarios S6 and S7 and considering from 2 to 5 UAVs in each scenario.

From the 120 cases (combination of values), only the ones which meet the safety distances constraints between every two UAVs are considered. Finally the minimum computation time is selected. Table 13 and Table 14 show the minimum computation time[12] from scenario S6 and S7 respectively (with the number of UAVs used in each scenario). The resulted values of number of collocation points used in the solution, S, E_t, and D are presented.

Table 13: Results of the real-time setting of hp-Adaptive method in the scenario S6.

UAVs	Minimum time (s)	Collocation points	S	E_t (m)	D (m)
2	0.082	21	2	0.01	1.0
3	0.300	21	1.2	0.01	1.2
4	0.525	21	1.1	0.05	1.2
5	0.880	17	1.1	0.1	1.3

Table 14: Results of the real-time setting hp-Adaptive method in the scenario S7.

UAVs	Minimum time (s)	Collocation points	S	E_t (m)	D (m)
2	0.315	37	2.5	0.005	1.0
3	0.550	29	1.2	0.01	1.2
4	0.7301	21	1.1	0.1	1.2
5	1.1706	25	1.0	0.01	1.3

All the computation times are below 1.5 second and only the time obtained in case of 5 UAVs and scenario S7 is a bit above 1 second (see Table 14). Then these configurations are good for real-time applications

One of these cases is presented to illustrate the solutions obtained. Figure 56 shows the

[12] Computation time of the algorithm in MATLAB

trajectories obtained in the scenario S7 with five UAVs. The execution of this example has been performed with the configuration ($S = 1.0$, $E_t = 0.01$, $D = 1.3$) given by Table 14.

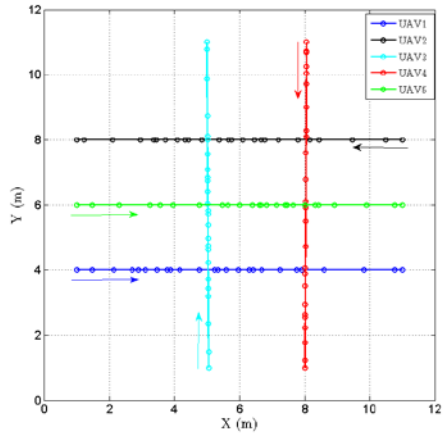


Figure 56: Solution of the scenario S7 with 5 UAVs. Trajectories.

Figure 57 depicts the speed profiles computed to avoid the collisions.

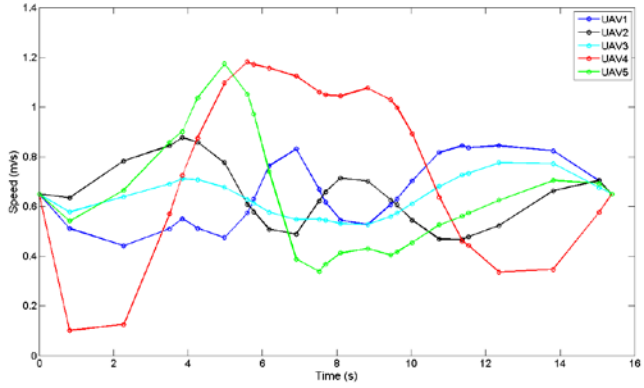


Figure 57: Solution of the scenario S7 with 5 UAVs. Speed profiles.

Finally the verification process of the solution is shown in Figure 58. It can be seen that all the distances are greater than 1.0 meter (the minimum safety distance), so the parameters of configuration are good.

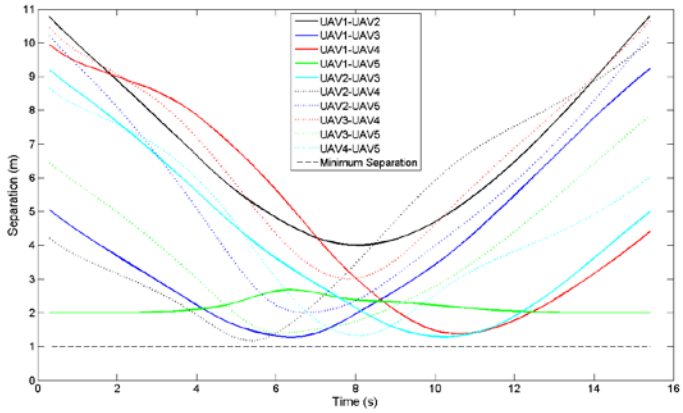


Figure 58: Solution of the scenario S7 with 5 UAVs. Safety distances.

5.3.3. Real-time configuration of LGL PS method

As seen in previous Section 5.3.1, the standard LGL PS method needs a high number of collocation points to obtain good solutions. However, in this section is presented a different point of view that provides a solution to this method. A rolling horizon configuration is proposed as novelty to solve the problem of computation time.

Rolling horizon is a global concept that proposes the generation of trajectories piece by piece. This is an iterative method such that it computes a solution (an optimal solution in this case) of a sub-problem in each iteration. The premise followed in this concept is that if the trajectory is solved piece by piece and every piece is solved by an optimal method, the entire trajectory should be close to the optimal solution.

The trajectory length in every sub-problem is limited by a suitable look-ahead time, T_{la}, which is determined by limits in the perception of UAV (vision sensors, radar, laser, etc.) Figure 59 shows an illustration of the method: Two UAVs have to fly from A to B and from C to D, respectively. Dashed lines represent the initial trajectories and the blue lines represent the trajectories flown until a certain time t. In this instant a new sub-trajectory is calculated (represented by arrows). Note that this sub-trajectories always point to their destinations. In the next instant $t + 1$, the UAVs will run this sub-trajectory while is computing the following sub-trajectory.

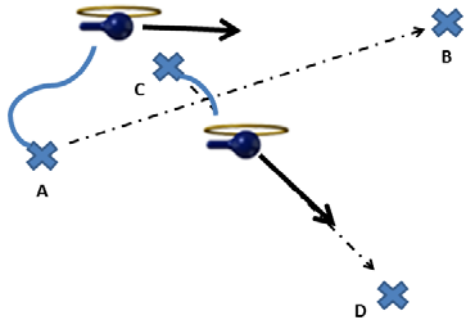


Figure 59: Rolling horizon strategy.

This rolling horizon strategy uses the LGL PS method to plan every sub-trajectory using a few collocation points. By this way, computation time is slow, trajectories converge to their goals, and a decentralized and reactive method is obtained.

Four parameters have to be studied for a rolling horizon configuration:

- **Computation time** (T_c): Is the time that algorithm need to obtain a sub-problem solution.
- **Frequency of execution** (T_f): Is the frequency which algorithm is launched.
- **Look-ahead time** (T_{la}): Is the time in which each UAV knows the information on the rest of UAVs. This variable can be translated to distance by considering the maximum speed of the vehicle (that is the worst case).
- **Number of collocation points** (N_c): Is the number of points used in the LGL PS collocation method to generate a trajectory. The more points are used, better is the quality of the trajectory and greater is the computation time.

Moreover, some constraints have to be met in order to ensure the safety of the vehicles in this implementation. The safety of this method is based on that the UAVs must not fly more distance than given by the look-ahead (T_{la}). For this the frequency of computation should be determined in order to ensure that each UAV does not fly the whole trajectory computed in the previous iteration. Then T_f should be larger than the computation time of the UAV trajectories:

$$T_f > T_c \tag{5.17}$$

Figure 60 illustrates the performance of the proposed method. Five collocation points have been considered in this example (blue points) as well as a suitable look-ahead time. First the trajectory of each UAV is computed within look-ahead time. The iterations are computed every T_f , so the computation time, T_c in the second iteration should be fulfilled:

$$T_{la} - T_f > T_c \tag{5.18}$$

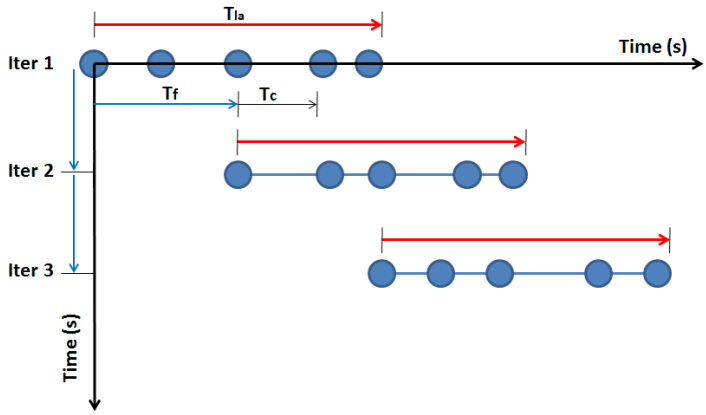


Figure 60: Description of the rolling horizon strategy.

Considering these constraints, the analysis of these parameters is critical to ensure the accuracy of this implementation. For this purpose, the scenario S8 in Figure 61 is considered. This scenario is composed by 4 UAVs and 5 meter trajectories randomly generated. The waypoints are defined by coordinates (x, y) and speed from that waypoint (v). The characteristics of the scenario are given in Table 15.

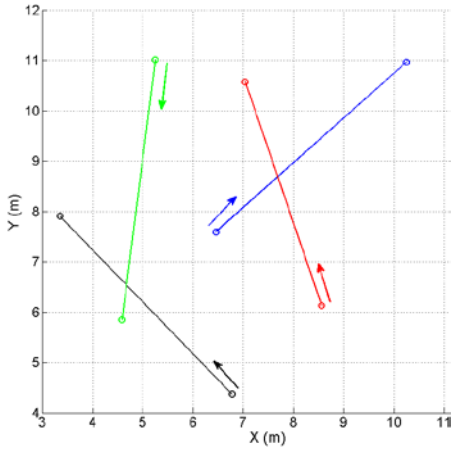


Figure 61: Scenario S8, collision avoidance with rolling horizon strategy.

Table 15: Specifications of the scenario S8.

Characteristics	
Area dimensions Scenario	9 x 9 m
Flight distance	5 m
Safety distance	1.10m
CDR[1]	Changes of speed & changes of heading

[1]CDR: Conflict Detection and Resolution.

Description of the study

The goal presented in this study is to find which values of T_{la} and N_c provide less computation time and accuracy in the solution. Rolling horizon strategy generates trajectories piece by piece, but in the verification process, the entire trajectory generated will be checked.

Taking into account the size of scenario S8, the following range of values for variables in the study have been considered. The maximum value of look-ahead for this vehicle is 3 second, then the following values are selected (1.0, 1.5, 2.0, 2.5 and 3.0). Then the number of used collocation points, N_c, are (3, 4, 5, 6, 7 and 8). Taking into account theses values, thirty possible combinations are explored. Moreover, one hundred random simulations are performed in each combination in order to obtain the average computation time and its standard deviation.

Table 16 shows the results of the study. Only eight of thirty cases have passed the verification process. The average computation time[13], the standard deviation and the number of colocation points are presented for these cases.

Table 16: Results of the real-time setting of LGL PS method with rolling horizon in the scenario S8.

T_{la} (s)	N_c	T_c (s)	σ_{Tc} (s)
2.5	4	0.808	0.207
2.5	3	0.819	0.212
1.0	4	0.815	0.258
3.0	3	0.824	0.304

[13] Computation time of the algorithm in MATLAB

1.5	3	0.837	0.197
2.5	6	0.847	0.219
1.0	7	0.851	0.207
2.0	6	0.873	0.234

The parameters that conform the rolling horizon constraints have to be selected. First parameter is the frequency of execution T_f which is chosen as the worst case of computation time in the table. In order to meet Equation (5.17), T_f should be greater than 1.107s (the larger value of $T_c + \sigma_{\mathrm{Tc}}$ in Table 16), then $T_f = 1.25$ seconds is considered.

It can be seen that only first, second, fourth and sixth cases in Table 16 meet Equation (5.18). Among these options, the minimum computation time is chosen to finish the study. Therefore, T_{la} = 2.5s and N_c = 4 are chosen as the most effective configuration. Therefore this configuration can be used in scenarios, as scenarios in which the number of UAVs is lower (from 1 to 4 UAVs).

Results of rolling-horizon LGL PS method

An example of execution of the method is presented to show its performance. The LGL PS method is executed every T_f with the parameters obtained in the study. Figure 62 shows the scenario S8 with four UAVs in four instants which correspond to t = 1.25, 3.75, 5.00, 6.25s. In the temporal evolution, it can be seen that four short trajectories converge to its goal waypoint.

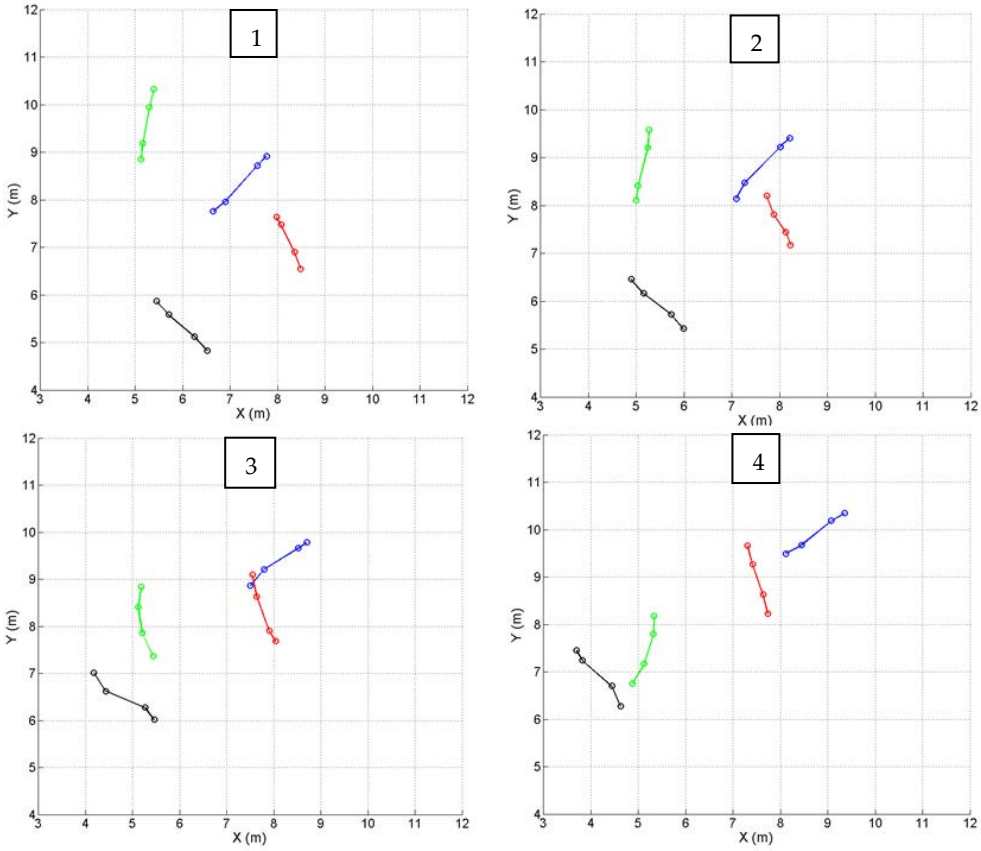


Figure 62: Solution of the scenario S8 with 4 UAVs. Ilustration of trajectories in four moments.

Figure 63 depicts the entire trajectory of the problem and Figure 64 shows the speed profile computed during the trajectory. Note that collisions are avoided by changes of speed and heading. Although trajectories are not optimal, all the trajectories converge to its goal as far as possible.

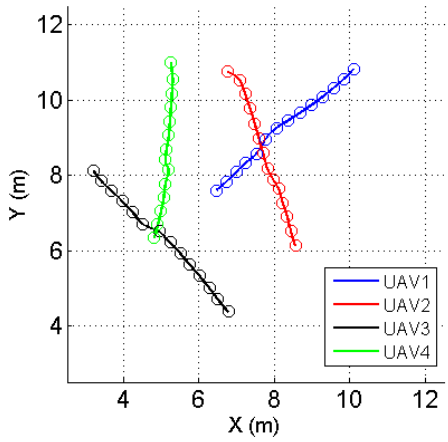


Figure 63: Solution of the scenario S8 with 4 UAVs. Full trajectories.

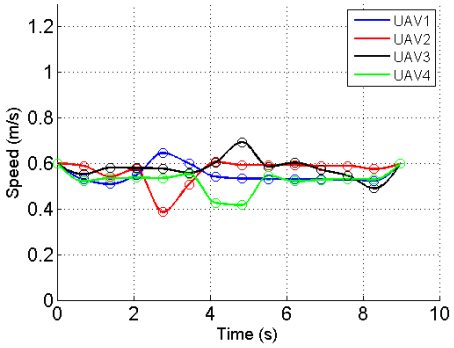


Figure 64: Solution of the scenario S8 with 4 UAVs. Full speed profiles.

Figure 65 presents an extreme case of scenario S8. This scenario considers a star configuration in which the collision conflict is the same for the four UAVs. In this case, it can be seen that the UAV 2 reduces the speed and is the last one to the collision point. UAV 2 changes the trajectory and the collision point as well. The other two UAVs change their speed and arrive to their goals without many problems.

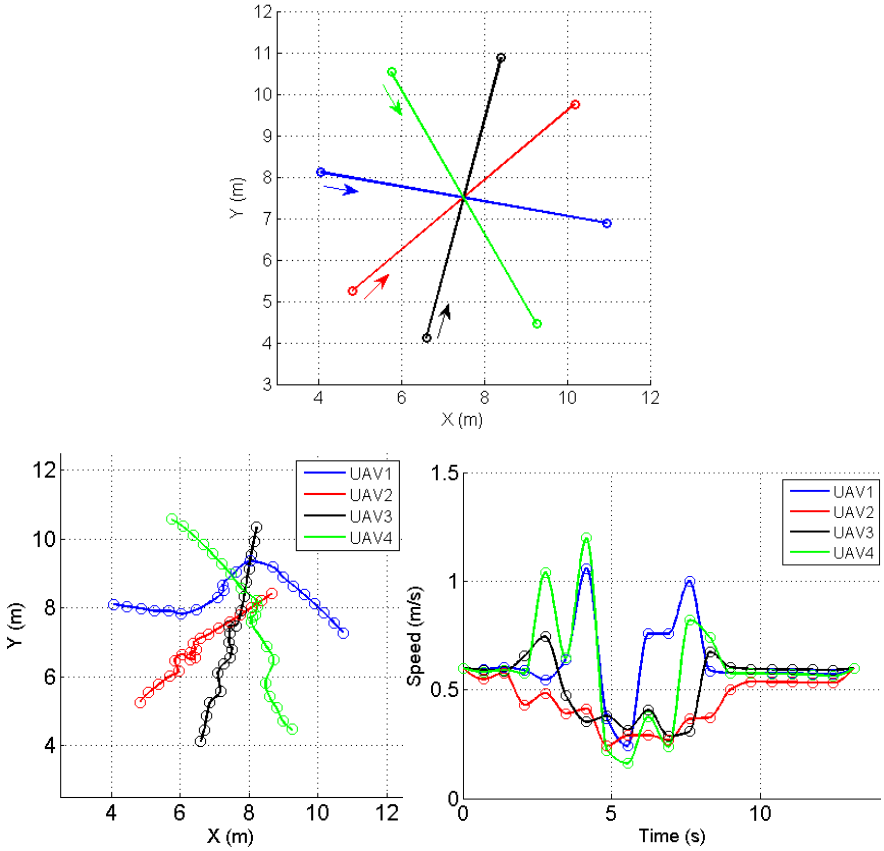


Figure 65: Scenario S8, star configuration with 4 UAVs. Trajectories and speed profiles.

In order to check the accuracy of the solution, the separation between every two UAVs is presented in Figure 66. All the distances are greater than 1.0 meter (the minimum allowed), Then it can be concluded that this specific configuration of rolling horizon with LGL PS collocation method is safe and efficient.

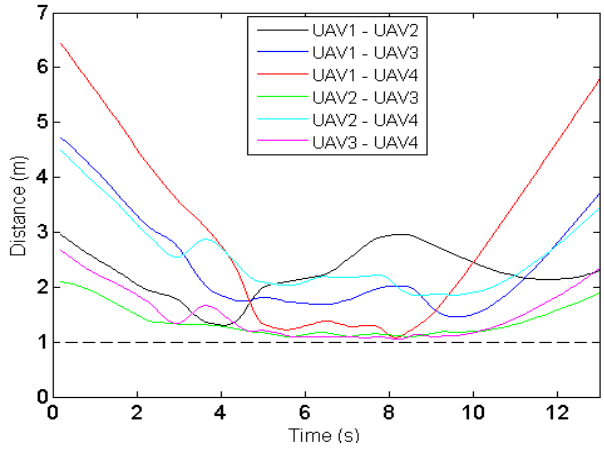


Figure 66: Solution of the scenario S8 (star). Safety distances of the full trajectories.

5.4. Teams of multiple UGVs

A trajectory planning application with autonomous ground vehicles is presented in this section. Particularly an automated warehousing as shown in Figure 67 is considered. This specific scenario corresponds to the scenario 4 developed in the EC-SAFEMOBIL European project. In this scenario a set of autonomous forklifts (AGVs) is used to transport pallets between storage areas, and for palletization stations and delivery trucks, all while sharing their workspace with human-operated vehicles and human workers. This is a real life application implemented by a company.

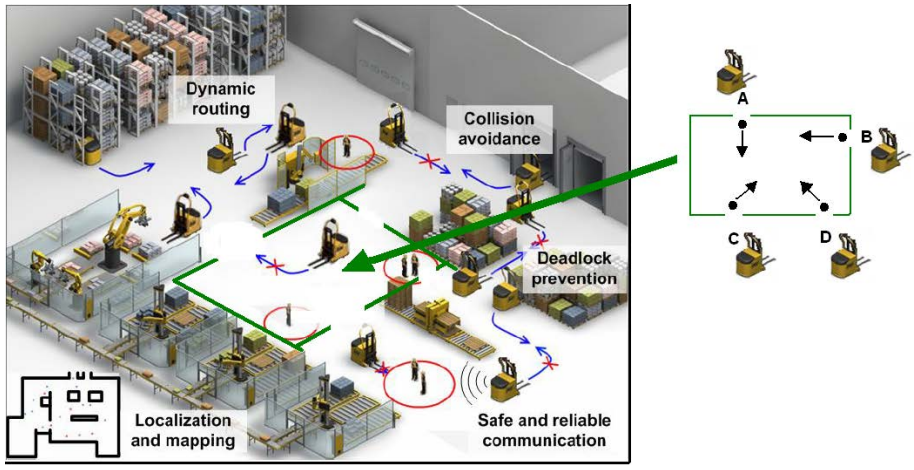


Figure 67: Context of the scenario S9, automated warehousing.

The fixed obstacles in trajectory planning scenarios solved by collocation method are presented in this section as a novelty. The resolution of optimal trajectory planning problems with fixed obstacles assumes a substantial increase in the difficulty of finding a solution. The number of constraints increases with the number of obstacles considered.

Three scenarios are presented for this application. First scenario represents the case in which several AGVs have to move in the same limited area without any collision. The second and third scenarios represent a similar scenario to the first one but with one and three obstacles respectively in the limited area.

Simulations shown in this section have been performed according to the model in Equation (5.11). This model has been completed with the parameter of the Pionner robot shown in Figure 68. The parameters needed to model the vehicle are given in Table 17. The S-Adaptive PS collocation is the method selected for solving this problem. This belongs to the sub-interval PS method defined in Section 3.4.3.

Figure 68: Pionner 3-DX used in the simulation.

Table 17: Parameters of the UGV: Pionner.

Characteristics	
Minimum Speed	0 m/s
Maximum Speed	1.2 m/s
Cruise Speed	0.5 m/s
Maximum angular speed (R and L)	12.30 rad/s
Horizon time	0.01 s

.

5.4.1. Collision avoidance without obstacles

The novelty in this section is given by the usage of UGVs in trajectory planning problems. In this case, the S-Adaptive method has been used due to the good and fast results obtained with UAVs.

Figure 67 shows the real scenario of warehousing problem. From this scenario, Scenario S9 has been modeled by extracting the shared area (see Figure 69). This scenario consists in 4 UGVs initially localized in each corner of the scenario. Waypoints are defined by 2D coordinates (x, y) and speed from that waypoint (v). The characteristics of scenario are given in Table 18.

The optimal criterion considered in this case is the minimum distance Equation (5.12). The goal is to find the conflict-free trajectories that minimize the optimal criteria and meet the safety distances constraints of the problem.

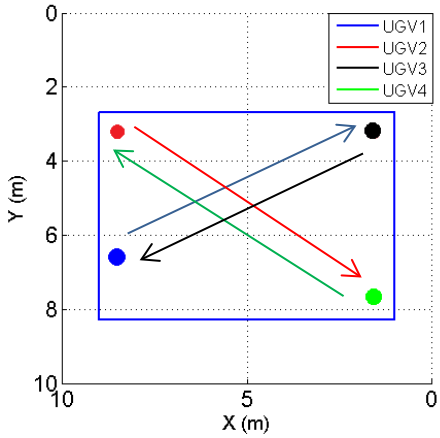


Figure 69: Scenario S9, modeled of the warehousing problem.

Table 18: Specifications of the scenario S9.

Characteristics	
Area dimensions Scenario	8 x 5 m
Distance of trajectory	9 m
Safety distance	1 m
CDR[1]	Changes of speed & changes of heading

[1]CDR: Conflict Detection and Resolution.

Results of S-Adaptive PS method

The trajectory planning results of the scenario S9 are given by Figure 70 and Figure 71. Figure 70 presents the trajectories given by the S-Adaptive method. The solution is comprised of by 14 waypoints. It can be seen how UGVs move to the center in this scenario, they move in a circle in order to avoid a collision in the center of the scenario. When the collision is avoided, each UAV moves to its goal point.

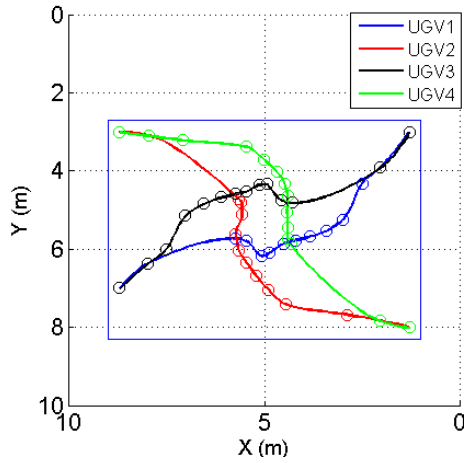


Figure 70: Solution of the scenario S9 with 4 UGVs. Trajectories.

Note that most of the collocation points are set in the center of the scenario, where the collision is avoided. This is because S-Adaptive method only increases the number of collocation points when it is necessary.

Figure 71 shows the speed profiles of the solution. In this case, profiles are similar. There are only some differences in some instant of time because all UGVs perform the same avoidance strategy and navigate the same distance.

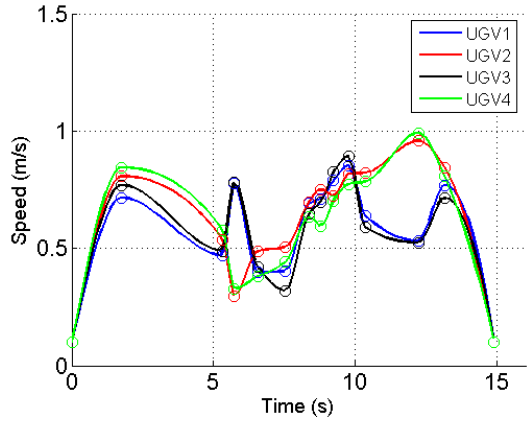


Figure 71: Solution of the scenario S9 with 4 UGVs. Speed profiles.

5.4.2. Collision avoidance with fixed obstacles

As far as can be ascertained, this is the first instance that fixed obstacles have been considered in collocation methods in order to solve a trajectory planning problem with ground vehicles. Two scenarios are presented in this section related with the collision avoidance application. The problem is more difficult to solve because the

safety distance has to be maintained between every pair of UGVs and between each UGV and the obstacles.

Obstacles are modeled by cylinders of one meter altitude and a half meter of radio. The distance between an UGV and an obstacle is calculated by the straight line between position of vehicle and position of obstacle.

Figure 72 shows the scenarios considered in this case (S10 and S11). Scenario S10 only has one obstacle in the middle while scenario S11 has 3 obstacles. The characteristics of scenario are given by Table 19.

The optimal criterion considered in this case is minimum distance given by Equation (5.12). The goal is to find the conflict-free trajectories that minimize the optimal criteria and to meet the safety distances constraints of the problem.

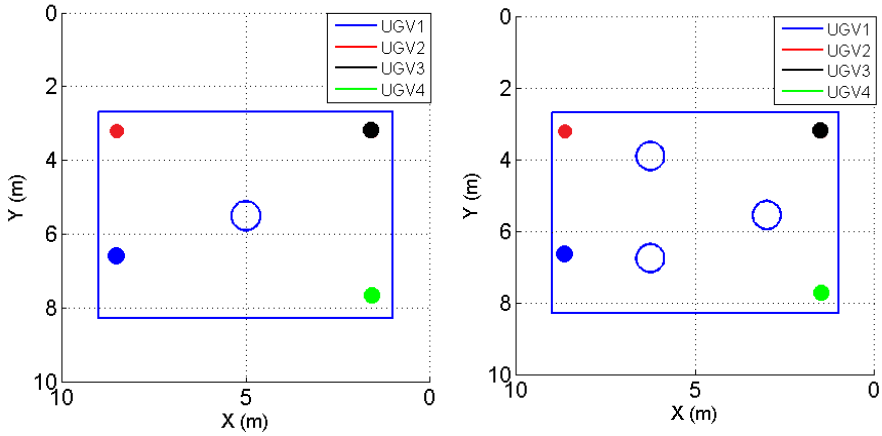


Figure 72: Scenario S10 and S11, modeled of the warehousing problem with fixed obstacle. In the left: scenario S10 one obstacle. In the right: scenario S11 three obstacles.

Table 19: Specifications of the scenarios S10 and S11.

Characteristics	
Area dimensions Scenario	8 x 5 m
Distance of trajectory	9 m
Safety distance	1 m
Obstacle diameter	0.5 m
Obstacle position of scenario S10	X=5, Y=5.50
Obstacle 1 position of scenario S11	X=3, Y=5.55
Obstacle 2 position of scenario S11	X=6.25, Y=3.9
Obstacle 3 position of scenario S11	X=6.25, Y=6.75
CDR[1]	Changes of speed & changes of heading

[1]CDR: Conflict Detection and Resolution.

Results of S-Adaptive PS method in scenario S10

The trajectory planning results of the scenario S10 are given by Figure 73 and the speed profiles of the UGVs in Figure 74. In the first picture depicts the trajectories of the solution. All UGVs moves to the center, as happens in the scenario S9, but in this case, they round around the obstacle, and keep moving to their goals. The safety distance with the obstacle is maintained at all times.

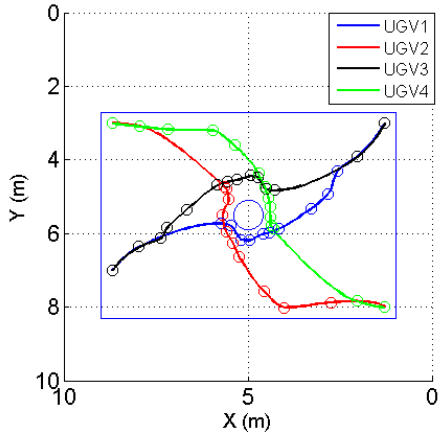


Figure 73: Solution of the scenario S10 with 4 UGVs. Trajectories.

Figure 74 shows the speed profiles of the solutions. As happen in scenario S9 speed profiles remain similar because they all utilize the same strategy.

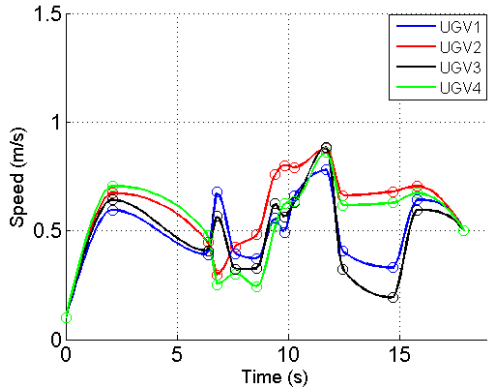


Figure 74: Solution of the scenario S10 with 4 UGVs. Speed profiles.

In order to check the safety distances of the solution, Figure 75 shows the distances between every two UGVs. It can be seen that the safety distance (1 meter) is constantly maintained. Only when the UGVs are in the middle of the scenario, the distances between vehicles are close to the limit.

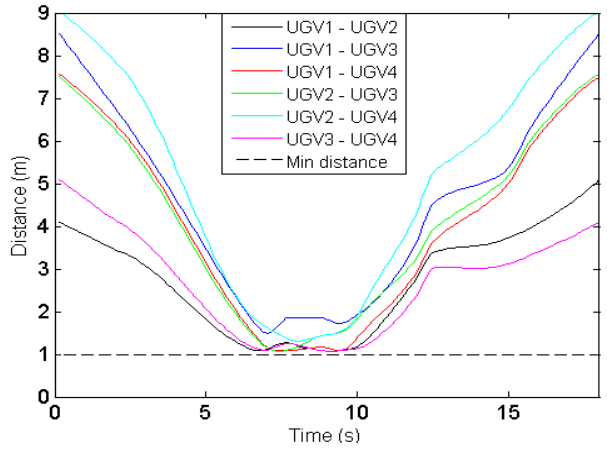


Figure 75: Solution of the scenario S10 with 4 UGVs. Safety distances between UGVs.

In this second picture (Figure 76), the distances between every UGV and the fixed obstacle are presented. In this case the safety distance is 0.5 meters. It can be seen in the picture that all the distances are greater than 0.65 meters then the safety constraints are met although the trajectories are very close to the object due to the optimal criterion (minimum distance traveled).

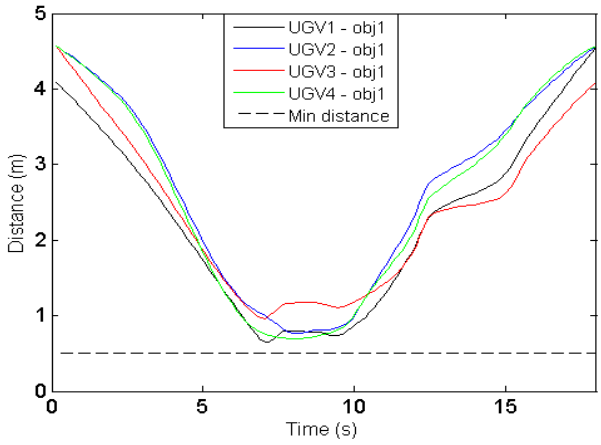


Figure 76: Solution of the scenario S10 with 4 UGVs. Safety distances between UGVs - obstacle.

Results of S-Adaptive PS method in scenario S11

This last scenario S11 presents an evolution with respect to scenario S10; that is, three obstacles are considered, increasing the number of problem constraints. The goal in this section is to analyze the behavior of S-Adaptive in this context. This scenario is much more difficult than scenario S10 because the free space is smaller than in previous scenario.

Solutions to the problem can be seen in Figure 77. Four smooth trajectories are obtained. It can be seen how UGVs move between the obstacles, looking for an optimal solution, until a destination is reached. A total of twenty collocation points have been needed to solve the problem.

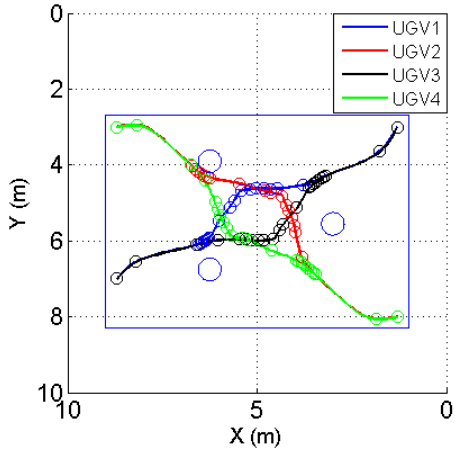


Figure 77: Solution of the scenario S11 with 4 UGVs. Trajectories.

Figure 78 shows the speed profiles of the solution. In this case the profiles are more abrupt than in scenario S10. Meaning that the solution is closer to the limits and confirms the scenario is more difficult. Anyway the speed profile is inside the limits considered for these vehicles.

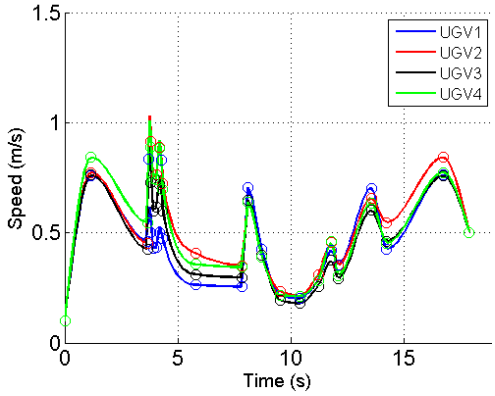


Figure 78: Solution of the scenario S11 with 4 UGVs. Speed profiles.

Finally, the verification process is presented in Figure 79 and Figure 80. In the first picture it can be seen that safety distances between UGVs are equal or greater than 1 meter, meeting the safety constraints. As happens in previous scenarios, during the collision avoidances in the center of the scenario, the UGVs are very close because of the optimal criterion.

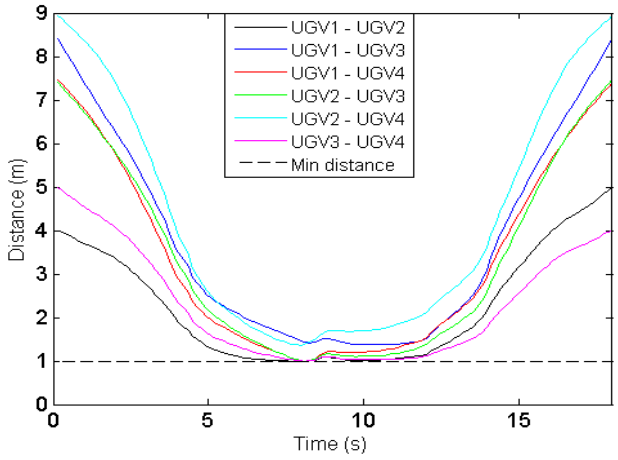


Figure 79: Solution of the scenario S11 with 4 UGVs. Safety distance between UGVs.

Figure 80 shows the distances between every UGV and every obstacle. The safety considered with the obstacles is 0.5 meters. It can be seen that all the UGVs move near one obstacle in order to perform the shortest trajectory. In the case of the UGV 1 and object 2 and UGV 3 and object 3, the distances are just in the limit, but in any case, all the safety constraints are met.

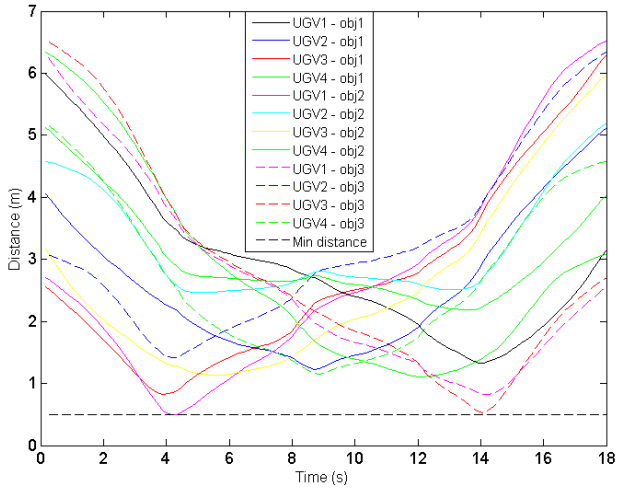


Figure 80: Solution of the scenario S11 with 4 UGVs. Safety distances between UGVs – obstacles.

5.5. Conclusions

Trajectory planning for four kinds of applications has been presented in this Chapter. Both hp-Adaptive and S-Adaptive PS collocation method have been used. In all cases, performances of accuracy, scalability or computation time have been analyzed.

First study focuses on fixed-wing aerial vehicles. In this section, performances of hp-Adaptive method are analyzed in three scenarios: collision free trajectory generation, an ATM converging air traffic problem and a landing of a UAV on a UGV problem. Quality results are obtained in these three cases. Specifically the scalability presented by hp-Adaptive is better than others optimal trajectory planning methods (a comparison between Heuristic Velocity Planning with Optimization Phase and hp-Adaptive PS method is presented). On the other hand, the number of collocation points used by hp-Adaptive is very high and a strong dependence between collocation points and computation time has been proven.

Secondly this study focuses on analyzing ways in which to reduce the computation time of collocation methods as much as possible. In this case, rotary-wing aerial vehicles have been used as well as different optimal criteria being tested. In this section, the efficiency of S-Adaptive algorithm is demonstrated first. Quality results have been obtained as well as very good scalability results. Later two specific configurations of the hp-Adaptive and LGL PS algorithms are presented. In the first case a specific setting is obtained from a deep study of the different tuning parameters of hp-Adaptive algorithm. In the second case a rolling horizon technique is implemented by using Legendre-Gauss-Lobatto PS method. Quality of the results in both algorithms with their specific configurations has been demonstrated.

Third and lastly this study focuses on the trajectory planning for multiple ground

vehicles in scenarios with obstacles. This study presents itself as a novelty in the collocation method literature. In this case, only the S-Adaptive algorithm has been used because of high level of scenario constraints (multiple UGVs and objects). Three different configurations of the same scenario have been tested with successful results.

The following chapter will present the experimental validation of some of the scenarios presented here. For this, two testbeds (one with Hummingbird quadrotors and the other with Pioneer 3DX) will be used.

6. EXPERIMENTAL VALIDATION

When everything seem to be going against you, remember that the airplane takes off against the wind, not with it.

Henry Ford

This chapter presents the validation of some of the methods and scenarios performed in the previews chapter. The goal in this chapter is the validation of the trajectories obtained by the different collocation methods with real aerial and ground vehicles. For this purpose, two indoor testbeds have been used.

6.1. Introduction

Different kinds of experiments are presented in this section in order to evaluate the behavior of the trajectories obtained by the different implementations of collocation methods used in the previous chapter. The validation of the trajectories focuses on checking the safety distance constraints before running the trajectories. Four sections are presented in this chapter following a similar order of the previous chapter. The first one introduces the two testbeds used for the different experiments. A full description of vehicles, software architecture and communication is presented. In Section 2 one fixed-wing UAVs experiment is presented. Although a model of fixed-wing UAV is used to obtain the trajectories, a rotary-wing UAV will be used in the validation, due to the difficulty to perform a 4 fixed-wing UAVs experiment. Then in Section 3, seven experiments with rotary-wing UAVs are presented. Three different scenarios are tested with autonomous quadrotors. Finally in Section 4, three experiments with UGVs are presented as well. Two of them consider fixed obstacles in the scenarios.

6.1.1. Description of testbeds

A complete description of the testbeds used in the following sections is presented here. The first one focuses on aerial vehicles (UAVs). This is an indoor testbed with some quadrotors. The second testbed focuses on ground robots (UGVs).

The use of fixed-wing UAVs has been restricted in the experiments because of the difficulty of using this kind of vehicles in an indoor scenario. These aircraft need a large scenario and a big communication infrastructure in the case of multiple UAVs flying at the same time.

Testbed of UAVs

The experimental validation of the proposed algorithms with aerial vehicles has been performed in the indoor testbed of Center for Advanced Aerospace Technologies (CATEC) in Seville (Spain). Figure 81 shows a picture of the scenario with 3 flying UAVs.

Figure 81: CATEC Testbed.

This testbed is based on a rectangular prism of 16x15x6 m^3. It uses a localization system based on 20 VICON cameras (like in Figure 82) which is able to offer the position and altitude of each object with centimeter accuracy in real time. Up to 20 mobiles objects can by tracked by the system. Wireless communication between the master computer and mobile vehicles is performed by the Zigbee network.

Figure 82:VICON system.

Up to 8 UAVs quadrotors like the one shown in Figure 83 can be used in this testbed. This is an AscTec Hummingbird quadrotor by Ascending Technologies with 200 gr of payload and up to 20 minutes of flight autonomy. Its maximum speed is 50 Km/h and

the minimum speed is 0 Km/h.

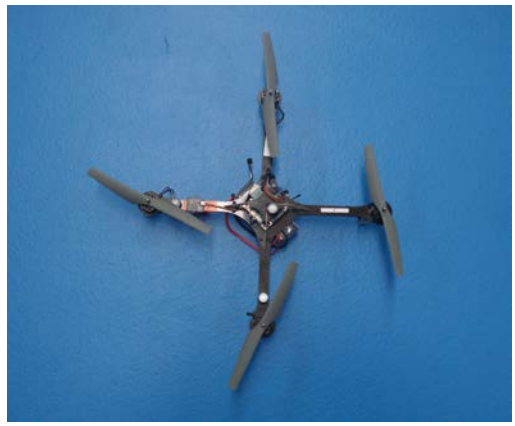


Figure 83: AscTec Hummingbird quadrotor.

The software architecture is based on a master computer and some slaves PCs. The master is a QNX Control Computer (real-time OS) responsible for the reception of data of position and altitude from VICON cameras, the flight plan, and run a software control and send command control to the vehicles through Zigbee network. Slaves PCs are connected by Ethernet to the master computer. These PCs have different software like ROS, Matlab, etc. which runs high level algorithms.

Testbed of UGVs

The experimental validation of the proposed algorithms with ground vehicles have been performed on the Laboratory for Robotics and Intelligence Control System (LARICS) of University of Zagreb, (Croatia). Figure 84 shows the scenario used with two UGVs.

Figure 84: LARICS Testbed.

This testbed is based on a rectangular area of 8 x 5 meter. It uses a global localization based on ROS architecture. This system uses the odometry as well as laser scan values in order to localize on a 2D map. The laser in Figure 85 is used. The system implements an adaptive Monte Carlo localization approach, which uses a particle filter to track the pose of a robot against a known map. Wireless communication between master computer and mobiles robots is performed by WiFi.

Figure 85: Sick LMS-500.

Up to 5 UGVs such as the ones in Figure 86 can be used in this testbed. These are Pioneer 3DX robots. They are equipped with an onboard computer running ROS on Linux OS, WiFi module for communication with other vehicles, as well as Sick laser and odometry sensors for localization purposes.

Figure 86: Team of 4 Pioneer 3DX.

All software architecture is centralized in a master PC. This computer has ROS software which is responsible of receive data of position and the trajectory plan. It runs control software and sends control commands to the vehicles through a WiFi network.

6.2. Teams of multiple fixed-wing UAVs (simulated with rotary-wing)

Execution of experiments with fixed-wing aircraft usually is difficult due to the dynamic of the vehicle. For example, a fixed-wing aircraft cannot stop and wait in the air for the experiment to start. Another problem is that a big outdoor scenario usually is needed to fly multiple aircraft like the vehicles modeled in Section 5.2.

One solution for this kind of experiments is to scale the scenario and run the experiments into a small testbed (indoor or outdoor). Rotary-wing UAVs can be used in order to simulate a fixed-wing UAVs. For this propose, the minimum speed of the rotary-wing should be limited. Thus, once started the experiment, the UAVs cannot stop. Such as happens with fixed-wing UAVs.

The trajectories presented in this section have been calculated according to a fixed-wing UAV model and the S-Adaptive algorithm. But the experimental validation has been performed by rotary-wing UAVs (quadrotors) in the CATEC indoor testbed.

6.2.1. Collision avoidance

The Scenario S2 of Figure 24 has been replicated at scale 1:600 inside of the multi-UAVs testbed presented in Section 6.1.1. The real scenario has a dimension of 10 x 10 m^2 and the flight distances are 7 meters and four UAVs are considered. The scenario is composed of 4 trajectories which replicate a star, so there is only one conflict in the scenario, and which is the same for all UAVs. The conflict resolution method used in this case is velocity planning, so no changes of heading are allowed. The safety distance considered between vehicles is 0.9 meter. This experiment has been tested with four Hummingbird quadrotors with 200gr. A minimum speed of 0.3 m/s has been set for every vehicle.

The experiment below validates the results obtained in the Section 5.2.1. In this case the hp-Adaptive PS method. Real trajectories of the experiment are shown in Figure 87. In the 3D view it can be seen that altitudes are almost constant during the flight (there are some millimeter of variation because of the altitude control of the quadrotor) and in the 2D view it can be seen that trajectories are straight paths. This effect is caused because velocity planning is the conflict resolution method used in the scenario.

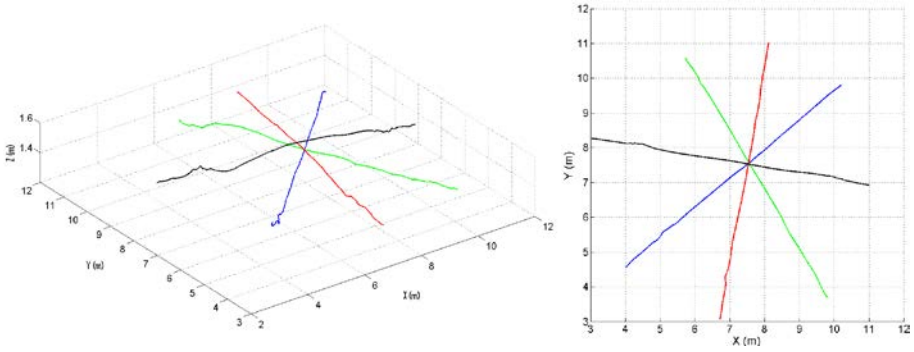


Figure 87: Experiment results of the scenario S2 with 4 UAVs. Trajectories.

In order to check the accuracy of the trajectories, the verification of the safety distances are presented in Figure 88. This figure shows the distance between every two UAVs in time. This is a real-time system and the sampling of the speed is every 0.10 seconds. Then the distances shown in the picture have the same resolution. Figure 88 shows that all distances are greater than 0.9 meter (the minimum distance allowed). Only the distance between UAV 1 and UAV 2 are very close to one meter around the instant 10 seconds (this corresponds approximately with the center of the star). The rest of distances are bigger than two meters, so the effectiveness of this method is proved.

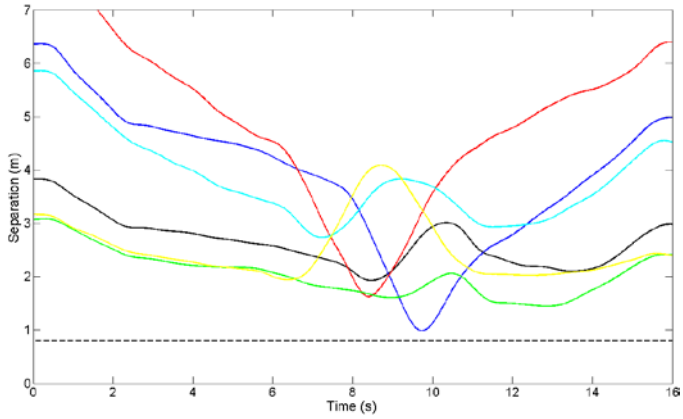


Figure 88: Experiment results of the scenario S2 with 4 UAVs. Safety distances.

6.3. Teams of multiple UAVs rotary-wing

Seven experiments related to the three case of study in Sections 5.3 are presented in this section. These experiments validate the use of collocation methods in real-time applications. First experiment focuses on the validation of the S-Adaptive algorithm. The second one focuses on the validation of the hp-Adaptive algorithm with a specific real-time configuration of its parameters. And finally, the third case focuses on the validation of the rolling horizon configuration of the LGL algorithm.

All the experiments have been performed in the same multi-UAVs testbed presented in Section 6.1.1 with four Hummingbird quadrotors with 200 gr of payload each.

6.3.1. Collision avoidance with S-Adaptive PS method

Four experiments have been performed in this section. The two variants of scenario S5 (see Figure 46) have been executed with two different cost functions each. Then scenario S5(a) is presented following while the scenario S5(b) and more experiments are presented in the Appendix. The goal in this section is to validate the trajectories obtained as well as the comparison of the trajectories when two different cost functions are used in the same scenario.

The real scenario has a dimension of 8 x 4 m^2 and the trajectories are 9 meters long. Four UGVs are considered. The conflict resolution method is based on changes of speed and heading, and the safety distance considered in this case is 1.2 meter. The minimum UAV speed is 0 and the maximum speed is 2.0 m/s. Two different optimal criteria are considered: J_d is the minimum distance traveled defined in Equation (5.12) and J_a obtains smooth trajectories and is defined in Equation (5.14).

Real trajectories of scenario S5(a) are represented in the following picture. Figure 89 shows the trajectories when J_d criterion is considered. It can be seen that the trajectories are almost straight lines because of the optimal criterion. In 3D view it can be seen that all UAVs fly in the same flight level (there are some millimeter of variation because of the altitude control of the quadrotor), so the conflict resolution is solved by velocity planning in this case.

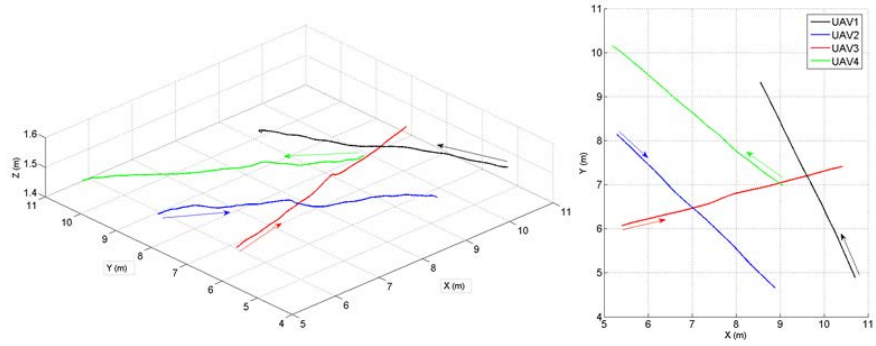


Figure 89: Experiment results of the scenario S5(a) with J_d optimal criterion. Trajectories.

In order to check the accuracy of the trajectories, the safety distances are presented in Figure 90. This picture shows the distance between every two UAVs in time. It can be seen that all distances are greater than 1.2 meter, so there is not any collision between vehicles. But it can be seen that distance between UAV 2 and UAV 3 (green line in Figure 90) is very close to the minimum allowed. This is because both vehicles have a small distance in front of them to avoid the collision without change the trajectory (see trajectory of UAV 2 and 3 in Figure 89). Something similar happens between UAV 1

and UAV 3, so in this scenario there is a strong connection between UAV 1, UAVs 2 and UAVs 3 and yet the solution is performed with success.

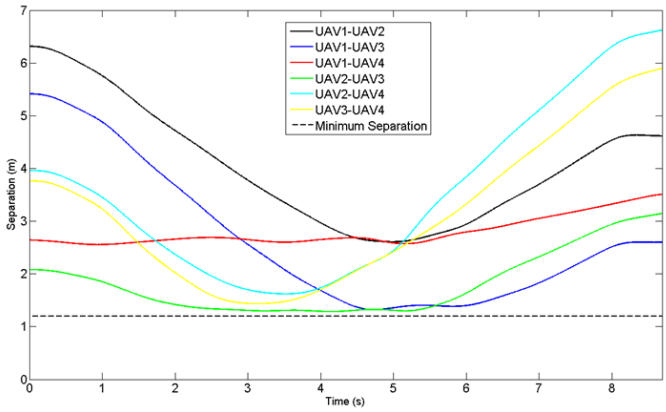


Figure 90: Experiment results of the scenario S5(a) with J_d optimal criterion. Safety distances.

In the second experiment, the same method is executed in the same scenario S5(a) but with the difference that the optimal criterion considered in this case is J_a (minimum changes in the UAV inputs controls) and the safety distance is 0.8 meter. Figure 91 shows the UAV real trajectories. In this case, changes of speed and heading for each UAV have taken place. In 2D view it can be seen that curvature of the trajectories are smooth due to the optimal criterion.

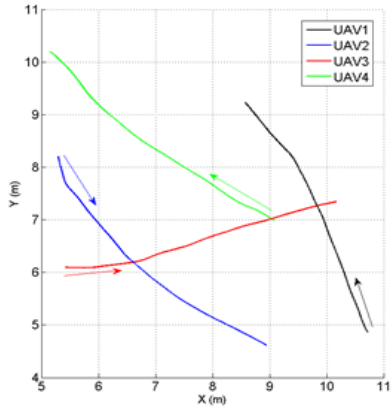


Figure 91: Experiment results of the scenario S5(a) with J_a optimal criterion. Trajectories.

Figure 92 shows three pictures with different moments of time of the experiment taken in the multi-UAVs testbed from CATEC. Above, the initial state of the experiment is presented. Then below, two moments of the navigation is presented as well.

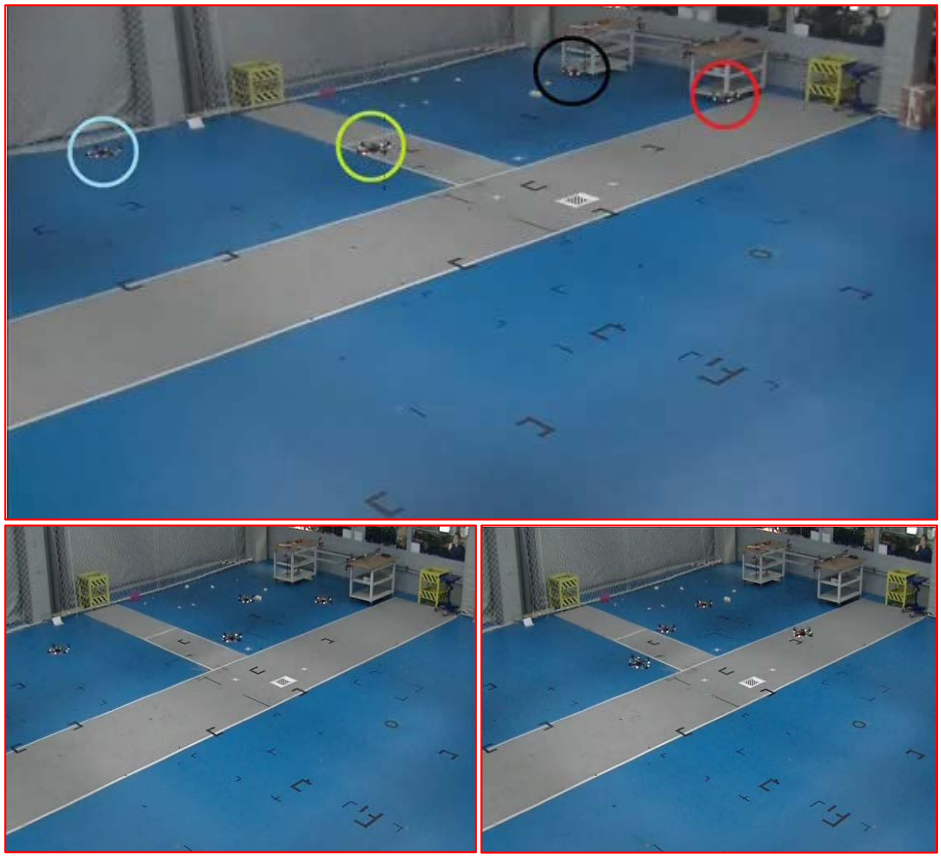


Figure 92: Snapshots of the multi-UAV testbed during the experiment of the scenario S5.

Figure 93 shows the distance between every two UAVs in time. As happens in the previous experiment, all the distances between the UAVs are greater than the limit, 0.8 meter in this case, so there is no collision.

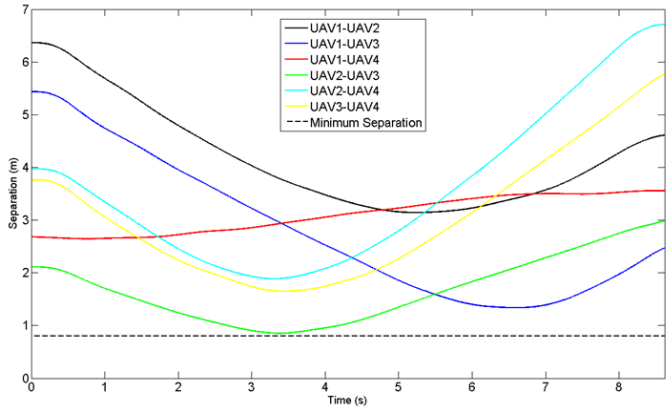


Figure 93: Experiment results of the scenario S5(a) with J_a optimal criterion. Safety distances.

If safety distances profiles in Figure 90 and Figure 93 are compared, it can be seen that the solution in the first case is more extreme (vehicles are closer each other during more time). This means that the optimal criterion selected affects to the results of the trajectory problem stronger.

6.3.2. Real-time configuration of hp-Adaptive PS method

Two experiments have been carried out considering the scenarios S6 y S7 of Figure 55. The goal in this section is to demonstrate the performance of the hp-Adaptive method with the configuration obtained in the study in Section 5.3.2. That configuration allows to execute the hp-Adaptive method in low computation time.

The real scenario has a dimension of 8 x 8 m^2 and the flight distances are 7 meters long. Four UAVs are considered. Conflict resolution is based only on changes of speed (velocity planning) and the safety distance considered in this case is 1.0 meter. The minimum UAV speed is 0 and the maximum speed is 2.0 m/s.

Real trajectories are represented in the following picture. Figure 94 shows the UAVs trajectories given by scenario S6. In the 2D view it can be seen than trajectories are straight paths, so the conflict resolution has been solved by velocity planning. It can be seen in the 3D view that a few millimeters of variation of altitude occur, but this is because of the altitude control of the quadrotor.

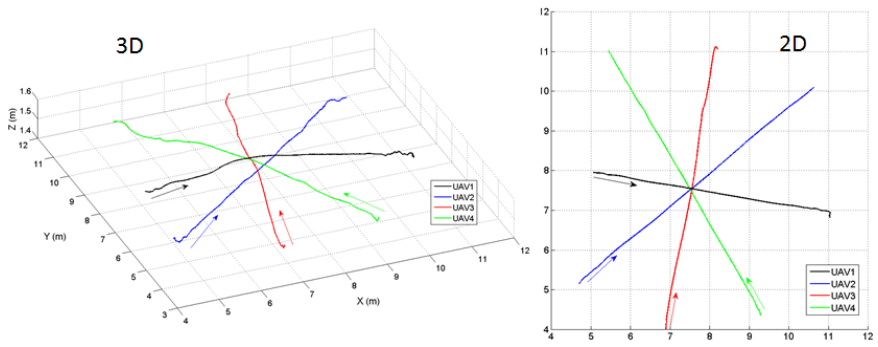


Figure 94: Experiment results of the scenario S6 with 4 UAVs. Trajectories.

In order to check the accuracy of the trajectories we can focus on the representation of the safety distances in Figure 95. This picture shows the distance between every two UAVs in time. It can be seen that all the distances are greater than 1 meter (the minimum distance required), so there is not any collision between UAVs. In this scenario the conflict is the same for all the UAVs so it can be seen that between instant 5 and 10, distances between UAV 1, AUV 4 and UAV 3 are very close to the limit (see lines red yellow and green). This interval of time is when the four UAVs are establishing an order of arrival to the center of the star. All the distances are increased after this point.

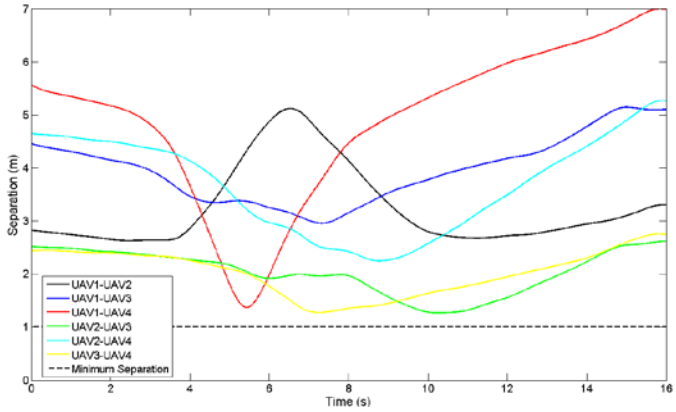


Figure 95: Experiment results of the scenario S6 with 4 UAVs. Safety distances.

In the second experiment four UAVs are considered in the scenario S7. Figure 96 shows the UAVs trajectories. Looking at the scenario two conflicts has been solved: first of them is between UAV 1 y UAV 4 and the second one is between UAV 2 y UAV 3.

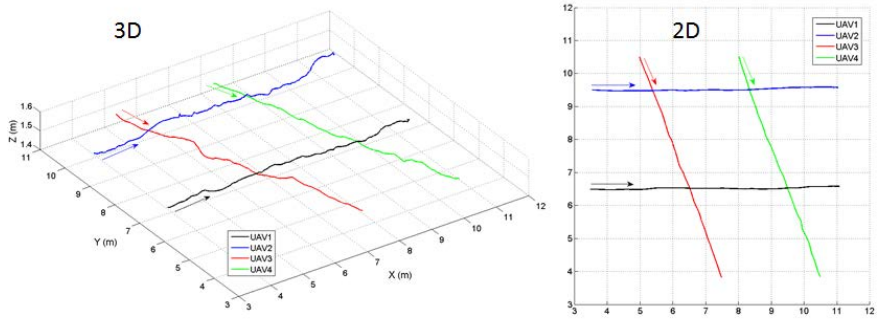


Figure 96: Experiment results of the scenario S7 with 4 UAVs. Trajectories.

In order to check the accuracy of trajectories, the safety distances are presented in Figure 97. It can be seen that all distances are greater than the limit (1 meter). However, it can be seen that the conflict resolution between UAV 2 and UAV 3 is very extreme in the second 4 (the distance reaches the limit). Then the safety distances between UAV 1 y UAV 3 and between UAV 1 y UAV 4 are also near to the limit for some seconds. This means that although the trajectories of the four UAVs are strongly coupled, the S-Adaptive PS method is able to find a solution of the problem.

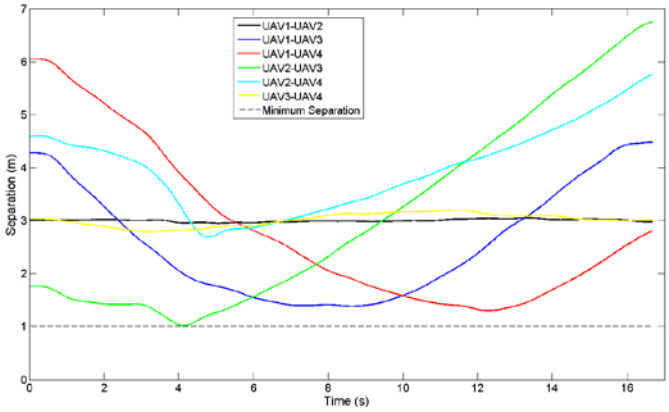


Figure 97: Experiment results of the scenario S7 with 4 UAVs. Safety distances.

6.3.3. Real-time configuration of LGL PS method

A validation experiment considering the scenarios S8 of Figure 61 is presented in the following. In this case, a rolling horizon technique with the LGL Pseudospectral algorithm is presented. Mainly two differences distinguish this method from the previous one: the first one is that this is a distributed algorithm while the others are centralized and the second one is that this is a local trajectory planning while the others are global.

Specifically the local characteristic of the trajectory planning method is proved here. In the Section 5.3.3 has been demonstrated the convergence of the vehicles to the global destination, as well the safety of the trajectories also has been demonstrated (in simulation). Now, this specific configuration of LGL PS method is validated.

The real scenario has a dimension of 8 x 8 m^2 and the flight distances are 5 meters long. Four UAVs are considered. The conflict resolution is based on changes of speed and heading and the safety distance considered in this case is 1.0 meter. The minimum UAV speed is 0 and the maximum speed is 2.0 m/s.

Real trajectories are represented in the following picture Figure 94. It can be seen that trajectories converge to their goals and also that the trajectories are almost straight lines, obtaining a result close to the minimum distance.

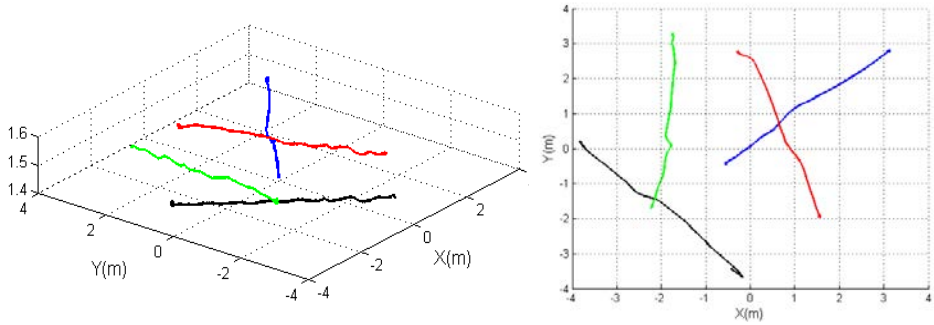


Figure 98: Experiment results of the scenario S8 with 4 UAVs. Trajectories.

Figure 99 shows the safety distances of the verification process. It can be seen that all distances are greater than the limit (1 meter). However, it can be seen that the conflict resolution between UAV 1 and UAV 2 and between UAV 3 and UAV 4 are very extreme (they reach the limits of 1 meter). Anyway it can be concluded that this specific rolling horizon configuration of basic PS collocation method can be used as real-time distributed trajectory planning method.

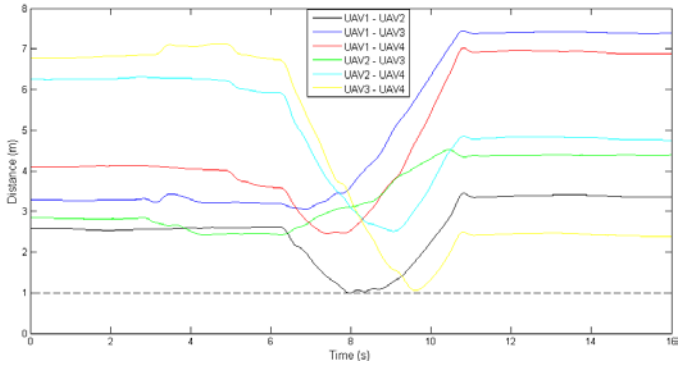


Figure 99: Experiment results of the scenario S8 with 4 UAVs. Safety distances.

6.4. Teams of multiple UGVs

Three new experiments related with the cases in Section 5.4 are presented in this section. These experiments attempt to validate the S-Adaptive algorithm with autonomous ground vehicles. Another novelty in collocation methods is the use of fixed objects in the scenario which affects to the trajectory planning. In these cases, the number of constraints in the problem increases proportionally with the number of

vehicles and objects. Thus the solution of the problem could be unreachable (or unreachable in an acceptable computation time). Secondly, when the number of obstacles increases in the same scenario, the free area is reduced and it is more difficult to find a trajectory solution.

The first experiment focuses on the validation of S-Adaptive algorithm in scenarios with multiple UGVs. The second and third experiments focus on the validation of the same scenario including different number of obstacles. The second experiment only considers one obstacle in the center of scenario while the third one considers three obstacles.

All the experiments have been carried out in the same multi-UGV testbed introduced in Section 6.1.1 with four Pioneer 3DX. The optimal criterion considered for the experiments are J_d, minimum distance of navigation, Equation (5.12). The navigation method used by the UGVs consists in the lineal interpolation of the waypoints. For the sake of the vehicles' security, these experiments have been executed 3 times slower than in simulation, but increasing the accuracy of the navigation system of the vehicles, the experiments could be executed without speed restrictions.

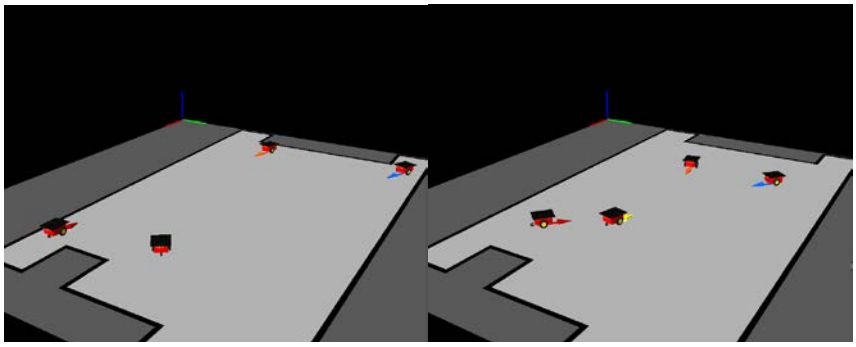
6.4.1. Collision avoidance without fixed obstacles

This experiment is based on the scenario S9 of Figure 70. The goal in this section is to demonstrate the performance of the S-Adaptive method with UGVs when criterion J_d is considered.

The real scenario has a dimension of 10 x 5 m² and the trajectories are 8 meter long. Four UGVs are considered. The conflict resolution is based on changes of speed and heading and the safety distance considered between vehicles is 1 meter. Then UGVs limits of speed are 0 and 0.5 m/s, as minimum and maximum speed respectively.

Results of simulation using ROS

In order to test this scenario with real vehicles (Pioneer robots), this scenario has been implemented in the Robot Operative System (ROS) and the results have been simulated by Gazebo ROS simulator. Figure 100 shows a sequence of six different instants of time of all vehicles moving in the scenario S9.



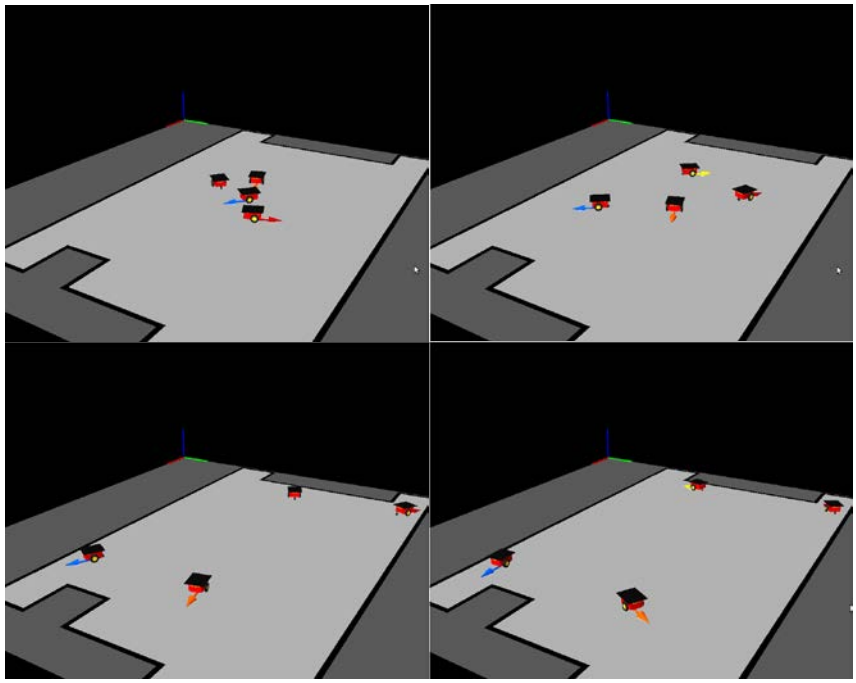


Figure 100: Results of simulation of the scenario S9 with 4 UGVs. Illustration of trajectories in eight moments.

The results of these simulations are quite realistic because the simulator considers the dynamic of vehicles, and also the communications between nodes and master are simulated. So the obtained trajectories are very close to the real experiment.

Figure 101 shows a picture of the real vehicles in the scenario. Then Figure 102 shows the real trajectories of the four UGVs.

Figure 101: Snapshot of the multi-UGV testbed during the experiment of the scenario S9.

It can be seen in Figure 102 that all the UGVs move to their destinations through the center because it is the shortest trajectory. Then, when all of them are in the center, they

interchange their positions maintaining the safety distance between them.

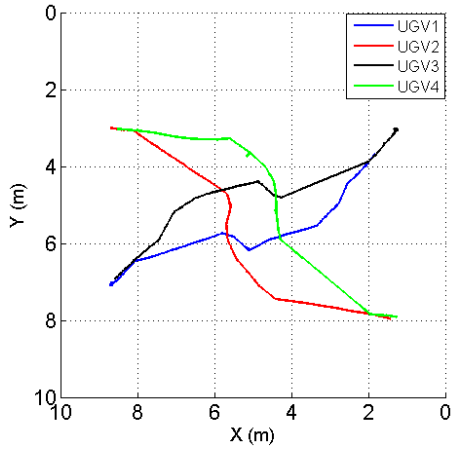


Figure 102: Experiment results of the scenario S9 with 4 UGVs. Trajectories.

Accuracy of trajectories is checked by Figure 103, which shows the separation between every two UGVs. It can be seen that each UGV maintains the minimum separation distance (in this case is 1.0 meter). Moreover, it can be seen that that the distances are very close to the limits during the interval defined from instant 30 to instant 60. This period of time correspond to the time in which all the UGV are in the center of scenario turning around, interchanging their positions.

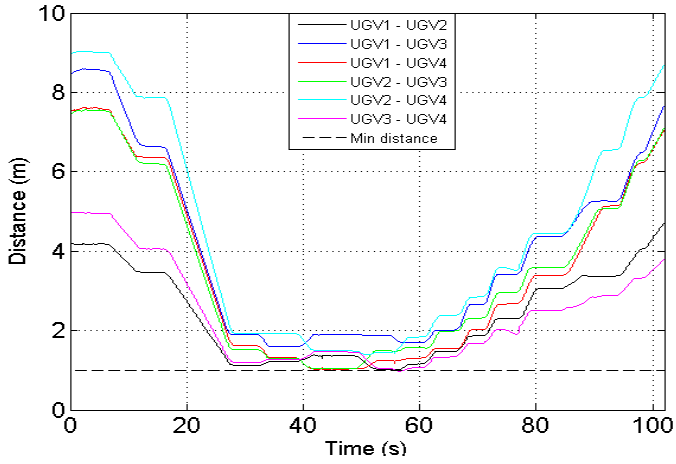


Figure 103: Experiment results of the scenario S9 with 4 UGVs. Safety distances.

Note that the curves of the distances are represented by jumps. This is due to the representation of interpolation in time of the waypoint of the navigation method of the vehicles. This method is based on two phases: in the first one, the UGV turns around its base looking for orientation to the next waypoint. In the second phase, when the UGV is orientated to the following waypoint, it starts to move to the waypoint with cruise speed and stops when it arrives.

6.4.2. Collision avoidance with fixed obstacles

Two experiments based on scenarios S10 and S11 of Figure 72 are presented in this section. The goal in this section is to demonstrate the performance of the S-Adaptive method with UGV when fixed obstacles are used.

The description of this scenario is similar to the previous experiment. Four UGVs in a 10 x 5 m^2 scenario, but now some fixed obstacles are introduced. Objects are defined by small boxes of 0.5 x 0.5 x 0.5 meter. Then, the safety distance considered between UGVs is 1 meter while in the case of the UGVs and the obstacles it is 0.5 meters. The optimal criterion is J_d, minimum distance of navigation.

Real trajectories are presented in following picture. Figure 104 shows the trajectories when scenario S10 is considered. One object is used in this experiment, and it is painted in its real position by a circle. The trajectories obtained are similar to the solution in the previous experiment, but in this case, the object is in the center of the scenario and each UGV rounds the object in order to interchange its positions. Then, every UGVs follow their trajectories to their destinations.

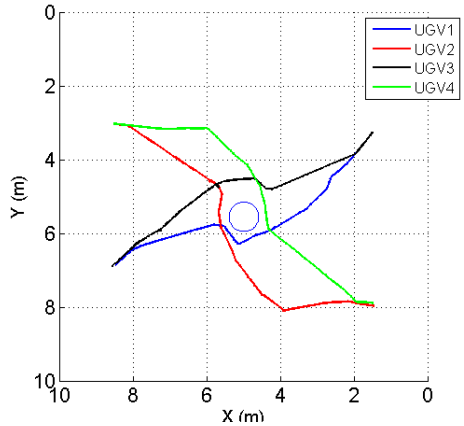


Figure 104: Experiment results of the scenario S10 with 4 UGVs. Trajectories.

Accuracy of trajectories is checked by Figure 105 which shows the separation between every two UGVs. It can be seen that each UGV maintains the minimum separation distance (1 meter) all the time. As occurred in the previous experiment, there is an interval of time in which the distances are very close to the limit, but in this case, this interval is also 20 seconds longer (from instant 20 to instant 70). This is because the object is in the middle of scenario.

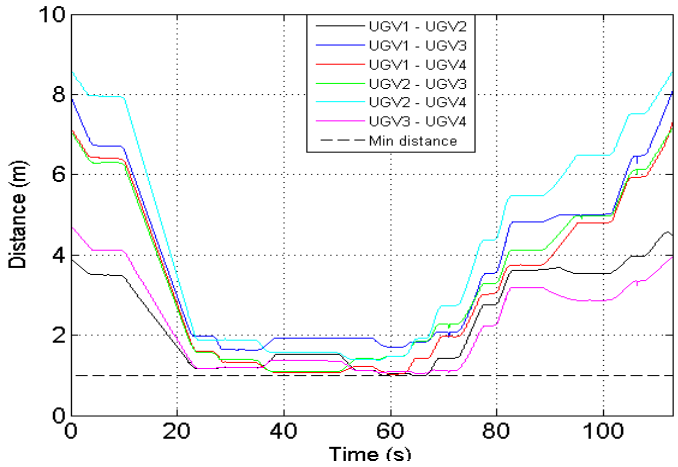


Figure 105: Experiment results of the scenario S10 with 4 UGVs. Safety distances between UGVs.

The safety distance between the UGVs and the obstacle has to be checked as well. In this case the safety distance is measured by the distance between the centers of vehicles and the fixed obstacle. Then this distance has to be greater than 0.5 meters (minimum distance considered in this scenario).

Figure 106 shows the distances between all UGVs and the object. It can be seen in the figure how all the UVGs move to the object in the beginning, so they are very close to the object (because the UAVs are moving around), and finally they move away from the object. In conclusion the safety distance is met at all time.

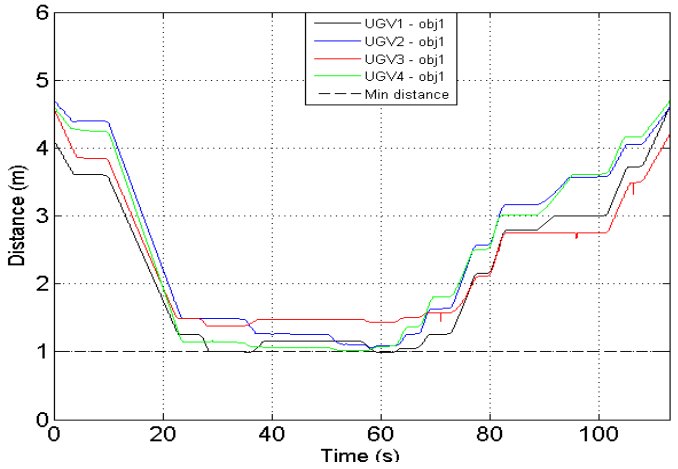


Figure 106: Experiment results of the scenario S10 with 4 UGVs. Safety distances between UGVs - obstacles.

Multiple fixed objects

In the second experiment three objects have been used in scenario S11. This scenario is more difficult than S10 because the free space is less. Then, the optimal control problem is more difficult to be solved. The optimal criterion is J_d, minimum distance of navigation.

Real trajectories are presented in the following Figure 107. The three objects are painted in their real positions. It can be seen that all UAVs move to the center of the scenario avoiding all obstacles. In the center, each UGV rounds in order to interchange its positions and finally every UGV moves to its destination.

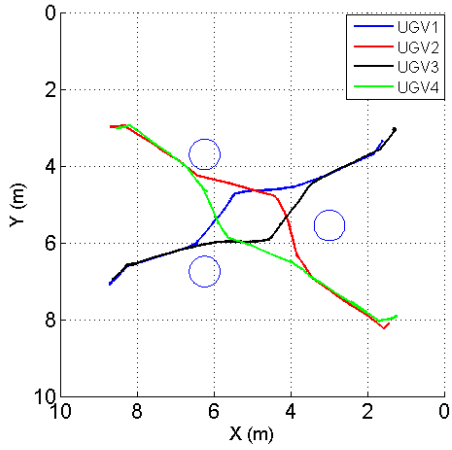


Figure 107: Experiment results of the scenario S11 with 4 UGVs. Trajectories.

Figure 108 shows two pictures with different instants of time of the experiment in the multi-UGVs testbed from LARICS. Above, the initial state of the experiment is presented. Then below, the exchange positions of four UGVs are presented.

Figure 108: Snapshots of the multi-UGV testbed during the experiment of the scenario S11.

Accuracy of trajectories is checked by Figure 109 which shows the separation between UGVs. It can be seen that each UGV maintains the minimum separation distance. As it happened in previous experiments, the distances between the UGVs are very close to the limit when all the UGVs are in the center interchanging their positions. In this case, this phase occurs later in time (from instant 50 to instant 90) because the UGVs have to avoid first the obstacles.

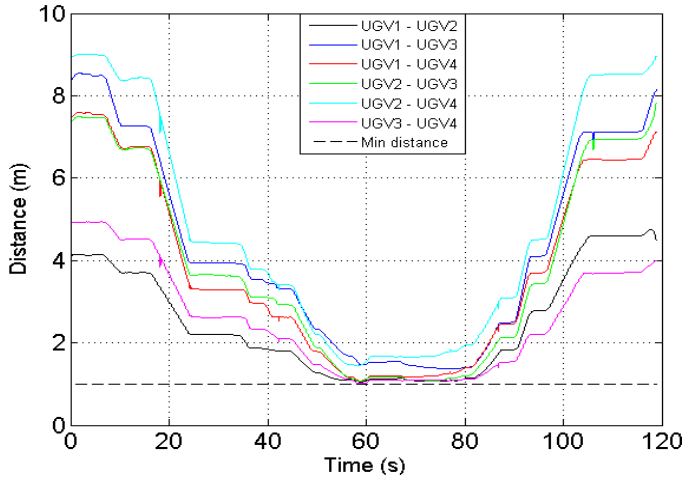


Figure 109: Experiment results of the scenario S11 with 4 UGVs. Safety distances between UGVs.

Figure 110 shows the distances between every UGV and all objects considered in this experiment. Two different phases can be seen in the picture: the first one between instants 20 and 40, and the second one between instants 90 and 100. The first occurs when the UGVs move from the initial positions to the center of the scenario (avoiding the obstacles). The second one occurs then UGVs move from the center to their destinations (avoiding the obstacles again).

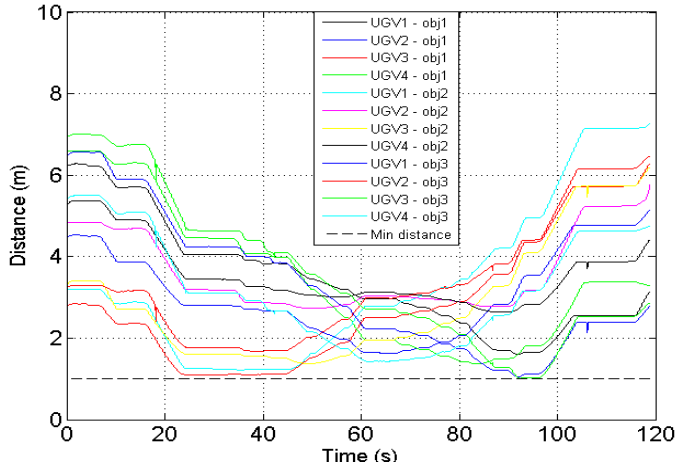


Figure 110: Experiment results of the scenario S11 with 4 UGVs. Safety distances between UGVs - obstacles.

6.5. Conclusions

The experimental validation of most of the scenarios presented in this thesis has been presented in this chapter. A total of eleven experiments have been performed considering aerial and ground vehicles. The executions of these experiments have been performed between the CATEC indoor aerial testbed and the LARICS indoor ground testbed.

Three blocks of experiments are distinguished. In the first block the trajectories of a fixed-wing is presented. In this case, the flight of the fixed-wing has been simulated by a quadrotor due to the difficulty of the scenario's setting. In the second block seven experiments with rotary-wing UAVs are presented. These experiments validate the resulted trajectories considering different optimal criteria and scenarios. Furthermore, the validation of real-time setting of hp-Adaptive PS method and the rolling horizon configuration of LGL PS method have been carried out. Finally in the third block, three experiments with UGVs are presented. Two of them consider fixed obstacles in the scenario for the first time in literature.

The following chapter will present the conclusions of the thesis with a small summary. Finally some future work will be also presented.

7. CONCLUSIONS AND FUTURE WORK

When you want something, all the universe conspires in helping you to achieve it.

Paulo Coelho

This chapter summarizes the main contributions of the Thesis and highlights its results. Advantages and drawbacks of the proposed method are discussed. Furthermore, in order to overcome these drawbacks, some guidelines are given for future extensions.

7.1. Framework

Control, coordination and autonomous cooperation of unmanned vehicles are promising areas of research that has gained a lot of attention. Their use is now no longer confined to military applications and an increasing number of civil applications are being developed worldwide.

Unmanned Aerial Vehicles (UAVs) offer advantages for many applications when comparing with their manned counterparts. They preserve human pilots of flying in dangerous conditions that can be encountered in scenarios involving operation in bad weather conditions, or near to buildings, trees, civil infrastructures and other obstacles. In case of Unmanned Ground Vehicles (UGVs), nowadays there are many applications in the industry and other sectors. Furthermore, the number of applications of mobile robot assistants in public places is increasing as well. In most cases it is essential for theses autonomous vehicles to move in a dynamic environment with other vehicles and obstacles. The development of a trajectory generation system for both kinds of vehicles is the main objective of this Thesis.

Most of the works has been developed in the framework of the European Project EC-SAFEMOBIL "Estimation and Control for Safe wireless high MOBility". One of the main objectives of EC-SAFEMOBIL is the safe cooperation, coordination and traffic control of multiple entities in applications such as automation of industrial warehousing or surveillance by means of aerial and ground vehicles.

7.2. Conclusions

A new point of view in multi-vehicle trajectory planning methods is presented in this Thesis. This considers solving the problem of trajectory planning for multiple vehicles as an optimal control problem. Thus the kinematics and dynamics of vehicles are considered, and more realistic trajectories are obtained.

The trajectory planning problem can be solved following an open loop terminal control problem. This strategy allows all the constraints acting on the dynamical system, including the dynamic constraints, to be taken into account in such a way that the resulting trajectory is admissible. However, this problem has an infinite number of solutions and an optimal criterion is used in order to eliminate this redundancy.

One of the main approaches to numerically solve continuous time optimal control problems is collocation methods. Collocation methods are based on modeling the problem obtaining a set of differential equations that describes the dynamics and several state and input constraints corresponding to physical constraints, which has to be met by the solution trajectory. Then, the differential equations are discretized at a set of collocation points, and the optimal control problem is transformed in a NLP problem, which is later solved using standard NLP solvers like SNOPT.

7.2.1. Collocation methods

Collocation methods have demonstrated through this Thesis being a good solution for solving trajectory planning problems. In chapter 3, a thorough survey of the different variants of collocation methods has been presented. In this chapter, the main collocation algorithms have been classified into three groups and a full description of these methods has been presented. Among them, PS collocation is one of the most widely used techniques. Chapter 5 presents a complete study of performances like accuracy, scalability and computation time. This chapter also presents several application of PS collocation to obtain optimal trajectories for multiple vehicles.

One important advantage of the use of collocation methods for trajectory generation is that obtained trajectories are very realistic, in the sense that when the vehicle will follow these trajectories they will be able to do it without large deviations, because vehicle's kinematic and dynamic models and actuator constraints are considered in the method. However, some drawbacks have been detected with PS collocation methods for multiple vehicles, which are mainly caused by the discretization process of the method. The trajectory constraints like safety distances with other vehicles and obstacles are only checked in the colocation points. This implies that a minimum number of collocation points have to be used in order to ensure that, after the interpolation of points, the trajectories are safe. Moreover, the computation time have a strong dependence of the number of collocation points. Thus, the quality of the solution and the computation time are conflicting.

In conclusion, classic PS collocation methods need a very high number of collocation

points in order to converge to a good quality solution. In some cases, when the scenario is complex (i.e. there are many vehicles flying in a small area), PS methods may not converge to a safe solution. The hp-Adaptive PS is an improvement of PS collocation which presents better results. This method uses a high number of collocation points, and depending of the complexity of the problem, the computation time can be so high that it cannot be used in a real application. However, the scalability presented by this method is very good in comparison with other optimal trajectory methods studied in this Thesis.

7.2.2. S-Adaptive Pseudospectral collocation method

The main contribution of the Thesis to the field of optimal trajectory planning for multiple vehicles is the S-Adaptive PS collocation algorithm described in chapter 4, which presents very good computation time sacrificing only slightly the quality of the trajectory solution. That algorithm presents several other advantages in comparison to classic PS collocation methods.

S-Adaptive presents a new method of segmentation of the problem taking into account the part of the trajectory in which safety distance constraints are not met. Only conflicting segments are recomputed in an iterative process. Thus, convergence of the problem is achieved by using less collocation points, as well as, less computation time is needed to solve the problem.

The S-Adaptive collocation method has been applied to obtain optimal trajectories for multiple vehicles, multiple ground vehicles and coordinated aerial and ground vehicles in chapter 5. Different scenarios and optimal criteria have been tested, obtaining good scalability and solution quality. The computation time, which is of the order of seconds, can be sufficient for many applications since it is able to optimize the trajectories.

Finally, the theoretical results presented in this Thesis have been validated with extensive experiments that have been performed in two aerial and ground vehicle testbeds, which are presented in chapter 6. The reliability of the trajectories has been demonstrated with real aerial and ground vehicles.

7.3. Future Developments

The research described in this Ph.D. Thesis presents a set of methods and tools to solve the optimal trajectory generation problem for multiple vehicles. Nevertheless, these solutions should be considered only as a first step in a long-term research effort. The answers provided generate new questions and open new lines of research that extend the scope of the proposed solutions and extrapolate the methods to different problems and technologies. Current and future research lines derived from this P.D. Thesis can be summarized as follows.

One of the future research lines is the extension of the application of the techniques

proposed in this Thesis to the Air Traffic Management domain, of which a converging traffic case study has been presented in section 5.2.2. This will include the adaptation of the aircraft Total Energy Model (TEM) which is standing behind the Base of Aircraft Data (BADA). BADA is an aircraft performance model with a corresponding database developed by the Eurocontrol Experimental Centre. The BADA release v3.7 contains performance and operating procedure data for 294 different aircraft types covering the huge part of all aircraft civilian traffic. The TEM model combines both potential and kinetic energy models supplied also with fuel consumption models. The use of the BADA models, which can be integrated perfectly in the pseudospectral collocation methods of this Thesis, will generate more realistic trajectories, and the proposed techniques can be integrated to the civilian air traffic management.

An interesting research line is the anytime formulation of the pseudospectral collocation methods in this Thesis. An algorithm is said to be anytime if it finds a feasible, non-optimal solution quickly and improves this solution towards an optimal one if allowed more computation time, which makes it more amenable to real-time implementations on resource-constrained robotic systems. For collocation methods this implies to force the NLP solver to provide a quick feasible solution and incremental updates when the solution converges to the optimum.

Another line deals with the implementation of the designed algorithms onboard the UAVs. For that purpose, an algorithm in C++ to be executed in onboard embedded computers will increase the efficiency.

APPENDIX

Two experiments are presented in this section. The first one presents the results of the scenario S5 case (b) of the Section 6.3.1. In the second experiment a long trajectory is flown following an iterative method that splits the initial trajectory into straight segments and generates a path for each.

Part I

The following experiment solves the collision avoidance problem of Scenario S5(b) in Figure 111. Such as in the Section 6.3.1, the goal in this section is to test the performances of the S-Adaptive PS method when two different optimal criteria are used: the optimal criterion J_d defines the minimum distance traveled, Equation (5.12) and the optimal criterion J_a defines the minimum changes of input controls, Equation (5.14).

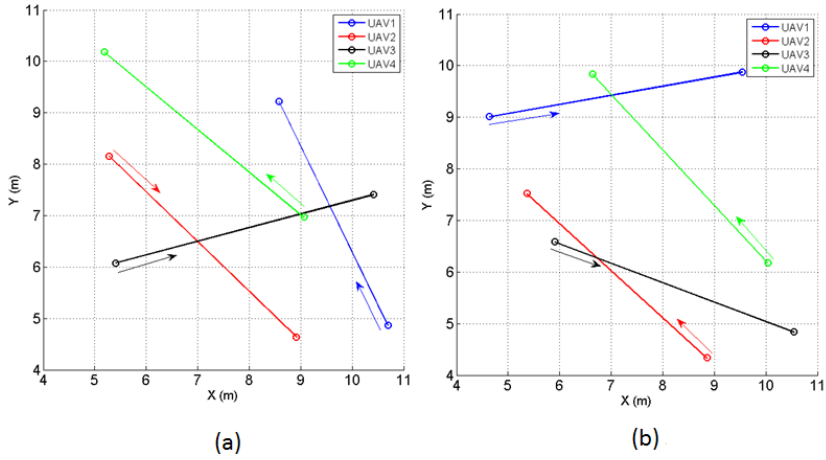


Figure 111: Scenario S5, comparison of two optimal criteria (Jd and Ja). Collision avoidance in the same flight level.

Figure 112 shows the real trajectories when J_d criterion is considered. In this figure, effects of the segmentation process of the S-Adaptive PS method are shown. It can be seen for example in the trajectory of the UAV 1, that only a piece of the trajectory has been changed in order to avoid the collision. The remainder of the trajectory follows a straight line. This effect is due to the optimal criterion. Something similar happens with the UAV 2 and UAV 3.

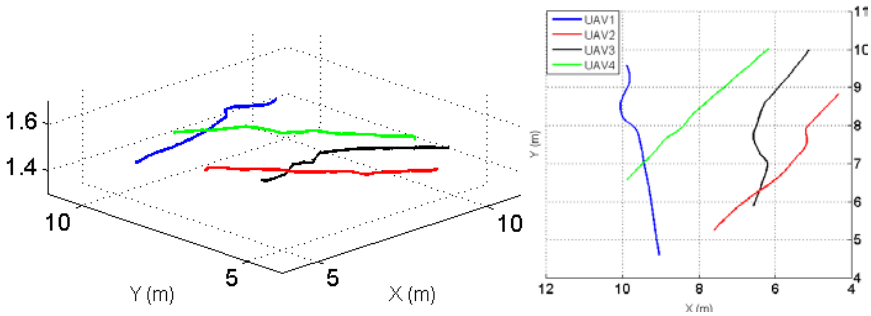


Figure 112: Experiment results of the scenario S5(b) with J_d optimal criterion. Trajectories.

In order to check the accuracy of the trajectories, the safety distances are presented in Figure 113. This picture shows the distance between every two UAVs in time. It can be seen that all distances are greater than 1.0 meter (the safety distance considered in this experiment). Then it can be concluded that the trajectories are safe.

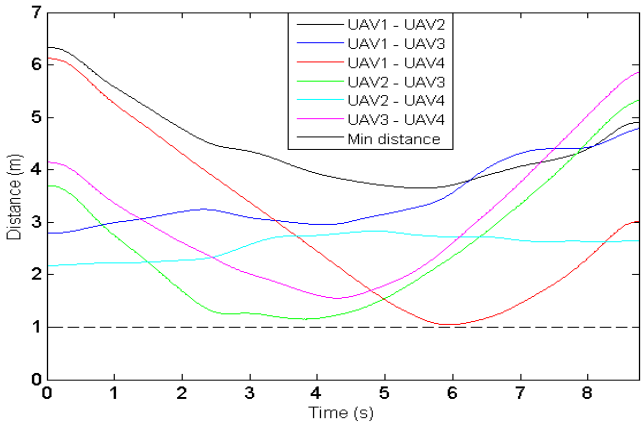


Figure 113: Experiment results of the scenario S5(b) with J_d optimal criterion. Safety distances.

The second experiment is executed in the same scenario S5(b) with the J_a optimal criterion. Figure 114 shows the UAV real trajectories. Four smooth trajectories are obtained in this case.

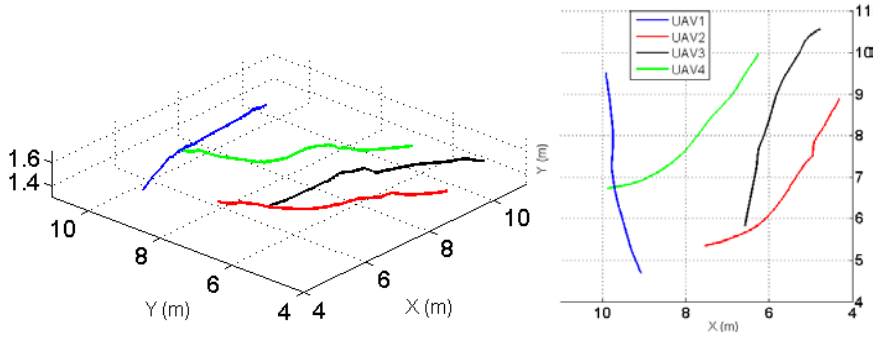


Figure 114: Experiment results of the scenario S5(b) with J_a optimal criterion. Trajectories.

Figure 115 shows the distance between every two UAVs in time. As happens in the previous experiment, all the distances are greater than 1.0 meters. If Figure 113 and Figure 115 are compared, it can be seen that in the first one the distances are closer to the limit than in Figure 115. It can be concluded that the curvature of the trajectories and the safety of the vehicles is strongly conditioned by the optimal criterion selected.

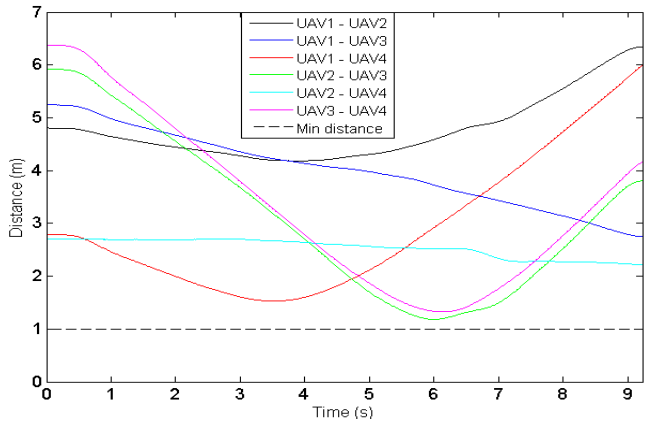


Figure 115: Experiment results of the scenario S5(b) with J_a optimal criterion. Safety distances.

Part II

An evolution of the previous scenario S5 is presented. The case of long trajectories can be solved with collocation method by splitting the initial trajectory into segments and solving the problem piece by piece.

Figure 116 shows a new scenario with four trajectories of 10 meters each. These trajectories have been split into 5 meters long trajectories by the addition of a new waypoint in the middle of it. The entire trajectory planning problem has been solved following an iterative philosophy.

The goal in this experiment is to demonstrate the performance of the S-Adaptive method in the resolution of long trajectory problem. The J_a optimal criterion has been used.

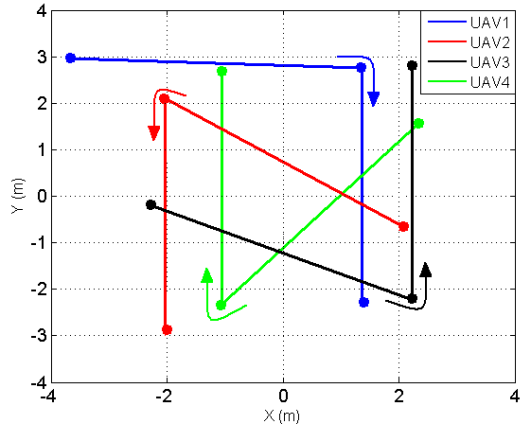


Figure 116: Scenario S12, long trajectories problem solved by multi-segments.

Figure 117 shows the real trajectories flown by the UAVs. It can be seen that the trajectories are continuous (There is no jumps during the transition of the segments). In the case of the UAV 1 and UAV3 it can be seen during a moment that they have to change their trajectories in order to meet the safety distance between them.

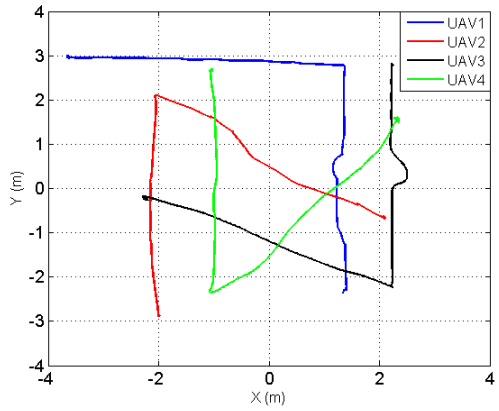


Figure 117: Experiment results of the scenario S12 with 4 UAVs. Trajectories.

Figure 118 shows the validation of the trajectories. All distances are greater than the minimum allowed (1 meter). In conclusion, S-Adaptive PS methods always meet the safety distances between vehicles, even when the trajectory is split into multiples segments.

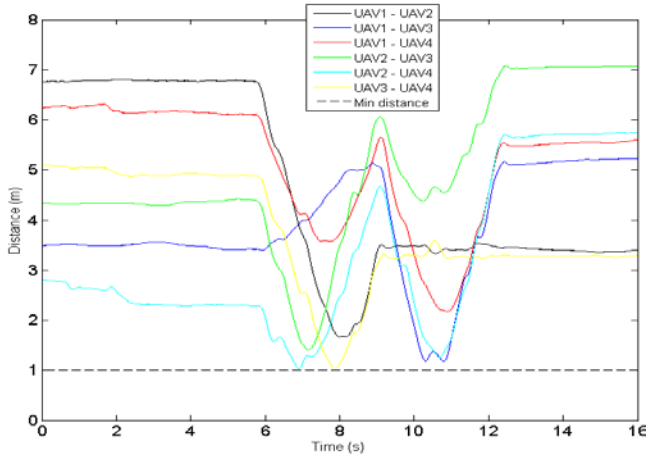


Figure 118: Experiment results of the scenario S12 with 4 UAVs. Safety distances.

REFERENCES

[1] Kevin P. Bollino and L. Ryan Lewis, "Collision-Free Multi-UAV Optimal Path Planning and Cooperative Control for Tactical Applications," in *(AIAA) Guidance Navigation and Control Conference and Exhibit*, Honolulu, Hawaii, 2008.

[2] Krithika Mohan, Michael A. Patterson, and Anil V. Rao, "Optimal Trajectory and Control Generation for Landing of Multiple Aircraft in the Presence of Obstacles," in *AIAA Guidance, Navigation, and Control Conference*, Minneapolis, Minnesota, 2012.

[3] Philip Gill, Walter Murray, Michael Saunders, and Margaret Wright, "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM Journal on Optimization*, vol. 12, no. 979-1006, 2002.

[4] S. Vera, J. A. Cobano, G. Heredia, and A. Ollero, "Collision Avoidance for Multiple UAVs using Rolling-horizon Policy," *Journal of Intelligent and Robotic Systems*, 2015.

[5] S. Vera, J. A. Cobano, G. Heredia, and A. Ollero, "Safe Trajectory Planning for Multiple Aerial Vehicles with Segmentation-Adaptive Pseudospectral Collocation," in *IEEE International Conference on Robotics and Automation*, 2015.

[6] S. Vera, J. A. Cobano, G. Heredia, and A. Ollero, "An Hp-Adaptative Pseudospectral Method for Conflict Resolution in Converging Air Traffic," *Springer Lecture Notes in Electrical Engineering, Portuguese Conference on Automatic Control CONTROLO'2014*, vol. 321, pp. 333-343, 2014.

[7] S. Vera, J. A. Cobano, G. Heredia, and A. Ollero, "An hp-adaptative pseudospectral method for collision avoidance with multiple UAVs in real-time applications," in *IEEE International Conference on Robotics and Automation*, Hong Kong, 2014, pp. 4717-4722.

[8] S Vera, J. A. Cobano, D. Alejo, G. Heredia, and A. Ollero, "Optimal conflict resolution for multiple UAVs using pseudospectral collocation," *Journal of Aerospace Information Systems*.

[9] S. Vera, "Rolling-horizon trajectory planning for multiple UAVs based on pseudospectral collocation," in *International Conference on Unmanned Aircraft Systems*, 2014, pp. 516-523.

[10] J.F. Canny, *The Complexity of Robot Motion Planning*. Cambridge Massachusetts: The MIT press, 1988.

[11] Steven M. LaValle, *Planning Algorithms*.: Cambridge University Press, 2006.

[12] J. H. Reif, "Complexity of the mover's problem and generalizations," in *IEEE Symposium on Foundations of Computer Science*, San Juan, Puerto Rico, 1979, pp. 421–427.

[13] Edsger. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, pp. 269-271, 1959.

[14] P. E.; Nilsson, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100-107, 1968.

[15] L.E. Kavraki, P. Svestka, J.-C. Latombe, and M.H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *Robotics and Automation, IEEE Transactions on*, vol. 12, no. 4, pp. 566-580, Aug 1996.

[16] Steven M. Lavalle, "Rapidly-Exploring Random Trees: A New Tool for Path Planning," Tech. rep. 1998.

[17] Steven M. Lavalle, James J. Kuffner, and Jr., "Rapidly-Exploring Random Trees: Progress and Prospects," , 2000, pp. 293-308.

[18] L. Jaillet, J. Cortes, and T. Simeon, "Transition-based RRT for path planning in continuous cost spaces," , Sept 2008, pp. 2145-2150.

[19] S Karaman and E Frazzoli, "Sampling-based algorithms for optimal motion planning," *International Journal of Robotics Research*, vol. 30, pp. 1-76, 2011.

[20] Charles Darwin, *On the Origin of Species by Means of Natural Selection*. London: Murray, 1859, or the Preservation of Favored Races in the Struggle for Life.

[21] David E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, 1st ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1989.

[22] Simon Kent, "Evolutionary Approaches to Robot Path Planning," Brunel University, Doctor of Philosophy 1999.

[23] O.K. Sahingoz, "Flyable path planning for a multi-UAV system with Genetic Algorithms and Bezier curves," in *International Conference on Unmanned Aircraft*

Systems, May 2013, pp. 41-48.

[24] Kalyanmoy Deb, Dilip Kumar Pratihar, and Amitabha Ghosh, "Learning to Avoid Moving Obstacles Optimally for Mobile Robots Using a Genetic-Fuzzy Approach," , 1998.

[25] N. Durand and J. Alliot, "Ant Colony Optimization for Air Traffic Conflict Resolution," in *USA/Europe air traffic management research and development seminar*, Napa, (CA, USA), 2009.

[26] P. Masci and A. Tedeschi, "Modelling and Evaluation of a Game-Theory approach for Airborne Conflict Resolution in Omnet++," in *Second International Conference on Dependability*, June 2009.

[27] Panfeng Huang, Gang Liu, Jianping Yuan, and Yangsheng Xu, "Multi-Objective Optimal Trajectory Planning of Space Robot Using Particle Swarm Optimization," in *Advances in Neural Networks - ISNN 2008*, Fuchun Sun et al., Eds.: Springer Berlin Heidelberg, 2008, vol. 5264, pp. 171-179. [Online]. http://dx.doi.org/10.1007/978-3-540-87734-9_20

[28] A.Z. Nasrollahy and H. Javadi, "Using Particle Swarm Optimization for Robot Path Planning in Dynamic Environments with Moving Obstacles and Target," in *Third UKSim European Symposium on Computer Modeling and Simulation*, Nov 2009, pp. 60-65.

[29] R. Bellman, *Dynamic Programming*. Princeton, NJ, USA: Princeton University Press, 1957.

[30] L. Pontryagin, V. Boltyanskii, R. Gamkrelidze, and E. Mishchenko, *The Mathematical Theory of Optimal Processes*. New York/London: Interscience Publishers, 1962.

[31] A. E. Bryson and Y. Ho, *Applied Optimal Control*. Abingdon, UK, 1975.

[32] W. Karush, "Minima of Functions of Several Variables with Inequalities as Side Constraints," in *PhD Thesis, Department of Mathematics, University of Chicago,* Chicago, 1939.

[33] H. Kuhn and A. Tucker, "Nonlinear Programming," in *2nd Berkeley Symposium on Mathematics, Statistics and Probability*, California, 1951, pp. 481–492.

[34] John T. Betts, "Survey of numerical methods for trajectory optimization," *Journal of Guidnance, Control,and Dynamics*, vol. 21, no. 2, pp. 193–207, August 1998.

[35] O. von Stryk and R. Bulirsch, "Direct and indirect methods for trajectory optimization," *Annals of Operations Research*, vol. 37, no. 1, pp. 357-373, 1992.

[36] E. D. Dickmanns and K. H. Well, "Approximate solution of optimal control problems using third order hermite polynomial functions," in *IFIP Technical Conference*, London, 1974, pp. 158-166.

[37] C. R. Hargraves and S. W. Paris, "Direct trajectory optimization using nonlinear programming and collocation," *Journal of Guidance, Control, and Dynamics*, vol. 10, no. 4, pp. 338-342, 1987.

[38] P. J. Enright and B. A. Conway, "Optimal finite-thrust space-craft trajectories using collocation and nonlinear programming," *Journal of Guidance, Control, and Dynamics*, vol. 14, no. 5, pp. 981-985, 1991.

[39] W. Roh and Y. Kim, "Trajectory optimization for a multi-stage launch vehicle using time finite element and direct collocation methods," *Engineering Optimization*, vol. 34, no. 1, pp. 15-32, 2002.

[40] A. L. Herman and B. A. Conway, "Direct optimization using collocation based on high-order gauss-lobatto quadrature rules," *Journal of Guidance, Control, and Dynamics*, vol. 19, no. 3, pp. 592-599, 1996.

[41] V. Coverstone-Carroll and J. E. Prussing, "Optimal cooperative power-limited rendezvous with propellant constraints," *Journal of the Astronautical Sciences*, vol. 43, no. 3, pp. 289-305, 1995.

[42] S. Tang and B. A. Conway, "Optimization of low-thrust interplanetary trajectories using collocation and nonlinear programming," *Journal of Guidance, Control, and Dynamics*, vol. 18, no. 3, pp. 599-604, 1995.

[43] P. J. Enright and B. A. Conway, "Discrete approximations to optimal trajectories using direct transcription and nonlinear programming," *Journal of Guidance, Control, and Dynamics*, vol. 15, no. 4, pp. 994-1002, 1992.

[44] Oliver Turnbull and Arthur Richards, "Collocation Methods for Multi-Vehicle Trajectory Optimization," in *European Control Conference (ECC)*, zürich, Switzerland, 2013.

[45] Inaki Rañó, "Direct Collocation for Two dimensional Motion Camouflage with Non-holonomic, Velocity and Acceleration Constraints," in *IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Shenyang, China, 2013.

[46] Qi Chen and Zhongyuan Wang, "Optimal Trajectory for Time-on-Target of a Guided Projectile Using Direct Collocation Method," in *International Conference on Mechatronic Sciences, Electric Engineering and Computer (MEC)*, Shenyang, China, 2013.

[47] M. Razzaghi and G. N. Elnagar, "Pseudospectral collocation method for the brachistochrone problem," *Mathematics and Computers in Simulation*, vol. 36, no. 3, pp. 241-246, 1994.

[48] Qi Gong et al., "Pseudospectral Optimal Control for Military and Industrial Applications," in *IEEE Conference on Decision and Control*, New orleans, Luisiana, 2007.

[49] Minxiu Kong, Zhengsheng Chen, Chen Ji, and Ming Liu, "Legendre Pseudospectral Computation of Optimal Speed Profiles for Vehicle Eco-Driving System," in *IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, Besançon, France, 2014.

[50] Fariba Fahroo and I. Michael Ross, "Advances in Pseudospectral Methods for Optimal," in *AIAA Guidance, Navigation, and Control Conference*, Honolulu, 2008.

[51] I. M. Ross, "User's manual for DIDO: A MATLAB application package for solving optimal control problems," Monterey, Department of Mechanical and Astronautical Engineering, Naval Postgraduate School.

[52] A. V. Rao et al., "GPOPS, A MATLAB software for solving multiple-phase optimal control problems using the Gauss pseudospectral method," *ACM Transactions on Mathematical Software*, vol. 37, no. 2, pp. 22:1–22:39, 2010.

[53] V. M. Becerra, "PSOPT Optimal Control Solver User Manual," United Kingdom, University of Reading 2009.

[54] S., W. Paris and C., R., Hargraves, "OTIS 3.0 Manual," Seattle, Boeing Space and Defense Group 1996.

[55] O. von Stryk, "User's Guide for DIRCOL (Version 2.1): A Direct Collocation Method for the Numerical Solution of Optimal Control Problems," Germany, Fachgebiet Simulation und Systemoptimierung (SIM) Technische Universität Darmstadt 2000.

[56] C. Jansch, K. H. Well, and K. Schnepper, "GESOP - Eine Software Umgebung Zur Simulation Und Optimierung," in *Société Française de Biophysique (SFB)*.

[57] P. Falugi, E. Kerrigan, and E Van wyk, "Imperial College London Optimal Control Software User Guide (ICLOCS)," Department of Electrical and Electronic Engineering, Imperial College London 2010.

[58] B. Houska, H. J. Ferreau, and M Diehl, "ACADO toolkit – an open-source framework for automatic control and dynamic optimization," *Optimal Control Applications and Methods*, vol. 32, no. 3, pp. 298–312, 2011.

[59] I. M. Ross, "A historical introduction to the covector mapping principle," *Advances in the Astronautical Sciences: Astrodynamics 2005*, vol. 122, pp. 05-332, 2005.

[60] John T. Betts, *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*. Philadelphia: SIAM, 2010.

[61] A. L. Dontchev and W. W. Hager, "The Euler approximation in state constrained optimal control," *Mathematics of Computation*, vol. 70, pp. 173-203, 2001.

[62] G. Elnagar, M. A. Kazemi, and M. Razzaghi, "The pseudospectral Legendre method for discretizing optimal control problems," *IEEE Transactions on Automatic Control*, vol. 40, no. 10, pp. 1793–1796, 1995.

[63] W. W. Hager, "Runge-Kutta methods in optimal control and the transformed adjoint system," *Numerische Mathematik*, vol. 87, pp. 247-282, 2000.

[64] Philip Gill, Walter Murray, Michael Saunders, and Margaret Wright, "User's Guide for NPSOL 5.0: A Fortran Package for Nonlinear Programming," Technical Report SOL 86-6 2001.

[65] Philip Gill, Walter Murray, Michael Saunders, and Margaret Wright, *Practical Optimization*.: Elsevier, 1982.

[66] O. Von Stryk, "Numerical solution of optimal control problems by direct collocation, Ed (1993)," *Optimal Control - ISNM International Series of Numerical Mathematics*, vol. 111, pp. 129–143, 1993.

[67] D Jain and P. Tsiotrea, "Trajectory optimization using multiresolution techniques," *Journal of Guidance, Control, and Dynamics*, vol. 31, no. 5, pp. 1424–1436, 2008.

[68] Y. Zhao and P. Tsiotras, "Density functions for mesh refinement in numerical optimal control," *Journal of Guidance, Control, and Dynamics*, vol. 34, no. 1, pp. 271–277, 2011.

[69] P. Williams, "Hermite-Legendre-Gauss-Lobatto direct transcription methods in trajectory optimization," *Advances in the Astronautical Sciences*, vol. 120, no. 1, pp. 465–484, 2005.

[70] C. De Boor and B. Swartz, "Collocation at Gaussian points," *SIAM Journal on Numerical Analysis*, vol. 10, no. 4, pp. 582–606, 1973.

[71] J. Villadsen and W. Stewart, "Solution of boundary-value problems by orthogonal collocation," *Chemical Engineering Science*, vol. 22, no. 11, pp. 1483–1501, 1967.

[72] G Elnagar and M. Razzaghi, "A collocation-type method for linear quadratic optimal control problems," *Optimal Control Applications and Methods*, vol. 18, no. 3, pp. 227–235, 1998.

[73] D. A. Benson, G. T. Huntington, T. P. Thorvalden, and A. V. Rao, "Direct trajectory optimization and costate estimation via an orthogonal collocation method," *Journal of Guidance, Control and Dynamics*, vol. 29, no. 6, pp. 1435–1440, 2006.

[74] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang, *Spectral Methods in Fluid Dynamics*. Heidelberg: Springer, 1988.

[75] B. Fornberd, *A Practical Guide to Pseudospectral Methods*, Cambridge University Press, Ed., 1999.

[76] L. N. Trefethen, *Spectral Methods Using MATLAB*. Philadelphia: SIAM, 2000.

[77] D. Garg, W. W. Hager, and A. V. Rao, "Pseudospectral methods for solving infinite-horizon optimal control problems," *Automatica*, vol. 47, no. 4, pp. 829–837, 2011.

[78] D., Patterson, M. GARG, W. W. Hager, A. V. Rao, D. A. Benson, and G. T. Huntington, "A unified framework for the numerical solution of optimal control problems using pseudospectral methods," *Automatica*, vol. 46, no. 11, pp. 1843–1851, 2010.

[79] S. Kameswaran and L. T. Biegler, "Convergence rates for direct transcription of optimal control problems using collocation at Radau points," *Computational Optimization and Applications*, vol. 41, no. 1, pp. 81–126, 2008.

[80] Qi Gong, I. Michael Ross, and Fariba Fahroo, "A Chebyshev Pseudospectral Method for Nonlinear Constrained Optimal Control Problems," in *Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference Shanghai*,

Shanghai, 2009.

[81] F. Fahroo and I. M. Ross, "Costate estimation by a Legendre pseudospectral," *AIAA Journal of Guidance, Control and Dynamics*, vol. 24, no. 2, pp. 270-277, 2001.

[82] Qi. Gong, Wei. Kang, and I. Michael Ross, "A pseudospectral method for the optimal control of constrained feedback linearizable systems," *IEEE Transactions on Automatic Control*, vol. 51, no. 7, pp. 1115-1129, 2006.

[83] Qi Gong, I. Michael Ross, Wei. Kang, and Fariba Fahroo, "On the pseudospectral covector mapping theorem for nonlinear optimal control," in *45th IEEE Conference on Decision and Control*, San Diago, 2006, pp. 2679-2686.

[84] W. Kang, M. Ross, and Q. Gong, "Pseudospectral Optimal Control and Its Convergence Theorems," *Analysis and Design of Nonlinear Control Systems*, pp. 109-124, 2008.

[85] C. L. Darby, W. W. Hager, and A. V. Rao, "An hp-adaptive pseudospectral method for solving optimal control problems," *Optimal Control Applications and Methods*, vol. 32, no. 4, pp. 476–502, 2010.

[86] Christopher L. Darby, "hp pseudospectral method for solving continuous-time nonlinear optimal control problems," in *PhD Thesis, Graduate School at the University of Florida*, 2011.

[87] Michael V. Cook, *Flight Dynamics Principles: A Linear Systems Approach to Aircraft Stability and Control*.: Elsevier aerospace engineering series, 2007.

[88] Randal, W Beard, "Quadrotor Dynamics and Control," Brigham Young University 2008.

[89] Aníbal Ollero, *Robótica: manipuladores y robots móviles*.: Marcombo, 2005.

[90] D. Alejo, J. M. Díaz-Báñez, J. A. Cobano, P. Pérez-Lantero, and A. Ollero, "The Velocity Assignment Problem for Conflict Resolution with Multiple Aerial Vehicles Sharing Airspace," *Journal of Intelligent & Robotic Systems*, 2012.