A Geometric Approach to Trajectory Planning for Underactuated Mechanical Systems

by

Fabrizio Boriero

Submitted to the Department of Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Computer Science
S.S.D. ING-INF05
Cycle XXX/2014

at the

Università degli Studi di Verona

July 2018

Author . . . . . .
Department of Computer Science
Jun. 25, 2018

Certified by . . . . . .
Prof. Paolo Fiorini
Full Professor
Thesis Tutor

Accepted by . . . . . .
Prof. Massimo Merro
Chairman of the PhD School Council

A Geometric Approach to Trajectory Planning for Underactuated Mechanical Systems – Fabrizio Boriero
PhD Thesis, Verona, June. 25, 2018.
ISBN: XXXXXXXXXXXXX

A Geometric Approach to Trajectory Planning for Underactuated Mechanical Systems

by

Fabrizio Boriero

Submitted to the Department of Computer Science
on Jun. 25, 2018, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Computer Science
S.S.D. ING-INF05
Cycle XXX/2014

Abstract

In the last decade, multi-rotors flying robots had a rapid development in industry and hobbyist communities thanks to the affordable cost and their availability of parts and components. The high number of degrees of freedom and the challenging dynamics of multi-rotors gave rise to new research problems. In particular, we are interested in the development of technologies for an autonomous fly that would allow using multi-rotors systems to be used in contexts where the presence of humans is denied, for example in post-disaster areas or during search-and-rescue operations. Multi-rotors are an example of a larger category of robots, called "under-actuated mechanical systems" (UMS) where the number of actuated degrees of freedom (DoF) is less than the number of available DoF. This thesis applies methods coming from geometric mechanics to study the under-actuation problem and proposes a novel method, based on the Hamiltonian formalism, to plan a feasible trajectory for UMS. We first show the application of a method called "Variational Constrained System approach" to a cart-pole example. We discovered that it is not possible to extend this method to generic UMS because it is valid only for a sub-class of UMS, called "super-articulated" mechanical system. To overcome this limitation, we wrote the Hamilton equations of the quad- rotor and we apply a numerical "direct method" to compute a feasible trajectory that satisfies system and endpoint constraints. We found that by including the system energy in the multi-rotor states, we are able to compute maneuvers that cannot be planned with other methods and that overcome the under-actuation constraints. To demonstrate the benefit of the method developed, we built a custom quad- rotor and an experimental setup with different obstacles, such as a gap in a wall and we show the correctness of the trajectory computed with the new method.

Thesis Advisor: Prof. Paolo Fiorini
Title: Full Professor

“Say what you know, do what you must, come what may”

Sofia Kovalevskaya

Acknowledgments

First of all, I want to thanks my advisor and mentor Prof. Paolo Fiorini, he supported me for more than ten years in the "Altair robotic lab" with his strong knowledge and patience, which was particularly high during the review of this thesis. I want also to thanks Prof. Joel Burdick for his advice that had a strong impact on this thesis and to hosted me during my internship at Caltech University.

My sincere thanks go to thesis committee Prof. Lorenzo Marconi and Prof. Paolo Robuffo Giordano for reviewing this thesis and for their comments and questions. I want to thanks all experts that shared their knowledge to help the development of the theoretical part of this thesis. A particular contribution has been given by the time that Nicola Sansonetto spent to introduce me to the magic world of Geometric mechanics and by Prof. Riccardo Muradore with his help in the control theory part. My PhD studies were also supported by the PhD school Chairmen, Prof. Massimo Merro and Prof. Gloria Menegaz and by the internal committee composed by Prof. Alessandro Farinelli and Prof. Giandomenico Orlandi.

A huge thanks to my old friends Diego Dall'Alba, Michela De Piccoli and to all members of the Altair robotic lab, in particular to Francesco Bovo that helped me a lot during the development of quad-rotors used in the experimental setup. On a personal side, I want to thanks Linda Tonolli for her love and to share this strange trip that life offers to us. Last but not least a warm thanks to my sister Giulia, my father Sergio and my mother Annalucia for all their support.

Contents

Chapter 1

Introduction

1.1 Summary of the Chapter

In this Chapter, we introduce the contest of the problem of the trajectory planning and the motivations that inspired this thesis. We present the current challenges in robotics, in particular the "multi-rotor" systems that have still interesting problems to overcome. After an overview of the problem and the methods present in literature, we will show the contribution of this thesis in the field of trajectory planning and the application of geometric mechanics to robotics.

1.2 Statement of the problem

The problem of this thesis is to find an optimal control input that drives an under-actuated mechanical system from an initial starting configuration to a final target configuration. In particular, using a geometric approach [8], we can state the problem has searching a control function $u(t)$ that minimize a certain cost function

$$J = \int_0^T g(x(t), u(t))dt \tag{1.1}$$

subject to the following conditions:

- the dynamic evolution of the state space is expressed as $\dot{x} = f(x(t), u(t))$ and is subject to constraints on the kinematics, the dynamics and on the controls
- f and $g \geq 0$ are smooth and $x \in M$ where M is a smooth manifold
- the endpoint conditions are $x(0) = x_0$ and $x(T) = x_T$

A detailed description of the problem is provided in Section 1.8 but the idea of this thesis is to use concepts coming from the geometric mechanics to generate an optimal trajectory for under-actuated mechanical systems. In particular, we

are interested to generate and execute on real quad-rotor a trajectory in the full configuration-space.

1.3 Motivations

Thanks to recent investments and public attention to the so-called "Industrial revolution 4.0", the modern society has a big expectation from robotics, which is seen as an investment to change everybody's lifestyle. This technological evolution will have a substantial impact on the way we produce goods and services, but there are also many open questions involving other aspects of human life such as philosophy, anthropology, ethics, sociology, law, etc.

Figure 1-1: Robot Yumi ABB: a popular robot used in industry mainly to assemble light products. Credits: www.abb.com

One of the most sensitive topics in this area is the addition of autonomy to robots with the aim to transfer the execution of the selected tasks from humans to machines avoiding tedious procedures and increasing the safety of the operators.

A concrete example of automation on an airplane is the "autopilot" mode which, since the 1912 assists the pilot in some phases of a regular journey. This feature allows pilots to focus on critical stages of the flight such as landing and take off, demanding to the automatic controller the tedious part of the journey

Another field where autonomy is playing an important role in the automotive industry. Since 2014, Tesla Model S offers a driver assist system composed of services such as speed control, autosteer, speed assist and others. This technology cannot be considered "fully autonomous", but many companies like Tesla, Google, and many others are trying to increase autonomation in cars and trucks. The automotive industry was also the first field that gave a precise definition of each level of autonomy in a car [80]. The levels of autonomy are:

- Level 0 -No driving automation: The entire driving task is under the driver control, also when the safety systems are active.
- Level 1 -Driver assistance: The car has a control on the lateral and longitudinal motion (not at the same time), and the driver controls everything else.
- Level 2 -Partial driving automation: a driving automation system control (ADS) controls lateral and longitudinal motion at the same time, and the driver supervises the driving automation system, detecting and responding to external objects or events.
- Level 3 - Conditional driving automation: ADS controls the entire dynamic driving task with the expectation that the driver is ready to be warned about any issues and he is ready to take control of the vehicle.
- Level 4 - High driving automation : Some driving subtasks (ex. parking, staying in the same lane in the freeway) are done by the automatic system control without the expectation of a human intervention
- Level 5 - Full driving automation: All the driving tasks are done by the automatic system control without user intervention

Increasing the level of autonomy is not easy because there are a lot of unknown external events that are difficult to forecast and handle safely.

There are particular environments or situations where the presence of humans is denied and the only way to operate is through fully autonomous robots. If we look at space exploration, for example, some projects push forward the use of autonomous robots on other planets or asteroids. In real missions, the robot receives the task

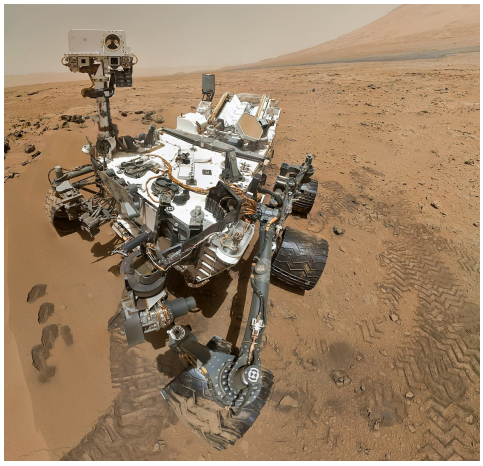

Figure 1-2: NASA/JPL Curiosity rover Credits: `https://www.jpl.nasa.gov`

plan from Earth and executes it in complete autonomy. After that, the results taken

from the onboard sensors are sent back to operators on Earth who analyze them and decide how to plan the next phase of the mission. In this case, the robot has to avoid obstacles and make a decision in complete autonomy so it is necessary to take into account both kinematic and dynamic constraints and merge this information with the real-time data acquired by the sensors.

There is a strong need for autonomy also when an accident happens and rescuers cannot search for survivors due to the presence of fire, avalanches, nuclear radiations or other external obstacles. After the Fukushima Dai-ichi nuclear accident in 2011, many robotic researchers focused their efforts to prepare robotic systems that can reach and operate in post-disaster areas. Two interesting projects that investigate the use of robots in post-disaster areas are the Sherpa project [44] and ERL Emergency Robots project [24]

In the Sherpa project, the goal is to support operators in search-and-rescue operations in the mountain area. In this project, a fixed-wing drone is used to patrol a mountain area to find people who are lost. Another case of study considered in this project was the rescue of people buried by an avalanche with the help of a multi-rotor that follows a magnetic signal coming from an anti-avalanche system (ARVA).

A similar project is the "ERL Emergency project" where the goal is to propose a robotic challenge in which an autonomous robot has to investigate a post-disaster area to search for possible survivors or to do some emergency tasks (for example closing a valve etc.).

1.4 Robotic systems classification

There are many methods to drive a robot and many model to represent them as we will see in the next Chapter. Refering to [61], in this Section we make a simple classification of the various robotic systems based on their main characteristics:

- Quasi-static system: is a robot described by a configuration space that consider only the position of the robot and it is also controlled using a set of Cartesian positions. In this case the dynamics does not influence the motion of the system.

- Kinematic system: is system controlled in velocity where the state space contains only a configuration point $q \in M$.

- Mechanical system: a system controlled in force and the state space contains positions and velocities $x = [q, \dot{q}]$.

- Under-actuated mechanical system (in the following "UMS" both for singular and plural devices): A system with less controllable degrees of freedom

than the observable ones where the control inputs are forces and the dynamic evolution involve positions and velocities.

Formal definitions, mathematical specifications and other characteristics of UMS are presented in more details in Chapter 3. In the next Section we present a particular robot that could be classified as an under-actuated mechanical system: the multi-rotor. We are interested in this particular robot because it contains heterogeneous degrees of freedom (rotational and translational) and it is influenced by the gravity force.

1.5 Multi-rotors

In the last decade most of the methods concerning the development of autonomous systems were applied to multi-rotors that "currently represent the best bet in terms of maneuverability and their ability to carry small payloads" [49]. Since the begin of the new millennium, multi-rotors had a rapid development in the hobbyist communities thanks to the affordable cost and the availability of parts and components, mainly present in the toy market. The high number of degrees of freedoms and the challenging dynamics of multi-rotors put new questions also in the research community.

In [39] authors proposed a first approach to control multi-rotors, finding good force/torques controls to stabilize the systems. A back-step approached was used to control the actuated degrees of freedom $\{roll, pitch, yaw, thrust\}$ in order to minimize the error between the desired trajectory and the position and the attitude estimated by sensors placed on-board. Two years later, in [18] the researchers propose a Lagrangian model of quadrotor describing the position and the attitude of the system using six coordinates $\{x, y, z, roll, pitch, yaw\}$. Applying a non-linear controller [41], the authors show that the multi-rotor is able to reach a target point in the Cartesian position $\{x, y, z, yaw\}$. A comparison between the two methods can be found in [12].

Since these early works, the literature counts many different approaches and variations to stabilize and control multi-rotors. In this thesis, we are interested in a particular sub-topic involving multi-rotors, which is the planning of trajectory of multi-rotors from an initial point to a target one, also considering dynamic and external constraints, as presented in the following Chapters.

The first approach to trajectory planning for multi-rotors was proposed in [21]. In this work authors minimize a cost function that involves the velocity of the multi-rotor, assuming that the energy consumption is proportional to the average velocity. The problem of finding an optimal trajectory was reduced to find a sequence of polynomial functions in the flat-outputs, which is the space of the states generated by the differential flat equations (see Section 2.5 for details). In a second phase, the method chooses a speed profile that allows following the computed trajectory

respecting the dynamics of the system. A very interesting approach was proposed in [13] where the authors applied the collocation methods to minimize the flying time. The idea is to approximate the trajectory with a sequence of polynomial functions parametrize by a sequence of variables (λ's). The authors present the problem of the under-actuation of multi-rotors showing two equations that represent the second-order non-holonomic constraints (that we present in Chapter 3). These equations are used to reduce the configuration space from a six-dimensional space $\{x, y, z, roll, pitch, yaw\}$ to a four dimensional configuration space $\{x, y, z, yaw\}$.

In [34] the authors introduce the concept of the Hamiltonian function (see Chapter 3 for details) to compute a reachable set thanks to the use of a specific toolbox. In this work, the system dynamics is not considered through the full trajectory but is approximated by the finite set of segments in which the trajectory is divided. For each segment, a different control algorithm is chosen, and the problem of finding the best switching conditions from one segment to the next is also solved. To drive the multi-rotor to its final state in a given time, the authors propose a validation method that tests if a sequence of states can reach the desired final configuration.

In [66] the authors use the influence of the dynamics to plan aggressive maneuvers, in contrast with the previous works where the focus was to stabilize the quadrotor around an equilibrium point. The main idea of the paper is to define a sequence of *keyframes*, which are common configuration points defined as $\{x, y, z, yaw\}$. The method builds piece-wise polynomial functions where the parameters are optimized using, as the cost function, the square of the norm of the jerk. Also in this work, the authors use the flatness property to control the quad-rotor in the space of the flat outputs.

In [42] there is an application of the *Pontryagin Maximum principle* (for details see Chapter 3). The authors propose to build the augmented Hamiltonian function considering also the state constraints and searching for good values for the adjoint variable to minimize the cost function. This is the application to quadrotor of the *direct adjoining approach* taken from [25]

The method described in [92] computes an optimized function using a *black-box* approach based on the pseudo-spectral method [93]. In this case, the best trajectories in flat outputs are searched to minimize the execution time and satisfy the multi-rotor dynamics. The benefit of this approach is the possibility of considering upper and lower bounds of the states and control functions.

The interest in solving the trajectory generation problem is also evident in two very recent results proposed in [59] and [27]. In [59] the trajectory computation method is similar to [49] but, thanks to a powerful onboard computer, the state estimation of the multi-rotor is computed using a stereo vision camera. In [27] the trajectory is computed using an interpolating polynomial function whose parameters are chosen to satisfy the multi-rotor dynamics and to avoid possible collisions with the obstacles, using a special vision sensor.

1.6 Multi-rotors control

To introduce the contribution of this thesis we presents in more details the control strategies for multi-rotors found in the literature. We can identify two main classes of controllers: the attitude controller and the position controller.

1.6.1 Attitude controller

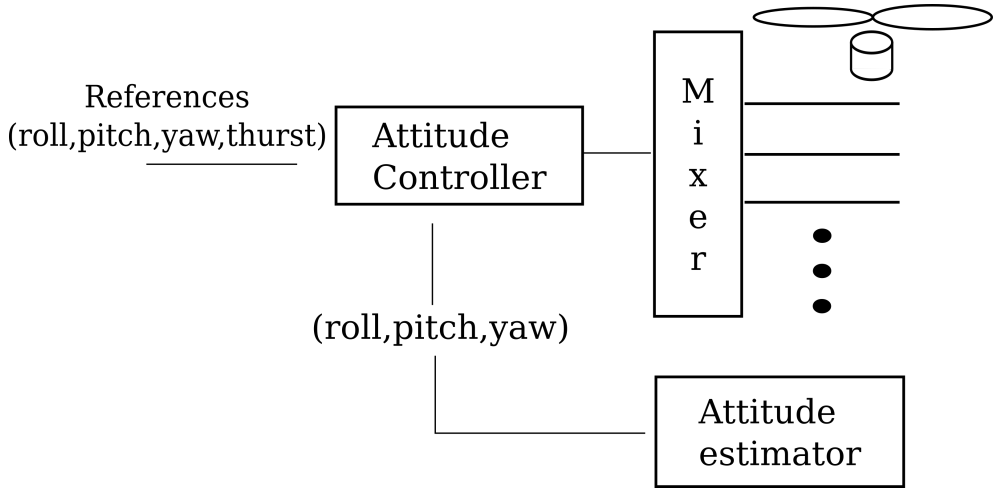

Figure 1-3: A schema of an attitude controller for multi-rotors

As we can see in figure 1-3, the attitude controller use an attitude estimator, usually placed on-board, which processes the data coming from different sensors to give the current attitude of the multi-rotor. The reference commands are provided by the pilot or from a "Ground station" computer, in the form of four angles $(roll, pitch, yaw, thrust)$. The attitude controller computes the error between commands and the current attitude and send forces and torques to another controller, called "Mixer". The role of the mixer is to transform the desired forces and torques to a current motors of the multi-rotor. As presented in Chapter 6, there are many ways to place motors on a frame and we can have a minimum of four motors up to hundred or more. In this thesis, we decided to work with four motors placed according to "X" configuration to simplify the mathematical formulation and reduce the cost of the experimental setup.

1.6.2 Position controller

An important task to autonomously is controlling a quad-rotor in Cartesian space, consider obstacles present in the environment, and decide to avoid or even to in-

teract with them. The simplest way to solve this problem is to use the "Flatness property" (presented in Section 2.5 to map the references commands from $r = (x, y, z, yaw)$ to $r = (roll, pitch, yaw, thurst)$.

In 1-4 we can see how it is possible to modify the attitude controller presented in the previous Section to include the Flatness property, changing the reference point from an attitude configuration to a position configuration.

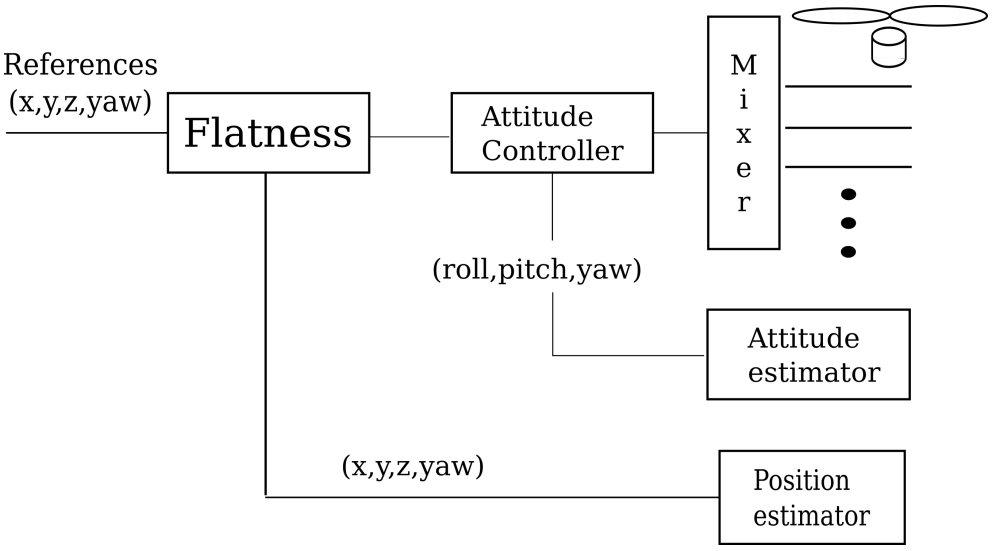

Figure 1-4: The schema of a position controller for multi-rotors

1.7 Discussion about current techniques

The Flatness property is useful from a practical point of view but in this way, we lose two degrees of freedom and we can plan the motion of the quad-rotor only in four degrees of freedom ($q = x, y, z, yaw$). In particular it is not possible to introduce constraints or target points with a configuration in six degrees of freedom ($q = x, y, z, roll, pitch, yaw$). In the Section 1.5 we presented an overview of all the different techniques and we saw that most of them are based on the Flatness property, which induce the authors to use a switching controller passing from an attitude controller to a position controller [66] [59] because it is the simplest way to control the multi-rotor in position putting also attitude constraints. This method allow to execute aggressive maneuvers, such as passing a generic oriented window, but are local methods so it is not possible to compute a global optimal trajectory, as shown in Chapter 2

It is also interesting to note that in [34] [42] the authors use optimal control techniques defining an augmented Hamiltonian function but it is not very clear how

it is implemented and how the algorithms use the “Pontryagin Maximum principle”.

In general, we can see the most of the works use old methods and apply them to this new interesting object, which is the multi-rotor, but there is still a lack of theory. For example, it is not clear how the constraints coming from the under-actuation affect the planning of a global trajectory.

1.8 Thesis contribution

Motivated by the challenge of trajectory planning for multi-rotors, in this thesis we are interested in developing a method that generates a feasible trajectory in six degrees of freedom ($q = x, y, z, roll, pitch, yaw$) overcoming the limitation of switching between the attitude controller and the ”Flatness” controller. Figure 1.8 shows an idea of the control schema that is useful to understand the contribution of the thesis

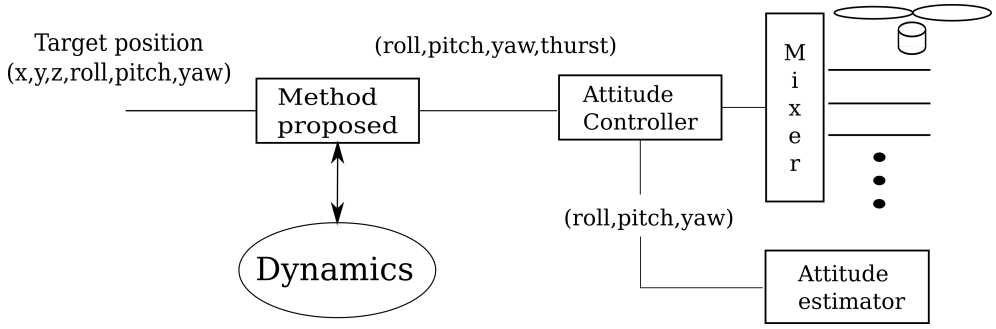

Figure 1-5: A schema of a position controller for multi-rotors

We want to generate a dynamically feasible trajectory which connects an initial point to a target one but in a six DoF configuration space. To do this we use the dynamics of the system and, when we find the desired trajectory, we pass the subset of actuated degrees ($q = roll, pitch, yaw, thurst$) to the attitude controller. Compared to the ”Flatness” property used in the other works present in literature, we are doing a map between $(x, y, z, roll, pitch, yaw) \rightarrow (roll, pitch, yaw, thurst)$ instead of $(x, y, z, yaw) \rightarrow (roll, pitch, yaw, thurst)$. The benefit is the possibility of overcoming constraints represented in six degrees of freedom.

We have carried out an interdisciplinary research that involves four disciplines: Robotics, Computer Science, Geometric mechanics and Control Theory. Mathematical tools give the possibility to see the concrete case studied in robotics through a lens which shows the nature of the issues to be solved in trajectory planning. In particular, Differential Geometry places the role of kinematic and dynamic constraints in the correct geometrical spaces, and help solving the challenges of trajectory planning.

To find a solution of this problem, we started by studying the works done between the 80's and the 90's (see Chapter 2) to find how it is possible to understand the characteristics of under-actuated mechanical systems from a mechanical and geometrical point of view. We saw many interesting works and methods which are not currently "popular" in the robotics community but that give an in-depth understanding of the problem that we want to study.

To deal with under-actuated mechanical systems, we need to know how kinematic and how potential energies affect the motion of the system.

In particular, we propose to use the Hamiltonian formulation that allows to explicitly consider the evolution of the energy in the system with the benefit of having a formal model, given by the differential geometry structures, which is useful as a link with the advanced mathematical methods, as presented in Chapter 4 and 5

Finally, we propose the implementation of our approach to a real case to execute a task never presented in the literature: using a quad-rotor to pass through a deep gap in a wall.

1.9 Organization

In Chapter 2 are presented the literature works about motion planning for multi-rotors. In Chapter 3 there are the theory useful to understand the characteristics of the under-actuated mechanical systems whereas in Chapter 4 we applied a method coming from the geometric mechanics to a simple under-actuated robot: the cart-pole. In Chapter 5 we presented a method to plan a feasible trajectory for quadrotor overcoming the limitation introduced by the under-actuation and proposing a task never done before: flying trough a small gap in a deep wall. In Chapter 6 we demonstrate the new method building a real setup and we analyzed the results. Finally in Chapter 7 we present some conclusions and future works.

Chapter 2

State of the art

2.1 Introduction

In this Chapter we want to present an overview of the methods presented in the literature about the motion planning problem. We want to understand what are the main techniques and their limitations to find a specific research area where we want to give our contribution.

2.2 Motion Planning

Automatic Motion planning is a discipline born in the late 90'[14] with the goal to give robots the ability to reach a target in an environment that could be partially known, sometimes with the presence of fixed or moving obstacles. If a robot is moving in a real world, it has to deal with physical laws and geometrical constraints so we can say that "motion planning" is an *interdisciplinary* research fields which include knowledge coming from analytical mechanics, control theory, differential geometry and computer science.

In this thesis, we consider a robot as a mechanical system composed of a set of actuators and sensors which can make changes to the device and to the real world. In particular, these physical changes will happen within a defined area, called *workspace*. In this area, we can avoid or interact with external objects and the laws of physics act in kinematic and dynamic properties of the body. In the workspace, a user plan the motion of the robot as a sequence of movements or goals to reach that are useful to achieve its needs. According to Jean-Claude LaTombe [51], we can define the *basic motion planning problem* as:

Assumption 2.1. *Let A be a single rigid object - the robot - moving in the Euclidean space W, called workspace, represented by R^N, with $N = 2$ or 3.*

Assumption 2.2. *Let $B_1 \dots B_q$ be fixed rigid objects distributed in W. The B_i's are called* obstacles

Assumption 2.3. *Assume that both the geometry of $A, B_1 \ldots B_q$ and the locations of the B_i's in W are accurately known. Assume further that no kinematic constraints limit the motion of A (we say that A is* free-flying object*)*

Definition 2.4. *The **basic motion planning problem** is: given an initial position and orientation and a goal position and orientation of A in W, generate a path τ specifying a continuous sequence of positions and orientations of A avoiding contact with the B_i's starting at the initial position and orientation, and terminating at the goal position and orientation. Report failure if no such path exists.*

To solve a motion planning problem, it is essential to know the *configuration* of a robot. In the simplest case, the configuration of a robot is a specification of an element of the robot relative to a fixed reference point [4]. The set of all possible configuration of the robots is defined, in this context, as the *configuration space*. As we can see in the next Section, there is a more precise definition of configuration space, but for the 80's works, this definition was enough to solve the fundamental motion planning problem. The dimension of the configuration space is equal to the number of independent variables used to represent the configurations, which are also called in literature *degrees of freedom (DOF)* of the robot.

This first approach to motion planning was studied and used to solve the "piano mover's problem" [60]. In this paper, the size of the robot is considerate as a geometric constraint that limits the possible directions of motion due to the presence of obstacles. The problem is to find a path from a starting configuration to a target one modifying position and orientation of the robot avoiding the obstacles as shown in figure 2.2

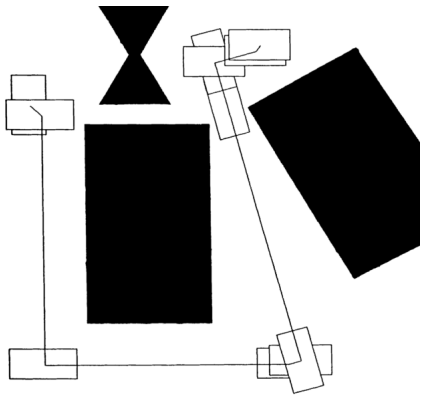

Figure 2-1: The piano mover's problem [60]

During the 80's, roboticists worked on this problem proposing many heuristics and approximated methods. The idea was to decompose the configuration space into simple cells finding which ones were inside the *free-space* defined as space

where there were no obstacles. Finding a possible *collision-free path* was the problem of finding a sequence of free cells connecting the cell with the initial configuration point to the final configuration point. For an overview of all this method, see the book "Robot motion planning" [51] and the references within.

In 1987, Jean-Paul Lamond in [53] said that finding a collision-free path along obstacles without considering the *kinematic constraints* was an approximation unacceptable when we have to deal with real robots, and he introduced new models and mathematical techniques to represent the reality and achieve more precise computation. Sastry and Murray, in 1989, introduced a new concept [81] in robotics which uses the kinematics constraints, i.e., constraints on robot velocities,such as for example *nonholonomic constraints*. This algorithm is based on differential geometry concepts, in particular, Lie Algebra, to find a sinusoidal input which drives the robot in directions that are not allowed by the kinematic constraints.

These approaches can be seen as the solution of the classical mathematical problem: the "rolling disk problem" which is the typically constrained motion planning. During the 90's many authors proposed different approaches to this problem [45, 9, 81, 69, 5, 38]

This application shows in a simple way that when a wheel touches the terrain if we are in the presence of the pure rolling condition, velocity in the point of contact has to be zero, which is an external kinematic constraint.

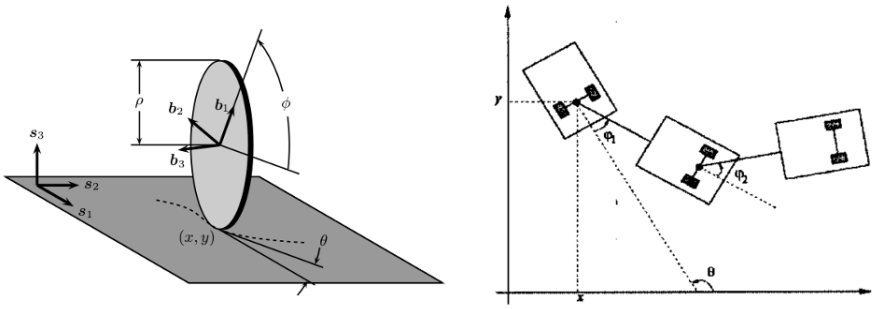

Figure 2-2: The "rolling disk" and the "car with trailers" examples

The rolling disk case started a long sequence of publications on *nonhonolonmic motion planning* with a lot of examples and algorithms. The most popular case is the "car-like" motion problem, that researchers solved by developing a different kind of methods to steering a car with [53, 9, 57, 76, 48] or without [81, 52, 75, 46] trailers.

Another step forward was done by J.Bobrow et al. in 1985 [10] with the introduction of the *Kinodynamic motion planning*. In this case, it was pointed out the importance of considering not only kinematic constraints, but also dynamic

constraints, in particular bounds on forces, velocities, and accelerations. For an overview of kynodynamic alghorithms refer to [15]

During the first part of the 90's many works address this problem, Donald et al. in 1993 [23] presented a solution to find a path that avoids many obstacles introducing velocities constraints. The technique is still a graph-based searching on a set of free-cell, but velocities are used as the weight to assure a safe path.

Fiorini and Shiller in 1993 [29] added the possibility of avoiding also moving the obstacle, with an algorithm, working in the space of velocities, which considers the dynamics of the robot and the obstacles and finds an optimal trajectory that avoids the collisions

2.3 Overview

In the previous Section, we presented a summary of approaches useful to solve a generic motion planning problem considering many aspects of robots. The aim of this Chapter is to present, by reviewing the methods and the techniques present in literature, the main classes of algorithms that solve the *motion planning problem*. Recalling the introduction, we describe the motion planning problem for under-actuated mechanical systems as [61]

Definition 2.5. ***Motion planning for UMS****: giving an initial state $x(0) = x_{start}$ and a goal state x_{goal}, the motion planning problem for a system $\dot{x} = f(x,u)$ is to find a control history $u(t)$ with $0 < t < T$ such that*

$$x_{goal} = x_0 + \int_0^T f(x(t),u(t))dt \tag{2.6}$$

while avoiding any obstacles that may be present on the environment and respecting the constraints introduced by the kinematics and the dynamics of the system. We also want to minimize some notion of cost

$$J = \int_0^T L(x(t),u(t))dt \tag{2.7}$$

Respect to the general definition of motion planning [51], we are not interested only in finding a path that is obstacle-free but a trajectory that is *admissible*. The difference is that we take into account the dynamic evolution of the system and we accept only accelerations and velocities which are feasible by our mechanical system. Between all possible solutions we are interested in a notion of "optimum" to choose the best trajectory in terms of the time, the space, the energy spent etc.

There are many works in literature that approach the problem of the trajectory planning for UMS. As shown in figure 2.3, we can propose a classification of these methods. The first classification that we do in this thesis is between local and global methods.

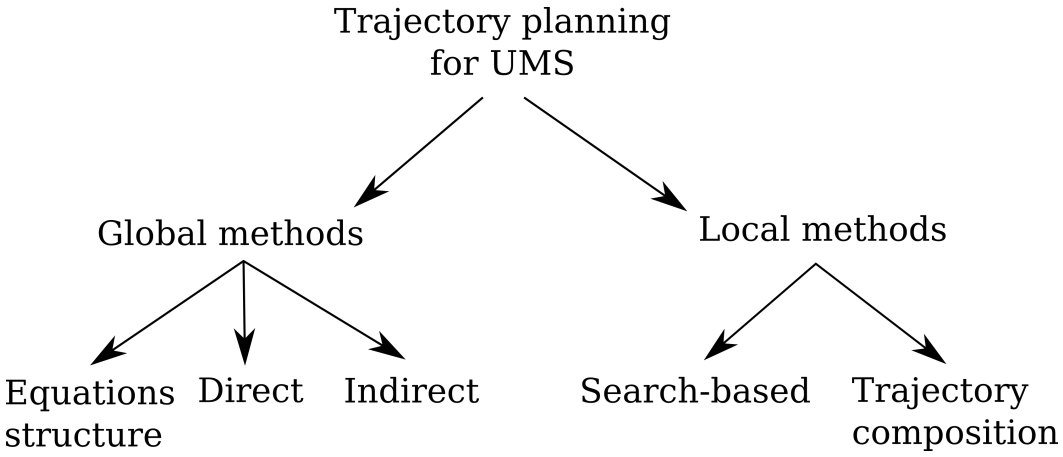

Figure 2-3: A simple categorization of trajectory planning methods

2.3.1 Local methods

Local methods started at the end of the 90's and are currently the most used techniques to find a trajectory in configuration space. The idea supporting these algorithms is to compose a trajectory as a sequence of "small" movements, checking if the path is free of external obstacle and ensuring that the dynamics of the system allows following the computed path. There are many approaches to find a solution using local methods but the most common are *search-based algorithms* and *trajectory composition*.

In the search-based methods, authors discretized the path in a set of many configuration points that are collocated into a search graph composed by *vertexes* and *edges*. To find a solution, authors use heuristic approaches to find a "good" sequence of edges which connects a set of vertexes which represent the configuration point of the desired robot. For a good overview of these methods, the reader could refer to [55] and references therein. Some examples of this kind of algorithms are Probabilistic Roadmap Planning (PRP) [47] or rapid-exploring random trees (RTTs) [56].

The trajectory composition is another approach, which consists of finding a sequence of trajectories which are, locally, *admissible* and obstacle-free. It is possible to drive a robot form an initial configuration to a final one by connecting all these pieces while respecting the kinematics and the dynamics of the system. A typical example could be found in [54] where a trajectory was generated respecting the nonholonomic nature of the system, whereas in [29] the dynamics of the system and of the obstacles were also considered.

It is important to notice that local methods cannot consider the dynamics of the system to find a "global" optimal trajectory, because when we switch from one piece to another, we lose the "history" of the dynamic evolution of the system.

2.3.2 Global methods

It is possible to approach the problem of trajectory planning from a global point of view, which means that we try to compute a continuous function representing the complete path in the configuration space connecting the starting point to the final point. To solve this problem, we are interested in exploiting the role of the dynamics of the system and we look for the solution of an *optimal control problem*. Optimal control is a large research area that addresses the computation of analytical solutions using the *Pontryagin Maximum Principle*

In the case of under-actuated systems, however, it is not possible to find an analytical solution of the control inputs and we need to use a numerical method to find the desired control inputs. For generic systems, solving the trajectory planning problem for UMS is equivalent to solve the *Two point boundary value problem (TPBVP)* .

In the literature, three main approaches are used to solve this problem:

- Specific equations structure: if it is true that it is not possible, in general, to find an analytical solution, for specific classes of under-actuated mechanical systems it is possible to exploit the properties of their dynamic equations to compute a specific control inputs or it is possible to find a way to linearize the system and apply the classical methods present in the optimal control theory for linear systems.

- Indirect methods: these methods use numerical algorithms to find a sub-optimal solution using techniques that yield local optimum solutions.

- Direct methods: these methods discretized the desired trajectory, fitting parametric curves inside the time intervals. The second step is applying a numeric algorithm to find the best parameters of the curves, connecting all the segments.

We are interested in this class of methods because we want to understand how to plan a trajectory that considers the dynamics of the system from the initial point to the final one. It is important to notice that the global methods can be seen from a geometrical point of view, which is useful to understand the problem and try to apply useful tools developed in the geometric mechanic community, as we will see in the next Chapters.

Before presenting the main global methods, in the next Section we propose how we can apply the notion of "controllability" to non-linear systems.

2.4 Controllability

In the case of linear systems that we can write system dynamics as

$$\dot{x} = Ax + Bu,\ x \in \mathbb{R}^n,\ u \in \mathbb{R}^m \tag{2.8}$$

and by the Kalman rank condition (KRC), the system controllability is defined as

Theorem 2.9. *It the rank of the matrix*

$$[B|AB|A^2B|\dots|A^{m-1}B] \tag{2.10}$$

is m, it is possible to move the system from any state to any other state in finite time

With nonlinear systems, such as most UMS, the previous notion of controllability is no longer valid so we need another definition. Some authors [17] [90] [70] [61] proposed other definition of controllability for nonlinear mechanicals systems, which is well explained by Sastry in [82]:

- *Small-time local accessibility (STLA) at x*: For any time $T > 0$, the reachable set starting from x at times $t < T$ contains a full-dimensions subset of the state space.
- *Small-time local controllability (STLC) at x: For any time $T > 0$, the reachable set starting from x at times $t < T$ contains a neighborhood of x*
- *Global controllability:*: The robot can reach any state from any other state

To demostrate the STLA condition, it is possible to build the Lie algebra A of the mechanical system and checks the *Lie Algebra Rank condition* (LARC) [17]. If a mechanical system has symmetric properties that allow it to move forward and backward along the Lie bracket directions, it is possible to demonstrate that the LARC condition implies the STLC but only if we consider *dift-less* mechanical systems (see Section 3.3 for details).

To understand this problem is useful to see [86], where the authors propose a geometric approach to motion planning for under-water UMS showing that is not possible to use bracketing techniques to demonstrate controllability in the presence of drift.

2.5 Specific equations structure

Since the early 90's [58] researchers have studied the structure of the equations describing the dynamic of UMS systems to find ad-hoc solutions for specific classes of systems. In this Section, we present some well-known result that are important

steps in planning a trajectory for a robot. It is important to notice the following techniques refer to the D'Alembert principle presented in Section 3.3.2 applied to Nonholonomic mechanical systems (NMS) since, as we will see later, there is a kinematic reduction from UMS to NMS.

2.5.1 Sinusoidal inputs and chained form

In 1990 Sastry and Murray presented a method [69] based on differential geometry tools. The idea was to consider a specific class of UMS, called *contact structures* known in the field of classical mechanics [4]. When considering these robots, it is possible to use the first level of Lie Bracketing which, in combination with the input vectors, span the tangent space of the configuration space. If the configuration manifold of the systems is $Q = \mathbb{R}^3$, it is possible to write the dynamic evolution of the system in a form called *chained form* that, in its simplest form, is:

$$\begin{aligned} \dot{q}_1 &= u_1 \\ \dot{q}_2 &= u_2 \\ \dot{q}_3 &= q_2 u_1 \end{aligned} \tag{2.11}$$

If we want to find the best trajectory that connects a given initial configuration point to a final configuration, we need to evaluate a cost function such as the energy consumption as

$$J = \int_0^1 u(t)^2 dt \tag{2.12}$$

The goal is to minimize J by finding the appropriate control sequence. In this special case, the solution computed in the input will be in the form $u(t) = e^{\Lambda t} u_0$ where Λ is a constant skew-symmetric matrix and the control inputs will be sinusoidal functions at different frequencies. The computation of Δ and the initial control u_0 is not a simple procedure and the reader is referred to [69] Using a similar approach, the work proposed in [87] extends the trajectory planning using the chained form to a car with n-trailers.

2.5.2 Kinematic reduction

Another approach used to deal with UMS is to apply a *kinematic reduction* (see [72] for an introduction to the theory), which is a method to transform a complex problem into an equivalent simpler problem. From a geometric point of view, the reduction is done using a map from the original configuration manifold Q to a distribution lying in the tangent space TQ. An interesting application of this method was proposed by Ostrowski and Burdick [73] where this theory was used to propose a way to control a *snake-board* robot [71]. In particular, in this work the kinematic reduction was used to simplify the configuration space avoiding the cycle variables

and, as a consequence, reducing the complexity of the problem. Another example of kinematic reduction for UMS could be done by mapping the input vectors and the accelerations constraints of the dynamic system to a tangent space of the configuration manifold in order to search for an optimal path considering the variables in the same geometrical space. An example of these reductions could be found in [64] where the *Boltzmann-Hamel equations* are studied in the context of optimal control for UMS. Another example is the application of the kinematic reduction to plan a trajectory of a suspend mass attached to a moving cart [83].

2.5.3 Differentially flat systems

The structure of the dynamic equations of a mechanical system can also be exploited by using the so-called "differential flatness" property. The first application of this method in control theory was proposed by Fliess in 1992 [30] and by Murray [94], showing that there is a direct equivalence between *differential flatness* and mechanical system linearized by feedback control. The idea is the possibility of linearizing a desired mechanical system introducing new variables, called *flat outputs*, that allow computing the control inputs of the system as functions depending on flat outputs. To clarify this concept, if an UMS has $q \in \mathbb{R}^n$ degrees of freedom and $u \in \mathbb{R}^m$ inputs, we can find a set of $y \in \mathbb{R}^m$ flat outputs which could be written in the form

$$y = y(q, u, \dot{u}, \ldots, u^p) \tag{2.13}$$

such that

$$\begin{aligned} q &= q(y, \dot{y}, \ldots, y^q) \\ u &= u(y, \dot{y}, \ldots, y^q) \end{aligned} \tag{2.14}$$

This properties was successfully applied to car with n-trails [79], Vertical Take-Off and Landing (VTOL) aircrafts [63], multi-body robots [50] and quadrotors [31][39]

2.6 Conclusions

In this Chapter started proposing the difference between local methods and global methods to solve a motion planning problem and after we shown the main techniques to do trajectory planning for Under-actuated mechanical systems. Today most of the work in motion planning are based on heuristic approaches which are easily computed by modern calculators and could deal with obstacles in a simple way. In general, global methods have the disadvantage that they do not work very well when there are obstacles on the trajectory. On the other hand, if we want to exploit the dynamics of the mechanical system, we need to considerate the behavior of the system as a continuous function otherwise there are points of discontinuity which affect the motion of the robot. In this thesis, we are interested in global

methods because, despite the fact that are not "popular" since the end of the 90's, we think that could have interesting points which can push forward the research in trajectory planning for under-actuated systems. In the following Section, we present a theoretical framework that will be the base for the application of mathematical methods to solve the problem of generating a feasible trajectory.

Chapter 3

Geometric mechanics and control

3.1 Introduction

In this Chapter we present the mathematical background that is useful to understand the trajectory planning problems for an under-actuated robotic system. We apply the techniques used in differential geometry to approach problems there are still open in the robotic community. The knowledge presented in this Chapter comes from a research area knows as "geometrical mechanics", in particular we refer to the work done by Marsden [62], Cortes [20], Bullo [17], Bloch [8] and others.

3.2 Simple mechanical system

3.2.1 Configuration space

If we want to describe the motion of an object, we need to introduce a common way to identify position and orientation. We start from the definition of **particle** which is an object with a mass but without volume. If we collect many particles in a way that the relation between their positions remains constant in time, we obtain a so-called **rigid body** which is an object with a position, orientation, and volume. Figure 3.2.2 shows the difference between a particle and a rigid body.

When we describe the position of a particle or a rigid body, we need first to define a **reference frame**, which allows measuring relative distances and orientations between the reference frame and the rigid body. In the rest of the thesis, we call $\Sigma_{world} = \{O, w_1, w_2, w_3\}$ the reference **inertial frame** that is composed by the origin O and a set of free orthonormal vectors $\{w_1, w_2, w_3\}$. With the term "inertial" we refer to a frame where the Newton-Euler laws hold, but for a more specific and formal definition please refer to [4].

Once we have defined the inertial frame, we can specify the position of a particle or the origin of a reference frame associated with a rigid body using a vector r which connect the origin O_{world} to particle on the body origin. If we consider also

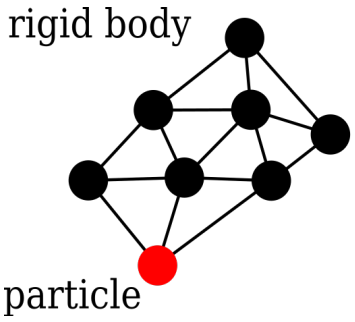

Figure 3-1: Rigid body and particles

the orientation of the rigid body, we have to introduce another frame, called **body frame** $\Sigma_{body} = \{O_{body}, b_1, b_2, b_3\}$ where O_{body} is the origin of the frame relative to the rigid body and the position of the particles which compose the rigid body defined with respect to the basis $\{b_1, b_2, b_3\}$. Now it is possible to define a vector r as $r = (O_{body} - O_{world})$ that allows measuring the distance between the origins of the two frames defined before. We also need to set a matrix $R \in SO(3)$ that describes the orientation of the rigid body respect the body frame Σ_{body}. It is important to notice that the orientation is related to the body frame and the position is associated to the inertial frame.

3.2.2 Generalized coordinates

To describe the behavior of a rigid body considering position and orientation, we need to a common reference frame that is usually the *inertial frame*. To this end, we introduce another rotational matrix W which converts the coordinates describing the rotation of the rigid body with respect to the body frame in coordinates describing the rotation with respect to the inertial frame.

This transformation between the coordinates in the body frame, called **local coordinates**, and the coordinates in the inertial frame, called **generalized coordinates** is essential because it is a fundamental concept of the *Lagrangian and Hamiltonian mechanics*.

Although most of the works in robotics use local coordinates to describe the motion of robots, in this thesis we will use the generalized coordinates because they establish a direct correspondence with the geometrical spaces that are the essential elements of the methods presented in Chapter 4 and Chapter 5

Definition 3.1. *A **free mechanical system** [17] is a collection* $\{P_\alpha\}_{\alpha \in 1,\ldots,N_p} \cup \{B_\beta\}_{\beta \in 1,\ldots,N_b}$ *of N_p particles and N_b rigid bodies. The possible positions of all particles and bodies are described by the set*

$$Q_{free} = ((SO(3) \times \mathbb{R}^3) \times \ldots \times (SO(3) \times \mathbb{R}^3)) \times \mathbb{R}^3 \ldots \mathbb{R}^3 \tag{3.2}$$

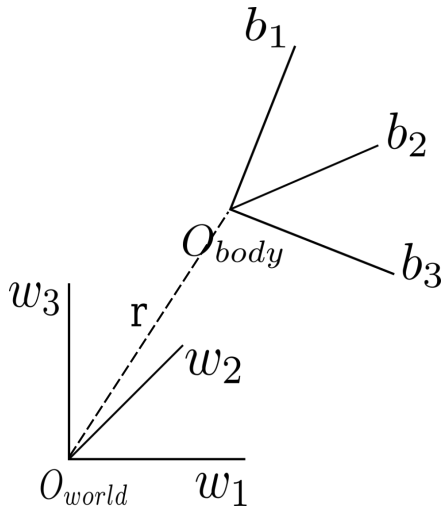

Figure 3-2: Inertial and body frames

Definition 3.3. *An **interconnected mechanical system** is a collection* $\{P_\alpha\}_{\alpha\in 1,\dots,N_p} \cup \{B_\beta\}_{\beta\in 1,\dots,N_b}$ *of* N_p *particles and* N_b *rigid bodies restricted to move on a **submanifold** (see definition in Appendix A)* Q *of* Q_{free}.

The submanifold $Q \subset Q_{free}$ is called **configuration manifold**, which is the geometrical space used to represent a simple mechanical system. This definition contains all the variables used to describe the position and the orientation of a rigid body and, in general, it is used to describe the kinematics and the dynamics of every kind of robots.

If we describe a generic robot using a specific configuration manifold Q, we are interested in finding how many links or joints we can move. We say that if the dimension $dim(Q)$ is equal to n, we have **n-degrees of freedom**.

3.2.3 Example: the two link manipulator

To better understand the role of the Configuration manifold to describe a mechanical system, we now present a simple example: the planar two-link robot. In this example, we consider a robot composed of two joints laying on the same plane. The first link J_1 is connected to a fixed base B and the second link J_2 is connected to the tip of the first link. Finally, we consider a point P placed on the tip of the second link as shown in figure 3-3. Our goal is to know the position of the point P considering the angular position of the joints J_1, J_2.

Starting from an initial inertial frame $\{O_{world}, w_1, w_2, w_3\}$, we want to identify the position (x, y) of P knowing the value of the joints J_1, J_2. As shown in figure 3-3, we can choose different *generalized coordinates*, in our example we can use either the angular position of the joints relative to the previous joints (local coordinates)

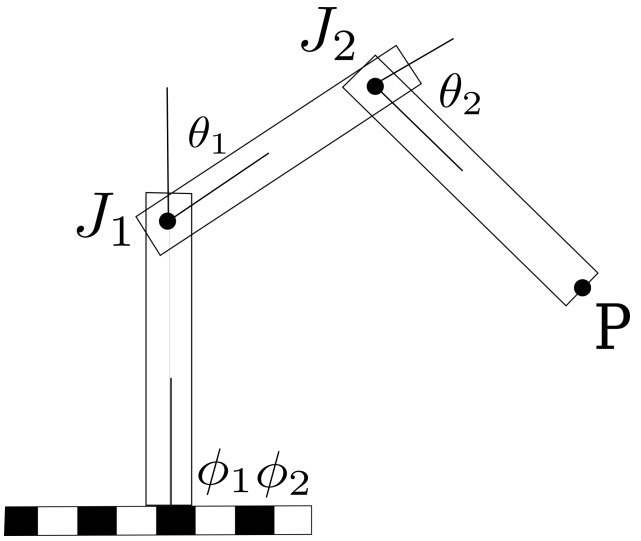

Figure 3-3: A simple model of two-link manipulator

or relative to the inertial frame (absolute coordinates). In this example we identify with (θ_1, θ_2) the relative coordinates and with (ϕ_1, ϕ_2) the absolute ones.

Based on the previous description we can represent the position (x,y) of the joint P, using the relative coordinates $(\theta_1.\theta_2)$ as

$$\begin{aligned} x &= r_1 \sin(\theta_1) + x_p sin(\theta_1 + \theta_2) + y_p \cos(\theta_1 + \theta_2) \\ y &= -r1 \cos(\theta_1) - x_p cos(\theta_1 + \theta_2) + y_p \sin(\theta_1 + \theta_2) \end{aligned} \tag{3.4}$$

and, if the choose the absolute coordinates, we obtain:

$$\begin{aligned} x &= r_1 \sin(\phi_1) + x_p sin(\phi_2) + y_p \cos(\phi_2) \\ y &= -r1 \cos(\phi_1) - x_p cos(\phi_2) + y_p \sin(\phi_2) \end{aligned} \tag{3.5}$$

In this example it is interesting to see that, in both cases, we are in the presence of a *configuration space* that lives in a configuration manifold $Q = S^1 \times S^1$. In this particular example we can visualize this configuration manifold as a geometrical space called "torus" which is possible to see in figure 3.2.3

Depending on the choice of local coordinates, the equations 3.4 and 3.5 are, from a geometrical point of view, two maps that, starting from the configuration manifold, return the desired position so $Q = S \times S \mapsto \mathbb{R}^2$.

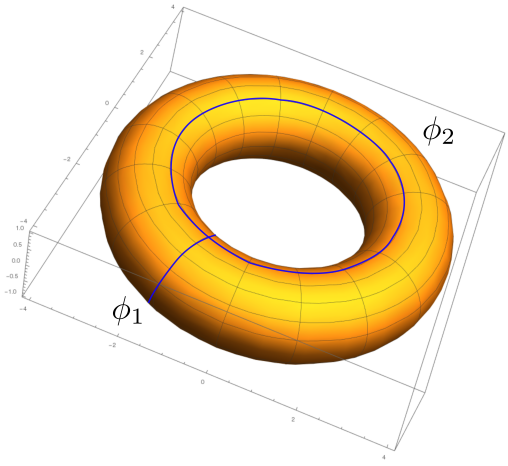

Figure 3-4: Visualization of the configuration manifold $Q = S^1 \times S^1$

3.2.4 Lagrangian function

When we know how to locate a point in space, and we can measure the distance between the point and a reference frame, we are ready to consider how to describe the motion of a particle or the rigid body. Given a configuration space Q with generalized coordinates $q^i, i = 1 \dots n$, we define a function $L(q^i, \dot{q^i})$ called *Lagrangian*. that returns real values and describes the difference between the energies acting on the system. We can write, in-fact, the Lagrangian as the difference between the *potential energy* $V(q, \dot{q})$ and the *kinetic energy* $K(q, \dot{q})$ of the systems $L(q, \dot{q}) = V(q, \dot{q}) - K(q, \dot{q})$.

We want to describe the motion of a rigid body using the previous given Lagrangian function and, to do that, we use the *Hamilton's principle* coming from analytical mechanics [28]:

Theorem 3.6. *The motion of a system from time t_1 to time t_2 is such that the line integral*

$$I = \int_{t_1}^{t_2} L dt \tag{3.7}$$

where $L = V - K$ has a stationary value for the correct path of the motion. The quantity I is also known as action *or* action integral

This principle describes the dynamics of a system as a *variational problem* of a single function, in our case the Lagrangian. If we apply Hamilton's principle, we obtain that a trajectory $q(t)$ is has to follow the *Euler-Lagrange equation*:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}^i} - \frac{\partial L}{\partial q^i} = 0 \tag{3.8}$$

The previous set of equations, one for each degree of freedom, is fundamental

and allows us to describe the free dynamic evolution of a mechanical system. In this case, the dynamics evolves without the presence of external forces and in the next Section we show how it is possible also to include forces and torques.

3.2.5 Lagrangian with external forces and torques

If we want to plan the motion of a rigid body, we need to apply a control input which drives a robot to a certain point in the state space. We are interested to analyze our systems using geometric mechanics so it is convenient to use force and torque controls. It is important to notice that the Lagrangian framework is based on the *tangent space* TQ and the Hamiltonian framework, that we will present in the next Section, is more appropriate to study the behavior of the forces and the constraints because they are components of the co-vector field that lives in the *co-tangent space* T^*Q

At this time, we just introduce the forces in the Euler-Lagrange equations as:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}^i} - \frac{\partial L}{\partial q^i} = F^i + u^i \tag{3.9}$$

where F^i denotes the external forces and u^i the control input given to execute the desired trajectory.

3.2.6 Example: the simple hovercraft

In this Section, we use the *simple hovercraft* example to show how it is possible to describe the dynamic evolution of a mechanical system using Lagrange equations 3.9. We consider a model of hovercraft as shown in figure 3.2.6

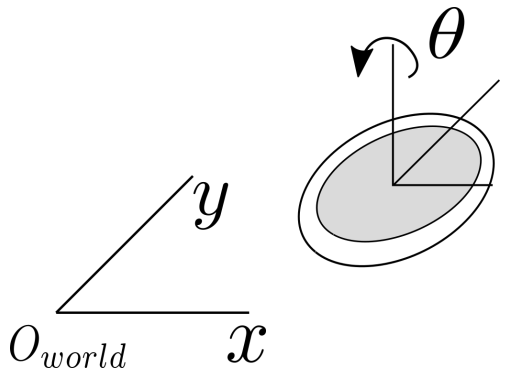

Figure 3-5: A simple model of hovercraft

The hovercraft floats using a constant force that we do not consider, and it can move in a planar space (x,y) and rotate of a certain angle θ around its center of mass. We define its configuration space as $q_H = (x,y,\theta)$ that can be seen, from

a geometrical point of view, as a point of a configuration manifold $Q = \mathbb{R}^2 \times \mathbb{S}^1$. We also consider a set of forces (u_x, u_y, u_θ) that can drive the hovercraft along all degrees of freedom.

In the model of this system, there is no potential energy and the Lagrangian can be written as the function

$$L(q_H, \dot{q_H}) = \frac{1}{2}(m\dot{x}^2 + m\dot{y}^2 + J\dot{\theta}^2) \tag{3.10}$$

recalling Lagrange's equations

$$\begin{aligned} \frac{d}{dt}\frac{\partial L}{\partial \dot{x}^i} - \frac{\partial L}{\partial x} &= F_x + u_x \\ \frac{d}{dt}\frac{\partial L}{\partial \dot{y}^i} - \frac{\partial L}{\partial y} &= F_y + u_y \\ \frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}^i} - \frac{\partial L}{\partial \theta} &= F_u + u_\theta \end{aligned} \tag{3.11}$$

considering that there are not external forces $F = \{F_x, F_y, F_\theta\}$ and substituting the Lagrangian $L(q_H, \dot{q_H})$ 3.10, we obtain the three equations that describe the dynamics of the hovercraft:

$$\begin{aligned} m\ddot{x} &= u_x \\ m\ddot{y} &= u_y \\ J\ddot{\theta} &= u_\theta \end{aligned} \tag{3.12}$$

3.3 Underactuated mechanical system

In the previous Section, we proposed two approaches to describe the dynamic evolution of mechanical systems. Most robots, in the real world, have to deal with *constraints* that affect the motion and that must be considered to generate a possible trajectory. In general, a mechanical system is affected by the constraints on position and velocity. If a robot, for example, is moving on a planar surface and finds an obstacle on its way, from a geometric view we are in the presence of a *holonomic constraint* that is a constraint on the configuration manifold. In the next Sections, we also see constraints on velocities and we will try to give some properties of under-actuated mechanical systems to understand how it is possible to plan a trajectory for this kind of systems.

The second type of constraint are called *non-holonomic constraint* and we will present in details in the next Section. If we work with under-actuated mechanical systems, we have other constraints.

3.3.1 Nonholonomic constraints

The existence of obstacle constraints implies the presence of constraints also on the tangent space. To explain this concept, we can think that its Cartesian space composes the configuration space of a robot. If there is an obstacle in the space of positions, it is natural that there will also be constraints in the space of velocities. From a geometrical point of view, velocities live in the space called *tangent-space*, the reader has an intuitive idea of this particular space looking at Figure 3.3.1 which represents the tangent space of the configuration space of the "two-link manipulator" shown at the beginning of this Chapter.

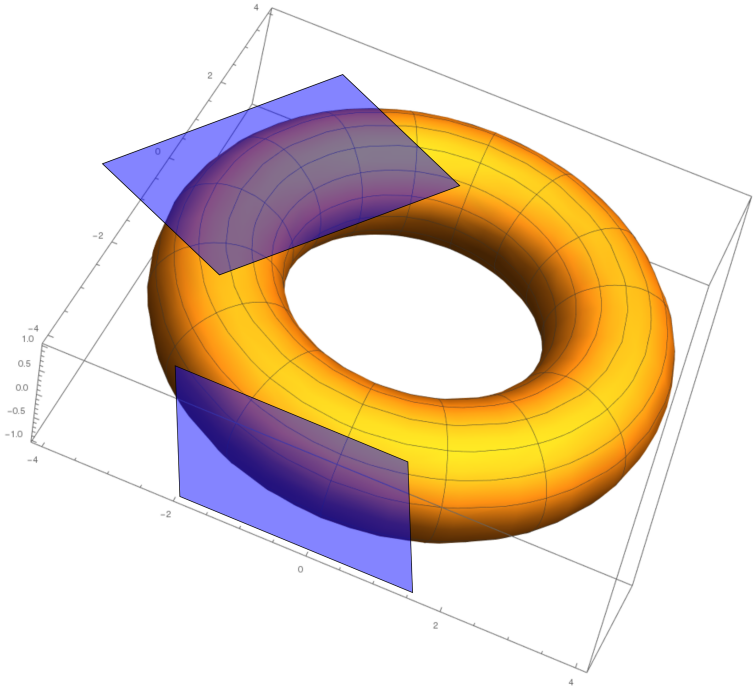

Figure 3-6: A graphical visualization of the tangent space of the two-link manipulator configuration space

From a geometrical point of view, velocity constraints generate a distribution (for definition see Appendix A) on the tangent space and are called, from a definition given by Heinrich Hertz in 1984, *Nonholonomic constraints*. If it is true that a holonomic constraint implies the presence of a nonholonomic constraint, as shown before, it is not true the opposite. In order to formalize this concept, we refer to [8] and we let a mechanical system have n degrees of freedom. We can now write $m < n$ velocity constraints $a_k(q_i)$ such that

$$\sum_{k=1}^{n} a_k^j(q_i)\dot{q}^k = 0 \tag{3.13}$$

where $j = 1, \ldots, m$. If the same system has also m position constraints that we can write as $b^j(q^i) = 0$, we can write their time derivatives

$$\sum_{k=1}^{n} \frac{\partial b^j}{\partial q^k} \dot{q}^k = 0 \tag{3.14}$$

At this point we have to possibilities, if the constraint distribution generated by equation 3.13 is the same of the one generated by 3.14, we are in presence of *holonomic constraints*, otherwise we could say that we are in the presence of *nonholonomic constraints*.

This concept has substantial implications in trajectory generation because if finding a trajectory for a *holonomic mechanical systems* (a system with holonomic constraints) is relatively simple, as we will show in the next Chapter, dealing with the motion of an *nonholonomic mechanical system* is significantly more complicated.

3.3.2 Lagrange-D'Alemebert equations

In Section 3.2.4 we introduced Hamilton's principle to derive the Lagrangian equations, and in the previous Section we saw that a particular class of mechanical systems, called *Nonholonomic mechanical systems* (NMS), have nonholonomic constraints that could interfere with the dynamics. It is possible to extend the Hamilton's principle (we leave the proof to Section 2-4 of reference [28]) to describe also the dynamic evolution of NMS using the *Lagrange-D'Alembert equation*

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = \sum_{j=1}^{m} \lambda_j a_i^j(q_i) \tag{3.15}$$

where $i = 1, \ldots, n$. The variable λ, are the *Lagrange multiplier* and $a(q_i)$ are functions describing the nonholonomic constraints. These numerical variables modify the effect of the nonholonomic constraints $a_i(q_i)$. In the next example we apply the Lagrange-D'Alembert equations and we write the equations of the nonholonomic constraints.

3.3.3 Example: The falling rolling disk

In order to understand the nonholonomic constraints, a classical example [20] [8] is the "falling rolling disk". Let we consider a disk with mass m and radius R rotating around its center of mass of a quantity θ. We also assume that there is a gravity force $F_g = m \cdot g$ acting on the center of mass. Let also J and I the moment of inertia with respect to the x and y axes. We can with respect to write the Lagrangian equation as the difference between the kinetic energy T and the potential energy V:

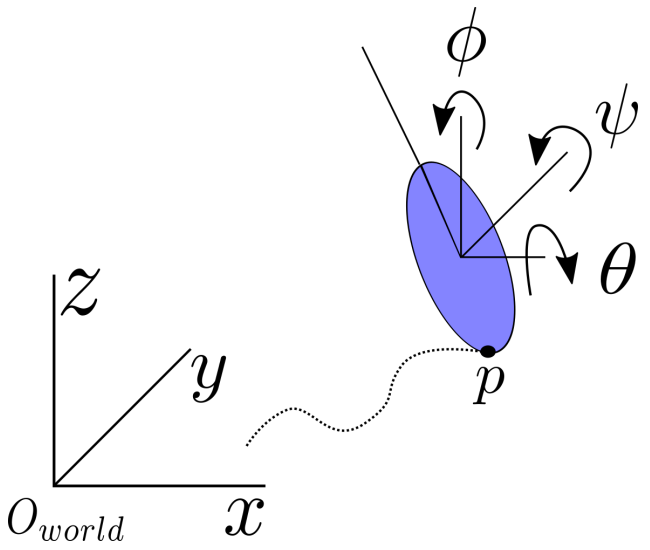

Figure 3-7: The common "falling rolling disk" example

$$
\begin{aligned}
T &= \frac{1}{2}m(\dot{x}^2+\dot{y}^2+R\dot{\psi}+R^2\dot{\phi}^2\sin^2(\psi)-mR(\psi^2\cos(\psi)(\dot{x}\sin(\phi)-\dot{y}\cos(\phi)) \\
&\quad +\dot{\phi}\sin(\psi)(\dot{x}\cos(\phi)+\dot{y}\sin(\phi))+\frac{1}{2}I(\dot{\psi}^2+\cos^2(\psi)) \\
&\quad +\frac{1}{2}J(\dot{\theta}+\dot{\phi}\sin(\psi))^2 \\
V &= mgRcos(\psi)
\end{aligned}
\tag{3.16}
$$

If we impose the condition that the point of contact between the disk and the terrain is not slipping, we are imposing the nonholonomic constraint

$$\dot{x}=0 \tag{3.17}$$

So we can write the two constraints a_1 and a_2 as

$$
\begin{aligned}
a_1 &= \dot{x}-(R\cos(\phi))\dot{\theta} \\
a_2 &= \dot{y}-(R\sin(\phi))\dot{\theta}
\end{aligned}
\tag{3.18}
$$

From a geometric point of view, the configuration manifold is $Q=\mathbb{R}^2\times\mathbb{S}^1\times\mathbb{S}^1\times\mathbb{S}^1$ that is the space representing the two translation variables (x,y) and the three angles (θ,ϕ,ψ). As described in the previous Section, the nonholonomic constraints generate a distribution on the tangent space TQ. This geometrical space represent a "zone" where we can search for an admissible trajectory, respecting the kinematic properties of the system. In this example the distribution is generated by the

In particular, the distribution is generated by a set of vectors

$$\mathbb{D} = \text{span}\left\{ \frac{\partial}{\partial \theta}, \frac{\partial}{\partial \phi}, R\cos(\phi)\frac{\partial}{\partial x} + R\sin\phi\frac{\partial}{\partial y} + \frac{\partial}{\partial \psi} \right\} \tag{3.19}$$

3.3.4 Underactuated mechanical systems

As discussed earlier, we refer to mechanical systems that have a force as the control input. In robotics and control theory, there is a special class of mechanical system defined as:

Definition 3.20. *Underactuated mechanical system (UMS) is a mechanical system where the number of degrees of freedom actuated m is less than the number n of available DoF.*

If we are working with UMS, the consequence is the presence of r acceleration constraints where $r = m - n$.

It is not simple to describe underactuated mechanical systems because they have a strong nonlinear dynamic evolution and there is not yet a general approach to model and control this kind of systems. Some authors tried to exploit the specific properties of specific UMS, for examaple K. Lynch in [61] propose to classify UMS in three categories:

1. Pure kinematic: if we are in the presence of an underactuated kinematic system and the control inputs are velocities. An example of this kind of system is the falling rolling disc presented in the previous Section.

2. Pure mechanical: if we have a UMS without nonholonomic constraints, such as a 3-links manipulator with a passive joint

3. Mixed kinematic and mechanical: if the UMS systems have nonholonomic constraints and also accelerations constraints. A common example is the *Snakeboard* [71] that is a particular skateboard where the input is given as two torques but the result is a translational movement.

Another interesting concept presented in [61], is that we can write the equation of motion as

$$\dot{q} = f(q) + \sum_{i=1}^{m} u_i g_i(q) \text{ where } m < n \tag{3.21}$$

where $f(q)$ is a *drift vector field* describing the free evolution of the system, $g_i(q)$ are *control vector fields* which describe how the controls, in case of UMS forces, influence the dynamic of the system and $u = [u_1, \ldots, u_m]$ is the control vector. In the case of UMS, we have $r = n - m$ acceleration constraints.

It is important also to point out that the works presented in literature assume *Drift-less UMS* which means that $f(q) = 0$. In this thesis, we are interested also

to consider underactuated mechanical systems that have a non-negative drift vector field so $f(q) \neq 0$.

We follow the Lagrangian models for UMS presented in [3], [77] and [88], to write the equations of motion of an UMS. Let $q^A := (q^a, q^\mu)$ the configuration of the system where where $q^a = [q_1, \ldots, q_m]$ are the actuated degrees of freedom, $q^\mu = [q_m + 1, \ldots, q_n]$ are the non-actuated degrees of freedom and $dim(q^A) = n$. In the case of drift-less UMS, we can write the Lagrangian equations of motion as

$$\begin{aligned} \frac{d}{dt}\frac{\partial L}{\partial \dot{q}^a} - \frac{\partial L}{\partial q^a} &= u \\ \frac{d}{dt}\frac{\partial L}{\partial \dot{q}^\mu} - \frac{\partial L}{\partial q^\mu} &= 0 \end{aligned} \tag{3.22}$$

From a geometrical point of view, the configuration manifold Q is composed by two sub-manifolds $Q = Q_1 \times Q_2$ where $q^a \in Q_1$ and $q^\mu \in Q_2$ so the Lagrangian L is a function $L : TQ := TQ_1 \times TQ_2 -> \mathbb{R}$. It is interesting to notice that the second equation could be seen as function which represent the acceleration constraints affecting UMS, in particular in [3] these are called *second-order non-holonomic constraints*.

The previous equations are not valid in general but only for a specific type of underactuated mechanical systems. If we consider underactuated robots such as quadrotors, submarines or spacecraft, in-fact, we are not able to describe the dynamic evolution of the system with the previous equations. To understand this fact, we have to refer to references [84] and [91] where there is a definition of *Super-articulated mechanical system*

3.3.5 Super-articulated mechanical systems

Definition 3.23. *A Super-articulated mechanical system is a system where some of the configuration variables (or degree of freedom) are directly controlled while the remaining variables evolve under the dynamic influence of the controlled degrees of freedom*

This definition introduces an important concept that inspires our work, despite the previous definition of the under-actuated system, here we see that there is the possibility of moving the non-actuated degrees of freedoms q^μ exploiting the dynamics of the system. Moreover, the concept of the super-articulated mechanical system introduced by Ballieul in [91], gives also a condition that the UMS must satisfy to deal with acceleration constraints. If we represent a Lagrangian function as $L = \frac{1}{2}\dot{q}^T M \dot{q} - V(q)$, the inertia matrix M is partitioned corresponding to (q^a, q^μ), we can write

$$M = \begin{bmatrix} \mathscr{N} & \mathscr{A} \\ \mathscr{A}^T & \mathscr{M} \end{bmatrix} \tag{3.24}$$

where $\mathscr{N}$, $\mathscr{M}$ and $\mathscr{A}$ are a invertible matrices. Looking at the matrix $\mathscr{N} = \mathscr{N}(q^a, q^\mu)$, a fundamental property arises:

Proposition 3.25. *If $\mathscr{N}(q^a, q^\mu)$ is an invertible matrix, we have an one-one correspondence between the control trajectory $u(t)$ and the trajectories of the actuated variables q^a.*

It the UMS considerate does not satisfy the previous property, cannot be classified as "super-articulated mechanical systems" and it is not possible to apply the method presented in the next Chapter.

3.4 Hamiltonian systems

As explained in Chapter 7.2 of [8], to find and optimized trajectory using a geometric approach could be done with the Hamiltonian framework. The difference between the Hamiltonian and the Lagrangian framework is how it is possible to describe the trajectories of a mechanical system. In the optimal control setting, in particular, applying the "Pontryagin Maximum principle," the trajectories are influenced by the controlled co-vector field whereas in the Lagrangian framework trajectories are just "constrained," as shown in Section 3.3.2. Moreover, following the definition of Under-actuated mechanical system given in the previous Section, the second-order non-holonomic constraints are vector fields that live in the second-order tangent space, but it is not possible to use them to generate a trajectory as in the first-order non-holonomic case (for example in [81]). The idea that inspires the method presented in Chapter 5, is that the UMS constraints live in the cotangent space, which is a geometrical space that could be used only working with the Hamiltonian framework. We are still not able to demonstrate this assertion, but the methods presented in Chapter 4 and 5 suggest us that this way could give good results in future works.

Also from a computational point of view, using the Hamiltonian we are able to search an optimal trajectory on the state space, considering also the energy spend by the system as presented in Chapter 5

In the Lagrangian framework, if we have n degrees of freedom, we need n equations of motion of the form

$$L(q_H, \dot{q_H}) = \frac{1}{2}(m\dot{x}^2 + m\dot{y}^2 + J\dot{\theta}^2) \tag{3.26}$$

To find a solution, we need $2n$ initial values because we are in the presence of second-order equations. In the Hamiltonian framework instead, we can describe the system using a set of $2n$ independent first-order equations. In this way we are representing the evolution of a point in the configuration space using the *phase space* defined as $(q, \dot{q})$.

3.4.1 Legendre transform

To find the Hamiltonian equations we apply the *Legendre transform* that allows to pass from the Lagrangian framework to the Hamiltonian equations.

Definition 3.27. *Starting from the Lagrangian function, we can define the conjugate momenta as*

$$p_1 = \frac{\partial L(q_l, \dot{q}_l, t)}{\partial \dot{q}_i} \tag{3.28}$$

where l and i are integer numbers with a value which is smaller than the degree of freedom of the system. Starting from the equation 3.28, the role of the *Legendre transform* is to change the variables of our mechanical system from $(q, \dot{q}, t)$ to (q, p, t).

For a proof of the Legendre transform please refer to [4], for our need we are interested only in a special function, called *Hamiltonian*, which relates the Lagrangian to the conjugate momenta:

$$H(q, p, t) = \dot{q}_i p_i - L(q, \dot{q}, t) \tag{3.29}$$

if we compute the differential of the Hamiltonian, that reads

$$dH = \frac{\partial H}{\partial q_i} dq_i + \frac{\partial H}{\partial p_i} dp_i + \frac{\partial H}{\partial t_i} dt_i \tag{3.30}$$

If we replace equation 3.29 in 3.30, we can write

$$dH = \dot{q}_i dp_i + p_i d\dot{q}_i - \frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} dq_i - \frac{\partial L}{\partial t} dt \tag{3.31}$$

By recalling the definition of momenta and Lagrange equations we can reduce the previous equation to:

$$dH = \dot{q}_i dp_i - \dot{p}_i dq_i - \frac{\partial L}{\partial t} dt \tag{3.32}$$

Finally, we can split equation 3.32 to obtain the *canonical Hamilton equations*, which are equivalent to the Lagrange equations presented in Section 3.2:

$$\begin{aligned} \dot{q}_i &= \frac{\partial H}{\partial p_i} \\ \dot{p}_i &= -\frac{\partial H}{\partial q_i} \end{aligned} \tag{3.33}$$

3.5 Conclusions

In this Chapter, we present an overview of the fundamental concepts used to describe the characteristics and limitations of the Under-actuated Mechanical Sys-

tems (UMS). Starting from basic geometric mechanics definitions, we introduced the Hamiltonian formalism from a geometric point of view. Hamiltonian equations are the fundamental tools to apply the methods proposed in the next Chapters. We discovered that it is not possible to apply the "Variational nonholonomic approach", proposed in the next Chapter, to all UMS so in this Chapter we show the properties of a sub-class of UMS, the *Super-articulated mechanical systems*. Another important concept presented in this Chapter is the possibility of using geometric spaces to solve an optimal control problem. In this way, it is possible to find particular equations of motion for Super-articulated systems and finding solutions applying numerical methods, as presented in the next Chapter.

Chapter 4

Variational constrained problem for a class of UMS

4.1 Introduction

In the previous Chapters, we presented a few tools from geometric mechanics useful to plan an optimal trajectory for underactuated mechanical systems. In this Chapter, we apply these concepts to plan a trajectory for under-actuated mechanical systems using the *variational constrained system approach* [2]. The benefit of this method is to "respect" the geometric structure of the system. In other words, it is possible to use the UMS constraints to define a distribution in the cotangent space and computing feaisible trajectories. Projecting this geometric structures in local coordinates allows applying numeric algorithms to compute local-optimal solutions [11]. The basic concepts of differential geometry are defined in Appendix A.

4.2 Problem statement

In Chapter 3 we presented a particular class of UMS called Super-articulated mechanical system. For these robots it is possible to define a configuration space $Q = Q_1 \times Q_2$ as the Cartesian product of two differentiable manifolds, Q_1 on which forces are applied, and Q_2 on which the dynamics evolve freely. To apply numerical algorithms to plan a trajectory, we need to project the geometric structures in local coordinates. If (q^a), $a = 1,\dots,r$ are local coordinates on Q_1, and (q^μ), $\mu = r+1,\dots,n$ are local coordinates on Q_2, composing the two sets of local coordinates, we obtain $q^A := (q^a, q^\mu)$, with $a = 1,\dots,r$ and $\mu = r+1,\dots,n$, which are the local coordinates on Q

This mapping allows to describe the dynamic evolution of a mechanical system using the Lagrangian function $L : TQ := TQ_1 \times TQ_2 \to \mathbb{R}$. In this case, external control forces are applied only to coordinates living in Q_1 so we can write the

Euler-Lagrange equations of motion as

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}^a}\right) - \frac{\partial L}{\partial q^a} = u^a \tag{4.1}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}^\mu}\right) - \frac{\partial L}{\partial q^\mu} = 0 \tag{4.2}$$

with $a = 1,\ldots,r$ and $\mu = r+1,\ldots,n$, and where u^a, $a = 1,\ldots,r$, are the external forces or control inputs.

To generate an optimal trajectory that drives an UMS from an initial point to a target point, we want to connects the initial configuration $(q^A(t_0), \dot{q}^A(t_0))$ to final configuration $(q^A(t), u^a(t))$. To choose one solution in the set of the all possible solutions, we minimize the functional

$$\mathscr{A}(q(\cdot), u(\cdot)) = \int_0^{t_f} C\left(q^a(t), q^\mu(t), \dot{q}^a(t), \dot{q}^\mu(t), u^a(t)\right)\, dt\,. \tag{4.3}$$

where $C()$ is a generic cost function.

4.3 Variational constrained control problem

Variational constrained problems are equivalent to optimal control problems but are based on geometrical structures. The advantage of this approach is the possibility to "incorporate" constraints inside the geometrical structure and not just "adjoining" them as in the case of the classical optimization techniques that use the Lagrangian multipliers. In particular, methods presented in Chapter 2 do not consider explicitly the cotangent space, which is the appropriate geometrical structure where we can plan a trajectory for UMS respecting the constraints caused by the under-actuation. In the following Sections, we apply the "Variational constrained systems problem" to plan an optimal trajectory that is not possible to compute with the standard methods. thus making clearer the advantages of the underlining geometric structure.

For proofs and details, please refer to [8].

The key point of the *variational constrained systems problem* [2], is the equivalence between minimize the functional 4.3 and the following cost function:

$$\tilde{\mathscr{A}}(q(\cdot)) = \int_0^{t_f} L(q^a(t), q^\mu(t), \dot{q}^a(t), \dot{q}^\mu(t), \ddot{q}^a(t), \ddot{q}^\mu(t))dt \tag{4.4}$$

subject to the constraints

$$\Phi^\mu(q^a(t), q^\mu(t), \dot{q}^a(t), \dot{q}^\mu(t), \ddot{q}^a(t)) := \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}^\mu}\right) - \frac{\partial L}{\partial q^\mu} = 0 \tag{4.5}$$

and to the boundary conditions.

It is important to notice that we can substitute the control input $u^a(t)$, used in the usual notion of cost functional, with the equivalent Lagrangian equation $\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}^a}\right) - \frac{\partial L}{\partial q^a}$.

The result is the Lagrangian function $\tilde{L} : T^2Q \to \mathbb{R}$ is defined on the second tangent space T^2Q by the equation

$$\begin{aligned} &\tilde{L}(q^a(t), q^\mu(t), \dot{q}^a(t), \dot{q}^\mu(t), \ddot{q}^a(t), \ddot{q}^\mu(t)) := \\ &C\left(q^a(t), q^\mu(t), \dot{q}^a(t), \dot{q}^\mu(t), \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}^a}\right) - \frac{\partial L}{\partial q^a}\right). \end{aligned} \tag{4.6}$$

Observing that the cost functional $\tilde{\mathscr{A}}$ is now independent of the controls $u(\cdot)$, the computation of the control problem will not give the control trajectory but only the optimal trajectory in the configuration space. If we are interested in the optimal control input trajectory, we need to evaluate equations (4.1) substituting optimal configuration variables $\hat{q}^a$ to compute the optimal controls $\hat{u}^a$.

According to the theory presented by Colombo et al. in [19], the dynamics of the higher–order constrained variational problem is determined by a pre–symplectic Hamiltonian system on a suitable fiber bundle W_0 over TQ. In the following, we recall the basic constructions of the fundamental geometric tools and the motion equations.

As described in the introduction, the main characteristic of this method is that is possible to use the explicit constraint function Φ^μ defined by equations (4.5) to represent the role of the acceleration constraints. In particular, Φ^μ generates a sub-manifold $\mathscr{M} \subset T^2Q$ where the admissible acceleration lives.

Recalling the Lagrangian equation describing the dynamic evolution of the non-actuated degrees

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}^\mu}\right) - \frac{\partial L}{\partial q^\mu} = 0 \tag{4.7}$$

and assuming the condition that the matrix $(W_{\mu\nu})$, $r+1 \le \mu, \nu \le n$, with coefficients given by

$$W_{\mu\nu} := \frac{\partial^2 L}{\partial \dot{q}^\mu \partial \dot{q}^\nu}$$

is non singular, then we can write the accelerations of the non-actuated degrees of freedom as

$$\ddot{q}^\mu = W^{\mu\nu} F_\nu(q^A, \dot{q}^A, \ddot{q}^a) =: G^\mu(q^A, \dot{q}^A, \ddot{q}^a), \tag{4.8}$$

where $(W^{\mu\nu})$ denotes the inverse of the matrix $(W_{\mu\nu})$ and

$$F_\nu(q^A, \dot{q}^A, \ddot{q}^a) = \frac{\partial^2 L}{\partial \dot{q}^a \partial \dot{q}^\nu}\ddot{q}^a + \frac{\partial^2 L}{\partial q^A \partial \dot{q}^\nu}\dot{q}^A - \frac{\partial L}{\partial q^\nu}.$$

Therefore $(q^A, \dot{q}^A, \ddot{q}^a)$, $A = 1, \ldots, n$ and $a = 1, \ldots, r$, defines local coordinates on $\mathscr{M}$. The previous assumption that the matrix $W_{\mu\nu}$ is nonsingular, is verified if L is a Lagrangian of the mechanical type such as we write it as the difference between the kinematic and potential energies

At this point, we have the correct geometrical space, which is the sub-manifold $\mathscr{M}$ of the second tangent space T^2Q and it is possible to define the restricted Lagrangian $\tilde{L}_{\mathscr{M}} := \tilde{L}|_{\mathscr{M}}$ which describes the energy evolution of a generic under-actuated mechanical system.

We need another step to put the UMS constraints in the correct geometric space. Skinner and Rusk in ([85]) proposed a formalism to define the appropriate geometrical space, summarized in figure 4-1, where we can search for a solution. Let $W_0 = T^*(TQ) \times_{TQ} \mathscr{M}$ be a fiber product over TQ, locally described by coordinates $(q^A, \dot{q}^A, p_A^0, p_A^1, \ddot{q}^a)$. The coordinates p_A^0 and p_A^1 are the conjugate momenta of q^A and $\dot{q}^A$, respectively. p_A^0 are the co-state on the tangent space that has the same role of the "Lagrange multipliers" in the Lagrangian frameworks. p_A^1 are other co-state, which are related to the evolution of the constraints given by the under-actuation.

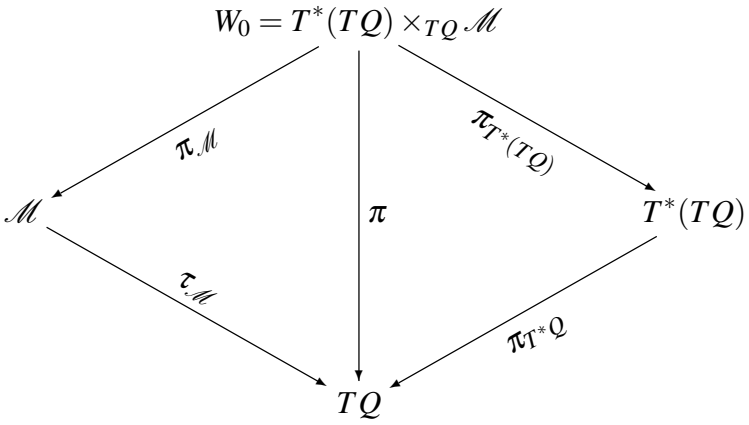

Figure 4-1: Skinner-Rusk formalism

In this case, the appropriate formalism to find a solution to our problem is the Hamiltonian formalism presented in Chapter 3. Let $\Omega_{W_0} = \pi_1^*(\omega_{TQ})$ be the pull-back on W_0 of the standard 2–form ω_{TQ} of TQ and $H_{W_0}(\alpha_x, v_x) := \langle \alpha_x, \imath_{\mathscr{M}}(v_x) \rangle - \tilde{L}_{\mathscr{M}}(v_x)$ the Hamiltonian on W_0, where $x \in TQ$, $v_x \in \mathscr{M}_x = \tau_{\mathscr{M}}^{-1}(x)$, $\alpha_x \in T_x^*TQ$ and $\langle \cdot, \cdot \rangle$ denotes the standard pairing of forms with vectors.

We are now ready to use the Hamiltonian formalism to find solutions to our problem. See Appendix A for details on how to compute solutions using the Hamiltonian framework. We can better understand the previous constructions using local coordinates, the 2–form Ω_{W_0} reads

$$\Omega_{W_0} = dq^A \wedge dp_A^0 + d\dot{q}^A \wedge dp_A^1 \tag{4.9}$$

and the Hamiltonian is

$$H_{W_0} = p^0_A \dot{q}^A + p^1_a \ddot{q}^a + p^1_\mu G^\mu(q^A, \dot{q}^A, \ddot{q}^a) - \tilde{L}_{\mathscr{M}}(q^A, \dot{q}^A, \ddot{q}^a) \tag{4.10}$$

The equations of motion of our constrained variational problem are Hamilton equations for H_{W_0}:

$$i_{X_{H_{W_0}}} \Omega_{W_0} = dH_{W_0}\,, \tag{4.11}$$

where $i_X\Omega$ denotes the contraction of the vector field X with the differential form Ω.

By construction, the 2–form Ω_{W_0} is a pre–symplectic 2–form, i.e. it is a closed, possibly degenerate, 2–form. We can simply understand this fact looking at the local coordinates where $\ddot{q}^a$ are not present the local representation (4.9) of Ω_{W_0}, moreover, its kernel is locally represented by

$$\ker \Omega_{W_0} = \mathrm{span}_{\mathbb{R}} \left(\frac{\partial}{\partial \ddot{q}^a} \right) \tag{4.12}$$

This is a pre-sympletic form but Hamiltonian functions live on a symplectic manifold. Tothis problem, we need to build another constraint, using the Gotay-Nester-Hinds's algorithm as explained in [36]. To this end, we consider a primary constraint:

$$dH_{W_0} \left(\frac{\partial}{\partial \ddot{q}^a} \right) = 0 \tag{4.13}$$

that in local coordinates is represented by the following constraint function:

$$\varphi^1_a := \frac{\partial H_{W_0}}{\partial \ddot{q}^a} = p^1_a + p^1_\mu \frac{\partial G^\mu}{\partial \ddot{q}^a} - \frac{\partial \tilde{L}_M}{\partial \ddot{q}^a} = 0\,. \tag{4.14}$$

The zero level set of the constraint φ^1_a defines a $4n$–dimensional manifold W_1 equipped with local coordinates $(q^A, \dot{q}^A, \ddot{q}^a, p^0_A, p^1_\mu)$, $A = 1, \ldots, n$, $a = 1, \ldots, r$ and $\mu = r+1, \ldots, n$. Denoting by $\imath_{W_1} : W_1 \longrightarrow W_0$ the canonical inclusion of W_1 in W_0, under some mild condition, namely the matrix $(\mathscr{R}_{ab})$, with coefficients given by

$$\mathscr{R}_{ab} = \frac{\partial^2 \tilde{L}_{\mathscr{M}}}{\partial \ddot{q}^b \partial \ddot{q}^a} - p^1_\mu \frac{\partial^2 G^\mu}{\partial \ddot{q}^b \partial \ddot{q}^a} \tag{4.15}$$

being not singular, the manifold (W_1, Ω_{W_1}) is a symplectic manifold, that is a manifold endowed with a closed and non–degenerate 2–form, where $\Omega_{W_1} := \imath^*_{W_1} \Omega_{W_0}$ is the pull-back of the 2–form Ω_{W_0} to W_1.

We are ready now to compute the *Hamilton equation* (4.11) in local coordinates. Let

$$X = X^{q^A} \frac{\partial}{\partial q^A} + X^{\dot{q}^A} \frac{\partial}{\partial \dot{q}^A} + X^{\ddot{q}^a} \frac{\partial}{\partial \ddot{q}^a} + X^{p^0_A} \frac{\partial}{\partial p^0_A} + X^{p^1_A} \frac{\partial}{\partial p^1_A} \tag{4.16}$$

be the generic vector field on W_0. We contract X with the pre–symplectic form Ω_{W_0}

$$i_X\Omega_{W_0} = X^{q^A}dp^0_A + X^{\dot{q}^A}dp^1_A - X^{p^0_A}dq^A - X^{p^1_A}d\dot{q}^A \tag{4.17}$$

and equating term by term the righthand side of (4.17) with the differential of H_{W_0}, we obtain the coefficients of the Hamiltonian vector field $X_{H_{W_0}}$:

$$\begin{aligned}
&X^{q^A} = \dot{q}^A, \quad X^{\dot{q}^A} = G^\mu + \ddot{q}^a, \\
&X^{p^0_A} = \frac{\partial \tilde{L}_{\mathscr{M}}}{\partial q^A} - p^1_\mu \frac{\partial G^\mu}{\partial q^A}, \quad X^{p^1_A} = \frac{\partial \tilde{L}_{\mathscr{M}}}{\partial \dot{q}^A} - p^0_A - p^1_\mu \frac{\partial G^\mu}{\partial \dot{q}^A}
\end{aligned}$$

Hamilton equation (4.11) in local coordinates reads:

$$\frac{dq^A}{dt} = \dot{q}^A, \qquad \frac{d^2q^a}{dt^2} = \ddot{q}^a \tag{4.18}$$

$$\frac{d^2q^\mu}{dt^2} = G^\mu\left(q^A, \frac{dq^A}{dt}, \frac{d^2q^a}{dt^2}\right) \tag{4.19}$$

$$\frac{dp^0_A}{dt} = \frac{\partial \tilde{L}_{\mathscr{M}}}{\partial q^A} - p^1_\mu \frac{\partial G^\mu}{\partial q^A} \tag{4.20}$$

$$\frac{dp^1_A}{dt} = \frac{\partial \tilde{L}_{\mathscr{M}}}{\partial \dot{q}^A} - p^0_A - p^1_\mu \frac{\partial G^\mu}{\partial \dot{q}^A} \tag{4.21}$$

$$p^1_a = \frac{\partial \tilde{L}_{\mathscr{M}}}{\partial \ddot{q}^a} - p^1_\mu \frac{\partial G^\mu}{\partial \ddot{q}^a} \tag{4.22}$$

Equation (4.22) is a condition on the vanishing of the coefficient of the differential of $\ddot{q}^a$, it defines the primary constraint φ^1_a and then the symplectic manifold W_1. Combining equations (4.21) and (4.22) we obtain an evolution equation for p^1_a:

$$\frac{d}{dt}p^1_a = \frac{d}{dt}\left(\frac{\partial \tilde{L}_{\mathscr{M}}}{\partial \ddot{q}^a} - p^1_\mu \frac{\partial G^\mu}{\partial \ddot{q}^a}\right) = -p^0_a - p^1_\mu \frac{\partial G^\mu}{\partial \dot{q}^a} + \frac{\partial \tilde{L}_{\mathscr{M}}}{d\dot{q}^a}\,.$$

Differentiating with respect to time and substituting the evolution equation (4.20) of p^0_a we obtain

$$\begin{aligned}
&\frac{d^2}{dt^2}\left(\frac{\partial \tilde{L}_{\mathscr{M}}}{\partial \ddot{q}^a} - p^1_\mu \frac{\partial G^\mu}{\partial \ddot{q}^a}\right) + \frac{d}{dt}\left(p^1_\mu \frac{\partial G^\mu}{\partial \dot{q}^a} - \frac{\partial \tilde{L}_{\mathscr{M}}}{\partial \dot{q}^a}\right) + \\
&+ \frac{\partial \tilde{L}_{\mathscr{M}}}{\partial q^a} - p^1_\mu \frac{\partial G^\mu}{\partial q^a} = 0\,.
\end{aligned} \tag{4.23}$$

The same procedure for p^1_μ gives

$$\frac{d^2p^1_\mu}{dt^2} = \frac{d}{dt}\left(\frac{\partial \tilde{L}_{\mathscr{M}}}{\partial \dot{q}^\mu} - p^1_\nu \frac{\partial G^\nu}{\partial \dot{q}^\mu}\right) + p^1_\nu \frac{\partial G^\nu}{\partial q^\mu} - \frac{\partial \tilde{L}_{\mathscr{M}}}{\partial q^\mu}\,. \tag{4.24}$$

We notice that solving equations (4.23) and (4.24) allows to find p^0_μ and p^0_a

using the equation 4.20. From equation 4.20 it is possible to get

$$p^0_\mu = \frac{\partial \tilde{L}_{\mathscr{M}}}{\partial \dot{q}^\mu} - p^1_\nu \frac{\partial G^\nu}{\partial \dot{q}^\mu} - \frac{dp^1_\mu}{dt},$$

combining equation 4.20 and the primary constraint φ^1_a, we can compute the momenta of the actuated degrees of freedom:

$$p^0_a = \frac{\partial \tilde{L}_{\mathscr{M}}}{\partial \dot{q}^a} - p^1_\nu \frac{\partial G^\nu}{\partial \dot{q}^a} - \frac{d}{dt}\left(p^1_\nu \frac{\partial G^\nu}{\partial \ddot{q}^a} - \frac{\partial \tilde{L}_{\mathscr{M}}}{\partial \ddot{q}^q} \right).$$

Finally, working on the manifold W_1 we can put equation 4.24 in normal form obtaining the ordinary differential equations which describe the dynamic evolution of our under-actuated mechanical system:

$$\begin{aligned}
\frac{d^4 q^a}{dt^4} &= \Gamma^a \left(q^A, \dot{q}^A, \ddot{q}^a, \dddot{q}^a, p^1_\mu, \dot{p}^1_\mu \right), \\
\frac{d^2 q^\mu}{dt^2} &= G^\mu \left(q^A, \dot{q}^A \ddot{q}^a \right), \\
\frac{d^2 p^1_\mu}{dt^2} &= \frac{d}{dt}\left(\frac{\partial \tilde{L}_{\mathscr{M}}}{\partial \dot{q}^\mu} - p^1_\nu \frac{\partial G^\nu}{\partial \dot{q}^\mu} \right) - \left(\frac{\partial \tilde{L}_{\mathscr{M}}}{\partial q^\mu} - p^1_\nu \frac{\partial G^\nu}{\partial q^\mu} \right)
\end{aligned} \tag{4.25}$$

where the function Γ^a is

$$\begin{aligned}
\Gamma^a \left(q^A, \dot{q}^A, \ddot{q}^a, \dddot{q}^a, p^1_\mu, \dot{p}^1_\mu \right) := \mathscr{R}^{ab} \Bigg[& \mathscr{H}_b + \frac{d}{dt}\mathscr{F}_b - \frac{d}{dt}\mathscr{L}_b \\
& - \dddot{q}^c \frac{d}{dt}\mathscr{R}_{bc} \Bigg]
\end{aligned}$$

with

$$\begin{aligned}
\mathscr{F}_a &= \frac{\partial \tilde{L}_{\mathscr{M}}}{\partial \dot{q}^a} - p^1_\mu \frac{\partial G^\mu}{\partial \dot{q}^a}, \\
\mathscr{H}_a &= p^1_\mu \frac{\partial G^\mu}{\partial q^a} - \frac{\partial \tilde{L}_{\mathscr{M}}}{\partial q^a} \\
\mathscr{L}_a &= \frac{\partial^2 \tilde{L}_{\mathscr{M}}}{\partial q^A \partial \ddot{q}^a} \dot{q}^A + \frac{\partial^2 \tilde{L}_{\mathscr{M}}}{\partial \dot{q}^b \partial \ddot{q}^a} \ddot{q}^b + \frac{\partial^2 \tilde{L}_{\mathscr{M}}}{\partial \dot{q}^\beta \partial \ddot{q}^a} G^\beta - \dot{p}^1_\mu \frac{\partial G^\mu}{\partial \ddot{q}^a} + \\
&\quad - p^1_\mu \left(\frac{\partial^2 G^\mu}{\partial q^A \partial \ddot{q}^a} \dot{q}^A + \frac{\partial^2 G^\mu}{\partial \dot{q}^b \partial \ddot{q}^a} \ddot{q}^a + \frac{\partial^2 G^\mu}{\partial \dot{q}^\beta \partial \ddot{q}^a} G^\beta \right)
\end{aligned} \tag{4.26}$$

where $(\mathscr{R}^{ab})$ is the inverse matrix of the matrix $(\mathscr{R}_{ab})$ defined in (4.15). Equations 4.25 generate a flow which allows to reconstruct the momenta p^0_A, and, in combination with the constraint equation (4.22) it also allows to generate the flow of the Hamiltonian vector field $X_{H_{W_1}}$.

The method proposed looks complicated but it allows, starting from the description of the Lagrangian function, to generate a set of equations that describe the dynamic evolution of a generic super-articulated mechanical system considering the acceleration constraints.

For practical use of this method, we implemented a Mathematica ®script that allows generating the previous equations of motion starting from the Lagrangian function.

4.4 Trajectory planning

In the previous Section we found a set of equation describing the evolution of the dynamics for generic super-articulated mechanical system. In this Section we use these equations to solve a *trajectory planning problem* i.e. computing an optimal trajectory $u^a(t)$ which drives an UMS system from a staring configuration $q^A(0) = q^A_{start}$ to a final configuration $q^A(T) = q^A_{goal}$. The first step is to write equation 4.25 in the ODE form $\dot{q} = h(q,p), \dot{p} = g(q,p)$ i.e.:

$$\begin{aligned}
&\frac{dq^a}{dt} = q \\
&\frac{d^2q^a}{dt^2} = \dot{q} \\
&\frac{d^3q^a}{dt^3} = \ddot{q} \\
&\frac{d^4q^a}{dt^4} = \Gamma^a\left(q^A, \dot{q}^A, \ddot{q}^a, \dddot{q}^a, p^1_\mu, \dot{p}^1_\mu\right), \\
&\frac{dq^\mu}{dt} = q^\mu \\
&\frac{d^2q^\mu}{dt^2} = G^\mu\left(q^A, \dot{q}^A\ddot{q}^a\right), \\
&\frac{dp^1_\mu}{dt} = p^1_\mu \\
&\frac{d^2p^1_\mu}{dt^2} = \frac{d}{dt}\left(\frac{\partial \tilde{L}_{\mathscr{M}}}{\partial \dot{q}^\mu} - p^1_\nu \frac{\partial G^\nu}{\partial \dot{q}^\mu}\right) - \left(\frac{\partial \tilde{L}_{\mathscr{M}}}{\partial q^\mu} - p^1_\nu \frac{\partial G^\nu}{\partial q^\mu}\right)
\end{aligned} \tag{4.27}$$

We can now apply a numerical method to search for a solution. As a first approach, we decided to use an indirect method, presented in the next Section, to find the best trajectory $\hat{q}^a(t)$, which connects the starting point to the final configuration. Since we are interested in computing the control input $u^*(t)$, so we need

another step. Recalling the Lagrange equation for the actuated degrees of freedom

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}^a}\right) - \frac{\partial L}{\partial q^a} = u^a \tag{4.28}$$

we can substitute the generic trajectory q^a with the solution q^{*a} computed by the indirect method $\hat{q}^a$ obtaining the desired best control trajectory $\hat{u}^a$. Finally we can apply the best controls $\hat{u}^a$ to actuated degrees of freedom driving the super-articulated system to the target configuration.

4.5 The cart-pole

We apply the method proposed in the previous Section to a standard example in control theory literature: the cart-pole system. In general, the problem is to move the cart from an initial position to a final one maintaining the pole around the equilibrium point. In our case, we want to move the cart but avoiding an obstacle put at a certain height in the final cart position x_f.

Formally, we want to plan an optimal trajectory $(x(t), \theta(t), u(t))$ of the configuration variables and of the controls that starting from a given initial configuration $(x(0), \theta(0), \dot{x}(0), \dot{\theta}(0))$, avoids the obstacle and stops at a given final position $(x(t_f), \theta(t_f), \dot{x}(t_f), \dot{\theta}(t_f))$.

To evaluate the best trajectory, we consider the following cost function:

$$\mathscr{A}(x(\cdot), \theta(\cdot), u(\cdot)) = \frac{1}{2}\int_0^{t_f} u^2 dt \tag{4.29}$$

The configuration space of the systems is $Q = \mathbb{R} \times \mathbb{S}^1$ equipped with local coordinates (x, θ), where x identifies the position of the center of mass of the cart and θ is the pole angle with respect to the vertical direction. The phase space is TQ with local coordinates $(x, \theta, \dot{x}, \dot{\theta})$. The Lagrangian of the system is :

$$\begin{aligned} &L(x, \theta, \dot{x}, \dot{\theta}) = \\ &\frac{1}{2}M\dot{x}^2 + \frac{1}{2}m(\dot{x}^2 + 2\ell\dot{x}\dot{\theta}\cos\theta + \ell^2\dot{\theta}^2) - mg\ell\cos\theta \end{aligned} \tag{4.30}$$

where M is the mass of the cart, m and ℓ are the mass and the length of the pendulum, respectively, and g is the gravity acceleration constant.

The system is subject to a control force $\vec{F} = (u, 0)$ along the x–axis and the degree of freedom defined by θ is not actuated. The equations of motion of the controlled system are then

$$\begin{aligned} &(M+m)\ddot{x} - m\ell\dot{\theta}^2\sin\theta + m\ell\ddot{\theta}\cos\theta = u, \\ &\ddot{x}\cos\theta + \ell\ddot{\theta} - g\sin\theta = 0\,. \end{aligned}$$

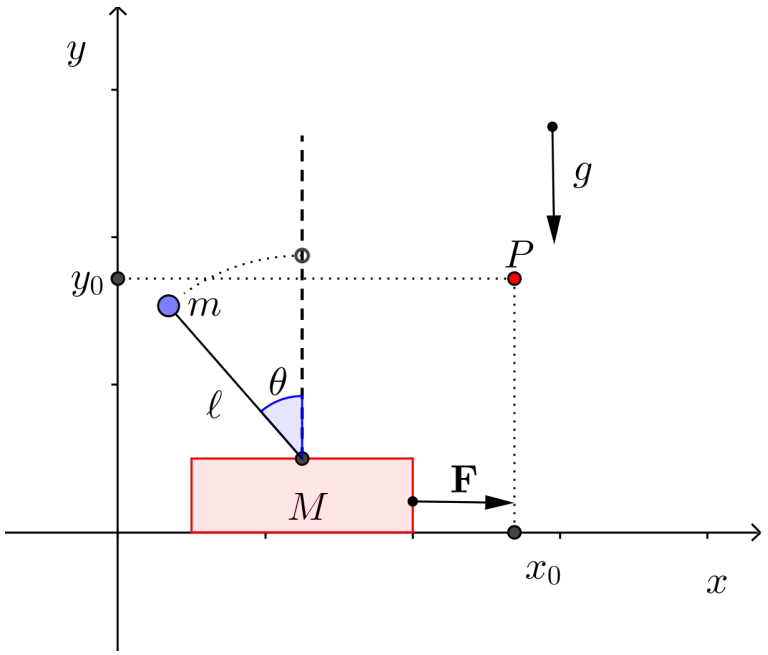

Figure 4-2: The cart-pole example

From the second equation we obtain

$$G^{\theta}(x,\theta,\dot{x},\dot{\theta},\ddot{x}) = \frac{g\sin\theta - \ddot{x}\cos\theta}{\ell} \tag{4.31}$$

and then the constrained Lagrangian $\tilde{L}\,|_{\mathscr{M}}$ is

$$\begin{aligned}\tilde{L}\,|_{\mathscr{M}}\,(x,\theta,\dot{x},\dot{\theta},\ddot{x}) = \frac{1}{2}[(M+m)\ddot{x} - m\ell\dot{\theta}^2\sin\theta \\ + mg\cos\theta\sin\theta - m\ddot{x}\cos^2\theta]^2\end{aligned} \tag{4.32}$$

where $\mathscr{M} = \left\{(x,\theta,\dot{x},\dot{\theta},\ddot{x},\ddot{\theta}) \in T^2Q \,|\, \ddot{\theta} = G^{\theta}(x,\theta,\dot{x},\dot{\theta},\ddot{x})\right\}$ is the constraint manifold.

The pre-symplectic 2-form Ω_{W_0} and the Hamiltonian H_{W_0} projected in the distribution W_0 are, respectively

$$\begin{aligned}H_{W_0} = p_x^0\dot{x} + p_\theta^0\dot{\theta} + p_x^1\ddot{x} + p_\theta^1 G^{\theta} - \\ - \frac{1}{2}\left[(M+m)\ddot{x} - m\ell\dot{\theta}^2\sin\theta + m\ell\ddot{\theta}\cos\theta\right]^2\end{aligned} \tag{4.33}$$

The primary constraint is

$$\varphi_x^1 = p_\theta^1 + p_\theta^1\frac{\partial G^{\theta}}{\partial\ddot{x}} - \frac{\partial\tilde{L}_{\mathscr{M}}}{\partial\ddot{x}} = 0\,. \tag{4.34}$$

The submanifold W_1 of W_0, locally defined by φ_x^1, equipped with the restriction Ω_{W_1} of Ω_{W_0} is a symplectic manifold resulting in

$$\mathscr{R} = (M + m\sin^2\theta)^2 \neq 0\,. \tag{4.35}$$

Using the Gotay-Neste-Hinds's algorithm we can find a unique vector field X_{W_1} on W_1 that satisfies the equation $i_{X_{W_1}}\Omega_{W_1} = dH_{W_1}$, where H_{W_1} denotes the restriction to W_1 of the Hamiltonian H_{W_0}. We are able now to find a unique control, given by equation (4.1), which minimizes the cost functional $\mathscr{A}$.

The theory presented in this Chapter guarantees the conservation of the Hamiltonian vector field X_{W_1} of the symplectic form Ω_{W_1} and of the Hamiltonian H_{W_1} along the flow on W_1.

Finally, after some symbolic computation we can write the equations of motions (4.25) for the controlled cart-pole:

$$\begin{aligned}
&\ddot{\theta} = G^{\theta}(x,\theta,\dot{x},\dot{\theta},\ddot{x})\\
&\frac{d^2 p_\theta^1}{dt^2} = \frac{d}{dt}\frac{\partial \tilde{L}_{\mathscr{M}}}{\partial \dot{\theta}} - \frac{\partial \tilde{L}_{\mathscr{M}}}{\partial \theta} + p_\theta^1 \frac{\partial G^\theta}{\partial \theta}\\
&\frac{d^4 x}{dt^4} = \mathscr{R}^{-1}\left[-\dddot{x}\frac{d}{dt}\mathscr{R} - \frac{d}{dt}\left(\frac{\partial^2 \tilde{L}_{\mathscr{M}}}{\partial\theta\,\partial\ddot{x}}\,\dot{\theta} - \dot{p}_\theta^1 \frac{\partial G^\theta}{\partial \ddot{x}} - p_\theta^1 \frac{\partial^2 G}{\partial\theta\,\partial\ddot{x}}\,\dot{\theta}\right)\right]
\end{aligned}$$

with $\mathscr{R}$ defined in (4.35).

4.5.1 Trajectory planning for the cart-pole system

Having computed the equations of motion for the cart-pole system, our goal is now to drive the cart-pole from an initial configuration point $q_0 = (x_0, \theta_0)$ to configuration $q_f = (x_f, \theta_f)$ avoiding the obstacle placed at point P of coordinates (x_f, y_P), with $y_P < \ell$ and with θ_f chosen so that $\ell\cos\theta_f < y_P$. We can see this problem as a *Two point boundary value problem* (see the book "Applied Optimal Control: Optimization, Estimation, and Control"[16]):

Given N first-order ordinary differential equations (ODE) in the canonical form

$$\frac{\partial y_i(x)}{\partial x} = g_i(x, y_1, y_2, \ldots, y_N) \text{ with } i = 1,2,\ldots N \tag{4.36}$$

and also have r_1 boundary conditions at the initial point x_1 and r_2 boundary conditions at the final point x_N, we write the boundary conditions as:

$$\begin{aligned}
&B_{1j}(x_1, y_1, y_2, \ldots, y_N) = 0 \text{ with } j = 1,\ldots r_1\\
&B_{2k}(x_2, y_1, y_2, \ldots, y_N) = 0 \text{ with } k = 1,\ldots r_2
\end{aligned} \tag{4.37}$$

As the first approach, we decided to solve using a standard *shooting method*. We use as cost function the difference between the reached position and the desired

position, after that, we minimize this cost function using the Levenberg-Marquardt algorithm [32]. As result of this computation, we find the best values for momenta p^1_θ, which are the momenta of the non-actuated degrees of freedom.

To understand the numerical algorithm, we present a pseudo-code which gives the idea of the procedure used to find solutions:

```
Function errorFun(par, q_0^a, q_0^μ, q_f^a, q_f^μ, time)
begin
    q_0= [q_0^a, 0, 0, 0, q_0^μ, 0, par[0], par[1]]
    q_curr(t)= odeSolv(dynEvo, q_0, time)
    q_curr^a = q_curr[0](t_f)
    q_curr^μ = q_curr[4](t_f)
    return [ q_0^a − q_curr^a, q_0^μ − q_curr^μ ]
end

Program cartPole
begin
    tMax= 1
    t= [0, 0.1, ..., t_f]
    optPar= optimize(errorFun(par, q_0^a, q_0^μ, q_f^a, q_f^μ, t)
    qInit_opt = [q_0^a, 0, 0, 0, q_0^μ, 0, par[0], par[1]]
    q_opt = odeInt(dynEvo, qInit_opt, time)
    u^a = d/dt ( ∂L/∂q̇^a_opt ) − ∂L/∂q^a
end
```

where function `dynEvo` $(q^a, \dot{q}^a, \ddot{q}^a, \dddot{q}^a, q^\mu, \dot{q}^\mu, p^1_\mu, \dot{p}^1_\mu)$ implements the system dynamic evolution as solutions of equations (4.25).

4.5.2 Numerical results

In this Section we can see the numerical results obtained using the method presented in the previous Section The physical parameters of the system are summarized , using the values in Table 4.1

The first plot in Figure 4-3 shows the evolution of the center of mass of the cart: we can observe that the cart goes ahead until reaches a maximum near $0.9\,s$ outlined by the dashed vertical line, and then it goes back to the prescribed final position. The second plot describes the evolution of the pole's angle θ. The pendulum rotates anticlockwise (over $1\,rad$) and then, with a (small) delay concerning the x maximum, stops increasing and rotates clockwise to the final position, without touching the obstacle that illustrates the time evolution of the height inverted pendulum-blue line-compared with the height of the (fixed) obstacle-horizontal red line). The plot describing the control input in Figure 4-3, shows the evolution of

Symbol	Description	Value
M	mass of the car	$1\,Kg$
m	mass of the pole	$0.01\,Kg$
ℓ	length of the pole	$1\,m$
g	gravity acceleration	$9.81\,ms^{-2}$
q_0	initial configuration	$(0;0)$
q_f	final configuration	$(1;1)$
t_0	initial time	$0\,s$
t_f	final time	$1\,s$

Table 4.1: Mechanical properties of the cart-pole example using the International system of unit

the optimal control: at the beginning, the applied control force is positive to move the cart toward the positive direction of the x axe, then, after $0.6\,s$ (and before $0.8\,s$) it changes sign and first slows down the cart-pole, then reverses the direction of the motion to reach the final position.

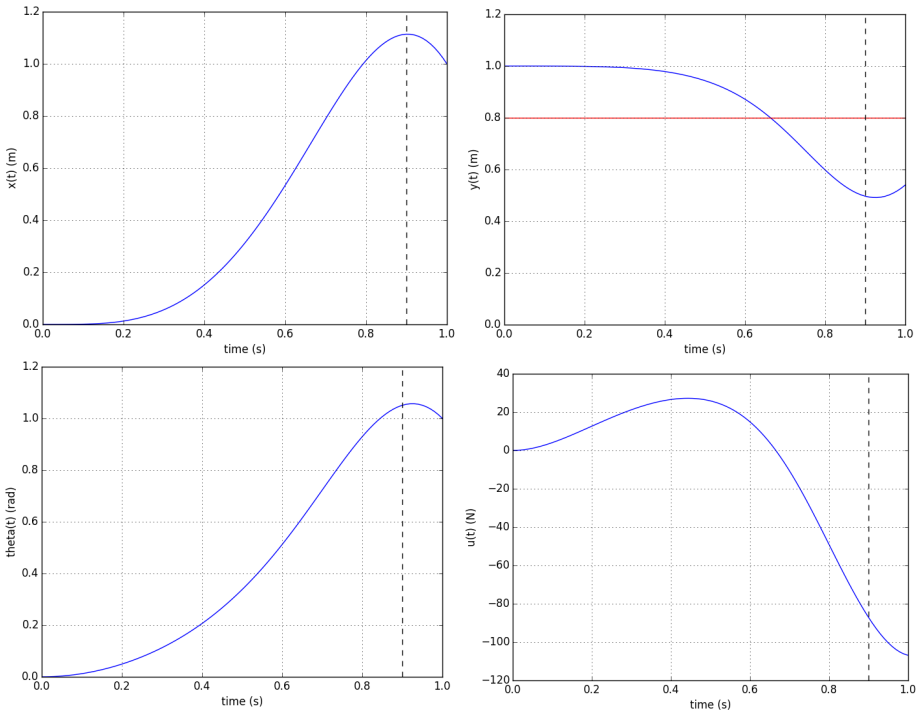

Figure 4-3: Time evolution of $x(t)$, $y(t)$, $\theta(t)$ and the control $u(t)$.

4.6 Quadrotor with a suspended load

Another case of study for the variational constrained system problem is a quadrotor with a suspend load. This problem could be useful in some practical applications as presented by various authors [74] [22] [35] [37].

If we control the quadrotor in position, using the *Flatness property*, as presented in Chapter 2, we have a similar model as the cart-pole proposed before and shown in figure 4-4 . For an application of the Flatness property to quadrotor with a suspended load refer to [89].

We suppose that the quadrotor maintains a certain height h and a link of length l suspends a mass m could freely swing. At a distance x_f from the staring point x_0p, there is an obstacle placed at height $d = (h-l)$.

Our aim is to plan a trajectory in such a way that the quadrotor with the pendulum, starting from a given position P_0 arrives at a prescribed final position P_f, avoiding the obstacle.

As first example we consider that the quadrotor is also constrained to move along a line that coincides with the x–axis of our inertial frame (see Fig. 4-4).

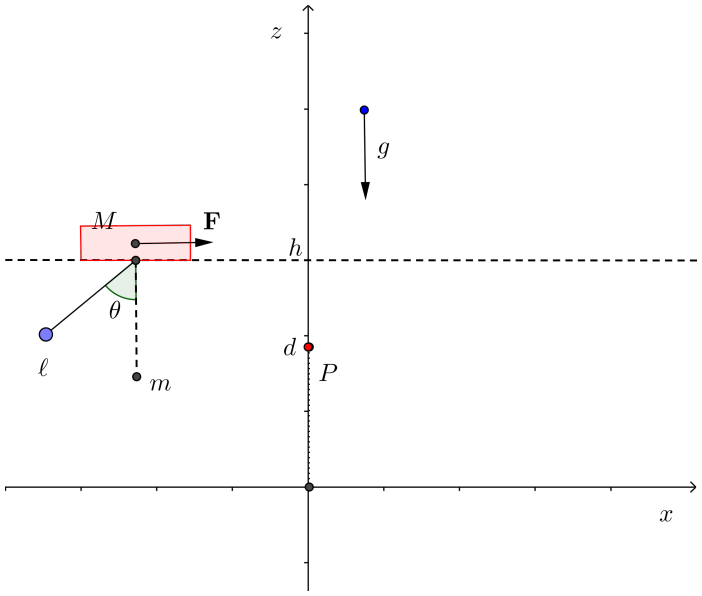

Figure 4-4: Simplified model of quadrotor with a suspend load moving along the x–axis.

We model our system as a mass M with pendulum of length ℓ and mass m constrained to move along a straight line at constant high h on which a control force $\mathbf{F}$ acts along the motion direction and that have to make the pendulum avoid the obstacle P fixed in space at the constant high $d > h - \ell$

The configuration space of the systems is $Q = \mathbb{R} \times \mathbb{S}^1$ equipped with local coordinates (x, θ), where x identifies the position of the center of mass of the cart and θ

is the pendulum angle with respect to the vertical direction. The phase space is the tangent bundle TQ with local coordinates $(x, \theta, \dot{x}, \dot{\theta})$. The Lagrangian of the system is :

$$\begin{aligned} &L(x, \theta, \dot{x}, \dot{\theta}) = \\ &\frac{1}{2}M\dot{x}^2 + \frac{1}{2}m(\dot{x}^2 + 2\ell\dot{x}\dot{\theta}\cos\theta + \ell^2\dot{\theta}^2) - mg\ell\cos\theta \end{aligned} \tag{4.38}$$

where M is the mass of the cart, m and ℓ are the mass and the length of the pendulum, respectively, and g is the gravity acceleration constant.

The system is subject to a control force $\mathbf{F} = (u, 0)$, applied parallel to the track, and we assume the degree of freedom defined by θ as underactuated. The equations of motion of the controlled system are then

$$\begin{aligned} &(M+m)\ddot{x} + m\ell\dot{\theta}^2\sin\theta - m\ell\ddot{\theta}\cos\theta = u, \\ &-\ddot{x}\cos\theta + \ell\ddot{\theta} + g\sin\theta = 0\,. \end{aligned} \tag{4.39}$$

4.6.1 Numerical results

As we did with the cart-pole system, we used the same numerical method to find a solution to this problem. In this case the parameters are shown in table 4.2.

Table 4.2: Mechanical properties of the cart-pendulum example using the International System of Unit

Symbol	Description	Value
M	mass of quadrotor	$1.3\,Kg$
m	mass of load	$0.01\,Kg$
ℓ	length of link	$0.15\,m$
g	gravity acceleration	$9.81\,ms^{-2}$
q_0	initial configuration	$(0m; 3.14rad)$
q_f	final configuration	$(0; 2.71rad)$
t_0	initial time	$0\,s$
t_f	final time	$1\,s$

Also in this study case, it is possible to plan a trajectory and decide the final configuration of the system. In particular, having the possibility of deciding the final value of the slung, we are able to avoid obstacles or allowing an automatic unload of the payload. In the next Section we present a real setup with some preliminary results.

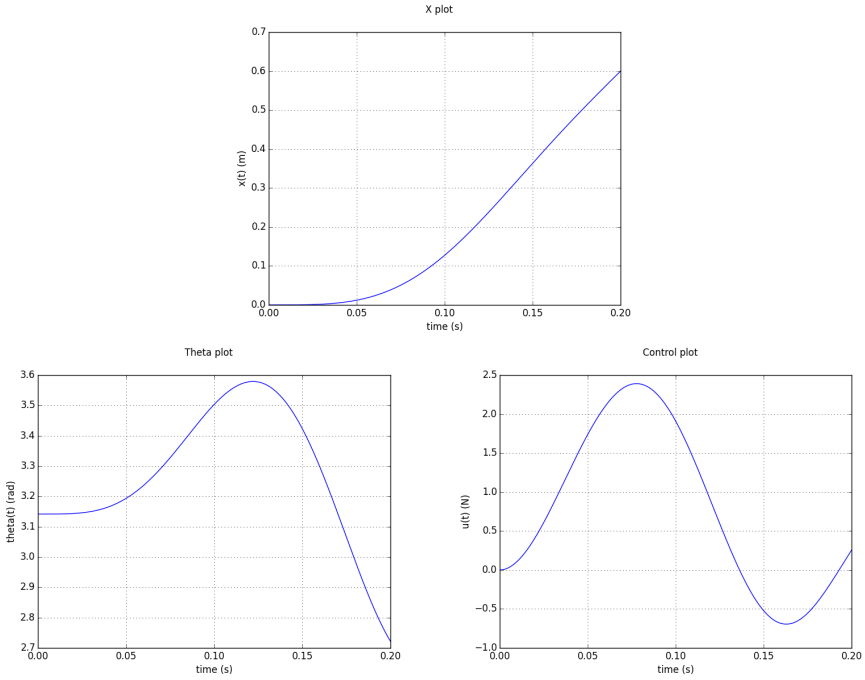

Figure 4-5: Time evolution of $x(t)$, $\theta(t)$ and the control $u(t)$.

4.7 Conclusions

In this Chapter, we applied a method, called "Variational constrained system problem" that use the Hamiltonian framework to solve the problem of trajectory planning for underactuated mechanical systems. It is interesting to notice that the method transform the Lagrangian equations of motion to Hamiltonian equations, including the constraints of the UMS. From a practical point of view, we apply a numerical method to search for admissible momenta that plan an optimal trajectory. This is the key point that supports the idea of the thesis presented in Chapter 1.

Using this method, we discovered that it could not be applied to all kind of UMS, but only to super-articulated system presented in Chapter 3. The primary case study of this thesis, presented in Chapter 1, is the multi-rotor but we discovered that the multi-rotor is not a "super-articulated system." because the model does not respect the properties defined by Ballieul in [91] so it is not possible to apply the "Variational constrained system problem." to multi-rotors.

We are interested in finding another method based on the Hamiltonian framework. Our idea is to see the evolution of momenta as a description of the energy that is stored inside the system, we want to search for a method that can use this energy to overcome the constraints imposed by the under-actuation. In the next Chapter, we will write the Hamiltonian equations describing the multi-rotor and we will apply a numerical method to plan feasible trajectories.

Chapter 5

Energy-based method for generic UMS

In this chapter, we use the Hamiltonian formalism presented in Chapter 3 to solve the trajectory planning problem for generic under-actuated mechanical systems (UMS). From the experience gained using the variational constrained systems method described in the previous Chapter, we want to investigate how the relationship between the evolution of robot configurations, energy, and input control can be used to plan a possible trajectory for generic UMS. The method proposed in the previous Chapter, in-fact, does not allow us to plan a trajectory for all kinds of UMS, but only for super-articulated mechanical systems. The case study proposed in this Chapter will be the quad-rotor, a six degrees of freedom flying robot with four actuated degrees if freedom that is of interest of many researchers, as discussed in the Introduction.

5.1 Proposed method

Our approach to overcame the constraints imposed by the under-actuation is to use the energy produced by the motion of the mechanical system. Recalling the concepts presented in Chapter 3, starting from the Lagrangian $L(q,\dot{q})$ we describe the Hamiltonian function as

$$H(q,p) = \dot{q}_i p_i - L(q,\dot{q}) \tag{5.1}$$

and the dynamic evolution of an UMS can be written in the following form:

$$\begin{aligned} \dot{q}_i &= \frac{\partial H(q,p)}{\partial p_i} \\ \dot{p}_i &= -\frac{\partial H(q,p)}{\partial q_i} \end{aligned} \tag{5.2}$$

We can see two first-order differential equations, one for each degree of freedom and one of which is the momenta evolution in explicit form. In the literature, the a direct numerical methods used to compute an optimal trajectory relay a cost function that includes position and control inputs of the studied mechanical system. The optimal trajectory $r(t)$ is a function of the positions x and the control input u:

$$r = (x, u) \tag{5.3}$$

In the case of UMS, we are interested in planning also the evolution of energy "stored" in the system by searching for a solution consider that not only positions and control inputs, but also the evolution of momenta p. Thus the function to optimize as

$$\tilde{r} = (x, p, u) \tag{5.4}$$

The benefit of this approach is the possibility of using the energy stored in the system to drive the quad-rotor in the degrees of freedom that are not directly controllable. To present this idea, we show an example of the Hamiltonian equations of a quad-rotor constrained to move along the x-direction and able to rotate along the y-direction only. Let $u = \tau_\theta$ the control along the x-axis then we obtain

$$\begin{aligned} \dot{x} &= \frac{p_x}{m}\,, \\ \dot{\theta} &= \frac{p_\theta}{I_3(1+\cos^2\theta)}\,, \\ \dot{p}_x &= -mg\tan\theta\,, \\ \dot{p}_\theta &= \frac{p_\theta^2\,\sin\theta\cos\theta}{I_3(1+\cos^2\theta)^2} + u\,. \end{aligned} \tag{5.5}$$

We can see how the control modifies the momentum p_θ and the evolution of the angle θ. As a consequence, the evolution of the momentum p_x is influenced by the angle θ and, finally, also the non-actuated degree of freedom x is actuated by the momentum. With this procedure, we are able to generate the desired quadcopter trajectory respecting the initial and final boundary conditions on x and p_x, which enables the computation of new maneuvers as presented in the next Section.

5.2 The quad-rotor

First of all we need to formalize a generic model to represent the quad-rotor using the generalized coordinates. We define a set of three oriented orthonormal vectors $T = \{t_x, t_y, t_z\}$ and we call it *Inertial reference frame* (or World frame). To describe the orientation, we need similar set of vectors $E = \{e_1, e_2, e_3\}$ where orientation remains parallel to T but origin is collocated at the center of mass of the quad-rotor (Body frame).

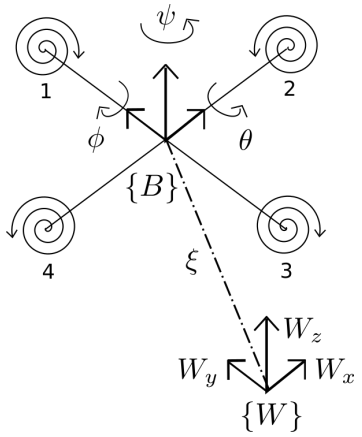

Figure 5-1: The quadorotor coordinates system

We can represent a point in the configuration space as:

$$q = (x, y, z, \psi, \theta, \phi) \in \mathbb{R}^6$$

where

- $\xi = (x, y, z)$ denotes the position relative to T
- $\eta = (\psi, \theta, \phi)$ are the rotation angles around E

Based on the previous points and considering $q = (\xi, \eta)$, we can model our system starting from Euler-Lagrange equation:

$$\frac{d}{dt}\left(\frac{\partial L(q,\dot{q})}{\partial \dot{q}}\right) - \frac{\partial L(q,\dot{q})}{\partial q} = F$$

whereas the Lagrangian is

$$L(q,\dot{q}) = T(q) - U(q)$$

with $T(q)$ is the kinetic energy function and $U(q)$ the potential energy.

We can separate translational part and rotational part of kinetic energy

$$L(q,\dot{q}) = T_{trans} + T_{rot} - U$$

where

- $T_{trans} = \frac{1}{2} m \dot{\xi}^T \dot{\xi}$ is kinetic energy related to $\{W\}$
- $T_{rot} = \frac{1}{2} \omega^T I \omega$ is rotational kinetic energy
- $U = -mgW_z$ is potential energy of the quadrotor

m is the mass, g is gravity acceleration, ω angular velocity respect to body frame $\mathscr{B}$ and I is inertial matrix expressed by

$$I = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}$$

$T_{rot} = \frac{1}{2}\omega^T I\omega$ is expressed relative to body frame and we re-write this equation using general coordinates. Variable ω is related to generalized coordinates $\dot{\eta}$ by the equation

$$\omega = W_\eta \dot{\eta}$$

where

$$W_\eta = \begin{bmatrix} -sin(\phi) & 0 & 1 \\ cos(\phi)sin(\theta) & cos(\theta) & 0 \\ cos(\phi)cos(\theta) & -sin(\theta) & 0 \end{bmatrix}$$

If we substitute last equations into $T_{rot} = \frac{1}{2}\omega^T I\omega$ we obtain:

$$\begin{aligned} T_{rot} = & \quad \tfrac{1}{2}\omega^T I\omega \\ T_{rot} = & \quad \tfrac{1}{2}(W_\eta \dot{\eta})^T I W_\eta \dot{\eta} \end{aligned}$$

We define $\mathbb{J}(\eta) = (W_\eta)^T I W_\eta$ as inertial matrix expressed in generalized coordinates to obtain

$$T_{rot} = \frac{1}{2}\dot{\eta}^T \mathbb{J}\dot{\eta}$$

that is the rotational kinematic energy of the quadrotor expressed to the inertial frame.

Summarizing, we can write the Lagrangian as

$$L(q,\dot{q}) = T_{trans} + T_{rot} - U = m\frac{1}{2}\dot{\xi}^T\dot{\xi} + \frac{1}{2}\dot{\eta}^T\mathbb{J}\dot{\eta} + mgW_z \tag{5.6}$$

Recalling the original equation

$$\frac{d}{dt}\left(\frac{\partial L(q,\dot{q})}{\partial \dot{q}}\right) - \frac{\partial L(q,\dot{q})}{\partial q} = F$$

we are now interested on working with forces and we separating the translational from rotational of L:

$$F = \begin{bmatrix} F_\xi \\ \tau \end{bmatrix}$$

For translational force we can write

$$F_\xi = R\hat{F} \in \mathbb{R}^3$$

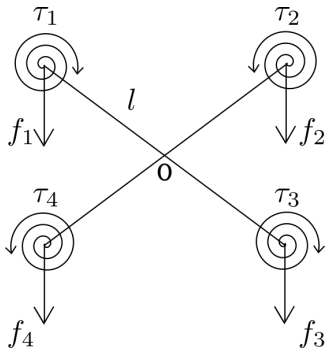

Figure 5-2: This picture shows the forces applied by rotors

where R represents orientation of the aircraft relative to world frame (using the Roll-Pitch-Yaw convention)

$$R = \begin{bmatrix} \cos\theta\cos\psi & \cos\psi\sin\theta\sin\varphi - \sin\psi\cos\varphi & \cos\psi\sin\theta\sin\varphi + \sin\psi\sin\varphi \\ \cos\theta\sin\psi & \sin\psi\sin\theta\sin\varphi + \cos\psi\cos\varphi & \cos\varphi\sin\theta\sin\psi - \cos\psi\sin\varphi \\ -\sin\theta & \cos\theta\sin\varphi & \cos\theta\cos\varphi \end{bmatrix} \tag{5.7}$$

and $\hat{F}$ is translational force applied to the z-axis in body frame

$$\hat{F} = \begin{bmatrix} 0 \\ 0 \\ u \end{bmatrix} \tag{5.8}$$

where $u = \sum_{i=1}^{4} f_i$ is the sum of all forces as shown in figure 5-2.

The generalized torques τ are expressed by

$$\tau = \begin{bmatrix} \tau_\psi \\ \tau_\theta \\ \tau_\phi \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{4} \tau_{M_i} \\ (f_2 - f_4)l \\ (f_3 - f_2)l \end{bmatrix} \tag{5.9}$$

where l is the distance between the motors and the center of gravity and τ_{M_i} is the moment produced by motor i about the center of gravity.

If we substitute the terms computed in equation 5.6 we obtain the equations of motion

$$\frac{d}{dt}\left[\frac{\partial L_{trans}}{\partial \dot{\xi}}\right] - \frac{\partial L_{trans}}{\partial \xi} = \hat{F} \rightarrow m\ddot{\xi} + mgW_z = \hat{F}$$

and

$$\frac{d}{dt}\left[\frac{\partial L_{rot}}{\partial \dot{\eta}}\right] - \frac{\partial L_{rot}}{\partial \eta} = \tau \rightarrow \mathbb{J}\ddot{\eta} + N(\eta, \dot{\eta}) = \tau$$

where $N(\eta, \dot{\eta}) = \left(\dot{\mathbb{J}} - \frac{1}{2}\frac{\partial}{\partial \eta}(\dot{\eta}^T \mathbb{J})\right)\dot{\eta}$ contains gyroscopic and centrifugal terms.

The extended dynamic equations are:

$$\begin{aligned} m\ddot{z} = &\quad u(\cos(\theta)\cos(\phi)) - mg \\ \mathbb{J}\ddot{\psi} + N(\psi, \dot{\psi}) = &\quad \tau_\psi \\ \mathbb{J}\ddot{\theta} + N(\theta, \dot{\theta}) = &\quad \tau_\theta \\ \mathbb{J}\ddot{\phi} + N(\phi, \dot{\phi}) = &\quad \tau_\phi \end{aligned}$$

We have also two equations that represent the dynamic evolution of the non-actuated degrees of freedom.

$$\begin{aligned} m\ddot{x} - u(\sin(\phi)\sin(\psi) + \cos(\phi)\cos(\psi)\sin(\theta)) = 0 \\ m\ddot{y} - u(\cos(\phi)\sin(\theta) - \cos(\psi)\sin(\phi)) = 0 \end{aligned}$$

In Chapter 4 we use the equations of the non-actuated DoF to build a distribution on the cotangent space. In the case of quadrotors this is not possible because it is not a "super-articulated mechanical system". In the future we want to understand the geometrical meaning of these equations. In the next Section we present our approach to apply geometrical insight to this problem.

5.3 Trajectory planning for quadrotors

In the previous section we presented a mathematical model to describe the quadrotor and computing the Lagrangian equations, which are useful to understand the equation describing the evolution of the non-actuated degrees of freedom. All the previous equations we described using the generalized coordinates. To reduce the complexity of the implementation and to use the commercial flight controller, in this section we model another time the quadrotor but we use the aeronautical convention Nord-Est-Down (NED) to orient the axes of the inertial frame. Another difference is that we the angular velocities are relative to the body frame and not relative to the inertial frame as in the Section before.

As shown in the previous section, the quadrotor equations can include the external forces and the effect of a constraint, for example to maintain a fixed quote h.

Let $T = (t_x, t_y, t_z)$ be an inertial reference frame oriented according the North-East-Down (NED) convention, and $E = (e_1, e_2, e_3)$ be a "body frame", with the unit vectors e_1, e_2 and e_3 oriented as the principal axes of inertia of the quadrotor (see Figure 5-10).

The SO(3) component of the configuration space is parametrized via the NED convention for the Euler's angles on SO(3) by $(q, \dot{q}) = (x, y, z, \psi, \theta, \phi, \dot{x}, \dot{y}, \dot{z}, \Omega_1, \Omega_2, \Omega_3)$, with $(x, y, z) \in \mathrm{R}^3$ the coordinates of the center of mass of the quadrotor with respect

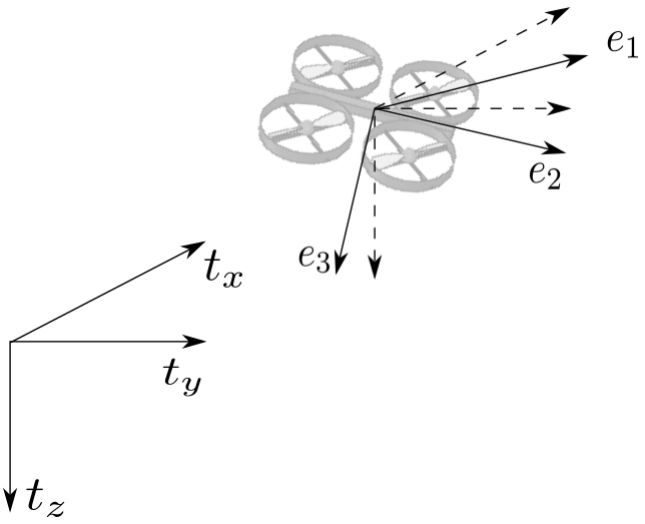

Figure 5-3: The figure shows the quadcopter and reference frames.

to the NED inertial frame, and with $\Omega = (\Omega_1, \Omega_2, \Omega_3) \in \mathbb{R}^3$ the angular velocities of the quadrotor in the body frame representation. The Lagrangian of the system is, in local coordinates:

$$L(q, \dot{q}) = \frac{1}{2} m(\dot{x}^2 + \dot{y}^2 + \dot{z}^2) + \frac{1}{2} \Omega \cdot I\Omega + mgz, \tag{5.10}$$

where g is the gravity acceleration constant, m denotes the mass and I is the tensor of inertia relative to the center of mass C of the quadrotor. In the body reference frame the inertia tensor is diagonal $\mathbb{I}_C = \text{diag}(I_1, I_2, I_3)$ and constant in time, where I_1, I_2 and I_3 are the diagonal components with respect to the principal axes of inertia. The quadrotor symmetry requires that $I_1 = I_2$. We compute the Hamiltonian formulation of the equations of motion via the Legendre transform. The Hamiltonian, denoting by p the conjugate momentum vector to q, is

$$H(q, p) = \frac{1}{2m}(p_x^2 + p_y^2 + p_z^2) + \frac{1}{2} \mathbb{I}_C^{-1} M \cdot M - mgz, \tag{5.11}$$

where $M = (M_1, M_2, M_3)$ denotes the angular momentum with respect to the center of mass C in the body representation frame, conjugated with Ω. Let $(F_x, F_y, F_z, \tau_1, \tau_2, \tau_3) \in \mathbb{R}^3 \times \mathbb{R}^3$ be the control forces acting on the quadrotor, where

$$\begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = R \begin{bmatrix} 0 \\ 0 \\ u \end{bmatrix}, \tag{5.12}$$

with R the transformation matrix that represents the quadrotor attitude:

$$R = \begin{bmatrix} \cos\theta\cos\psi & \cos\psi\sin\theta\sin\varphi - \sin\psi\cos\varphi & \cos\psi\sin\theta\sin\varphi + \sin\psi\sin\varphi \\ \cos\theta\sin\psi & \sin\psi\sin\theta\sin\varphi + \cos\psi\cos\varphi & \cos\varphi\sin\theta\sin\psi - \cos\psi\sin\varphi \\ -\sin\theta & \cos\theta\sin\varphi & \cos\theta\cos\varphi \end{bmatrix} \tag{5.13}$$

We assume a linear model of air resistance, in which the resisting force is proportional to the velocity. Let κ_x, κ_y and κ_z be the resistance coefficients.

The Hamilton's equations for the quadrotor are

$$\begin{aligned} &\dot{x} = \frac{p_x}{m}, \quad \dot{y} = \frac{p_y}{m}, \quad \dot{z} = \frac{p_z}{m}, \\ &\dot{p}_x = \frac{\kappa_y}{m} p_x + F_x, \quad \dot{p}_y = \frac{\kappa_y}{m} p_y + F_y, \quad \dot{p}_z = -mg + \frac{\kappa_z}{m} p_z + F_z, \\ &M_1 = I_1(\dot{\varphi} - \dot{\psi}\sin\theta), \quad M_2 = I_1(\dot{\psi}\cos\theta\sin\varphi + \dot{\theta}\cos\varphi), \\ &M_3 = I_3(\dot{\psi}\cos\theta\cos\varphi + \dot{\theta}\sin\varphi) \\ &\dot{M}_1 + \frac{M_2 M_3}{I_1 I_3}(I_3 - I_1) = \tau_1, \quad \dot{M}_2 + \frac{M_1 M_3}{I_1 I_3}(I_1 - I_3) = \tau_2, \\ &\dot{M}_3 = \tau_3, \end{aligned} \tag{5.14}$$

In this thesis we make some simplifications to enlighten the problem aspects but the method can used without simplification. We start assuming the resistance coefficients equal to 1 and that the quadrotor does not rotate about the z direction and maintaining a fixed altitude. By imposing a fixed altitude, $\dot{z}$ and $\dot{p}_z$ must vanish, yielding:

$$u = \frac{mg}{\cos\theta\cos\varphi}. \tag{5.15}$$

By imposing that the quadrotor can not rotate about the z direction, we require that ψ, $\dot{\psi}$ and $\dot{M}_3$ vanish, therefore M_3 is constant. Hamilton's equations for the constrained system then become:

$$\begin{aligned} &\dot{x} = \frac{p_x}{m}, \quad \dot{y} = \frac{p_y}{m}, \\ &\dot{p}_x = -\frac{mg}{\cos\theta}\sin\theta, \quad \dot{p}_y = -\frac{mg}{\cos\varphi\cos\theta}\sin\varphi, \\ &M_1 = I_1\dot{\varphi}, \quad M_2 = I_1\dot{\theta}\cos\varphi, \\ &\dot{M}_1 + \frac{M_2 M_3}{I_1 I_3}(I_3 - I_1) = \tau_1, \quad \dot{M}_2 + \frac{M_1 M_3}{I_1 I_3}(I_1 - I_3) = \tau_2. \end{aligned} \tag{5.16}$$

If we impose a further constraint that the quadrotor cannot rotate about the x direction, we deal with a system with two degrees of freedom $(x, \theta) \in \mathbb{R}\times]0, \pi[$.

Hamilton's equation for this systems are then:

$$\begin{aligned} \dot{x} &= \frac{p_x}{m}, \quad \dot{\theta} = \frac{p_\theta}{I_3(1+\cos^2\theta)}, \\ \dot{p}_x &= -mg\tan\theta + \frac{\kappa}{m}p_x, \quad \dot{p}_\theta = \frac{p_\theta^2 \sin\theta\cos\theta}{I_3(1+\cos^2\theta)^2} + \tau_\theta, \end{aligned} \tag{5.17}$$

where p_θ is the conjugate momentum to θ.

5.4 Implementation

In the previous Section we presented the Hamiltonian equations describing the dynamic evolution of the quadrotor. We now want apply a numerical method to compute the best trajectory for configurations q_i, momenta p_i and control input u_i. We use a direct method, called *collocation method* presented in the next Section.

5.4.1 Direct collocation method

The *direct collocation method* was proposed in [40] with the objective to transform an optimization problem into a general nonlinear programming problem (NLP) that can be solved by computing n vectors that solve the equation:

$$min_{\mathbf{x}} F(\mathbf{x}) \tag{5.18}$$

subject to the m constraints:

$$c_L \leq c(\mathbf{x}) \leq c_U \tag{5.19}$$

and bounds:

$$\mathbf{x}_L \leq \mathbf{x} \leq \mathbf{x}_U \tag{5.20}$$

We can transform the equations describing the dynamic evolution of the system into a set of constraints imposing $c_L = c_U$. In this way, the trajectory generation problem is transformed into an NLP with a level of approximation that depends on the numerical method used in the solution. There are many numerical algorithms that can be used to solve this problem, the most common are: *Hermite-Simpson Method* [40], *High Order Gauss-Lobatto Method* [96] and the *Pseudospectral Method* [26]. A survey of these methods and their performance are presented in [93] and [6].

The steps performed in the method are as follows. The first step is to consider a generic state trajectory $x(t)$ and a control trajectory $u(t)$, and to split them into N events $e_i \in E[t_0, t_f]$ where $i \in [1, N]$, ad shown in Figure 5-4. This step produces two piecewise functions $x(t) = \sum_{i=0}^{N} x(e_i)$ and $u(t) = \sum_{i=0}^{N} u(e_i)$.

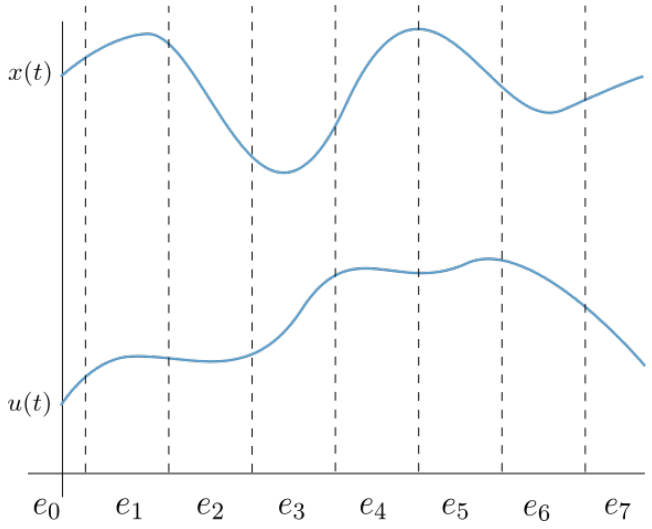

Figure 5-4: Schematic representation of the direct collocation method

The i-event e_i is assumed to represent the dynamical system subject to the differential constraint:

$$x'(t) = f^i(x,u,\omega,t), t \in [e_i, e_i+1] \tag{5.21}$$

where ω is a vector containing the kinematic and dynamic parameters of the system (size, mass, inertia etc.)

Then, we consider only one event $[e_i, e_{i+1}]$, and we fit a cubic spline parametrized as $spl(s_1, s_2, \Delta)$. As shown in [40], we can describe the cubic spline in the interval $[e_i, e_{i+1}]$ using a unique variable Δ_i. To connect all the segment we must satisfy a linear equation in which all the independent variables are grouped into a single vector:

$$P = [Z, E, \omega] \tag{5.22}$$

where $Z = \sum_{i=0}^{N}(x_i, u_i)$.

To complete the problem description we add the nonlinear boundary conditions

$$B_N = \sum_{i=0}^{N} l_{BE} \leq b^i(x(e_i), e_i, \omega) \leq u_{BE} \tag{5.23}$$

and the path and control constraints

$$H_N = \sum_{i=0}^{N} l_{PC} \leq h^i(x(e_i), u(e_i), \omega, t) \leq u_{PC} \tag{5.24}$$

where b, h are assumed to be smooth ($\in C^2$). If all the constraints are grouped into

a unique vector equation

$$C = [\Delta, B_N, H_N] \tag{5.25}$$

the nonlinear programming problem is defined as:

$$\text{minimize } J = \Phi(P)$$

subject to

$$l \leq \begin{pmatrix} P \\ AP \\ C(P) \end{pmatrix} \leq u \tag{5.26}$$

Given this formulation, we can now solve the trajectory computation problem by using a numerical solver such as the proprietary library SNOPT [33] or the open-source project IPOPT[7]. In this thesis we decided to use the "Pytrajectory" toolbox [1] that implements the Hermite-Sympson method [40].

5.5 Results

To test the method proposed, as a first step we make some simulations and, in the next Section, we demonstrate the method on a real setup. To produce a realistic trajectory, we identify the dynamic parameters of the custom made quadrotor, class 280, shown in Figure 5-5 that are summarized in Table 5.1

Body mass	0.89 Kg
Moment of inertia about w_x	0.0048 $kg \cdot m^2$
Moment of inertia about w_y	0.0048 $kg \cdot m^2$
Drag coefficent	0.5

Table 5.1: Table of the dynamic parameters used in the simulations.

We do not have a real setup running to identify the drag coefficient so we decided a reasonable value for this value looking at results in the literature. To give a simple visualization of results and to allow an easy testing of the method, we connected the computation to the visualization tool V-REP [78]. We use V-REP only to define, in a simple and intuitive way, the initial configuration of the quadrotor and to place the trajectory end point. In the first example we use as trajectory target a sequence of windows randomly oriented in the space, whereas in the second example the desired target is the entrance of the narrow gap in a thick wall. In these example we do not use the dynamic simulation capabilities offered by V-REP.

Figure 5-5: Quadrotor used to find the values of the dynamic parameters used in the simulation.

5.5.1 Task 1: flying trough a sequence of generic oriented windows

In the first example, we want to pass through a sequence of randomly oriented windows. Passing through a window is a task that has already been presented in other papers, e.g. [59], [27] and [67]. We present these simulations to show that our approach can handle some of the most recent solved problems in trajectory generation.

The quadrotor starts from a hovering position and we want to compute a trajectory that drives the flying robot to a certain final configuration, which is equal to the window position and orientation. To drive the quadcopter through a sequence of windows, we assign as the initial condition of the second trajectory the terminal conditions of the first trajectory and continue with the other windows. By imposing equal boundary conditions also to the momenta, we are able to ensure a smooth transition between the segments that compose the complete trajectory. Please note that this is equivalent to impose continuity of velocity between the segments as done in previous solutions to this problem. In this case, the quadcopter keeps its center of mass at a fixed hight, according to equation (5.16). The trajectory computed for this example is shown in Figure 5-10.

In this example we can place a final configuration for momenta along x and y directions. In this way we are able to pass trough horizontal windows with a diagonal movements.

Figures 5-7 and 5-8 show the behavior of the quadrotor during the task described above. The graphs show 3 dotted lines which represent the instant of time when the quadrotor passes through a target window. The first window is placed horizontally

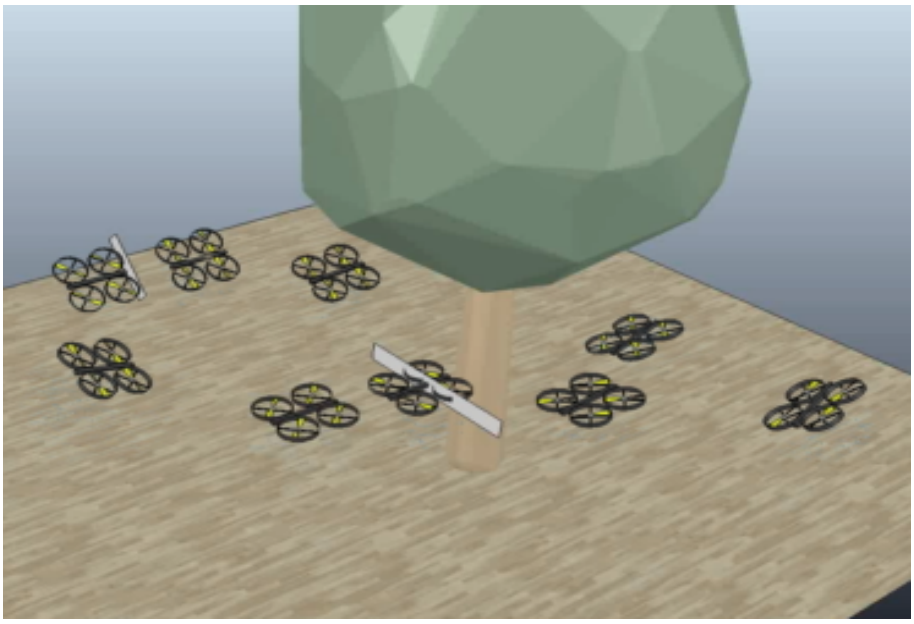

Figure 5-6: Visualization of the trajectory driving the quadrotor trough a sequence of generic oriented windows

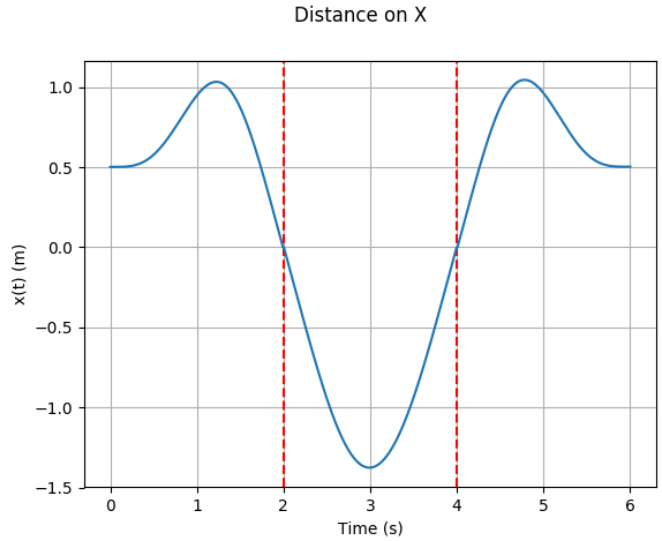

Figure 5-7: X trajectory that avoid an obstacle after the first window

(roll angle equal to zero) and we imposed positive momenta $px = 2$ and $py = 2$ along x and y direction so that the quadrotor *slides* through the window following a diagonal direction with respect to the window face. The second window is placed at a generic position of the world space but is rotated by 0.7 radiants with respect to the line of the horizon.

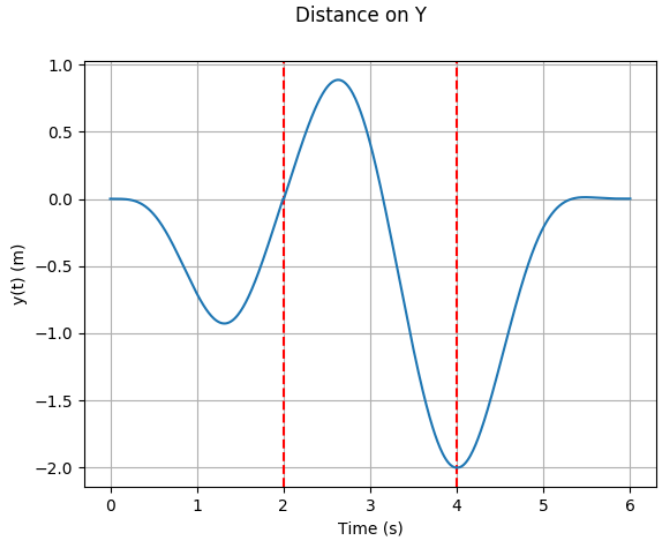

Figure 5-8: Y trajectory that avoid an obstacle after the first window

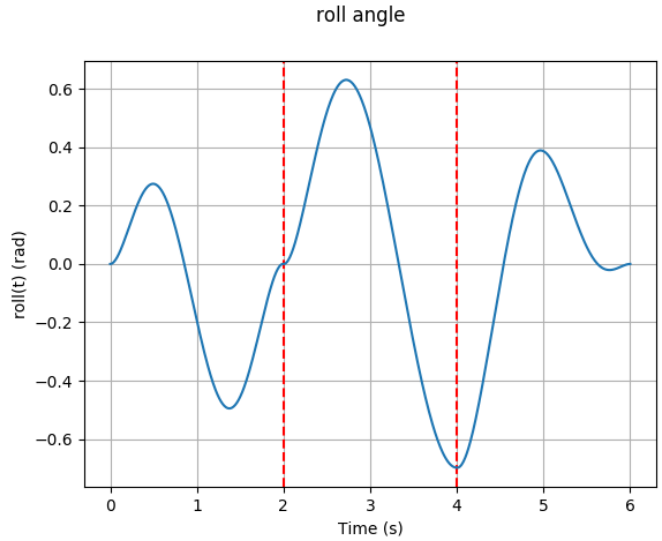

Figure 5-9: Roll trajectory that avoid an obstacle after the first window

5.5.2 Task 2: flying trough a deep gap in the wall

A new maneuver that can be computed using the method developed in the previous sections and not with the other methods described in the literature, is to pass through a small gap in a thick wall. By imposing a non-zero value to energy as terminal boundary condition, in fact, we use the momenta available in the system as *fictitious* control inputs to move the quadrotor in the directions that cannot be directly actuated, i.e. (x and y).

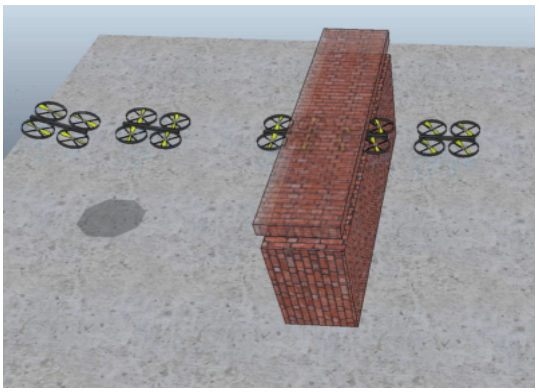

Figure 5-10: Visualization of the trajectory driving the quadrotor passing trough a deep gap in the wall

As in the previous example, we make the quadrotor start from a generic position in space, selected by the user, and the algorithm computes the trajectory which takes it to the target, i.e. the entrance of the gap in the wall, where it arrives with a non-zero energy. In this case, the quadcopter moves along the x direction according to equation (5.17). To see the effect of the quadrotor energy, we implement a forward integration using as initial conditions the terminal conditions of the trajectory. During this phase where the system dynamics evolves, the drag force induced by the air resistance is more evident and the velocity of the quadrotor decreases as shown in Figure 5-11. To check whether a real quadrotor would be able to traverse a gap in the wall with a specific thickness, we can compute the position of the quadrotor when the effect of the energy is finished. This aspect of trajectory computation will be investigated in our future works.

Figure 5-11 shows the trajectory generated for the quadrotor, starting from an initial point $x = -1$, the quadrotor makes a loop to acquire enough momentum to satisfy the terminal constraint decided by the user (in this case 2 $m \cdot s^{-1}$). After the passage through the entrance of the wall, the quadcopter dynamics evolves as describe above, with a logarithmic decay of the velocity until reaching the hovering condition. In this example we decided to stop the simulation after 2 seconds.

5.6 Conclusions

Based on the results achieved in Chapter 4 and motivated by the geometric mechanic theory, in this Chapter we used the Hamiltonian formalism to handle the constraints affecting a generic under-actuated mechanical system. The idea is that the Hamiltonian equations can explicitly describe the evolution of the energy stored in the system and make it possible to use a numerical method to compute an optimized trajectory, not only for state variables but also for momenta. The practical

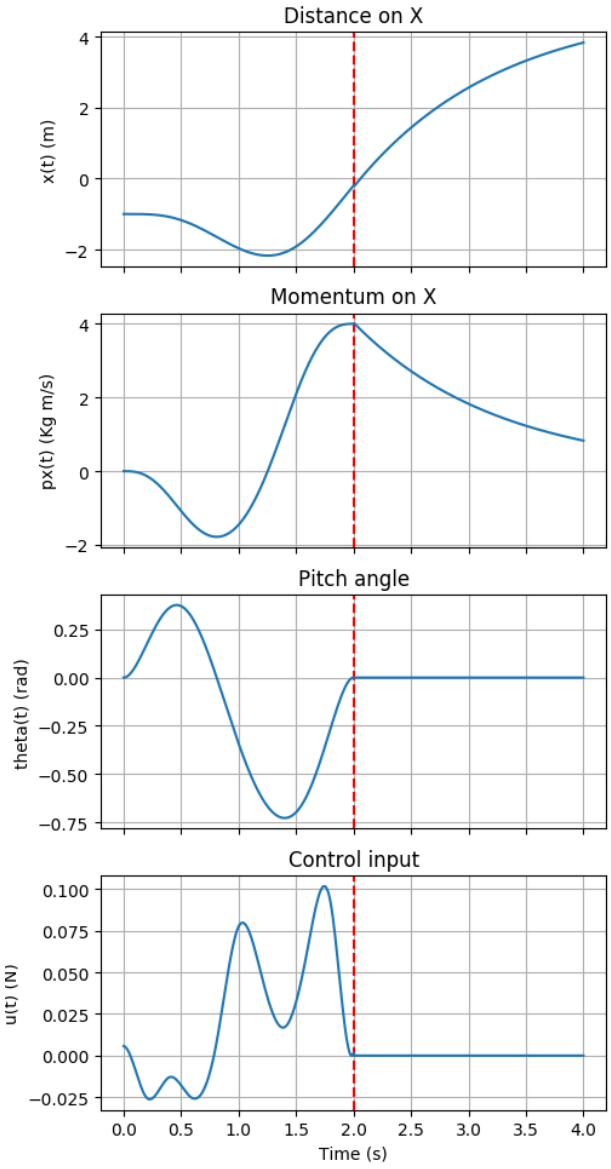

Figure 5-11: Trajectory generated to pass trough a thick wall

result is the possibility to define a final value for the energy, which is useful to do maneuvers that are not possible to do otherwise.

We do not have mathematical proofs, but our intuition is that the underactuated constraints can be represented as co-vector fields that make a distribution on the cotangent space. In this way, it would be possible to build a geometrical method, as we shown in Chapter 4, that plan an optimal trajectory.

In the next Chapter, we apply the technique proposed in this Chapter to a real experimental setup.

Chapter 6

Experimental results

After the presentation of a method to plan a feasible trajectory for quadrotors in Chapter 5, in this Chapter we present an application of the method to a real case. We want to verify that the computed planned trajectory computed drives the quadrotor trough a deep gap using a flight controller present on a custom unmanned aerial vehicle (UAV).

As presented in the Introduction, our goal is to overcome the limits imposed by the under-actuation by planning a trajectory for the actuated DoF, using the energy acquired during the motion and move the quadrotor along a non-actuated DoF. To propose a concrete task, we plan a trajectory that drives the quadrotor through a small gap in a deep wall. The trajectory has constant values for $roll, yaw$ angles attitude and y, z positions, we plan a feasible trajectory and only the $pitch$ Dof is used to complete the trajectory that is not possible with any other techniques.

6.1 Hardware architecture

A quad-rotor is a flying robot composed of many elements that need to operate together to fly. In figure 6-1 we present, an overview of the hardware components and the connections between the various elements.

There are four motors that apply a particular force along the rotational axes. To power the motors, we use the power provided by the battery pack. In Figure 6-1, we represent with dotted lines the current and with continuous lines the control signals. The battery pack also applies current to the flight controller but the voltage is lower (typical 5 volts), so we need a voltage converter called *BEC*. To support all these elements we need a rigid structure, called *frame* that could hold the motors using two kinds of configuration: X or +. Figure 6-2 shows the two possible frame configurations. The combination of the forces generated by the motors induces forces and torques to the body frame and the frame configuration must be considered when computing the motor control inputs.

There is not a big difference between the two configurations, historically the

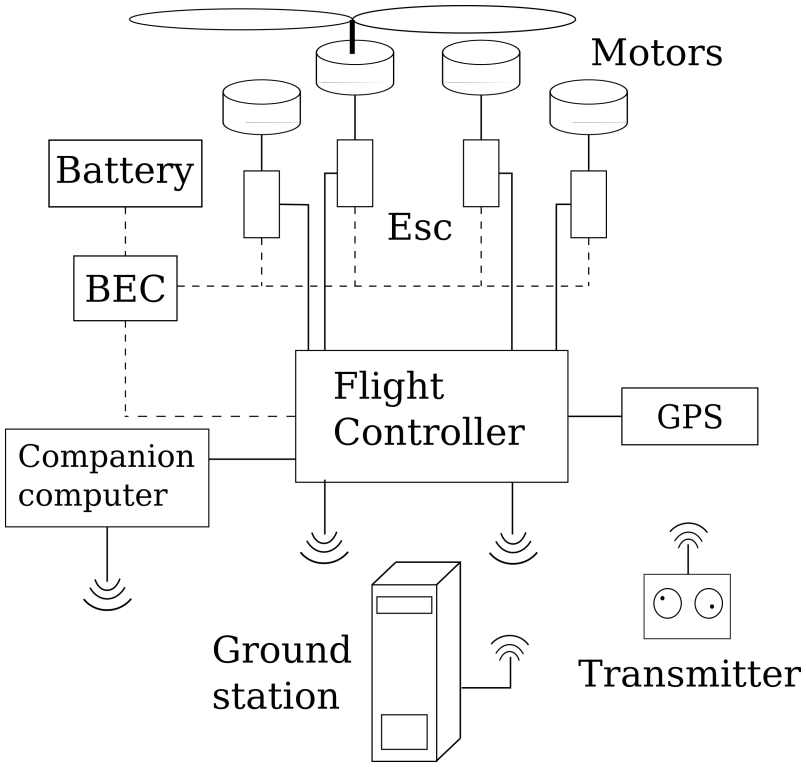

Figure 6-1: An overview of the main hardware components

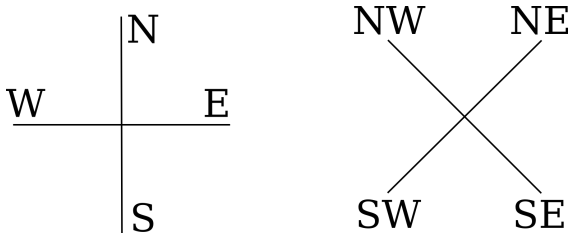

Figure 6-2: The possibile frame configurations

first quad-rotors developed by hobbyists, research centers and universities had a +-configuration because it was more natural to generate a horizontal motion (i.e., it is the difference between the front and back motors). When cameras and sensors were mounted on quad-rotors, it was more convenient to use the X-configuration because there was more space in the front direction. If we assume that the front of the frame is heading North, it is possible to identify a specific motor in a unique way using the first letter of the cardinal points (North, South, North-East, South-East etc.).

6.1.1 ESCs

We can control the forces generated by the propellers setting the right amount of current to the motors. The role of the *ESC* (Electronic Stability Control) is to give a constant amount of current to the motors proportional to the voltage coming from the control signal also considering the voltage level of the battery pack. Figure 6-3 shows a typical ESC connected to a battery and to generic flight controller that we will present in the next Section. ESCs need to be calibrated to set control voltages corresponding the maximum or the minimum amount of current to motors so usually the firmware present in the flight control implements also a method to calibrate it. To control these modules, the flight controller sends a numerical value (usually a positive integer), and it is not possible to control directly the forces and the velocities of the propellers.

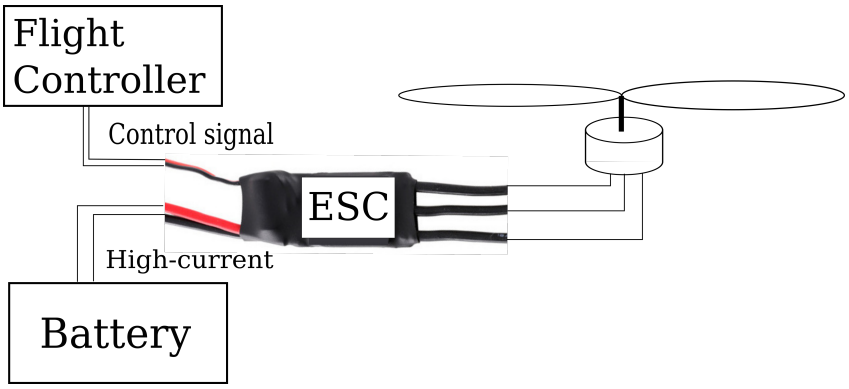

Figure 6-3: A tipical ESC connected to a motor

In Section 6.3 we present a bench that we built to identify the relationship between the control inputs and the forces applied by the propellers to the quadrotor frame.

6.1.2 Flight Controller

The *flight controller* is the "core" of the systems and implements the control strategy to maintain a desired attitude or position. The firmware runs on a real-time embedded system with sensors and communication devices. All the information coming from the sensors are processed by this software that implements device drivers and control loops for the ESC as presented earlier.

With the support of Figure 6-4, we present now a generic functional architecture showing the modules implemented on a typical research flight controller.

- Mixing and ESC control: in this part, there are functions that translate the command coming from the controller into a control signal to the ESCs. In

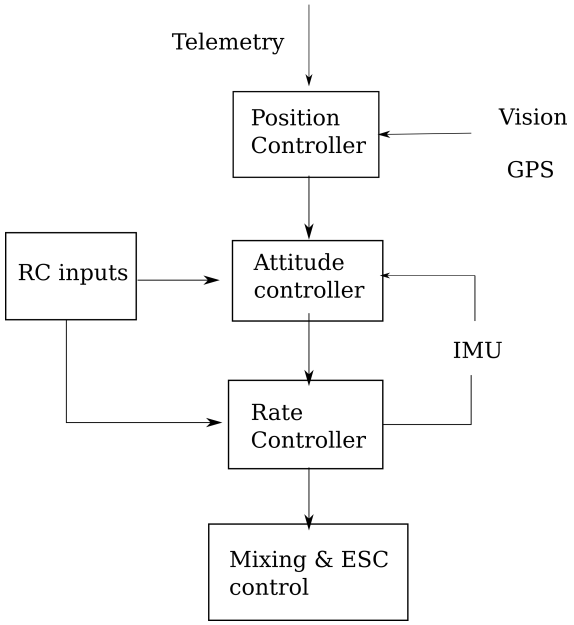

Figure 6-4: A functional architecture of a typical flight controller

particular, based on the configuration frame the mixing function translates the roll, pitch, yaw, thrust rate commands into control commands for the NE, NW, SE, SW motors (in the case of the X configuration).

- Rate controller: the aim of this feedback controller is converting the velocity commands coming from the attitude controller into inputs to pass to the mixer and the EScs. The velocities of the quadrotor are estimated using IMU, in particular, the gyroscope. We have only a proportional constant, Kr, which relate the error between the desired velocities and the velocities estimated by the IMU. It is also possible to bypass the attitude controller and give the desired velocities directly from the transmitter used by the operator or the ground-station.

- Attitude controller: The attitude controller is a position/orientation feedback controller that converts quadrotor positions into reference velocities for the rate controller. The desired position comes from the ground-station which gives the plan of the mission, usually through GPS waypoints. The current position of the quadrotor could be estimated using the GPS or vision systems if the drone is used in an outdoor environment, otherwise it is possible to use a visual tracker such as the Optitrack or Vicon systems.

- Position controller: the position controller use the feedback data from an

external sensor (see sec 6.1.4 for details) computing commands to send to the Attitude controller as shown in the picture.

To support many flight modes the commercial flight controllers implement all the previous modalities but, in our experimental setup, we use only the attitude controller to apply the trajectory planned for the "pitch" degree of freedom and the position controller maintaining a constant value for y, z, yaw DoF.

6.1.3 Ground-station

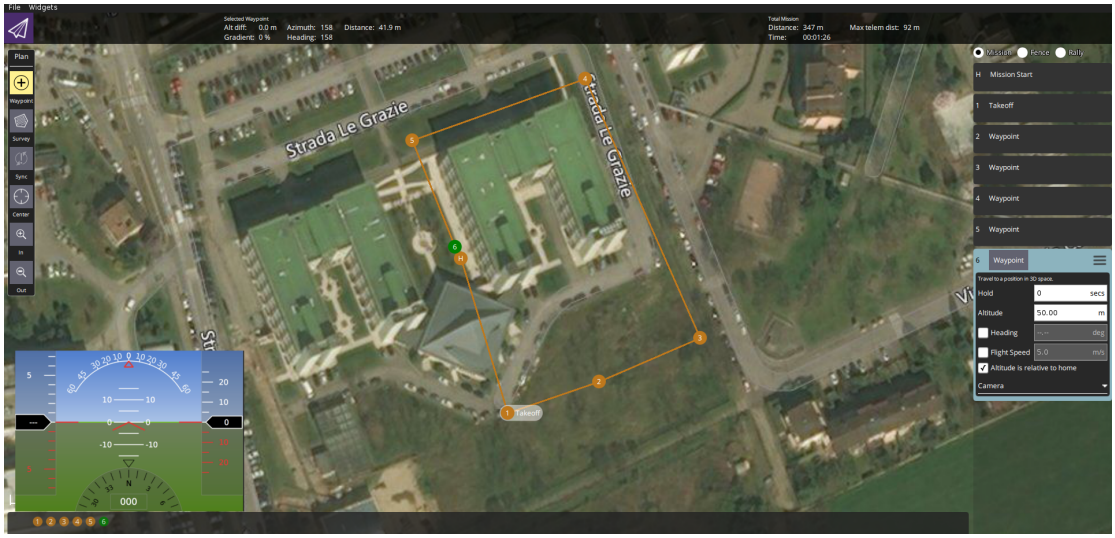

Figure 6-5: A screen-shot of the QGroundstation software

With *Groundstation* we identify a computer or a tablet used to monitor the parameters of the quadrotor and to execute a teleoperated or autonomous flight planned using a sequence of GPS waypoints. The connection between the Ground-station and the flight controller is called *telemetry* and usually is a radio connection (450 or 900 MHz) but it is possible to make a telemetry connection using Bluetooth, X-Bee or Wi-Fi (2.4 GHz) protocols. Trough the telemetry it is possible to receive all the onboard data such as the state of IMU, battery voltage, error checks but is also possible to over-write the on-board parameters. In the research area, it is common to use the telemetry to overwrite the control commands and to do autonomous flights.

6.1.4 Position estimation

The Global Position System (GPS) is a technology that is used to estimate the outdoor position of people of devices such as quadrotors. These systems are not precise

Figure 6-6: The Px4Flow visual odometry device and the OptiTrack system. Credits: www.optitrack.com

enough (around 2m) to be used for feedback in position controller. For this reason, researchers use two different methods to improve the position estimation:

- Visual tracker: a visual tracker system is a set of infra-red cameras (usually ten or more) that identify a pattern of markers put on an object, estimating position and orientation of a rigid body with a precision of $\tilde{0}.2$ mm. The most used tracker camera systems are the VICON and the Optitrack.

- Visual devices: Another way to estimate the position of a quadrotor is to use "simple" cameras and to do visual odometry implementing algorithms which run on the companion computers or on-chip. Examples of this kind of devices are the Px4FLow [43], which is possible to see in Figure 6-6 or the new advance flight controllers such as the Qualcomm Snapdragon Flight Kit or the Intel Aero Computer Board that have onboard cameras.

In our setup we use a visual tracker system, in particular, the "Optitrack", to receive the feedback for positions (y,z) and we also use it to log the dynamic evolution of the axes "x", under the effect of the controller trajectory in "$(pitch)$".

6.2 Thesis setup

After presenting the elements that compose a general setup, in this Section, we present the custom setup built to test the method proposed in Chapter 5. After many trials with popular "open-source project" such as PX4 [65], Ardupilot [1] etc. We found that these projects are too complex and have many bugs and issues. For this reason, we decided to join the "Librepilot" project, presented in the next section, contributing to the project with a custom plugin. With the help of Librepilot community, we build a custom quadrotor, called "Chirottero", presented in the next Section.

Figure 6-7: The quadrotor used in the experimental phase

6.2.1 Chirottero miniquad

Using the OpenScad software [2], we designed the frame, making the necessary plastic pieces using a 3D printer. All the pieces are linked together using carbon fiber sticks and the result is a class 150 quadrotor, which means that the distance between the NE and SW motors is 150 mm.

All files and the instructions to build Chirottero are present on the website of the author of this thesis [3] but we show the parts lists so that the reader can buy the components from the toy market and reproduce the experiment:

- Motors: 8520 8.5x20mm 53500RPM Coreless Motor
- Telemetry: HC-06 bluetooth module
- Propellers: 65mm blade propellers
- Flight controller: CC3D brushed FC board.
- Battery: 3.7V 500mAh 20C Battery pack

The total cost of the quadrotor is around 50 €, which is an affordable cost for an experimental setup usable in many contexts (research, education etc.).

[1] `http://ardupilot.org/`
[2] `www.openscad.org/`
[3] www.boriero.it

6.2.2 Librepilot project

To control Chirottero, we need a firmware that implements the control algorithms presented in Chapter 6.1 and a Ground-station that communicates, using the Telemetry connection, with the on-board controller.

Figure 6-8: The Librepilot user interface

The Librepilot project in an open-source project active since 2015. The goal of the project is to provide an open environment where the contributors can add new projects and support new hardware for unmanned autonomous vehicle and robots. The Librepilot share its code under the GPLv3 license and it is a friendly environment where it is possible to develop a research activity, as we do in this thesis.

In Figure 6-9 it is possible to see the user interface of the plugin wrote to implement the method presented in this thesis. The plugin receives the position of the quadrotor using the "Natural Point Optitrack system" visual tracker trough the VRPN [4] communication protocol and visualize the useful information in on the top-left corner. In the upper part of the plugin is also present a Position controller where is possible to set a desired position and "Yaw" orientation and where it is possible to tune the PID parameters in a simple way.

In the lower part of the screen, it is possible to load a comma-separated-value (CSV) file containing the optimized trajectory with the desired values for the "pitch" DoF. When the user presses the "Start" button, the Position controller controls the axis (y, z, yaw) meanwhile the *pitch* angle is controlled by the values loaded from

[4] `https://github.com/vrpn/vrpn/wiki`

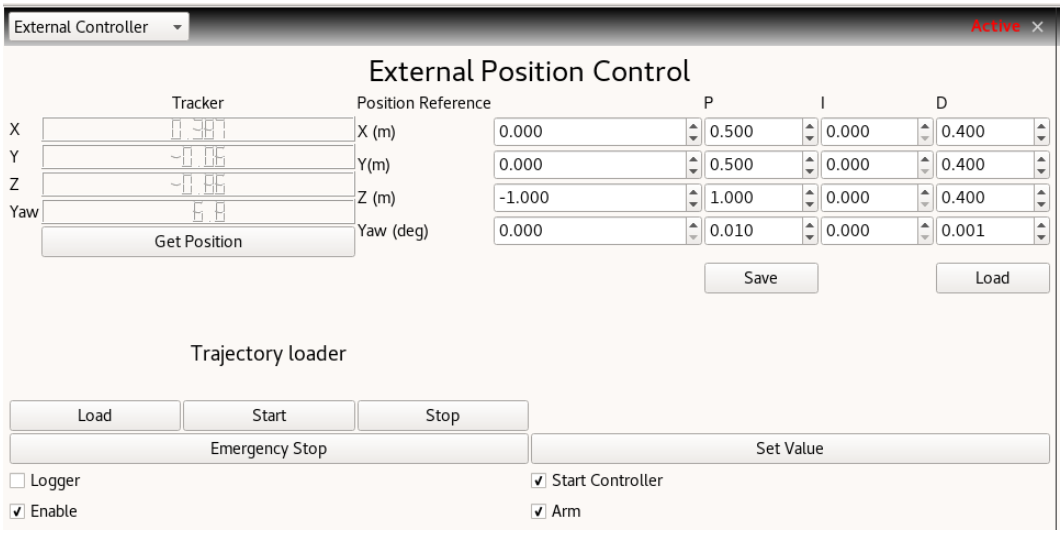

Figure 6-9: The plugin interface used in the experimental phase

the planned trajectory file. The evolution of the x DoF is output of the system dynamics.

6.3 Dynamic parameters identification

When the hardware and the software one ready, we need to identify the dynamic parameters of the flying robot to specify the dynamic moodel of the trajectory planner presented in Chapter 5. In particular, we need to know the mass of the quadrotor and the inertia matrix.

6.3.1 Dynamic identification

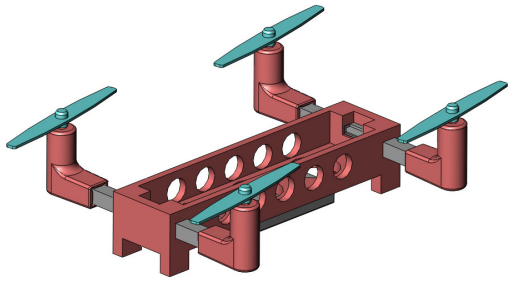

Figure 6-10: The rendering of the CAD model generated to identify the inertia matrix

To identify the mass m we used a precise balance board and the result is a mass of 0.08 kg

The inertia matrix I is more complex to identify and there are many techniques but, in our case, we decided to compute it using the CAD software "SolidWorks". The 3D files that we printed to obtain the quadrotor frames, were put in the software (see Figure 6-10), including the correct mass of each piece.

The result of the inertia matrix calculation is:

$$I = \begin{bmatrix} 76591.06 & 0.00 & -91.57 \\ 0.00 & 235968.68 & 0.00 \\ 91.57 & 0.00 & 183829.45 \end{bmatrix} \tag{6.1}$$

The values are expressed in $g * mm^2$

6.3.2 Motor identification

After the characterization of the dynamic parameters of the experimental setup, we need a method that converts the numerical values sent by the FC to the ESC to set the motor values. To this end, we prepared a benchmark composed of a load cell and the same flight controller mounted on the Chirottero mini-drone.

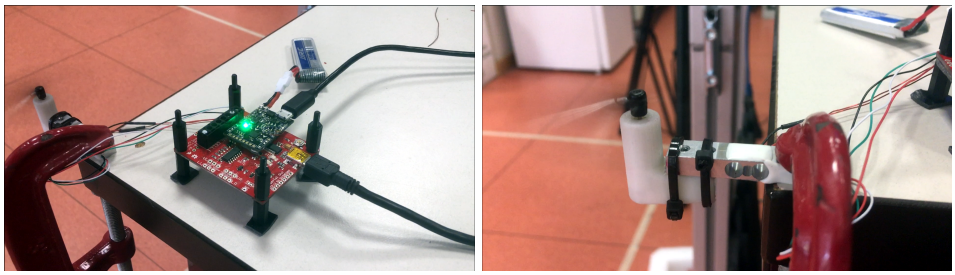

Figure 6-11: The benchmark to identify the relationship between command values and forces/torques applied

The FC and the load-cell are connected to a standard Linux-based PC to log the data during the identification process. We divided the range of all possible output values $[0-255]$ in 17 samples and we sent these numerical input values to the ESC. For each control value, we measured, using the load cell, the force applied by the rotating blade. Finally, we fit a linear function estimating a constant C_f that convert output commands in forces. Figure 6-12 shows the results. After estimating the dynamic parameters of Chirottero, in the next Section we use them to demonstrate that the method proposed in Chapter 5 can be applied to a real setup.

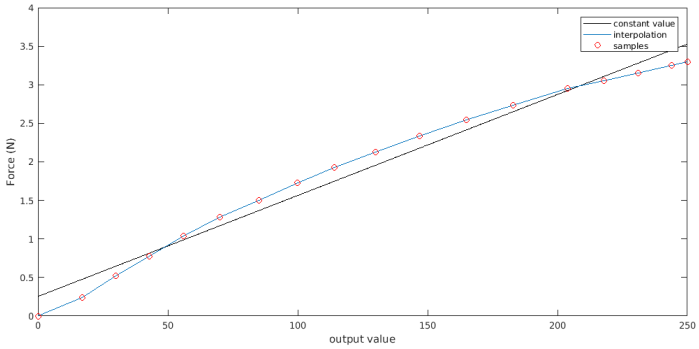

Figure 6-12: This figure shows the relation between the input control values and the forces applied by the rotors

6.4 Task results

6.4.1 Task description

To test the task proposed in Section 5.5.2 we build the following setup. We prepared the flying arena shown in Figure 6-13 where a visual tracker is able to acquire the position and the orientation of the Chirottero mini-drone. The task is to start from an initial position $x_0 = 0.0m$ that is 1 meter far from the obstacle, which simulates a deep gap in a wall. The quadrotor has to start from the initial configuration and acquire enough energy to arrive just before the gap with a pitch angle equal to zero and with the desired momentum px_f. The initial configuration point is $x_0 = 0$ m, $\theta_0 = 0$ deg and the final configuration is $x_f = 1.04$ m and $\theta_f = 0$ deg. We also want to have a final momenta $px_f = 0.016\ K \cdot \frac{m}{s}*$. Remembering that the momenta is $px = m \cdot \dot{x}$, the final desired velocity is $\dot{x}_f = 0.2\frac{m}{s}$.

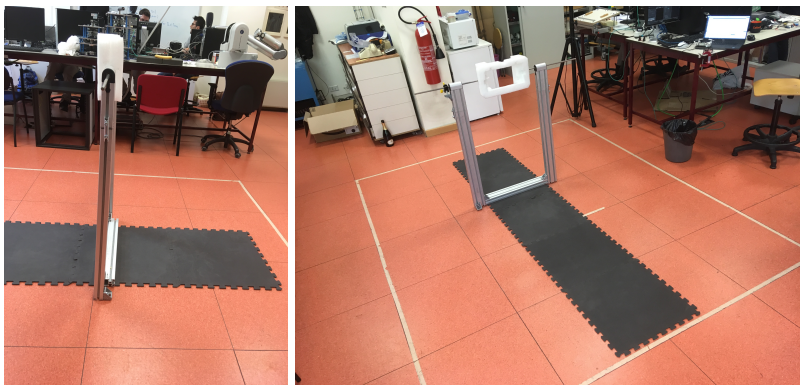

Figure 6-13: The flying arena with the obstacle simulating a deep gap in a wall

The initial conditions, the final conditions and the dynamic parameters are used

with the Python program described in Section 5.5.2 to plan a feasible trajectory that we use to control the pitch angle. In particular, using the telemetry connection from the Librepilot plugin we send the pitch trajectory to the attitude controller implemented on the flight control board of the "Chirottero" quadrotor. In the next Section, we present the results obtained.

6.4.2 Results

In this Section, we present the results of the task of trajectory generation. Figure 6-14 shows a snapshot of the video of the quadrotor passing through the gap using the energy acquired with the motion.

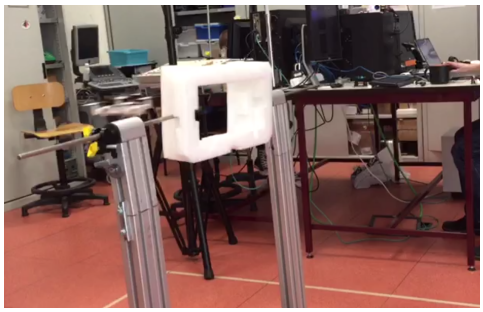

Figure 6-14: Snapshot of the video showing the task proposed (`https://youtu.be/F96YewACfdU`)

Figure 6-15 display plots of the results. In all plots, there are two curves, one blue and one orange. The blue curve is the trajectory computed by the Python script implementing the method presented in Chapter 5 and the orange curve is the position of the quadrotor recorded using the visual tracker. To have a common timeline, the two data are synchronized according to the timeline given by the tracker. In the first figure, it is possible to see the planned and the resulting pitch, in the second and third figures the evolution of the x axis of the momenta. The last Figure shows a lot of noise affecting the orange curve because we had to compute the momenta from the position trajectory logged by the visual tracker so using numerical derivation with respect the time and this operation introduces noise.

6.5 Conclusion

In this Chapter, we demonstrate that the theory presented in Chapter 5 can be applied to a real experimental setup. After building a custom mini quad-rotor, we used the trajectory planned to drive the flying robot through a deep gap. The novel idea is to impose a final condition on the momenta so it is possible to use the energy stored

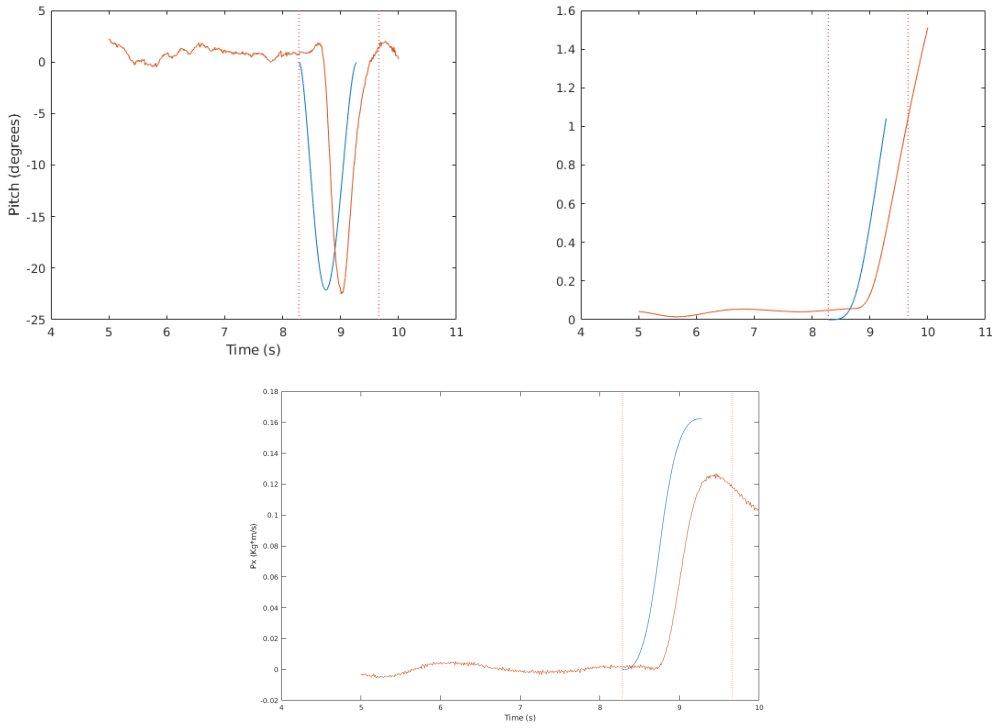

Figure 6-15: The results of the task for the pitch, x and px variables. In blue it is possible to see the reference trajectory and in orange the trajectory logged by the visual tracker

to drive the quadrotor along the non-actuated degrees of freedom x maintaining a *pitch* angle equal to zero.

Chapter 7

Conclusions and future works

7.1 Motivations

In the last decade, robotic autonomy has had a big impact on industrial and academic research. In particular, since the begin of the new millennium, a new flying robot has been developed that will give a strong contribution to autonomous maneuvers: the multi-rotor. A multi-rotor is a small and relatively cheap unmanned aerial vehicle (UAV) that can be a valid replacement for helicopters in many fields such as video-shooting or when small payloads needed to be moved. The main problem of multi-rotors is the high number of Dof and has challenging dynamics, introduced by the fact that is an Under-actuated Mechanical System (UMS). To increase the autonomy of multi-rotors, we are interested to plan trajectories that can do maneuvers that are difficult for a human to command, for example passing through gaps that are generically oriented in the space. In our literature review, we identify two main areas that can be improved:

- Most of the works are based on the "Flatness property" that uses external position sensors, such as the GPS system or visual trackers, to transform the controlled attitude degrees of freedom in controlled position DoF $(roll, pitch, yaw, z) \rightarrow (x, y, z, yaw)$. This is useful if we assume that the multi-rotor has a constant value for the "roll" and "pitch" angles but if we move far from the hovering condition, for example, to do aggressive maneuvers, we need to switch between attitude and the position controllers.

- When the authors plan a trajectory from an initial configuration to a final configuration, in general, they use local methods that connect, using probabilistic or numerical algorithms, a sequence of feasible segments. The problem of these methods is that the planner does not consider the dynamics of the system and the dynamic constraints along the whole path, and it is only possible to do an analysis of the constraints in the aftermath.

Motivated by these points, we decided to see the problem of planning a feasible trajectory of an under-actuated mechanical system from a global perspective. To this end, we study the tools and the methods used in geometric mechanics, in particular, the Hamiltonian formalism, to understand the geometry of the problem and to propose a novel approach.

7.2 Summary of the thesis

In the first Chapter, we presented the introduction to the trajectory planning and the motivations that inspired this thesis, showing the current challenges in robotics and the "multi-rotor" systems that have still interesting problems to overcome.

In the Chapter 2 there is an overview of the methods presented in the literature about the motion planning problem. In this Chapter, we showed the difference between local methods and global methods to solve a motion planning problem and the main techniques to do trajectory planning for Under-actuated mechanical systems.

In this Chapter 3 there is the mathematical background useful to understand the trajectory planning problems for UMS. The knowledge presented comes from a research area knows as "geometrical mechanics" refering to the work done by Marsden [62], Cortes [20], Bullo [17], Bloch [8] and others. Starting from basic geometric mechanics definitions, we introduced the Hamiltonian equations that are fundamental tools to apply the methods proposed in the following Chapters.

Chapter 4 proposes the application of the "Variational constrained system approach" using the Hamiltonian framework to solve the problem of trajectory planning for underactuated mechanical systems. The example proposed to show the benefit of the method is the classical "cart-pole" system and to find a solution, we applied an indirect method to search for admissible momenta that plan an optimal trajectory.

In Chapter 5 we use the Hamiltonian formalism to solve the trajectory planning problem for generic under-actuated mechanical systems (UMS). The Hamiltonian equations can explicitly describe the evolution of the energy stored in the system and make it possible to use a numerical method to compute an optimized trajectory, not only for state variables but also for momenta. As a practical example, we proposed the problem of passing with a quadrotor through a small gap in a deep wall.

In Chapter 6 we tested the method proposed in Chapter 5 building a real setup scenario. We built a custom low-cost quadrotor and a software plugin that implement a position controller and can load and execute the planned trajectory. Finally, we analyzed the results showing that imposing final conditions to momenta, it is possible to drive the quadrotor in the directions that are not directly controllable.

7.3 Summary of contributions

In the Chapters of this thesis we gave the following contributions to the problem of planning a feasible trajectory for UMS:

7.3.1 Study of the problem

Our first contribution, is the analysis of literature regarding trajectory planning for mechanical systems, a research field active since the early 80's. In Chapter 2 we classified the trajectory panning methods in "Global" and "Local" methods and we decided to focus on the global methods. In this sub-class of algorithms, we propose another classification based on the approaches used by the authors: methods using specific equation structures, indirect methods and direct methods.

7.3.2 Application of geometric methods to a classic control problem

After finding the issues introduced by the UMS in the trajectory planning problem, we found an approach called "variational constrain problem" that was proposed in the field of geometric mechanics to solve the problem of UMS constraints. In Chapter 4, we applied this method to the problem of planning a feasible trajectory for a cart-pole system. The results of Chapter 4 were published in [11].

However we found that is is not possible to extend the method to multi-rotors because they do not satisfy the necessary conditions imposed by the method proposed in Chapter 4. As consequence we discovered that this method can be applied only to "super-articulated mechanical systems" and multi-rotors are not part of this class of UMS.

7.3.3 A novel method to plan a trajectory for generic UMS

From the experience gained with the cart-pole example, we understood that the Hamiltonian formalism can be used to describe the evolution of the energy stored in the UMS. For this reason in Chapter 5 we wrote the Hamiltonian equations of quadrotor and we applied a classical numerical method, called "direct collocation method", to plan a feasible trajectory to drive the quadrotor from an initial configuration to a final configuration. To demonstrate the benefits of the method, in Chapter 6 we built an experimental setup and carry out a task never done before: "flying through a small gap in a deep wall". This method uses the energy stored during the motion of the quadrotor to overcome the limitations introduced by the under-actuation constraints.

7.4 Future work

Despite the efforts spent to understand the geometric nature of the planning feasible trajectory for generic UMS, it is still not clear why the method proposed in Chapter 4 cannot be applied to all kind of UMS, In particular, we think that the under-actuated constraints live in the cotangent space of the tangent space $T^*(TQ)$ where it is possible to search for an optimal trajectory. Some authors use the Lagrangian formalism and they consider the UMS constraints as "second-order nonholonomic constraints", living in the second-tangent space T^2Q but a method that works with these kinds of constraints has not been proposed yet. In the future, we are interested in discussing with the robotic and geometric mechanic communities to search for a correct geometric description of the UMS constraints and we want to search an optimal control strategy to solve the problem of trajectory planning for generic UMS.

In Chapter 5 we applied a numerical method that returns feasible solutions but does not search for optimal solutions using a functional cost. In future work, we want to use the optimal control techniques, such as the "Pontryagin Maximum Principle", to compute optimal solutions. The methods presented in this thesis consider only constraints affecting the dynamics of the system, we plan to consider obstacles that could be present in the space between the initial configuration and the final configuration.

The flight arena presented in Chapter 6 has a small (less than 6 m^2) surface and we could not test aggressive maneuvers that need more space to be executed. We plan to find a bigger arena or using a quadrotor that can fly outdoor to have enough space to test the method presented in Chapter 5.

Appendix A

Basic concepts of Differential geometry

Differential geometry is a mathematical area that solves problems with the help of geometric spaces allowing also to reach computational solutions. In the context of this thesis, this tool is useful to find a smooth curve that connect two points considering the dynamic constraints of a system.

This section presents the mathematical basis of Topology and differential geometry useful to understand a geometric approach to nonholonomic motion planning. Section A.1 presents some topology notions taken from the book [68]. Section A.3 presents differential geometry concepts taken from the book [95] but with a modified notation that could be compatible with "A.Block-Nonholonomic mechanics and control" book [8].

A.1 Topological space

Definition A.1. *A* topology *on a set X is a collection $\mathscr{T}$ of a subset of X having this properties:*

- $\emptyset$ *and X are in* $\mathscr{T}$
- *The union of the elements of any sub-collection of* $\mathscr{T}$ *is in* $\mathscr{T}$
- *The intersection of the elements of any finite sub-collection of* $\mathscr{T}$ *is in* $\mathscr{T}$

A set X for which a topology $\mathscr{T}$ *is specified is called a* topologycal space

Definition A.2. *If X is a topological space with topology* $\mathscr{T}$*, we say that a subset U of X is and* open set *of X if U belongs to the collection* $\mathscr{T}$

Definition A.3. *let* $\pi_1 : X \times Y \to X$ *be defined by equation*

$$\pi_1(x,y) = x; \tag{A.4}$$

and let $\pi_2 : X \times Y \to Y$ *be defined by equation*

$$\pi_2(x,y) = x; \tag{A.5}$$

The maps π_1 *and* π_2 *are called the* projections *of* $X \times Y$ *onto its first and second factors, respectively.*

Definition A.6. *If* X *is a set, a* basis *for a topology on* X *is a collection* $\mathscr{B}$ *of subset of* X *such that*

- *For each* $x \in X$*, there is at least one basis element* B *containing* x*.*
- *If* x *belongs to the intersection of two basis elements* B_1 *and* B_2*, then there is a basis elements* B_3 *containing* x *such that* $B_3 \subset B_1 \cap B_2$

Definition A.7. *A subset* U *of* X *is said to be an* open subset *in* X *if, for each* $x \in U$*, there is a basis element* $B \in \mathscr{B}$ *such that* $x \in B$ *and* $B \subset U$*. If* U *is an open set containing* x*, we can said that "*U *is a* neighbourhood *of* x*"*

Definition A.8. *Let* X *and* Y *be topological spaces. The* product topology *on* $X \times Y$ *is a topology having as basis the collection* $\mathscr{B}$ *of all set*

Definition A.9. *Let* X *be a topological space with topology* $\mathscr{T}$*. If* Y *is a subset of* x*, the collection*

$$\mathscr{T}_Y = \{Y \cap U | U \in \mathscr{T}\} \tag{A.10}$$

is a topology of Y*, called* subspace topology*. With this topology,* Y *is called* subspace *of* X

Definition A.11. *A subset* A *of a topological space* X *is said to be* closed *it the set* $X \setminus A$ *is open.*

Definition A.12. *A topological space* X *is called* Hausdorff space *if for each pair* x_1, x_2 *of distinct points of* X*, there exits neighborhoods* U_1 *and* U_2 *of* x_1 *and* x_2*, respectively, that are disjoint.*

A.2 Continuous functions

Definition A.13. *Let* X *and* Y *be topological spaces. A function* $f : X \to Y$ *is said to be* continuous *if, for-each open subset* V *of* Y*, the set* $f^{-1}(V)$ *is an open subset of* X

Definition A.14. *Let* X *and* Y *be topological spaces; let* $f : X \to Y$ *be a bijection. If both the function* f *and the inverse function*

$$f^{-1} : Y \to X \tag{A.15}$$

are continuous, the f *is called* homomorphism*.*

Definition A.16. *Let $U \subset \mathbb{R}^n$ be open, and let $f : U \to \mathbb{R}$. We say that f is a* differentiable of class C^k *on U if the partial derivative $\frac{\partial^i}{\partial r^i}$ exists and are continuous on U for $i = 1 \ldots k$. We say that f is C^∞ and a* smooth function *if is it C^k for all $k \geq 0$.*

A.3 Differentiable manifolds

Definition A.17. *A* locally Euclidean space *M* of dimension *d is a Hausdorff topological space M for which each point has a neighbourhood homomorphic to an open subset of Euclidean space R^d. If φ is a homomorphism of a connected open set $U \subset M$ onto a open subset of R^d, φ is called* coordinate map*, the functions $x_i = r_i \circ \varphi$ are called* coordinate functions *and the pair (U, φ) is called a* coordinate system*.*

Definition A.18. *A* structure $\mathscr{F}$ of class C^k *on a locally Euclidean space M is a collection of coordinate systems $\{(U_\alpha), \varphi) : \alpha \in A\}$ satisfying the following properties:*

1. $\cup_{\alpha \in A} U_\alpha = M$
2. $\varphi_\alpha \circ \varphi_\beta$ *is C^k for all $\alpha, \beta \in A$.*
3. *The collection $\mathscr{F}$ is maximal with respect to point 1 and 2;*

Definition A.19. *a* d-dimensional differentiable manifold of class C^k *is a pair $(M, \mathscr{F})$ consisting of a d-dimensional, second countable, locally Euclidean space M with a differentiable structure $\mathscr{F}$ of class C^k.*

A.4 The tangent space

Definition A.20. *Let $m \in M$. Functions f and g defined on open sets containing m are said to have the same* germ at *m, if they agree on some neighborhood of m.*

Definition A.21. *Two functions being* equivalent *if and and only if they have the same germ. The equivalence classes are called* germs *and we denote them with $\tilde{F}_m$. If f is a smooth function on a neighborhood of m, we denote* f *its germ.*

Definition A.22. *A* tangent vector *v at a point $m \in M$ is a linear derivation of the algebra $\tilde{F}_m$, That is, for all f, $g \in \tilde{F}_m$ and $\lambda \in \mathbb{R}$,*

1. $v(\mathbf{f} + \lambda \mathbf{g}) = v(\mathbf{f}) + \lambda v(\mathbf{g})$
2. $v(\mathbf{f} \cdot \mathbf{g}) = \mathbf{f}(m) v(\mathbf{g}) + \mathbf{g}(m) v(\mathbf{f})$

T_mM denotes the set of tangent vectors to M at m and is called tangent space.

Corollary A.23. *$dim(M_m) = dim(M)$*

Definition A.24. *Let (U,φ) be a coordinate system with coordinate functions $x_1,\ldots,x_d$ and let $m \in U$. For each $i \in (1,\ldots,d)$, we define a tangent vector $(\partial/\partial x_i)|_m \in M$ by setting*

$$\left.\frac{\partial f}{\partial x_i}\right|_m = \left.\frac{\partial (f\circ\varphi^{-1})}{\partial r_i}\right|_{\varphi(m)} \tag{A.25}$$

for each function f which is smooth in a neighborhood of m. We interpreter the previous equation as the directional derivative *of f at m in the x_i coordinate direction.*

A.5 Tangent vectors

Definition A.26. *Let $f : M \to N$ be a smooth function and let $m \in M$. The* derivative of *f is the linear map*

$$T_mf : T_mM \to T_{f(m)}N \tag{A.27}$$

defined as follows. If $v \in T_mM$, then $T_mf(v)$ is to be a tangent vector at $f(m)$, so we describe how it operates on functions.

Definition A.28. *Let $(U,x_1,\ldots,x_d)$ and $(V,y_1,\ldots,v_l)$ be coordinates systems about m and $\psi(m)$ respectively. Following the previous definition we can write*

$$d\psi\left(\left.\frac{\partial}{\partial x_y}\right|_m\right) = \sum_{i=1}^{l} \left.\frac{\partial (y_i\circ\psi)}{\partial x_j}\right|_m \left.\frac{\partial}{\partial y_i}\right|_{\psi_m} \tag{A.29}$$

The matrix $\partial(y_i\circ\psi)\ \partial x_i$ is called the Jacobian *of the map ψ respect the coordinate system*

Definition A.30. *A smooth mapping $\sigma : (a,b) \to M$ is called a* smooth curve *in M. Let $t \in (a,b)$, then the* tangent vector *to the curve σ at t is the vector*

$$d\sigma\left(\left.\frac{d}{dr}\right|_t\right) \in T_{\sigma(t)}M$$

we denote the tangent vector to σ at t by $\dot{\sigma(t)}$

Definition A.31. *Let M a smooth manifold with differentiable structure $\mathscr{F}$. Let*

$$TM = \cup_{m\in M} T_mM$$

with smooth map called natural projection*:*

$$\tau : TM \to M,\ \tau(v) = m \text{ if } v \in T_m M$$

Let $(U,\varphi) \in \mathscr{F}$ with coordinate functions $x_1,\ldots,x_d$. Define $\tilde{\varphi} : \tau^{-1}(U) \to \mathbb{R}^{2d}$ by

$$\tilde{\varphi} = (x_1(\tau(v)),\ldots,x_d(\tau(v)),dx_1(v),\ldots,dx_d(v)) \tag{A.32}$$

for all $v \in \tau^{-1}(U)$. We can now construct a topology and a differentiable structure on TM:

1. *if (U,φ) and $(V,\psi) \in \mathscr{F}$, then $\tilde{\psi} \circ \tilde{\varphi}^{-1}$ is smooth.*
2. *the collection $\{\tilde{\varphi}^{-1}(W) : W \text{ open in } \mathbb{R}^{2d}, (U,\varphi) \in \mathscr{F}\}$ forms a basis for a topology on TM*
3. *Let $\tilde{\mathscr{F}}$ be a maximal collection, respect to point 2, containing $(\tau^{-1}(U),\tilde{\varphi}) : (U,\varphi) \in \mathscr{F}$ then $\tilde{\mathscr{F}}$ is a differentiable structure on TM.*

TM with this differentiable structure is called tangent bundle.

Definition A.33. *Let $\psi : M \to N$ be a smooth function.*

1. *ψ in an* immersion *if $d\psi_m$ in non-singular for each $m \in M$*
2. *The pair (M,ψ) is a* submanifold *of N if ψ is a one-to-one immersion*
3. *ψ in an* imbedding *if ψ is a one-to-one immersion which is also a homeomorphism into; that is, ψ is open as a map into $\psi(M)$ with the relative topology.*
4. *ψ is a* diffeomorphims *if ψ maps M one-to-one onto N and ψ^{-1} is smooth*

A.6 Vector fields

Definition A.34. *A mapping $\sigma : [a,b] \to M$ is a* smooth curve *in M if σ extend to be a smooth mapping of $(a-\varepsilon,b+\varepsilon)$ into M for some $\varepsilon > 0$. If $\sigma : [a,b] \to M$ is a smooth curve in M, then its tangent vector*

$$\dot{\sigma}(t) = d\sigma\left(\left.\frac{d}{dr}\right|_t\right) \in T_{\sigma(t)}M$$

Definition A.35. *A vector field X along a curve $\sigma : [a,b] \to M$ is a mapping $\mathscr{X} : [a,b] \to TM$ which* lifts *ω; that is $\tau \circ \mathscr{X} = \tau$. A vector field $\mathscr{X}$ is called a* smooth vector field *along τ if the mapping $\mathscr{X} : [a,b] \to TM$ is C^∞ that is a map $\mathscr{X} : U \to TM$ such that*

$$\tau \circ \mathscr{X} = \text{identity map on } U \tag{A.36}$$

Definition A.37. *If $\mathscr{X}$ and $\mathscr{Y}$ are smooth vector fields on M, we define a vector field $[\mathscr{X},\mathscr{Y}]$ called* the Lie bracket *of $\mathscr{X}$ and $\mathscr{Y}$ by setting*

$$[\mathscr{X},\mathscr{Y}]_m(f) = \mathscr{X}_m(\mathscr{Y}f) - \mathscr{Y}_m(\mathscr{X}f) \tag{A.38}$$

Proposition A.39. *Lie bracket have the following properties:*

1. *$[\mathscr{X},\mathscr{Y}]$ is ineeded a smooth vector field on M*
2. *if $f,g \in C^\infty(M)$, then $[f\mathscr{X},g\mathscr{Y}]_m(f) = fg[\mathscr{X},\mathscr{Y}] + f\mathscr{X}g\mathscr{Y} - g(\mathscr{Y}f)\mathscr{X}$*
3. *$[\mathscr{X},\mathscr{Y}] = -[\mathscr{Y},\mathscr{X}]$*
4. *$[[\mathscr{X},\mathscr{Y}],\mathscr{Z}] + [[\mathscr{Y},\mathscr{Z}],\mathscr{X}] + [[\mathscr{Z},\mathscr{X}],\mathscr{Y}] = 0$ for all smooth vector fields $\mathscr{X},\mathscr{Y}$ and $\mathscr{Z}$ on M*

A.7 Distributions and Frobenius theorem

Definition A.40. *Let c an integer such that $1 \leqslant c \leqslant d$. A* c-dimensional distribution *$\mathscr{D}$ on a d-dimensional manifold M is a choice of a c-dimensional subspace $\mathscr{D}(\Updownarrow)$ of T_mM for each $m \in M$.*

Definition A.41. *A distribution $\mathscr{D}$ is* smooth *if for each $m \in M$ there is a neighborhood U of m there are c vector fields $\mathscr{X}_\infty,\ldots,\mathscr{X}_\rfloor$ of class C^∞ on U which span $\mathscr{D}$ at each point of U.*

Definition A.42. *A vector field $\mathscr{X}$ on M is said to* belong to (or lie to) *the distribution $\mathscr{D}(\mathscr{X} \in \mathscr{D})$ if $\mathscr{X}_m \in \mathscr{D}(m)$ if $\mathscr{X}_m \in \mathscr{D}(m)$ for each $m \in M$.*

Definition A.43. *A smooth distribution $\mathscr{D}$ is called* involutive or completely integrable *if $[\mathscr{X},\mathscr{Y}] \in \mathscr{D}$ whenever $\mathscr{X}$ and $\mathscr{Y}$ are smooth vector fields lying on $\mathscr{D}$.*

Theorem A.44. ***Frobenius theorem:*** *Let $\mathscr{D}$ be a c-dimensional, involutive, smooth distribution on M^d. Let $m \in M$. Then there exists an integral manifold of $\mathscr{D}$ passing through m. Indeed. there exists a cubic coordinate system (U,φ) which is centered at m, with coordinate functions $x_1,\ldots,x_d$ such that the slices*

$$x_i = \text{constant for all } i \in \{c+1,\ldots,d\}$$

are integral manifold of $\mathscr{D}$; and if (M,ψ) is connected integral manifold of $\mathscr{D}$ such that $\psi(N) \subset U$, then $\psi(N)$ lies in one of this slices.

Bibliography

[1] PyTrajectory.

[2] a.M. Bloch and P E Croach. Reduction of Euler Lagrange problems for constrained variational problems and relation with optimal control problems. *Proceedings of 1994 33rd IEEE Conference on Decision and Control*, 3(December), 1994.

[3] Hirohiko Arai, Kazuo Tanie, and Naoji Shiroma. Nonholonomic control of a three-DOF planar underactuated manipulator. *IEEE Transactions on Robotics and Automation*, 14(5):681–695, 1998.

[4] V Arnold. *Mathematical methods of classical mechanics*, volume 49. 1978.

[5] a.W. Divelbiss and J. Wen. A global approach to nonholonomic motion planning, 1992.

[6] John T. Betts. Survey of Numerical Methods for Trajectory Optimization. *Journal of Guidance, Control, and Dynamics*, 21(2):193–207, 1998.

[7] L. T. Biegler and V. M. Zavala. Large-scale nonlinear programming using IPOPT: An integrating framework for enterprise-wide dynamic optimization. *Computers and Chemical Engineering*, 33(3):575–582, 2009.

[8] A.M. Bloch. *Nonholonomic Mechanics and control*. Springer, 2015.

[9] A.M. Bloch and N.H. McClamroch. Control of mechanical systems with classical nonholonomic constraints. *Proceedings of the 28th IEEE Conference on Decision and Control*, (December):201–205, 1989.

[10] J E Bobrow, S Dubowsky, and J S Gibson. Time-Optimal Control of Robotic Manipulators Along Specified Paths. *The International Journal of Robotics Research*, 4(3):3–17, 1985.

[11] Fabrizio Boriero, Nicola Sansonetto, Antonio Marigonda, Riccardo Muradore, and Paolo Fiorini. Optimal Solution of Kinodynamic Motion Planning for the Cart-Pole System. *IFAC-PapersOnLine*, 50(1):6308–6313, 2017.

[12] S. Bouabdallah, a. Noth, and R. Siegwart. PID vs LQ control techniques applied to an indoor micro quadrotor. *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, 3:2451–2456, 2004.

[13] Y. Bouktir, M. Haddad, and T. Chettibi. Trajectory planning for a quadrotor helicopter. *2008 16th Mediterranean Conference on Control and Automation*, pages 1258–1263, 2008.

[14] Michael Brady, J Hollerbach, T Johnson, T Lozano-Perez, and M Mason. *Robot Motion: Planning and Control*. 1982.

[15] Lukas Brantner. On the complexity of sails. In *Pacific Journal of Mathematics*, volume 258, pages 1–30, 2012.

[16] Arthur E. Bryson and Ho Yu-Chi. *Applied Optimal Control: Optimization, Estimation, and Control*. CRC Press, 1979.

[17] Francesco Bullo and Andrew D. Lewis. Geometric Control of Mechanical Systems, 2005.

[18] P. Castillo, R. Lozano, and A. Dzul. Stabilization of a mini-rotorcraft having four rotors. *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, 3(October 2002):2693–2698, 2004.

[19] Leonardo Colombo, David Martín De Diego, and Marcela Zuccalli. Optimal control of underactuated mechanical systems: A geometric approach. *Journal of Mathematical Physics*, 51(8):1, 2010.

[20] Jorge Cortes. *Geometric, Control and Numerical Aspects of Nonholonomic Systems*. 2002.

[21] Ian D. Cowling, Oleg a. Yakimenko, James F. Whidborne, and Alastair K. Cooke. A Prototype of an Autonomous Controller for a Quadrotor UAV. *European Control Conference*, pages 1–8, 2007.

[22] Shicong Dai, Taeyoung Lee, and Dennis S. Bernstein. Adaptive control of a quadrotor UAV transporting a cable-suspended load with unknown mass. In *Proceedings of the IEEE Conference on Decision and Control*, volume 2015-Febru, pages 6149–6154, 2014.

[23] Bruce Donald, Patrick Xavier, John Canny, and John Reif. Kinodynamic motion planning. *Journal of the ACM*, 40(5):1048–1066, 1993.

[24] Eurobotics. ERL Emergency Robots, 2017.

[25] Hartlan Richard F., Suresh P.Sethi, and Vickson G. Raymond. A Survey of the Maximum Principles for Optimal Control Problems with State Constraints. *SIAM Review*, 37(2):181–218, 1995.

[26] Fariba Fahroo and I. Michael Ross. Costate Estimation by a Legendre Pseudospectral Method. *Journal of Guidance, Control, and Dynamics*, 24(2):270–277, 2001.

[27] Davide Falanga, Elias Mueggler, Matthias Faessler, and Davide Scaramuzza. Aggressive quadrotor flight through narrow gaps with onboard sensing and computing using active vision. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 5774–5781, 2017.

[28] John Michael Finn. *Classical mechanics*, volume 70. Addison-Wesley Publishing Company, Inc., 2002.

[29] Paolo Fiorini and Zvi Shiller. Motion planning in dynamic environments using velocity obstacles. *International Journal of Robotics Research*, 17(7):760–772, 1998.

[30] M. Fliess, J. Levine, P. Martin, and P. Rouchon. On differentially flat nonlinear systems. *IFAC Symposia Series*, (7):159–163, 1993.

[31] Simone Formentin and Marco Lovera. Flatness-based control of a quadrotor helicopter via feedforward linearization. *Proceedings of the IEEE Conference on Decision and Control*, pages 6171–6176, 2011.

[32] Henri Gavin. The Levenberg-Marquardt method for nonlinear least squares curve-fitting problems. pages 1–15, 2015.

[33] Philip E. Gill, Walter Murray, and Michael A. Saunders. SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization. *SIAM Journal on Optimization*, 12(4):979–1006, 2002.

[34] Jeremy H. Gillula, Haomiao Huang, Michael P. Vitus, and Claire J. Tomlin. Design of guaranteed safe maneuvers using reachable sets: Autonomous quadrotor aerobatics in theory and practice. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1649–1654, 2010.

[35] Farhad A. Goodarzi, Daewon Lee, and Taeyoung Lee. Geometric control of a quadrotor UAV transporting a payload connected via flexible cable. *International Journal of Control, Automation and Systems*, 13(6):1486–1498, 2015.

[36] Mark J. Gotay and James M. Nester. Presymplectic Lagrangian systems I: the constraint algorithm and equivalence theorem. *Ann. Inst. H. Poincare, A*, pages 129–142, 1979.

[37] M E Guerrero, D a Mercado, R Lozano, and C D Garc. Passivity Based Control for a Quadrotor UAV Transporting a Cable-Suspended Payload with Minimum Swing. *Conference on Decision and Control (CDC)*, (Cdc):1–6, 2015.

[38] Leonid Gurvits. Averaging approach to Nonholonomic Motion Planning. *IEEE International Conference on Robotics and Automation*, pages 2541–2546, 1992.

[39] Tarek Hamel, Robert Mahony, Rogelio Lozano, and James Ostrowski. Dynamic modelling and configuration stabilization for an X4-flyer. *IFAC Proceedings Volumes (IFAC-PapersOnline)*, 15(1):217–222, 2002.

[40] C. R. Hargraves and S. W. Paris. Direct trajectory optimization using nonlinear programming and collocation. *Journal of Guidance, Control, and Dynamics*, 10(4):338–342, 1987.

[41] John Hauser, Shankar Sastry, and George Meyer. Nonlinear control design for slightly non-minimum phase systems: Application to V/STOL aircraft. *Automatica*, 28(4):665–679, 1992.

[42] Markus Hehn and Raffaello D'Andrea. *Quadrocopter trajectory generation and control*, volume 18. IFAC, 2011.

[43] Dominik Honegger, Lorenz Meier, Petri Tanskanen, and Marc Pollefeys. An open source and open hardware embedded metric optical flow CMOS camera for indoor and outdoor applications. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1736–1741, 2013.

[44] Http://www.sherpa.ac.uk/projects/sherpa.html. SHERPA Project, 2006.

[45] P. Jacobs and J. Canny. Planning smooth paths for mobile robots. *Proceedings, 1989 International Conference on Robotics and Automation*, pages 2–7, 1989.

[46] P. Jacobs, J.-P. Laumond, and M. Taix. Efficient motion planners for nonholonomic mobile robots. *Proceedings IROS '91:IEEE/RSJ International Workshop on Intelligent Robots and Systems '91*, (91):1229–1235, 1991.

[47] Lydia E. Kavraki, Petr Švestka, Jean Claude Latombe, and Mark H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, 1996.

[48] M. Khatib, H. Jaouni, R. Chatila, and J.P. Laumond. Dynamic path modification for car-like nonholonomic mobile robots. *Proceedings of International Conference on Robotics and Automation*, 4(April):2920–2925, 1997.

[49] Vijay Kumar and Nathan Michael. Opportunities and challenges with autonomous micro aerial vehicles. *The International Journal of Robotics Research*, 31(11):1279–1291, 2012.

[50] Florent Lamiraux and Jean Paul Laumond. Flatness and small-time controllability of multibody mobile robots: application to motion planning. *IEEE Transactions on Automatic Control*, 45(10):1878–1881, 2000.

[51] Jean-Claude Latombe. *Robot motion planning*, volume 54. 1991.

[52] J.-P. Laumond, M. Taix, and P. Jacobs. A motion planner for car-like robots based on a mixed global/local approach. *EEE International Workshop on Intelligent Robots and Systems, Towards a New Frontier of Applications*, pages 765–773, 1990.

[53] Jean-Paul Laumond. Finding Collision-Free Smooth Trajectories for a Non-Holonomic Mobile Robot. *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1120–1123, 1987.

[54] Jean Paul Laumond, Paul E. Jacobs, Michel Taïx, and Richard M. Murray. A Motion Planner for Nonholonomic Mobile Robots. *IEEE Transactions on Robotics and Automation*, 10(5):577–593, 1994.

[55] S M LaValle. *Planning Algorithms*. Cambridge University Press, 2006.

[56] Steven M. LaValle. Rapidly-exploring random trees: A new tool for path planning. *Technical Report*, (TR 98-11), 1998.

[57] Sheng Li, Guoliang Ma, Qingwei Chen, Xiaobei Wu, and Weili Hu. Nonholonomic motion planning: Steering using bang-bang control. *Proceedings of the American Control Conference*, 5(1):4653–4656, 2004.

[58] X Li and J Canny. Motion of two rigid bodies with rolling contstraint. *IEEE Trans on Robotics and Automation*, 6(1):62–72, 1990.

[59] Giuseppe Loianno, Chris Brunner, Gary McGrath, and Vijay Kumar. Estimation, Control, and Planning for Aggressive Flight With a Small Quadrotor With a Single Camera and IMU. *IEEE Robotics and Automation Letters*, 2(2):404–411, 2017.

[60] Tomás Lozano-Pérez. Spatial Planning: A Configuration Space Approach. *IEEE Transactions on Computers*, C-32(2):108–120, 1983.

[61] Kevin M. Lynch. Underactuated Robots. (1), 2013.

[62] J E Marsden and T S Ratiu. Introduction to Mechanics and Symmetry. *Physics Today*, 48(July):70, 1998.

[63] Philippe Martin, Santosh Devasia, and Brad Paden. A different look at output tracking: Control of a VTOL aircraft. In *Automatica*, volume 32, pages 101–107. IEEE, 1996.

[64] Jared M. Maruskin and Anthony M. Bloch. The Boltzmann-Hamel equations for optimal control. *Proceedings of the IEEE Conference on Decision and Control*, pages 554–559, 2007.

[65] Lorenz Meier, Dominik Honegger, and Marc Pollefeys. PX4: A node-based multithreaded open source robotics framework for deeply embedded platforms. *Proceedings - IEEE International Conference on Robotics and Automation*, 2015-June(June):6235–6240, 2015.

[66] Daniel Mellinger and Vijay Kumar. Minimum snap trajectory generation and control for quadrotors. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 2520–2525, may 2011.

[67] Daniel Mellinger, Nathan Michael, and Vijay Kumar. Trajectory generation and control for precise aggressive maneuvers with quadrotors. *Springer Tracts in Advanced Robotics*, 79(5):361–373, 2014.

[68] James R. Munkres. *Topology*. Prentice Hall, 2nd editio edition, 2002.

[69] R.M. Murray and S.S. Sastry. Steering nonholonomic systems using sinusoids. *29th IEEE Conference on Decision and Control*, (December):2097–2101 vol.4, 1990.

[70] Henk Nijmeijer and Arjan van der Schaft. *Nonlinear Dynamical Control Systems*. Springer New York, New York, NY, 1990.

[71] J Ostrowski, A Lewis, R Murray, and J Burdick. Nonholonomic mechanics and locomotion: the snakeboard example. *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, 3(May 1994):2391–2400, 1994.

[72] James P. Ostrowski. Computing reduced equations for robotic systems with constraints and symmetries. *IEEE Transactions on Robotics and Automation*, 15(1):111–123, 1999.

[73] Jim Ostrowski and Joel Burdick. The geometric mechanics of undulatory robotic locomotion. *International Journal of Robotics Research*, 17(7):683–701, 1998.

[74] Ivana Palunko, Rafael Fierro, and Patricio Cruz. Trajectory generation for swing-free maneuvers of a quadrotor with suspended payload: A dynamic programming approach. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 2691–2697, 2012.

[75] Diego Pardo, Lukas Möller, Michael Neunert, Alexander W. Winkler, and Jonas Buchli. Evaluating direct transcription and nonlinear optimization methods for robot motion planning. *ICRA. IEEE International Conference on Robotics and Automation*, 2015.

[76] M. Reyhanoglu and E. Al-Regib. Nonholonomic motion planning for wheeled mobile systems using geometric phases. *Proceedings of 1994 9th IEEE International Symposium on Intelligent Control*, pages 4–9, 1994.

[77] M. Reyhanoglu, A. van der Schaft, N.H. McClamroch, and I. Kolmanovsky. Nonlinear control of a class of underactuated systems. In *Proceedings of 35th IEEE Conference on Decision and Control*, volume 2, pages 1682–1687, 1996.

[78] Eric Rohmer, Surya P.N. Singh, and Marc Freese. V-REP: A versatile and scalable robot simulation framework. *IEEE International Conference on Intelligent Robots and Systems*, pages 1321–1326, 2013.

[79] P Rouchon, M Fliess, J Lévine, and P Martin. Flatness and motion planning: the car with n trailers. *Proc. ECC'93, Groningen*, 33(1):1–6, 1993.

[80] SAE. DRAFT Taxonomy and Definitions for Terms Related to On-Road Automated Motor Vehicles. Technical report, 2012.

[81] S. Sastry and Z. Li. Robot motion planning with nonholonomic constraints. *Proceedings of the 28th IEEE Conference on Decision and Control,*, (December), 1989.

[82] Shankar Sastry. *Nonlinear Systems*, volume 10 of *Interdisciplinary Applied Mathematics*. Springer New York, New York, NY, 1999.

[83] Jarvis Schultz and Todd Murphey. Trajectory generation for underactuated control of a suspended mass. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 123–129, 2012.

[84] D Seto and J Baillieul. Control problems in super-articulated mechanical systems. *Automatic Control, IEEE Transactions on*, 39(12):2442–2453, 1994.

[85] Ray Skinner. Generalized Hamiltonian dynamics. I. Formulation on TQŁTQ. *Journal of Mathematical Physics*, 24(11):2589, 1983.

[86] Ryan N. Smith, Monique Chyba, George R. Wilkens, and Christopher J. Catone. A geometrical approach to the motion planning problem for a submerged rigid bodySmith, R. N., Chyba, M., Wilkens, G. R., & Catone, C. J. (2009). A geometrical approach to the motion planning problem for a submerged rigid body. International Journal of Con. *International Journal of Control*, 82(January 2015):1641–1656, 2009.

[87] O J Sørdalen. Conversion of the Kinematics of a Car with n Trailers into a Chained Form. *International Conference on Robotics and Automation*, pages 382–387, 1993.

[88] Mark W. Spong. Underactuated mechanical systems. *Control Problems in Robotics and Automation, Springer*, pages 135–150, 1998.

[89] Koushil Sreenath, Taeyoung Lee, and Vijay Kumar. Geometric control and differential flatness of a quadrotor UAV with a cable-suspended load. *Proceedings of the IEEE Conference on Decision and Control*, pages 2269–2274, 2013.

[90] H. J. Sussmann. A General Theorem on Local Controllability. *SIAM Journal on Control and Optimization*, 25(1):158–194, jan 1987.

[91] Edited M Thoma and A Wyner. *Mathematical Control Theory II*, volume 462. 2015.

[92] Teodor Tomić, Moritz Maier, and Sami Haddadin. Learning quadrotor maneuvers from optimal control and generalizing in real-time. *Proceedings - IEEE International Conference on Robotics and Automation*, (Section III):1747–1754, 2014.

[93] F. Topputo and C. Zhang. Survey of direct transcription for low-thrust space trajectory optimization with applications. *Abstract and Applied Analysis*, 2014:1–15, jun 2014.

[94] M van Nieuwstadt, M Rathiam, and R M Murray. Differential flatness and absolute equivalence of nonlinear control systems. $\{SIAM\}$ *J. Control Optim.*, 36(4):1225–1239, 1998.

[95] Frank W. Warner. *Foundations of Differentiable Manifolds and Lie Groups*. 1971.

[96] Paul Williams. Hermite-Legendre-Gauss-Lobatto Direct Transcription in Trajectory Optimization. *Journal of Guidance, Control, and Dynamics*, 32(4):1392–1395, 2009.