Alma Mater Studiorum – Università di Bologna

DOTTORATO DI RICERCA IN

Automatica e Ricerca Operativa

Ciclo XXVIII

Settore Concorsuale di afferenza: 09/G1 - AUTOMATICA

Settore Scientifico disciplinare: ING-INF/04 - AUTOMATICA

MOBILE ROBOTS CONTROL AND PATH PLANNING STRATEGIES

Presentata da: **Michele Furci**

Coordinatore Dottorato **Relatore**

Prof. Daniele Vigo **Prof. Lorenzo Marconi**

Esame finale anno 2016

Abstract

Mobile robots gained lots of attention in the last decades, and it is estimated that their applications will increase exponentially in the next years. Because of its flexibility and increased capabilities of automation, mobile robots are used in many different applications: from domotic, to search and rescue missions, to exploration in hostile environments, agriculture, environment protection and many more. The main capability of mobile robots to accomplish a typical mission is the mobility in the work environment. It may seems a simple task for humans, but for mobile robots there are many challenges. To move in a certain environment the robots should achieve: guidance, navigation and control. Guidance refers to the capability of generating a suitable path(s) for the mission. It refers to what in robotic is called path-planning or motion planner. A suitable path should be generated based on the environment information and its state, to safely move the robot through the obstacles. Navigation refers to the ability of determining the robot/vehicle state with the use of sensors and information. Finally control refers to the ability of generating appropriate input to the system, in terms of actuators, to steer it to the desired path and to guarantee stability to the system. This thesis focuses on guidance and control of mobile robots, with application to certain classes of robots. In particular the application concerns Vertical Take Off and Landing Unmanned Aerial Vehicles (VTOL UAV) and Differential Wheel robots or Car-like robots. The first class is a good benchmark because of its underactuation and it's dynamic model, while the second is a good benchmark because of its underactuation and non-holonomic constraints. The contribution of this thesis is on modeling and control of the two classes of robots, and on novel strategies of combined control and motion planning for kinodynamic systems. On modeling, a new approach to model a class of multi-propeller VTOL is proposed, with the aim of generating a general model for a system seen as a composition of elementary

modules such as actuators and payloads. On control, two control law for VTOL vehicles and differential-wheel robot are proposed. The goal of the first is to generate a simple yet powerful control to globally asymptotically stabilize a VTOL for acrobatic maneuvers. The second is a simple saturated input control law for trajectory tracking of a differential wheel robot model in 2D . About planning, first a novel approach to generate non-feasible trajectories for robots that still guarantees a correct path for kinodynamic planning is proposed. The goal of this approach is to reduce the runtime of planners to be used in real-time and realistic scenario. Moreover an innovative general framework for mobile robots motion planning with the use of Discrete Event Systems theory is introduced. The two proposed approaches together allow to build a global, real-time, quasi-optimal, kinodynamic planner suitable for replanning in unknown environments. The background and motivation of the thesis is the European project SHERPA, where a heterogeneous robotic platform for search and rescue mission in alpine scenario is developed. The proposed approaches are supported throughout the thesis with simulations and applications on real platforms.

Abstract (Italiano)

I robot e la robotica mobile hanno guadagnato molte attenzioni durante gli ultimi anni, ed è stimato che le applicazioni nella robotica aumenteranno esponenzialmente nei prossimi anni. Grazie alla loro flessibilità e ai miglioramenti nel campo dell'automazione, i robot mobili sono usati in molte applicazioni: domotica, missioni di ricerca e soccorso, esplorazioni di ambienti ostili, agricoltura, protezione ambientale e molti altri. Le capacità di base che un robot mobile deve compiere per una missione ÃÍ la possibilitÃă di muoversi agilmente nell'ambiente circostante. PuÃš sembrare un task semplice per gli uomini, ma per un robot ci sono molte difficoltÃă tecniche. La capacitÃă di movimento di un robot mobile si suddivide nelle cosidette: guidance, navigation and control. Guidance si riferisce alla capacità di generare una traiettoria consona per la missione. La traiettoria generata, di solito deve utilizzare le informazioni dell'ambiente circostante e dello stato del robot e deve permettere di muovere il robot verso un obiettivo in maniera sicura evitando gli ostacoli ed eventuali pericoli. Navigation si riferisce invece all'abilità di determinare lo stato del robot a partire dalle informazioni dei sensori di bordo. Per esempio lo stato del robot potrebbe essere la sua posizione e velocità nello spazio, ma potrebbe anche rappresentare stati di alto livello come per esempio guasti ad attuatori o sensori. Infine Control riguarda l'abilità di generare appositi input agli attuatori del robot, per farlo muovere sulla traiettoria generata e per garantire stabilità del sistema. Per un'automobile autonoma per esempio gli input potrebbero essere le velocitÃă o le coppie alle ruote. Questa tesi si focalizzerà su guidance e control per robot mobili, con applicazioni ad alcune classi di robot. In particolare, le applicazioni riguardano Vertical Take Off and Landing Unmanned Aerial Vehicles (VTOL UAV) e differential-wheel robot o robot car-like. La prima classe di robot (VTOL) è un perfetto benchmark a causa della sottoattuazione che lo caratterizza e per il particolare modello dinamico, mentre la seconda è perfetta ancora per la sottoattuazione e perchè soggetta a vincoli non olonomi. Il

contributo di questa tesi riguardano la modellazione e il controllo delle due classi di interesse e lo sviluppo di nuove strategie di controllo e pianificazione del moto combinate per sistemi cinematici/dinamici. Per quanto riguarda la modellazione, viene proposto un nuovo approccio per modellare VTOL con eliche multiple, con il goal di generare un modello generico visto come composizione di singoli elementi attuatore/payload. Per quanto riguarda il controllo, sono proposte due nuove leggi di controllo per VTOL e differential-wheel robot. La prima con lo scopo di generare una legge di controllo semplice ma potente in grado di stabilizzare asintoticamente e globalmente il VTOL per compiere manovre acrobatiche. La seconda invece è una semplice legge di controllo saturata in velocità per il differential-wheel robot che permette l'inseguimento di traiettorie in un piano 2D. Per quanto riguarda la pianificazione del moto, innanzitutto viene proposto un approccio innovativo per generare traiettorie semplici ma non fattibili, che garantiscono comunque una traiettoria corretta per robot di cui si tiene conto del modello cinematico/dinamico. L'obiettivo di questo approccio è di ridurre il tempo di computazione richiesto dalla pianificazione della traiettoria, per essere utilizzato in applicazioni realistiche real-time. Inoltre viene presentato un innovativo framework di pianificazione del moto per robot mobili basato sulla teoria dei DES (Discrete Event System). I due approcci proposti consentono assieme di sviluppare un pianificatore di moto globale, real-time, con traiettorie quasi ottime, per modelli cinematici/dinamici ed adatto alla ripianificazione del moto per ambienti non noti e dinamici. Gli approcci proposti saranno supportati durante tutta la tesi con simulazioni ed esperimenti con applicazioni a piattaforme robotiche vere.

Aknowledgements

First of all I'd like to thanks my supervisor and co-supervisors Lorenzo Marconi, Roberto Naldi and Andrea Paoli from which I really learned a lot. Then a big thanks to all my colleagues and lab mate, in particular Zack, Dani and Fra who shared discussions, papers, lunches, experiments, ping-pong matches, aperitivi, jokes and advice. Of course the family deserves a huge hug and thanks. In particular to my parents who supported me, my girlfriend that went through my geekiness, my brother that read all my thesis and my red cat that woke me up all the morning biting.

Contents

Abstract **iv**

Notation **1**

1 Introduction **7**
1.1 Motivation 7
1.2 Organization 8

2 Robots Model **11**
2.1 VTOL UAV Model 12
2.1.1 Contribution 12
2.1.2 The Modular Multi-Propeller Aerial Vehicle 13
2.1.3 Control Strategy 17
2.1.4 Control Allocation 18
2.1.5 Application: Modeling of a Quad-Rotor Helicopter 21
2.2 Differential Wheel Robot Model 23

3 Robots Control **25**
3.1 VTOL UAV Control 26
3.1.1 Problem Formulation 28
3.1.2 Inner-Outer Loop Control Strategies 31
3.2 Differential Wheel Robot Control 52
3.2.1 Bound with quadratic function 54
3.2.2 Closed Form Parameters 55
3.2.3 Simulations 55

4 Path Planning Strategies **59**
4.1 Introduction . . . 61
4.1.1 Feasibility . . . 61
4.1.2 Optimality . . . 63
4.1.3 Runtime . . . 63
4.1.4 Completeness . . . 64
4.1.5 Globality . . . 64
4.1.6 Robustness . . . 65
4.1.7 Replanning . . . 65
4.1.8 Incremental . . . 65
4.2 Piece-Wise Controllable Trajectories and Practical Tracking . . . 66
4.2.1 Piece-wise Continuous Reference and Integrator Chain System . . . 66
4.2.2 Piece-wise Controllable Reference and Non-Linear System . . . 71
4.2.3 Piece-wise Controllable Reference and Non-Linear System with Disturbances . . . 75
4.2.4 Application to VTOL and Car-like Robot . . . 80

5 DESP: Discrete Event System Planner **93**
5.1 DESP: Discrete Event System Planner . . . 94
5.1.1 Map Automaton . . . 94
5.1.2 Specification Automaton . . . 95
5.1.3 Agent Automaton . . . 97
5.1.4 Swath and Collision Checking . . . 101
5.1.5 Supervisor and Reachability Graph . . . 103
5.1.6 Exploiting DES capabilities . . . 106
5.1.7 Advantages of DESP . . . 107
5.2 Applications and Simulations . . . 108
5.2.1 Double integrator and Uncontrollable Events . . . 108
5.2.2 Differential Wheel Robot and Replanning . . . 117

6 Experiments and Applications **123**
6.1 Application to the control of a quadrotor aerial vehicle . . . 124
6.1.1 Simulations . . . 125
6.1.2 Experiments . . . 128
6.2 Architecture for Control and Coordination of a Swarm of Micro-Quadrotors 134
6.2.1 Crazyflie Platform . . . 135
6.2.2 Motivation and Control Architecture . . . 136
6.2.3 Global Trajectory Tracking Control Law . . . 140

6.2.4 Experiments and Results 140
6.3 Other applications 146

7 Conclusion and Future Works **147**
7.1 Future Works 148

A Appendix **149**
A.1 Robots Control 149
A.1.1 Hybrid Systems: Definitions and Stability Notions 149
A.1.2 Proof of Lemma 3.1 151
A.1.3 Proof of Lemma 3.2 152
A.1.4 Computation of the rotation matrix 152
A.2 Path Planning Strategies 153
A.2.1 Discrete Event Systems and Automata 153

Bibliography **166**

Notation

- $\mathbb{R}$, $\mathbb{R}_{>0}$, $\mathbb{R}_{\geq 0}$ denote the set of real, positive real and non-negative real numbers, respectively.
- $I_n \in \mathbb{R}^{n \times n}$ denote the n-dimensional identity matrix.
- For a matrix $M^{n \times m}$, the ith row and the jth column, with $1 \leq i \leq n$ and $1 \leq j \leq m$, are denoted as $M|_{(i,:)}$ and $M|_{(:,j)}$, respectively.
- For a vector $v \in \mathbb{R}^n$, the ith element, with $1 \leq i \leq n$, is denoted as $v(i)$ or, equivalently, as $v|_{(i)}$.
- For a matrix $B \in \mathbb{R}^{m \times n}$, $n \geq m$, we denote with $B^+ \in \mathbb{R}^{n \times m}$ the generalized pseudo-inverse of B, i.e., when $\text{rank}(M) = m$, $BB^+ = I_m$.
- $\omega\times := \text{Skew}(\omega)$ where $\text{Skew}(\text{col}(x_1, x_2, x_3))$ denote the skew-symmetric matrix with the first, second and third row respectively given by $[0, -x_3, x_2]$, $[x_3, 0, -x_1]$ and $[-x_2, x_1, 0]$.
- $\mathcal{F}_i$ and $\mathcal{F}_b$ denote, respectively, an inertial reference frame and a reference frame attached to the center of gravity of the vehicle.
- For $x \in \mathbb{R}^n$, $|x|$ denotes the Euclidean norm and, given a closed set $\mathcal{A} \subset \mathbb{R}^n$, $|x|_{\mathcal{A}} = \inf_{y \in \mathcal{A}} |x - y|$.
- For a function $f : [0, \infty) \to \mathbb{R}^k$, $k > 0$, we define with $|f|_\infty := \sup_{t \in [0, \infty)} |f(t)|$ and $|f|_a := \limsup_{t \to \infty} |f(t)|$.
- Given a set $\mathcal{M}$, $\overline{\mathcal{M}}$ denotes its closure.

- Given sets $\mathcal{S}_1$ and $\mathcal{S}_2$, the notation $f : \mathcal{S}_1 \rightrightarrows \mathcal{S}_2$ denotes a set-valued map mapping subsets of $\mathcal{S}_1$ onto subsets of $\mathcal{S}_2$.

- With $\mathcal{B}_r^n$ we denote the closed ball of radius r centered at the origin of $\mathbb{R}^n$, namely $\mathcal{B}_r^n = \{x \in \mathbb{R}^n : |x| \leq r\}$.

- Given a function $f : \mathbb{R} \to \mathbb{R}$ and $i \in \mathbb{N}$, we use the notation $f(t)^{(i)} := \frac{d^i}{dt^i} f(t)$ to denote derivative of f with respect to t.

- We denote the unit vectors as $e_1 := [1, 0, 0]^\top$, $e_2 := [0, 1, 0]^\top$, and $e_3 := [0, 0, 1]^\top$.

- For any $x \in \mathbb{R}^3$, we let $S(x)$ - with the first, second and third rows given by $[0, -x_3, x_2]$, $[x_3, 0, -x_1]$ and $[-x_2, x_1, 0]$ - be a skew-symmetric matrix and we denote with $^\wedge$ the inverse operator such that $S(x)^\wedge = x$. Let $SO(3)$ denote the *special orthogonal group* of order three, i.e., $SO(3) = \{R \in \mathbb{R}^{3\times 3} : R^\top R = RR^\top = I_3, \det R = 1\}$.

- Given a rotation matrix $R \in SO(3)$, $\Theta(R) := \frac{1}{2}\text{trace}(I_3 - R)$.

- We denote the n-dimensional unit sphere as $\mathbf{S}_n := \{x \in \mathbb{R}^{n+1} : |x| = 1\}$.

- A unit quaternion $q \in \mathbf{S}_3$ is defined as a pair $q = [\eta, \epsilon^\top]^\top$ in which $\eta \in \mathbb{R}$ and $\epsilon \in \mathbb{R}^3$ are denoted, respectively, as the scalar and vector part. Given unit quaternions $q_1 = [\eta_1, \epsilon_1^\top]^\top$ and $q_2 = [\eta_2, \epsilon_2^\top]^\top$, the standard quaternion product is defined as

$$q_1 \otimes q_2 = \begin{bmatrix} \eta_1 & -\epsilon_1^\top \\ \epsilon_1 & \eta_1 I_3 + S(\epsilon_1) \end{bmatrix} \begin{bmatrix} \eta_2 \\ \epsilon_2 \end{bmatrix}.$$

- With $\mathbf{1} = [1, 0, 0, 0]^\top \in \mathbf{S}_3$ we denote the identity quaternion element and, for a quaternion $q = [\eta, \epsilon^\top]^\top \in \mathbf{S}_3$, with $q^{-1} = [\eta, -\epsilon^\top]^\top$ the inverse, so that $q \otimes q^{-1} = q^{-1} \otimes q = \mathbf{1}$.

- A rotation matrix parameterizing attitude can be expressed in terms of a unit quaternion $q \in \mathbf{S}_3$ through the mapping $\mathcal{R} : \mathbf{S}_3 \to SO(3)$ (known as Rodrigues formula [104]) defined as

$$\mathcal{R}(q) = I_3 + 2\eta S(\epsilon) + 2S(\epsilon)^2.$$

- The mapping $\mathcal{R}$ is such that $\mathcal{R}(q) = \mathcal{R}(-q)$, namely the two quaternions q and $-q$ correspond to the same rotation matrix.

- A *saturation function* as a mapping $\sigma : \mathbb{R}^n \to \mathbb{R}^n$ such that, for $n = 1$,

1. $|\sigma'(s)| := |d\sigma(s)/ds| \leq 2$ for all s,
2. $|\sigma''(s)| := |d^2\sigma(s)/ds^2| \leq \bar{d}$ for some $\bar{d} > 0$, for all s,
3. $s\sigma(s) > 0$ for all $s \neq 0$, $\sigma(0) = 0$,
4. $\sigma(s) = \text{sgn}(s)$ for $|s| \geq 1$,
5. $|s| < |\sigma(s)| < 1$ for $|s| < 1$.

For $n > 1$, the properties listed above are intended to hold componentwise.

- For a matrix, A^T defines the transpose of the matrix A.

Acronyms

A list of abbreviations and acronyms used among the thesis are reported in the following table.

Table 1: Acronyms

UAV	Unmanned Aerial Vehicle
VTOL	Vertical Take Off and Landing
DWR	Differential Wheel Robot
AHRS	Attitude and Heading Reference System
IMU	Inertial Measurements Unit
MP	Multi Propeller
GAS	Globally Asymptotically Stable or Global Asymptotic Stability
LAS	Locally Asymptotically Stable or Local Asymptotic Stability
BLDC	Brushless Direct Current (Motor)
ISS	Input to State Stability
DES	Discrete Event System
DESP	Discrete Event System Planner
RRT	Rapidly Exploring Random Tree
PRM	Probabilistic Roadmap

1

Introduction

In this introduction we present the motivation behind the topics choice of this thesis and the general organization of the latter.

1.1 Motivation

The background and motivation of this thesis is the European project SHERPA: Smart collaboration between Humans and ground-aErial Robots for imProving rescuing activities in Alpine environments [2] [1]. The goal of the project is the development of a heterogeneous robotic platform for search and rescue missions in hostile environment such as Alpine scenario, with 10 partners from all over Europe. The platform includes multi-rotors UAV for low altitude and precise operations, a fixed wing UAV and a big helicopter UAV for high altitude operations and a ground rover that is a differential track robot with a manipulator on top, used as base station and for battery replacement operations. This thesis focuses on the control and motion planning algorithms for the robotic platforms, in particular for the multi-propeller UAV and the differential track robot. The hostile and unknown environment requires highly specialized algorithms for both control and path planning, to be robust to external disturbances and to be reactive to the unknown scenario. On the control side, in particular for the multi-rotors, it requires a robust control able to handle uncertainties in the model and robustness to

disturbances because of real sensors usage and disturbances from the environment such as wind gusts. Moreover one of the tasks in the project includes the aerial and hand deployment operation, where the UAV has to be deployed by a human operator or from an helicopter in overturned initial condition. For this task, the need arises for a globally stabilizing controller able to execute acrobatic maneuvers as attitude recovery and to stabilize the system from any initial configuration, avoiding dangerous crash. On the path planning side, the requirements for the project are: real-time and low computation algorithm for real on-board application, a robust algorithm able to take into account disturbances and to plan safe paths accordingly and suitable to replanning because of the unknown environment. Moreover a general framework for multiple heterogeneous robots had to be develop and shared between different agents. Me and my group were responsible for the multi-propeller UAV and the ground rover, hence the focus of this thesis is on control and path planning with applications on those platform, but suitable for other mobile robots. The two models of interest will be used throughout the whole thesis for examples and applications.

1.2 Organization

In chapter 2 are presented the dynamic models of VTOL UAV and differential wheel robot. The VTOL UAV is suitable to model the multi-propeller UAV used in the project, and we propose a novel strategy to model this class of vehicles with a modular approach, where the UAV is seen as composition of payload and actuators modules. This will allow to model general multi-propeller vehicles with non-standard configurations. About the differential wheel robot, the model used is a standard unicycle model to define the kinematic, which is often sufficient to build accurate models and control strategies.
In chapter 3 we propose the control law for the two robots of interest. For the VTOL UAV we propose an hybrid control law with a cascade approach, where the continuous position controller is in cascade with the hybrid attitude controller. It allows to overcome a topological obstruction for continuous law on compact manifolds such as SO(3) for the attitude, and allows to robustly execute acrobatic maneuvers such as flips and attitude recovery, useful for the aerial deployment task of the project. About the differential wheel robot, a *vectored speed* control is proposed, with saturation on the velocity input to represent the constraints on the actuator. The proposed control allows to reach a waypoint in a 2D plane starting from any initial configuration or to track a desired position trajectory.
In chapter 4 we propose a combined planning and control strategy to generate kinodynamic feasible paths, without considering the dynamic model with its differential constraints in real-time, but with a priori analysis with Lyapunov tools. This allows a

fast planner, suitable for real-time application. Moreover the robust case is considered when dealing with disturbances in the system, such as parametric uncertainties in the model or exogenous disturbances from the environment. With Lyapunov analysis and ISS theory it is possible to take into account the disturbances in the motion planner to plan a safe feasible path.
In chapter 5 we introduce a novel framework for path planning of mobile robots, based on DES (Discrete Event System) theory and tools. Both the dynamic model of the agent and the environment are discretized and converted into symbolic description by mean of automata. The framework is suitable to build a kinodynamic, real-time, global planner, suitable to replanning for unknown environments.
Finally in chapter 6 are presented some experiments and applications to validate part of the proposed approaches. The experiment focus more on the control law, while planning strategies are supported by mean of simulation and examples throughout the whole thesis.

2

Robots Model

In this chapter we present the models for two families of robot we use throughout the thesis: VTOL UAV (Vertical Take-Off and Landing Unmanned Aerial Vehicle) and differential wheel robot. UAV are recently under high investigation by many authors because of the reliability of the hardware and the reduced cost to build a platform suitable for experiments. Moreover because of its underactuated nature, it is an interesting topic of interest for control Differential wheel and car-like robots (or unicycle, bicycle) are topic of research from decades. The investigation topics lie in control and robotics fields. The main contribution of this thesis in modelling, is a novel strategy to model a wide class of VTOL UAV. In particular we can model all multi-propeller UAV with actuators that generate the trust in the same direction. This includes quad-rotors, exa-rotors, octa-rotors with generic configuration in space. For what concern the differential wheel robot, we limit to report models from the literature, because they will be used in other sections of the thesis.

2.1 VTOL UAV Model

For VTOL UAV we consider all the unmanned aerial vehicles that can takeoff vertically by generating a single total trust in one direction, while completely actuated in attitude dynamic. The VTOL UAV family, that includes among others helicopters and ducted fans, also includes the multi-propellers vehicles. Multi-propeller UAV are in general composed by a frame and a certain number of actuators with propellers capable of producing forces and torques. Multi-propeller aerial vehicles have revealed to be effective aerial platforms for accomplishing a large variety of different tasks, ranging from aerial survey, exploration of populated areas and search and rescue missions [32, 8, 29, 88]. One reason for this large success is the high level of maneuverability which allows to safely perform flight missions in densely populated environments [8] or to accomplish complicated robotics tasks [85, 36]. Moreover, these configurations are characterized by reduced mechanical complexity when compared to other vehicles, including in particular helicopters where the attitude is governed by complex mechanical mechanisms such as cyclic and collective pitches [44].

Several contributions document the interest for this particular configuration by presenting both modeling and control design and by showing the performances in practical applications [101] [47] [18] [77]. As far as the dynamical model is concerned, miniature multi-propeller vehicles can be considered as rigid bodies affected by a certain number of aerodynamic forces and torques [77]. In the presence of non negligible relative wind, the aerodynamic model of the propeller, which can be derived using momentum theory or blade-element theory [69], appears to play an important role to determine the dynamic behavior of the system [46] [3]. On the other side, when the flight is maintained stationary, most aerodynamic effects can be reasonably neglected [77].

Recent contribution are also considering the role of the geometric characteristics of the vehicle both in the modeling and control design. In [56] the analysis of generalized multi-rotor aerial configuration is proposed showing the link between mechanical design and dynamical performances. Modular aerial systems have appeared in [94], were the design and decentralized control for a system composed of a number single rotor modules (the Distributed Flight Array) is proposed, and in [89] [33], where each module is given by an autonomous ducted-fan aerial vehicle.

2.1.1 Contribution

By considering the vehicle as a modular system composed of i) actuator modules, i.e. fixed-pitch propellers capable of producing aerodynamic control forces and torques, and of ii) payload modules, i.e. equipment or mechanical devices characterized by non-

negligible mass, this work derives a control strategy capable of controlling a class multi-propeller aerial vehicles [90]. The geometry of the modular vehicle, namely the relative position between the different actuator and payload modules, and the aerodynamic properties of each single module are employed to derive a parametric dynamical model of the overall system. With this model at hand, the control law is obtained in two different steps. In the first step, control allocation algorithms [17] [57] are synthesized to resolve possible actuator redundancy. The goal is to obtain *virtual-vehicle* characterized by a *vectored-thrust* under-actuated dynamic behavior [51] where one force component (the main thrust) and three torque components are available for feedback. Since fixed-pitch propellers are employed, the control allocation algorithms proposed in this work are designed to take into account for the fact that each module can produce forces and torques only in one direction.

A mathematical model for the VTOL UAV system can be derived using the Newton-Euler equations of motion of a rigid body in the configuration space $SE(3) = \mathbb{R}^3 \times SO(3)$. By considering the inertial coordinate frame $F_i = \{O_i, \overrightarrow{i}_i, \overrightarrow{j}_i, \overrightarrow{k}_i\}$ and assuming that the body frame F_{b_m} has its axis aligned with the principal axis of inertia of the rigid body. The dynamical model of a generalized VTOL UAV with respect to the inertial frame is described by [91] [90], :

$$\begin{aligned} M\ddot{p} &= Ru_f + Mge_3 + u_d \\ J\dot{\omega} &= -\omega \times J\omega + u_\tau + u_{\tau,d} \end{aligned} \tag{2.1}$$

where M is the mass of the vehicle, J denotes the inertia of the vehicle, $p = \text{col}(x, y, z)$ is the position of the center of mass, ω the angular velocity expressed in the body frame F_{b_m}, R the rotation matrix relating the reference frames F_{b_m} and F_i, and e_3 the unit vector $e_3 := [0, 0, 1]^T$. Moreover u_d and $u_{\tau,d}$ represent respectively the force and a torque disturbances. This general model doesn't take into account how u_f, u_d, u_τ and $u_{\tau,d}$ are generated and considers the inertia parameters M and J as a single rigid-body object. In the following section is presented a more general dynamical model for a class of multi-propeller aerial vehicles that can be described by (2.1).

2.1.2 The Modular Multi-Propeller Aerial Vehicle

The main idea is to consider the vehicle as the interconnection of $N > 0$ single actuator modules and $P \geq 0$ payload modules. Each actuator module is given by a fixed-pitch propeller driven by a motor. This subsystem is capable of generating a number of control forces and torques in order to actuate the overall vehicle. The payload module, on the other hand, consists of a mass (e.g. a battery, a camera, etc.) which cannot generate any control force. The forces and torques generated by the modules as well as the main

aerodynamic effects characterizing such subsystems are described in the following two subsections.

The Fixed-Pitch Propeller Module: Control Wrench

A fixed-pitch propeller module is given by an electric motor equipped with a fixed-pitch propeller. In some configurations, the electric motor can be replaced by an endothermic engine. The forces and torques components generated by each $i-th$ single module are expressed in the reference body frames $F_{b_i} = \{O_{b_i}, \overrightarrow{i}_{b_i}, \overrightarrow{j}_{b_i}, \overrightarrow{k}_{b_i}\}$, $i \in \{1, 2, ...N\}$, attached to the center of mass of each module (which corresponds, approximately, to the motor position). Each body reference frame is fixed so as the z-axis is aligned with the propeller spin axis. Following [106], [69] and assuming stationary flight (i.e., zero relative wind speed), the force-torque vector produced by each fixed-pitch propeller can be then approximated as

$$f^{b_i} = \begin{bmatrix} 0 \\ 0 \\ -K_T w_{e,i}^2 \end{bmatrix}, \quad \tau^{b_i} = \begin{bmatrix} 0 \\ 0 \\ s_i K_Q w_{e,i}^2 \end{bmatrix} \tag{2.2}$$

where $w_{e,i}$ is the angular speed of the propeller, $s_i \in \{-1, 1\}$ denotes the spin direction, $i \in \{1, 2, ..., N\}$, K_T and K_Q collect all constant aerodynamic coefficients. It is assumed that the angular speed represents the available input for control, hence we define

$$u_i := w_{i,e}^2 \,, \tag{2.3}$$

where the spin direction is assumed to be fixed for each given actuator module.

The Fixed-Pitch Propeller and the Payload Module: Disturbance Vector

In the previous subsection, the control wrench vector produced by each propeller module has been derived by assuming zero relative wind speed. This assumption, however, is not satisfied when the presence of wind disturbances and/or the motion of the propeller with respect to an inertial reference frame are not negligible. In particular, following momentum theory [110], the presence of wind directed perpendicular to the propeller disk may reduce or increase the total thrust produced by the propeller. On the other side, when the wind speed is directed along the x_{b_i}-y_{b_i} axis, induced-drag phenomena and blade flapping (see [69] [77]) create force components not directed along the body z_{b_i} axis of the module. Finally, in the presence of nonzero relative wind, the payload module is a source of aerodynamic drag [106]. In summary, for each module a disturbance wrench vector modeling the effects of relative wind can be considered. In this

work, for sake of simplicity, these aerodynamic effects will be simply modeled as disturbances, namely for each module $i \in \{1, ..., N+P\}$ the force and torque disturbances $f_{d,i}$ and $\tau_{d,i}$ are defined.

Modular Configuration

Let us consider now a modular system composed of a number $N > 0$ of equal fixed-pitch propellers modules and of a number $P \geq 0$ of payload modules rigidly connected together. Let the coordinate frame $F_{b_m} = \{O_{b_m}, \overrightarrow{i}_{b_m}, \overrightarrow{j}_{b_m}, \overrightarrow{k}_{b_m}\}$ be attached to the center of mass of the modular system. Let the unit vectors $\overrightarrow{k}_{b_i}$, namely the propellers spin axis of each actuator module i, $i \in \{1, 2, ...N\}$, be fixed in order to point in the same direction, so that the propeller thrust of each module counteracts the gravity force at hover. Without loss of generality, let also the unit vectors $\overrightarrow{k}_{b_i}$, $i \in \{1, ...P\}$, attached to the payload modules be directed as the ones of the actuator modules. Finally, for all the modules in the group in which the body z-axis does not intersect the center of mass of the formation, let the x-axis pointing towards the z-axis of the reference frame F_{b_m}, while, for the remaining actuator modules, let the x-axis be aligned with the one of the frame F_{b_m}. For the above choice of reference frames, the vectors $\ell_i^{b_i}$, which denote the position of the center of mass of each module with respect to the center of mass of the overall formation expressed in the reference frame F_{b_i}, are given by

$$\ell_i^{b_i} = [r_i,\, 0,\, h_i]^T$$

in which, by construction, $r_i \in \mathbb{R}_{\geq 0}$ and $h_i \in \mathbb{R}$ denote respectively the horizontal and vertical displacement of each module in the group, with $i \in \{1, 2, ...N+P\}$. Finally, let us denote the relative orientation between the reference frames F_{b_i} and the reference frame F_{b_m} by ψ_i, $i \in \{1, 2, ..., N+P\}$. Indeed, ψ_i denotes the angle by which the reference frame F_{b_i} should be rotated around the z-axis to align the x-axis with the one of the frame F_{b_m}. Accordingly, for a vector $v^{b_i} \in \mathbb{R}^3$ defined in the frame F_{b_i}, $R_{\psi_i}^T v^{b_i}$ represents the same vector in the reference frame F_{b_m}, having defined

$$R_{\psi_i} := \begin{bmatrix} \cos\psi_i & -\sin\psi_i & 0 \\ \sin\psi_i & \cos\psi_i & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Since modules are physically separated, the distance between the center of mass of any two different modules should be greater than zero. Accordingly, if $N + P > 1$, the following constraint on system parameters holds

$$\|R_{\psi_i}^T \ell_i^{b_i} - R_{\psi_j}^T \ell_j^{b_j}\| > 0 \;\forall\; i \neq j,\; i, j \in \{1, 2, ..., N+P\}\,. \tag{2.4}$$

With the above construction and notation at hand, we are now able to give a definition of the class of modular system of interest.

Definition 2.1. *A modular multi-propeller aerial robot $\mathcal{MP}$ is given by the 6-tuple $(N, P, \mathcal{S}, \Psi, \mathcal{R}, \mathcal{H})$ where*

- *N is the number of equal actuator modules with input $u_i \in \mathbb{R}_{\geq 0}$ given by (2.3);*
- *P is the number of payload modules, each one characterized by a mass $M_{P,i}$, $i \in \{1, ..., P\}$;*
- *$\mathcal{S} := \{s_i \in \{-1, +1\} \,|\, i = 1, 2, ...N\}$ is the set of spin directions of the different propeller modules;*
- *$\Psi := \{\psi_i \in \mathbb{R} \,|\, i = 1, 2, ...N + P\}$ is the set of the orientations ψ_i of each frames F_{b_i} with respect to the frame F_{b_m} (namely the angle required to align frame F_{b_i} with F_{b_m});*
- *$\mathcal{R} := \{r_i \in \mathbb{R} \,|\, i = 1, 2, ...N + P\}$ is the set of horizontal distances r_i between each module and the center of gravity of the formation;*
- *$\mathcal{H} := \{h_i \in \mathbb{R} \,|\, i = 1, 2, ...N + P\}$ is the set of vertical distances h_i between each module and the center of gravity of the formation.*

Let $\mathbf{u} := [u_1, u_2, ..., u_N]^T$ be the vector of all the force-torque components u_i produced by all the N actuator modules in the formation. The resultant control force and torque vectors $f_c \in \mathbb{R}^3$ and $\tau_c \in \mathbb{R}^3$ applied by all the modules to the center of mass of the formation are given by

$$f_c = B_f \mathbf{u} \tag{2.5}$$

$$\tau_c = B_\tau(\Psi, \mathcal{S}, \mathcal{R})\mathbf{u} \tag{2.6}$$

where

$$\begin{aligned} B_f &:= [G_f, ..., G_f], \\ B_\tau(\Psi, \mathcal{S}, \mathcal{R}) &:= [R_{\psi_1}^T G_\tau(s_1, r_1), ..., R_{\psi_N}^T G_\tau(s_N, r_N)], \end{aligned}$$

having defined

$$G_f := \begin{bmatrix} 0 \\ 0 \\ -K_T \end{bmatrix}, \; G_\tau(s_i, r_i) := \begin{bmatrix} 0 \\ -r_i K_T \\ s_i K_Q \end{bmatrix}.$$

Vehicle Dynamics

A mathematical model for the multi-propeller vehicle $\mathcal{MP}$ system can be derived using the Newton-Euler equations of motion of a rigid body in the configuration space

$SE(3) = \mathbb{R}^3 \times SO(3)$. By considering the inertial coordinate frame $F_i = \{O_i, \overrightarrow{i}_i, \overrightarrow{j}_i, \overrightarrow{k}_i\}$ and assuming that the body frame F_{b_m} has its axis aligned with the principal axis of inertia of the rigid body, the dynamical model of the modular multi-propeller aerial robot $\mathcal{M}$ with respect to the inertial frame is described by

$$\begin{aligned} M_m \ddot{p} &= R f_c + M_m g e_3 + u_d \\ J_m \dot{\omega} &= -\omega \times J_m \omega + u_\tau + u_{\tau,d} \end{aligned} \tag{2.7}$$

where M_m is the total mass of the modular system, J_m denotes the inertia of the modular vehicle, $p = \text{col}(x, y, z)$ is the position of the center of mass, ω the angular velocity expressed in the body frame F_{b_m}, R the rotation matrix relating the reference frames F_{b_m} and F_i, and e_3 the unit vector $e_3 := [0, 0, 1]^T$. Moreover u_d and $u_{\tau,d}$ represent respectively the resultant force and a torque disturbance deriving from the disturbance wrench $[f_{d,i},\ \tau_{d,i}]$ produced by each single module $i \in \{1, 2..., N + P\}$.

By denoting with M the mass of a single actuator module, with $J_{A,i}$ the inertia matrix of a single actuator module expressed in its reference F_{b_i} and with $M_{P,i}$ and $J_{P,i}$, $i \in \{1, ..., P\}$, the mass and the inertia of each payload module, in the case in which the mass of the links connecting the modules can be neglected, applying Huygens-Steiner theorem we have[1]

$$\begin{aligned} J_m = \textstyle\sum_{i=1}^{N} R_{\psi_i} J_{A,i} R_{\psi_i}^T + M \sum_{i=1}^{N} J_i(r_i, h_i, \psi_i) + \\ + \sum_{i=1}^{P} R_{\psi_{N+i}} J_{P,i} R_{\psi_{N+i}}^T + \sum_{i=1}^{P} M_{P,i} J_{N+i}(r_{N+i}, h_{N+i}, \psi_{N+i}) \end{aligned}$$

with

$$J_i(r_i, h_i, \psi_i) = \begin{bmatrix} r_i^2 \sin^2 \psi_i + h_i^2 & 0 & 0 \\ 0 & r_i^2 \cos^2 \psi_i + h_i^2 & 0 \\ 0 & 0 & r_i^2 \end{bmatrix}$$

and $M_m = NM + \sum_{i=1}^{P} M_{P,i}$.

2.1.3 Control Strategy

For systems characterized by actuators redundancy the design of the control law may be simplified by employing a control allocation schemes [17], [23], [57]. The idea is to divide the controller in two different parts, namely the feedback control law and the control allocation algorithm. Goal of the feedback law is to compute the resultant forces and torques to be applied to the modular vehicle in order to obtain the desired

[1] For sake of compactness, we denote $M_{P,0} = 0$ and $J_{P,0} = 0$.

closed-loop behavior. These control forces and torques are then obtained trough the control allocation algorithm by combining the effects of all the actuators available on the modular system.

2.1.4 Control Allocation

This subsection introduces the main results pertaining control allocation. Before introducing the specific control allocation problem for the class of vehicles of interest, the special case of positive inputs is considered in successive Subsection. This is in fact the distinguishing feature of system (2.7) due to the definition of the inputs in (2.3) characterizing each actuator module. In a first step, we show how, under certain conditions, standard linear control allocation techniques can be adopted also for this class of systems neglecting the positiveness constraint. With this result at hand, the specific control allocation problem for the multi-propeller modular vehicles is presented.

Control Allocation for Systems with Positive Inputs

For systems characterized by positive inputs, we consider the following result:

Proposition 2.1. *Let $B \in \mathbb{R}^{m\times n}$, $n > m > 0$. If*

1) rank$(B) = m$, and

2) there exists $u_p \in \mathbb{R}^n_{>0}$ such that $Bu_p = 0$,

then there exists $u \in \mathbb{R}^n_{\geq 0}$ such that

$$Bu = v, \tag{2.8}$$

for all $v \in \mathbb{R}^m$.

Proof. Under the above assumptions, a feasible solution to (2.8) can be computed as

$$u = B^+ v + \lambda u_p \tag{2.9}$$

where $\lambda \geq 0$ is given by

$$\lambda = \underset{i\in\{1,\,N\}\,:\,(B^+v)(i)<0}{\operatorname{argmin}} \lambda u_p(i) + (B^+v)(i) = 0\,. \tag{2.10}$$

□

As shown in Proposition 2.1, the case in which the inputs are constrained to be positive represents a particular case of general constrained control allocation [17]. In fact a solution can be derived neglecting the presence of such constraint provided that the

kernel of the B matrix contains a vector given by positive values. Such a vector can be in fact employed to "adjust" the sign of the inputs obtained applying standard techniques, such as the generalized pseudo-inverse.

The Vectored-Thrust Control Allocation Problem

In the control allocation problem for the multi-propeller vehicle $\mathcal{MP}$ it is of interest to assign a desired torque vector and only the force component directed along the body z-axis. The idea is in fact to govern the vehicle by using vectored-thrust control paradigms - see among others [51] - for thrust-propelled aerial robots. For this reason, the problem is referred to as *vectored-thrust* control allocation problem (VT-CAP) and it is formulated as follows.

Problem VT-CAP: Given $u_f^\star \in \mathbb{R}_{\geq 0}$ and $u_\tau^\star \in \mathbb{R}^3$, find a value of $\mathbf{u} \in \mathbb{R}^N_{\geq 0}$ such that

$$B_{VT}(\Psi, \mathcal{S}, \mathcal{R})\mathbf{u} = \begin{bmatrix} -u_f^\star \\ u_\tau^\star \end{bmatrix} \tag{2.11}$$

in which $B_{VT}(\Psi, \mathcal{S}, \mathcal{R}) \in \mathbb{R}^{4\times N}$ is given by

$$B_{VT}(\Psi, \mathcal{S}, \mathcal{R}) = \begin{bmatrix} B_f|_{(3,:)} \\ B_\tau(\Psi, \mathcal{S}, \mathcal{R}) \end{bmatrix}. \tag{2.12}$$

□

To derive a solution to $VT-CAP$, the construction proposed in Proposition 2.1 will be taken into account. More specifically, the following proposition can be stated:

Proposition 2.2. *Given $u_f^\star > 0$ and $u_\tau^\star \in \mathbb{R}^3$ there exists $\epsilon(u_f^\star) > 0$ such that if $\|u_\tau^\star\| \leq \epsilon$ and*

a) there exists $\mathbf{u}'_p \in \mathbb{R}^N_{>0}$ such that $B_\tau(\Psi, \mathcal{S}, \mathcal{R})\mathbf{u}'_p = 0$, and

b) rank $(B_{VT}(\Psi, \mathcal{S}, \mathcal{R})) = 4$

then problem $VT-CAP$ admits a solution such that $\mathbf{u} \in \mathbb{R}^N_{>0}$.

Proof. To prove the result we compute a solution to $VT-CAP$ by employing the assumptions in the statement of the proposition.

1) Note that $B_\tau(\Psi, \mathcal{S}, \mathcal{R}) \in \mathbb{R}^{3\times N}$ satisfies constraint 1) and 2) of Proposition 2.1. In fact, from b), rank$(B_\tau(\cdot)) = 3$, and, from a), constraint 2) in Proposition 2.1 holds true with $u_p = \mathbf{u}'_p$. Accordingly a solution $\mathbf{u}^\star \in \mathbb{R}^n_{\geq 0}$ such that $B_\tau(\Psi, \mathcal{S}, \mathcal{R})\mathbf{u}^\star = u_\tau^\star$ can be obtained from the algorithm (2.9), namely

$\mathbf{u}^\star := B_\tau(\Psi, \mathcal{S}, \mathcal{R})^+ u_\tau^\star + \lambda \mathbf{u}'_p$, with $\lambda \geq 0$ computed as in (2.10);

2) given $u^\star \in \mathbb{R}^n_{\geq 0}$, we denote with $\sigma^\star := \sum_{i=1,..N} \mathbf{u}^\star(i)$. Since $B_\tau(\cdot)$ is a linear map, for a sufficiently small value of ϵ then $\sigma^\star \geq 0$ is arbitrary small;

3) let $\sigma^{thrust} := u_f^\star / K_T$, $\sigma' := \sum_{i=1,..N} \mathbf{u}'_p(i)$ and assume that $\sigma^\star <= \sigma^{thrust}$ (this condition can be satisfied by requiring ϵ small). Let be $\lambda' := (\sigma^{thrust} - \sigma^\star)/\sigma'$. Then a solution to $VT - CAP$ is given by

$$\mathbf{u} := B_\tau(\Psi, \mathcal{S}, \mathcal{R})^+ u_\tau^\star + (\lambda + \lambda')\mathbf{u}'_p \,. \tag{2.13}$$

□

Proposition 2.2 shows how standard linear control allocation techniques can be adopted for the multi-propeller configuration $\mathcal{MP}$ provided that the desired torque vector is small compared to the magnitude of desired force $u_f^\star$. The result, following the framework presented in Proposition 2.1, relies on the existence of a positive input vector, $\mathbf{u}'_p \in \mathbb{R}^n_{>0}$ in the statement of the proposition, contained into the kernel of the matrix $B_\tau(\cdot)$. Interestingly enough, the computation of such element can be simplified for some important configurations of practical interest. In particular, the following result holds true.

Proposition 2.3. *Let be $\mathcal{MP}$ a multi-propeller with $N \geq 4$ and $P \geq 0$. If the following conditions hold true*

- $\sum_{i=1}^N s_i = 0$;
- *balanced payload: $\sum_{i=1}^N R_{\psi_i}^T [r_i,\, 0,\, 0]^T = [0,\, 0,\, 0]^T$, namely the lateral / horizontal position of the center of gravity of the system coincides with the one of the centroid formed by the actuator modules,*

then $u^{bal} = [1,\, 1,\, ...,\, 1]^T$ is such that $B_\tau(\Psi, \mathcal{S}, \mathcal{R}) u^{bal} = 0$.

Proof. From the first item and the definitions of G_τ and R_ψ it holds that $\sum_{i=1}^N (R_{\psi_i} G_\tau)|_{(3)} = 0$. From the second item, since $\sum_{i=1}^N R_{\psi_i}^T [r_i,\, 0,\, 0]^T = [0,\, 0,\, 0]^T$, it follows that

- $\sum_{i=1}^N \cos\psi_i r_i = 0$;
- $\sum_{i=1}^N \sin\psi_i r_i = 0$,

and hence, from the definition of G_τ, $\sum_{i=1}^N R_{\psi_i}^T G_\tau(s_i, r_i) = [0,\, 0,\, 0]^T$. □

Remark: The above proposition shows that, when the system is balanced i.e. the payload does not affect the position of the center of gravity, a simple positive input can

be obtained by requiring all the actuator modules to generate the same amount of thrust. Interestingly enough, following the construction of the solution in Proposition 2.2, this solution allows to equally distribute among the different modules the thrust required to obtain the final desired control force $u_f^\star$.

□

Remark: Note that, in the general case in which the balanced payload property may not be satisfied, the positive input can be computed by defining a convex optimization problem, e.g.

$$\begin{aligned} \min\, & g(\mathbf{u}'_p) \\ & B\tau(\Psi, \mathcal{S}, \mathcal{R})\mathbf{u}'_p = 0 \\ & \mathbf{u}'_p \in \mathbb{R}^N_{>0} \end{aligned}$$

for some convex cost function $g : \mathbb{R}^N \to \mathbb{R}_{\geq 0}$.

□

2.1.5 Application: Modeling of a Quad-Rotor Helicopter

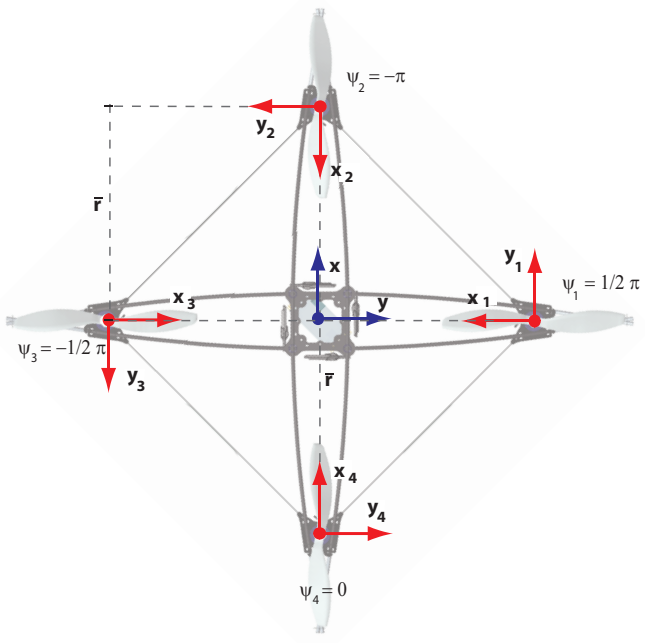

Figure 2.1: Quadrotor as a multi-propeller modular vehicle

For the quadrotor depicted in Figure 2.1 the system is assumed to be composed by 4 actuator modules and one single payload module. The actuator modules are disposed at a distance $\bar{r}$ from the center of gravity, the payload module is assumed to be located at the center of mass of the vehicle. As far as the control wrench generation is concerned,

we obtain

$$B_f = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -K_T & -K_T & -K_T & -K_T \end{bmatrix},$$

$$B_\tau(\Psi, \mathcal{S}, \mathcal{R}) = \begin{bmatrix} -K_T\bar{r} & 0 & K_T\bar{r} & 0 \\ 0 & K_T\bar{r} & 0 & -K_T\bar{r} \\ K_Q s_1 & K_Q s_2 & K_Q s_3 & K_Q s_4 \end{bmatrix}, .$$

When $s_1 = s_3$, $s_2 = s_4$ and $s_1 \neq s_2$ we have that rank$(B_{VT}) = 4$. Moreover, since

$$\sum_{i=1}^{4} R_{\psi_i}^T [r_i,\, 0,\, 0]^T = [0,\, 0,\, 0]^T\,,$$

namely the system is balanced, we have that $\mathbf{u}'_p = [1,\, 1,\, 1,\, 1]^T$, is such that $(B_\tau)\mathbf{u}'_p = 0$. In case the payload is not located in the center of mass the system may fail to be balanced, since the center of gravity may not coincide with the centroid formed by the four actuator modules. In this other case a different solution to $(B_\tau)\mathbf{u}'_p = 0$, with $\mathbf{u}'_p \in \mathbb{R}^4_{>0}$ can be computed using numerical techniques.

2.2 Differential Wheel Robot Model

The models of unicycles and car-like robots are well known in robotic literature [73] [6] [93]. They fall under the category of nonholonomics models due to the non-holonomic constrain they are subject to. This kind of constraints is typical of wheeled robots. What is constrained are the velocities of the robots, that cannot assume arbitrary and independent values.

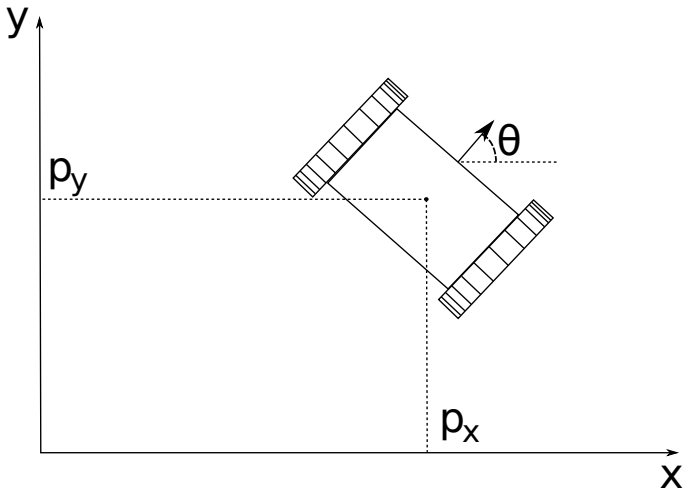

Figure 2.2: Differential wheel robot in 2D and its states.

We report the kinematic model of a unicycle in 2D that can model a differential wheel robot (Figure 2.2), a robot with two independent wheels:

$$\begin{cases} \dot{p}_x = v\ \cos\theta \\ \dot{p}_y = v\ \sin\theta \\ \dot{\theta} = \omega \\ v > 0 \end{cases} \tag{2.14}$$

in which p_x and p_y are the position in a cartesian plane, θ the heading angle of the robot, v and ω the linear and angular velocities, respectively. The linear and angular velocities are a combination of the velocities of the two independent wheels. If the velocities of the wheels are w_1, w_1, the distance of the two wheels is given by d_w and the radius of the wheel is R_w, the linear and angular velocities are given by:

$$\begin{cases} v = \frac{w_1+w_2}{2} \\ \omega = \frac{R_w}{d_w}\frac{w_1-w_2}{2} \end{cases} \tag{2.15}$$

The system (2.14) is characterized by a state $x = [p_x,\, p_y,\, \theta] \in \mathbb{R}^3$, input $u = [v,\, \omega] \in \mathbb{R}^2$.

Following [73], this model is equivalent, with some input transformation to the model of a car-like robot with front steering wheels, which model is given by:

$$\begin{cases} \dot{p}_x = v \ \cos\theta\cos\phi \\ \dot{p}_y = v \ \sin\theta\cos\phi \\ \dot{\theta} = v\tan\phi/l \\ \dot{\phi} = \omega' \end{cases} \tag{2.16}$$

where in this case ϕ is the steering angle of the front wheel, l is the distance from front and rear wheels and ω' is the steering velocity input. Hence, the model and control proposed in this thesis can be adapted to car-like front steering wheel robots.

I can control my destiny, but not my fate. Destiny means there are opportunities to turn right or left, but fate is a one-way street. I believe we all have the choice as to whether we fulfill our destiny, but our fate is sealed.

Paulo Coelho

3

Robots Control

In this chapter we present the control law for trajectory tracking of the two families of robots of interest: VTOL UAV and differential wheel robot. The aim is to build state-feedback control to track a desired trajectory. Both robots result underactuated, so the trajectory to track is only in a subset of the configurations space. The dynamical models used to build the control law are the ones presented in the previous chapter.

3.1 VTOL UAV Control

In this section we want to provide a suitable control law [91] for the VTOL system, compatible to the model proposed in the previous chapter. The general architecture for the control can be seen in Figure 3.1.

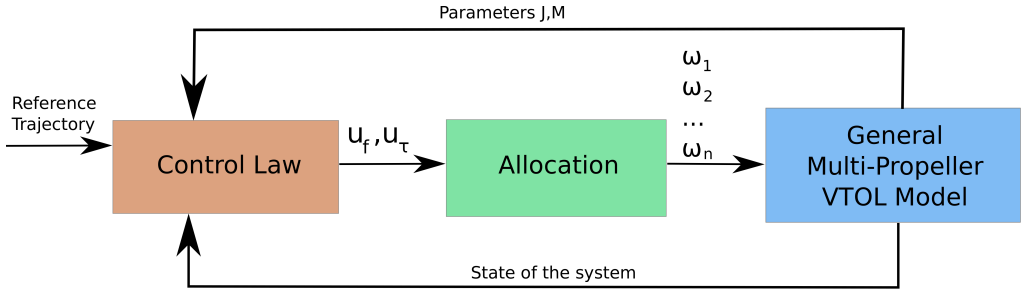

Figure 3.1: Control, model and allocation architecture for VTOL.

The general multi-propeller VTOL model and allocation were described in previous Chapter. The Control Law block is what will develop in this Chapter. The basic idea is that the control law generates suitable inputs in terms of T, τ (thrust and torques) to track a suitable trajectory. The control law is the same for the class of VTOL we model, where the only difference is on some tuning parameters that depend on M, J. The control inputs are then converted by the allocation block into suitable inputs for the VTOL, i.e. rotational speed of propellers $\omega_1, \omega_2, ..., \omega_n$. Miniature Vertical Take-Off and Landing (VTOL) aerial systems are currently employed successfully in a large number of applications including, among others, surveillance, aerial photography and search and rescue operations [32]. One reason for this large success is the high level of maneuverability, which allows to safely perform flight missions in densely cluttered environments [8] or to perform advanced robotic tasks [80]. Among the different configurations, VTOL class of aerial systems includes helicopters [44] ducted-fan tail-sitters [96] [92] and multi-propeller helicopters [101] [18] [26]. All of these vehicles are under-actuated mechanical systems, in which the number of available control inputs is less than the number of degrees-of-freedom (d.o.f.). As a consequence, to achieve the high level of agility required by real-world applications, the feedback control design plays a central role.

Several contributions [49] [81] [63] [75] document different approaches to the control design for such a class of under-actuated systems.

In [52], almost-global stability results are demonstrated by means of Lyapunov based techniques. Results therein show robustness also in the presence of aerodynamic drag disturbances that typically affect aerial systems. Trajectory tracking in the absence of linear velocity measurements has been considered in [4] where a hierarchical controller has also been proposed. In [68], almost-global stability results are achieved by consid-

ering geometric methods and then applied to the control of a quadrotor aerial vehicle. Backstepping control design has been proposed in [34] in order to perform aggressive maneuvers by considering the dynamics of a model helicopter, and in [22] by considering a hybrid controller able to globally stabilize a desired trajectory. In [79] and [54], inner-outer loop control strategies have been employed to stabilize the dynamical model of a miniature helicopter. The proposed methodology takes into account for the feedback interconnection between the inner attitude and the outer position control loops. In particular, nested saturations and high-gain control techniques are used to show stability of the overall closed-loop system under some limitations in term of the initial attitude configuration. More recently, a survey describing feedback control design for under-actuated VTOL systems has appeared in [51].

In this chapter, hierarchical control strategies for a miniature VTOL vehicle to track a desired trajectory globally with respect to the initial position and attitude configuration are proposed. In particular, drawing inspiration from recent results pertaining to attitude control of rigid bodies [83], hybrid control techniques [45] are used to overcome the topological obstruction affecting continuous-time globally stabilizing control laws [12]. Robustness with respect to possibly large exogenous disturbances and parametric uncertainties is the main contribution. This is achieved by combining total stability tools for nonlinear control systems [53] with a suitable design of the hybrid control law.

Two different hierarchical control approaches are employed and compared. The first approach is based on the idea of “breaking the loop" between the attitude and the position closed-loop dynamics through a suitable choice of the control torques. The overall closed-loop system can be considered as a cascade connection in which the attitude and the position controllers can be tuned independently to achieve the desired stability properties. However, this control design relies upon the perfect knowledge of the vehicle dynamics and then it may not be effective in the general real-world scenario in which uncertainties and disturbances, including wind and aerodynamic drag forces, affect the dynamics of the vehicle.

To overcome this important limitation, a second approach is proposed. Since the lack of full knowledge of the system dynamics prevents one to compensate the influence of the position dynamics on the attitude of the vehicle, we propose an attitude controller that combines a linear hybrid feedback law and a feed-forward law obtained by nominal model inversion. The resulting control law is more suitable for practical implementation on a real autopilot due to robustness properties. Such a control law, however, leads to a more involved feedback interconnection between the hybrid attitude and the continuous-time position closed-loop subsystems and, in turn, to a more complex closed-loop analysis.

For definition and notion of stability in Hybrid Systems, that are used in this Chapter,

the reader can refers to the Appendix at the end of this thesis.

3.1.1 Problem Formulation

Dynamical Model

The dynamics of a large class of miniature Vertical Take-Off and Landing (VTOL) aerial vehicles, including helicopters, ducted-fan and multi-propeller configurations, can be described by considering the following dynamic model (see among others [51], [4])

$$\begin{aligned} M\ddot{p} &= -u_f R e_3 + M g e_3 + d_f \\ \dot{R} &= RS(\omega) \\ J\dot{\omega} &= S(J\omega)\omega + u_\tau + d_\tau \end{aligned} \tag{3.1}$$

in which $p = [x,\, y,\, z]^\top \in \mathbb{R}^3$ denotes the position of the center of gravity of the system expressed in the inertial reference frame $\mathcal{F}_i$, $\omega = [\omega_x,\, \omega_y,\, \omega_z]^\top \in \mathbb{R}^3$ is the angular speed expressed in the body frame $\mathcal{F}_b$, $R \in SO(3)$ is the rotation matrix relating vectors in $\mathcal{F}_b$ to vectors in $\mathcal{F}_i$, $M \in \mathbb{R}_>$ and $J \in \mathbb{R}^{3\times 3}$ (with the property that $J = J^\top > 0$) are the mass and the inertia matrix of the system, $u_f \in \mathbb{R}_{\geq 0}$ denotes the control force that, by construction, is directed along the body z axis and $u_\tau \in \mathbb{R}^3$ is the control torque vector. The force and torque vectors $d_f \in \mathbb{R}^3$ and $d_\tau \in \mathbb{R}^3$ are bounded unknown exogenous signals modeling the effects of aerodynamic drag and wind disturbances.

To model actuator limitations, the control force and torques are required to satisfy

$$u_f \in \Omega_f, \quad u_\tau \in \Omega_\tau \tag{3.2}$$

where the compact sets $\Omega_f \subset \mathbb{R}_{\geq 0}$ and $\Omega_\tau \subset \mathbb{R}^3$ define the attainable force and torques for the specific vehicle.

Besides the presence of the exogenous disturbances d_f and d_τ, the further source of uncertainty considered in the paper is the inertia matrix J. More specifically, it is assumed that only a nominal value $J_0 \in \mathbb{R}^{3\times 3}$, with $J_0 = J_0^\top > 0$, and an "upper-bound" $J^U \in \mathbb{R}^{3\times 3}$, i.e., such that

$$|x^\top J^U x| \geq |x^\top J x| \tag{3.3}$$

for all $x \in \mathbb{R}^3$, are known. This uncertainty on the value of J reflects the fact that, for a physical system having a complex mass distribution, an exact inertia matrix may not be available.

Remark 3.1. Instead of using a rotation matrix R, the attitude in (3.1) can be parameterized by means of the unit quaternion $q \in \mathbf{S}_3$. In this case, in the first equation in (3.1) the rotation matrix is replaced by the Rodrigues map $\mathcal{R}$ and the kinematic equations, which

are given by the second equation in (3.1), are replaced by

$$\dot{q} = \frac{1}{2} q \otimes \begin{bmatrix} 0 \\ \omega \end{bmatrix} . \tag{3.4}$$

In many applications, quaternion parametrization of attitude is often preferred due to the small number of parameters (4 with respect to 9 required by a rotation matrix) and the computationally simple quaternion algebra [104]. △

Control Problem and Nominal System Inversion

This work focuses on the problem of *global* tracking by state feedback for system (3.1). More specifically, the goal is to asymptotically track a given time reference position and orientation

$$t \mapsto p^*(t) \in \mathbb{R}^3, \quad t \mapsto R^*(t) \in SO(3) \tag{3.5}$$

for all possible initial conditions $p(0) \in \mathbb{R}^3$, $\dot{p}(0) \in \mathbb{R}^3$, $R(0) \in SO(3)$, $\omega(0) \in \mathbb{R}^3$, by assuming full knowledge of the state of the system.

Two different scenarios will be considered. The first scenario, referred to as the *nominal case*, is when the force and the torque disturbances are neglected, i.e., $d_f \equiv d_\tau \equiv 0$, and the inertia matrix is perfectly known, i.e., $J^U \equiv J_0 \equiv J$. The second scenario, referred to as the *robust case*, takes into account all of the uncertainties specified in Section 3.1.1. Due to the presence of disturbances, a *practical* tracking result will be derived in the *robust case*, while *asymptotic* tracking results are obtained in the *nominal case*.

The desired references (3.5) are required to satisfy *functional controllability* constraints that are described below. The first constraint derives from the under-actuated nature of system (3.1) by which the reference position and orientation cannot be assigned independently. As the aircraft position assumes a major role in real-world applications [32], the attitude reference is required to satisfy some constraints to meet the requirements posed by the position tracking objective. Specifically, let $t \mapsto p^*(t)$ be the desired position reference and let $t \mapsto v_f^*(t)$ be the *nominal reference control force vector* defined as

$$v_f^*(t) := Mge_3 - M\ddot{p}^*(t) \quad \forall t \geq 0 \, . \tag{3.6}$$

The function in (3.6) represents the force vector yielding the desired acceleration $\ddot{p}^*(t)$ in the *nominal case* (i.e., when $d_f \equiv 0$). The enforcement of such a $v_f^*(t)$ necessarily requires that the body z-axis of the vehicle, i.e., the thrust direction, is aligned with $v_f^*(t)$ at each t. This requires that the reference attitude $t \mapsto R^*(t) \in SO(3)$ satisfies the following

constraint

$$R^*(t)e_3 = \frac{v_f^*(t)}{|v_f^*(t)|} \quad \forall t \geq 0\,. \tag{3.7}$$

Implicit in the previous expression is the requirement that $t \mapsto \ddot{p}^*(t)$ is such that

$$|v_f^*(t)| = M|ge_3 - \ddot{p}^*(t)| > \underline{v}, \;\; \forall t \geq 0 \tag{3.8}$$

for some $\underline{v} \in \mathbb{R}_{>0}$, which is assumed hereafter.

The problem of computing a rotation matrix satisfying (3.7) has been considered, for instance, in [68] using differential geometric tools, and in [4] using a unit quaternion parametrization of R^*. Details on the computation of a rotation matrix R^* fulfilling (3.7) are presented in Appendix A.1.4.

With $t \mapsto v_f^*(t)$ and $t \mapsto R^*(t)$ fulfilling (3.7) and (3.8) in hand, the nominal system inversion can be accomplished by defining the reference force and torque control inputs as

$$u_f^*(t) \;\; := \;\; |v_f^*(t)| \tag{3.9}$$

and

$$u_\tau^*(t) \;\; := \;\; J\dot{\omega}^*(t) - S(J\omega^*(t))\omega^*(t)\,,$$

for each $t \geq 0$, where $\omega^*(t) := (R^{*T}(t)\dot{R}^*(t))^\wedge$ is the reference angular velocity. Note that u_τ^* depends on the uncertain parameter J; hence, it can be computed only in the *nominal case*. In the following, for the sake of clarity, we shall denote by $u_{\tau 0}^*$ the nominal value of u_τ^*, namely

$$u_{\tau 0}^* \;\; := \;\; J_0\dot{\omega}^* - S(J_0\omega^*)\omega^*\,. \tag{3.10}$$

The reference angular velocity ω^* and its time derivative along the body x and y axis can be easily derived as functions of p^* and its time derivatives. As a matter of fact, using the second equation in (3.1), it follows that $R^{*\top}\dot{R}^* e_3 = S(\omega^*)e_3$ by which

$$\begin{bmatrix} \omega_x^* \\ \omega_y^* \end{bmatrix} := W_{xy} R^{*\top} \frac{d}{dt} \frac{v_f^*}{|v_f^*|}\,, \tag{3.11}$$

$$\begin{bmatrix} \dot{\omega}_x^* \\ \dot{\omega}_y^* \end{bmatrix} := W_{xy} \left(-S(\omega^*) R^{*\top} \frac{d}{dt} \frac{v_f^*}{|v_f^*|} + R^{*\top} \frac{d^2}{dt^2} \frac{v_f^*}{|v_f^*|} \right)$$

where $W_{xy} \in \mathbb{R}^{2\times 3}$ is the matrix with the first and second rows given by $[0,\, -1,\, 0]$ and $[1,\, 0,\, 0]$, respectively. On the other hand, the angular speed and acceleration along the body z-axis, namely $t \mapsto \omega_z^*$ and $\dot{\omega}_z^*$, are not subjected to constraints deriving from the position tracking objective.

Further constraints on the reference position $t \mapsto p^*(t)$ and the reference orientation

$t \mapsto R^*(t)$ must be chosen to let the control force and torques computed in (3.9) and (3.10) satisfy the actuator limitations (3.2), namely

$$u_f^*(t) \in \Omega_f, \;\; u_\tau^*(t) \in \Omega_\tau \;\; \forall t \geq 0\,. \tag{3.12}$$

In particular, the reference derivatives $p^{*\,(1)}$, $p^{*\,(2)}$, $p^{*\,(3)}$, $p^{*\,(4)}$, ω_z^* and $\omega_z^{*\,(1)}$ are required to be bounded functions of time satisfying appropriate bounds.

Remark 3.2. Let $R_1 \in SO(3)$ be such that (3.7) holds with $R^* = R_1$. Then (3.7) also holds by picking $R^* = R_1 R_z$ for any $R_z \in SO(3)$ such that $R_z e_3 = e_3$ (i.e., R_z represents an elementary rotation around the e_3 unit vector). This fact shows that the relation given in (3.7) fixes only two of the three degree of freedom of R^*. The third degree of freedom, which is the rotation around the vector $v_f^*(t)$, can be arbitrarily assigned according to attitude tracking objectives. △

3.1.2 Inner-Outer Loop Control Strategies

This section presents control strategies that solve the global tracking problem in the *nominal* and *robust* case. The proposed solutions rely upon a hierarchical control structure having the attitude and the position closed-loop dynamics playing the role of the *inner loop* and of the *outer loop*, respectively. A *vectored-thrust* control paradigm (see [51]) is followed in the design of the control law. In this respect, a crucial role in avoiding singularities is played by the use of saturation functions in the outer loop that naturally lead to the design of a "control vectored-thrust" whose amplitude never vanishes regardless of the values assumed by the position error. This feature, in turn, enables the adoption of vectored-thrust design paradigms in setting up references signal for the attitude dynamics on which the inner loop is built.

As far as the inner loop is concerned, two different control strategies are presented to address the *nominal* and *robust cases*, respectively. In the *nominal case*, the torque control input is synthesized as a "feedback linearizing" control law able to decouple the closed-loop attitude dynamics from the position dynamics. The resulting control loop, which is depicted Figure 3.2, is a *cascade* interconnection between the attitude and the position loops. In the next subsections, it will be shown that stability of the overall interconnected system does not impose constraints on the tuning of the position and attitude controllers. The above property can be also achieved by designing a torque control input able to only partially decouple the closed-loop attitude and position dynamics. This leads to the feedback interconnection depicted in Figure 3.3, in which, due to a suitable design of the torque control input, the influence of the position dynamics on the attitude dynamics does not affect the stability properties of the overall closed-loop system.

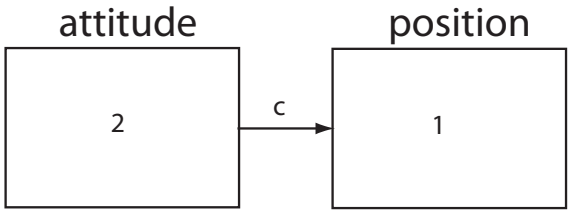

Figure 3.2: Nominal Case: Cascade Interconnection

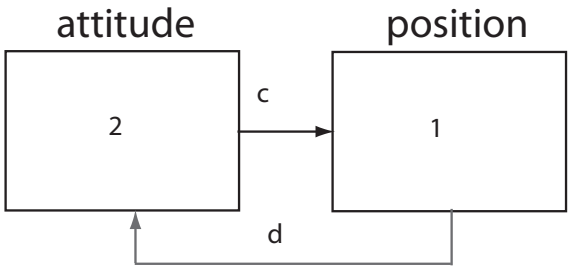

Figure 3.3: Nominal Case: Feedback Interconnection

In the *robust case*, a complete decoupling of the closed-loop attitude and position dynamics is not possible. In this case, the proposed attitude controller combines only a linear feedback law, driven by a hybrid system to overcome the topological obstruction, and a feed-forward law obtained by model inversion.

This control law will lead to a feedback interconnection between the attitude (inner) and the position (outer) loop (see Figure 3.4). In the stability analysis of this feedback interconnection, a crucial role is played by the use of nested saturation functions (used in the outer loop) introducing a “decoupling effect" between the two interconnected dynamics. Such an effect, in turn, is crucial to show that the attitude loop, which is a hybrid system, has solutions that, after some finite time, only flow. This property of solutions is instrumental to establish asymptotic properties of the feedback interconnection.

Stabilization of the translational motion

By letting

$$\tilde{\mathbf{p}} := \begin{bmatrix} \tilde{p} \\ \dot{\tilde{p}} \end{bmatrix} := \begin{bmatrix} p - p^* \\ \dot{p} - \dot{p}^* \end{bmatrix}$$

and bearing in mind the first equation in (3.1), the position error dynamics are described by

$$M\ddot{\tilde{p}} = -u_f R e_3 + v_f^* + d_f \tag{3.13}$$

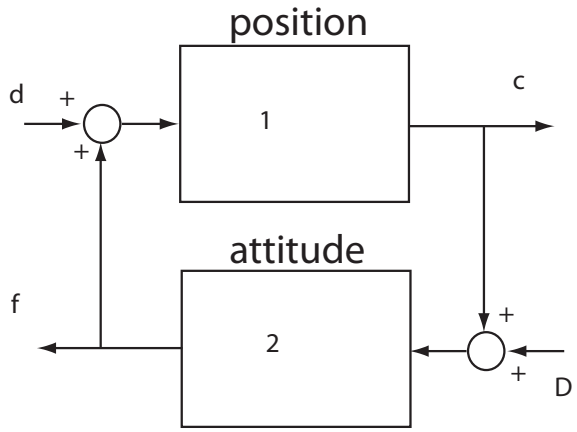

Figure 3.4: Robust Case: Feedback Interconnection

with v_f^* defined in (3.6). A vectored-thrust control strategy is employed to stabilize system (3.13). To this end, let the *control force vector* v_c be defined as

$$v_c(\tilde{\mathbf{p}}, t) := v_f^*(t) + \kappa(\tilde{\mathbf{p}})\,, \tag{3.14}$$

where κ is a state feedback law satisfying $\kappa(0) = 0$ and

$$|\kappa(\tilde{\mathbf{p}})| \leq \bar{\kappa} \qquad \forall\, \tilde{\mathbf{p}} \in \mathbb{R}^6 \tag{3.15}$$

with $0 < \bar{\kappa} < \underline{v}$. Property (3.15), which will be fulfilled in the following by designing $\kappa(\cdot)$ as a saturated function, guarantees that

$$|v_c(\tilde{\mathbf{p}}, t)| \geq |v_f^*(t)| - |\kappa(\tilde{\mathbf{p}})| \geq \underline{v} - \bar{\kappa} > 0 \tag{3.16}$$

for all $\tilde{\mathbf{p}} \in \mathbb{R}^6$ and $t \geq 0$. The form of (3.13) suggests to design the force control input $u_f \in \mathbb{R}_{>0}$ and a desired reference attitude $R_c \in SO(3)$ in such a way that $u_f(t) R_c(\tilde{\mathbf{p}}, t) e_3 = v_c(\tilde{\mathbf{p}}, t)$, namely

$$R_c(\tilde{\mathbf{p}}, t) e_3 \;=\; \frac{v_c(\tilde{\mathbf{p}}, t)}{|v_c(\tilde{\mathbf{p}}, t)|} \tag{3.17}$$

and

$$u_f = u_{fc}(\tilde{\mathbf{p}}, t) := |v_c(\tilde{\mathbf{p}}, t)|\,. \tag{3.18}$$

Note that (3.17), (3.18) are well defined for all $\tilde{\mathbf{p}} \in \mathbb{R}^6$ and for all $t \geq 0$ by virtue of (3.16) and of (3.15). By bearing in mind the discussion in Remark 3.2, relation (3.17) fixes two of the three degree of freedom characterizing $R_c \in SO(3)$. The third degree of freedom

can be fixed by enforcing the constraint

$$R_c(0,t) = R^*(t) \quad \forall t \geq 0 \tag{3.19}$$

which, along with (3.17), uniquely defines the control reference attitude $R_c \in SO(3)$.

By adding and subtracting the term $u_f R_c e_3$ in (3.13), the position error dynamics read as

$$M\ddot{\tilde{p}} = -\kappa(\tilde{\mathbf{p}}) + \Gamma(R_c, R) + d_f \tag{3.20}$$

where Γ is defined as

$$\Gamma(R_c, R) := u_{f_c} (R_c - R)\, e_3\,. \tag{3.21}$$

The design of $\kappa(\cdot)$ must be conceived to stabilize the origin of (3.20), which is a double integrator forced by the exogenous inputs Γ and d_f, and to fulfill the crucial requirement (3.15). Nested saturations can be used for such a purpose ([109, 54]). Among the possible nested saturation design solutions available in literature (see, for instance, [109], [7], [72]), the approach in [54, Appendix C] yields the following control law

$$\kappa(\tilde{\mathbf{p}}) := \lambda_2 \sigma \left(\frac{k_2}{\lambda_2} \left(\dot{\tilde{p}} + \lambda_1 \sigma \left(\frac{k_1}{\lambda_1} \tilde{p} \right) \right) \right) \tag{3.22}$$

in which σ is a saturation function defined in Chapter while λ_1, λ_2, k_1, and k_2 are chosen as

$$\lambda_i = \varepsilon^{(i-1)}\, \lambda_i^\star, \qquad k_i = \varepsilon\, k_i^\star, \qquad i = 1, 2 \tag{3.23}$$

where $k_i^\star$, $\lambda_i^\star$ are positive constants (fixed as Proposition 3.1 below) and $\varepsilon > 0$. Note that, by the definition of saturation function,

$$|\kappa(\tilde{\mathbf{p}})| \leq \sqrt{3}\, \lambda_2^\star \varepsilon\,.$$

Property (3.15) is thus fulfilled by fixing ε as

$$0 < \varepsilon \leq \frac{\bar{\kappa}}{\sqrt{3}\lambda_2^\star}\,. \tag{3.24}$$

The asymptotic properties of the closed-loop position system (3.20), (3.22) are detailed in the following proposition.

Proposition 3.1. *Consider the closed-loop position error dynamics (3.20)-(3.22) with λ_i and k_i, $i = 1, 2$, chosen as in (3.23), (3.24) and $\lambda_i^\star$, $k_i^\star$ taken as*

$$\frac{\lambda_2^\star}{k_2^\star} < \frac{\lambda_1^\star}{4}, \qquad 4k_1^\star\lambda_1^\star < \frac{\lambda_2^\star}{4}, \qquad 6\frac{k_1^\star}{k_2^\star} < \frac{1}{24}\,. \tag{3.25}$$

Then, there exist $R_\Gamma > 0$, $R_{d_f} > 0$ *and* $\gamma_{\rm pos} > 0$ *such that the system is Input-to-State Stable with respect to the inputs* (Γ, d_f) *without restrictions on the initial state, restrictions* $(R_\Gamma,\ R_{d_f})$ *on the inputs and asymptotic gain* $\gamma_{\rm pos}$*. In particular, for all* (Γ, d_f) *such that* $|\Gamma|_\infty \leq R_\Gamma$, $|d_f|_\infty \leq R_{d_f}$ *and for all initial conditions* $\tilde{\mathbf{p}}(0) \in \mathbb{R}^6$*, the resulting trajectories are bounded and the following asymptotic bound holds true*

$$|\tilde{\mathbf{p}}|_a \leq \gamma_{\rm pos}\, \max\{\, |\Gamma|_a\,,\ |d_f|_a\,\}\,.$$

Proof. System (3.20)-(3.22) can be rewritten as

$$\begin{aligned} \dot{\zeta}_1 &= -\lambda_1 \sigma\left(\tfrac{k_1}{\lambda_1}\zeta_1\right) + \zeta_2 \\ M\dot{\zeta}_2 &= -\lambda_2 \sigma\left(\tfrac{k_2}{\lambda_2}\zeta_2\right) + M k_1 \sigma'\left(\tfrac{k_1}{\lambda_1}\zeta_1\right)\dot{\zeta}_1 + \gamma(R_c, R) + \\ &+ d_f \end{aligned}$$

where $\zeta_1 := \tilde{p}$, $\zeta_2 := \dot{\tilde{p}} + \lambda_1 \sigma\left((k_1/\lambda_1)\zeta_1\right)$. Then the result follows from [54, Lemma C.2.1] and [54, Proposition C.2.2] since the system can be written as [54, (C.7)] with $n = 2$, $q_1 = 1$, $q_2 = 1/M$, $v_1 = 0$ and $v_2 = (\Gamma(\cdot) + d_f)/M$. □

If $R = R_c$ then $\Gamma \equiv 0$ and the previous result shows that the position tracking error has an asymptotic bound upper bounded by a function of the disturbance d_f. Due to the fact that saturation functions are used in the design of $\kappa(\cdot)$, the class of force disturbances is required to fulfill the restriction $|d_f|_\infty \leq R_{d_f}$. In particular, if $d_f \equiv 0$, global asymptotic tracking is guaranteed. In the next subsection, we show how the attitude dynamics of the vehicle can be controlled in order to asymptotically enforce the condition $R = R_c$. This is accomplished through appropriate design of the inner loop, in which the rotation matrix R_c plays the role of reference signal for the attitude dynamics.
Instrumental to the design of the inner loop is the computation of the angular velocity ω_c associated to the rotation matrix R_c, defined as

$$\omega_c = (R_c^\top \dot{R}_c)^\wedge\,.$$

The next two lemmas exploit the particular choice of the position control law (3.22) to highlight some properties of ω_c and $\dot{\omega}_c$ that will play a key role in the subsequent analysis.

Lemma 3.1. *The angular velocity* ω_c *can be expressed as*

$$\omega_c = \Omega_1(\tilde{\mathbf{p}}, t) + \Omega_2(\tilde{\mathbf{p}}, t)\,\Gamma(R, R_c) + \Omega_2(\tilde{\mathbf{p}}, t)\,d_f$$

where Ω_1 and Ω_2 are smooth functions satisfying $\Omega_1(0,t) = \omega^\star(t)$ for all $t \geq 0$ and

$$|\Omega_i(\tilde{\mathbf{p}}, t)| \leq \bar{\Omega} \qquad \forall\, \tilde{\mathbf{p}} \in \mathbb{R}^6\,,\ t \geq 0$$

$i \in \{1,\, 2\}$, with $\bar{\Omega}$ a positive constant.

Proof. See Appendix A.1.2. □

Lemma 3.2. *If $d_f \equiv 0$ then there exist $\bar{\Omega}_{c1},\, \bar{\Omega}_{c2} \in \mathbb{R}_{\geq 0}$ such that*

$$|\omega_c|_\infty \leq \bar{\Omega}_{c1} \text{ and } |\dot{\omega}_c|_\infty \leq \bar{\Omega}_{c2}\,.$$

Proof. See Appendix A.1.3. □

The next two subsections present the attitude stabilization in the nominal and robust case, respectively. The analysis in those sections is based on a quaternion parametrization of the attitude. In this respect, we denote by $q_c \in \mathbf{S}_3$ a *control quaternion* associated to R_c, namely $\mathcal{R}(q_c) = R_c$ with $\mathcal{R}(\cdot)$ the Rodrigues map. Due to topological reasons, the computation of q_c from R_c requires lifting continuous paths from $SO(3)$ to $\mathbf{S}_3$. In this paper, this has been achieved by employing the path-lifting mechanism proposed in [82], which ensures that $t \mapsto q_c(t)$ is a continuous function of time.

Attitude Stabilization: the Nominal Case

Let us consider the problem of attitude stabilization in the *nominal case* in which $d_f \equiv 0$, $d_\tau \equiv 0$, and $J \equiv J_0$. We start by defining attitude error coordinates as

$$\begin{array}{rcl} \tilde{q} & := & q_c^{-1} \otimes q \\ \tilde{\omega}_c & := & \omega - \bar{\omega}_c \end{array} \tag{3.26}$$

with $\bar{\omega}_c := R(\tilde{q})^\top \omega_c$. From (3.26), bearing in mind (3.4) and the last equation in (3.1), the following error attitude dynamics can be computed

$$\begin{array}{rcl} \dot{\tilde{q}} & = & \dfrac{1}{2}\tilde{q} \otimes \begin{bmatrix} 0 \\ \tilde{\omega}_c \end{bmatrix} \\ J\dot{\tilde{\omega}}_c & = & \Sigma(\tilde{\omega}_c, \bar{\omega}_c)\tilde{\omega}_c + S(J\bar{\omega}_c)\bar{\omega}_c - J\mathcal{R}(\tilde{q})^\top \dot{\omega}_c + u_\tau\,, \end{array} \tag{3.27}$$

having defined by Σ the skew-symmetric matrix

$$\Sigma(\tilde{\omega}_c, \bar{\omega}_c) := S(J\tilde{\omega}_c) + S(J\bar{\omega}_c) - S(\bar{\omega}_c)J - JS(\bar{\omega}_c)\,.$$

Hereafter, the scalar and vector part of the error quaternion $\tilde{q}$ are denoted, respectively, by $\tilde{\eta}$ and $\tilde{\epsilon}$.
Inspired by [83], the following two controllers are designed

$$u_\tau = u_{\tau,FF}(\tilde{q}, \omega_c, \dot{\omega}_c) + u_{\tau,FB}(\tilde{q}, \tilde{\omega}_c, h)\,, \tag{3.28}$$

$$u'_\tau = u'_{\tau,FF}(\tilde{q}, \omega_c, \dot{\omega}_c) + u_{\tau,FB}(\tilde{q}, \tilde{\omega}_c, h) \tag{3.29}$$

where $u_{\tau,FF}(\cdot)$ and $u'_{\tau,FF}(\cdot)$ are the "feedforward terms", which are given by

$$u_{\tau,FF}(\tilde{q}, \omega_c, \dot{\omega}_c) = J\mathcal{R}(\tilde{q})^\top \dot{\omega}_c - S(J\bar{\omega}_c)\bar{\omega}_c \tag{3.30}$$

and

$$u'_{\tau,FF}(\tilde{q}, \omega_c, \dot{\omega}_c) = u_{\tau,FF} - \left(\Sigma(\tilde{\omega}_c,\, \bar{\omega}_c) - S(J\tilde{\omega}_c)\right)\tilde{\omega}_c\,, \tag{3.31}$$

while $u_{\tau,FB}$ is the "hybrid feedback term"

$$u_{\tau,FB}(\tilde{q}, \tilde{\omega}_c, h) = -k_p\, h\, \tilde{\epsilon} - k_d\, \tilde{\omega}_c \tag{3.32}$$

in which k_p, k_d are positive gains and where $h \in \{-1,\, 1\}$ is a logic variable with hysteresis governed by the hybrid dynamics

$$\begin{cases} \dot{h} = 0 & h\,\tilde{\eta} \geq -\delta \\ h^+ \in \overline{\text{sgn}}(\tilde{\eta}) & h\,\tilde{\eta} \leq -\delta \end{cases} \tag{3.33}$$

where $\delta \in (0,\, 1)$ is the hysteresis threshold and $\overline{\text{sgn}} : \mathbb{R} \rightrightarrows \{-1, 1\}$ is the set-valued function

$$\overline{\text{sgn}}(s) = \begin{cases} \text{sgn}(s) & |s| > 0 \\ \{-1, 1\} & s = 0\,. \end{cases}$$

The goal of the control law (3.29) is to completely decouple the attitude from the position dynamics in order to obtain the cascade connection given in Figure 3.2. The control law (3.28), on the other hand, is such that the skew-symmetric term $\Sigma(\tilde{\omega}_c, \bar{\omega}_c)\tilde{\omega}_c$ in (3.27) is not canceled, leading to the interconnection shown in Figure 3.3.

By considering the control torques (3.28) and (3.29), the corresponding closed-loop attitude error systems, denoted respectively as $\mathcal{H}_{\text{nom}}$ and $\mathcal{H}'_{\text{nom}}$, are given by

$$\mathcal{H}_{\text{nom}} \begin{cases} \dot{\tilde{x}} = F_{\text{nom}}(\tilde{x}, \omega_c) & \tilde{x} \in C_{\text{nom}} \\ \tilde{x}^+ \in G_{\text{nom}}(\tilde{x}) & \tilde{x} \in D_{\text{nom}} \end{cases} \tag{3.34}$$

and

$$\mathcal{H}'_{\rm nom} \left\{ \begin{array}{ll} \dot{\tilde{x}} = F'_{\rm nom}(\tilde{x}) & \tilde{x} \in C_{\rm nom} \\ \tilde{x}^+ \in G_{\rm nom}(\tilde{x}) & \tilde{x} \in D_{\rm nom} \end{array} \right. \tag{3.35}$$

where $\tilde{x} = \mathrm{col}(\tilde{q},\, \tilde{\omega}_c,\, h)$,

$$\begin{aligned} F_{\rm nom}(\tilde{x}, \omega_c) &:= \begin{bmatrix} \frac{1}{2}\tilde{q} \otimes \begin{bmatrix} 0 \\ \tilde{\omega}_c \end{bmatrix} \\ J^{-1}\left(\Sigma(\tilde{\omega}_c, \bar{\omega}_c)\, \tilde{\omega}_c - k_p\, h\, \tilde{\epsilon} - k_d\, \tilde{\omega}_c \right) \\ 0 \end{bmatrix}, \\ F'_{\rm nom}(\tilde{x}) &:= \begin{bmatrix} \frac{1}{2}\tilde{q} \otimes \begin{bmatrix} 0 \\ \tilde{\omega}_c \end{bmatrix} \\ J^{-1}\left(S(J\tilde{\omega}_c)\tilde{\omega}_c - k_p\, h\, \tilde{\epsilon} - k_d\, \tilde{\omega}_c \right) \\ 0 \end{bmatrix}, \\ G_{\rm nom}(\tilde{x}) &:= [\tilde{q}^\top,\, \tilde{\omega}_c^\top,\, \overline{\rm sgn}(\tilde{\eta})]^\top, \\ C_{\rm nom} &:= \{\tilde{x} \in \mathcal{X}_{att} \;:\; h\,\tilde{\eta} \geq -\delta\}\,, \\ D_{\rm nom} &:= \{\tilde{x} \in \mathcal{X}_{att} \;:\; h\,\tilde{\eta} \leq -\delta\} \end{aligned}$$

having defined $\mathcal{X}_{att} := \mathbf{S}_3 \times \mathbb{R}^3 \times \{-1, 1\}$. Note that the attitude error system (3.34) is affected by the position error system through the input ω_c, whereas the attitude error system (3.35), as a consequence of the choice (3.31), is an autonomous system. Also note that $F_{\rm nom}(0, \omega_c) = 0$ for all $\omega_c \in \mathbb{R}^3$.

For the closed-loop autonomous system (3.35), inspired by the ideas in [83], we have the following result.

Proposition 3.2. *Consider the hybrid systems $\mathcal{H}'_{\rm nom}$ in (3.35). For all $k_p > 0$, $k_d > 0$, and $\delta \in (0,\, 1)$, the compact set*

$$\mathcal{A} = \{\tilde{x} \in \mathcal{X}_{att} \;:\; \tilde{q} = h\mathbf{1}\,,\; \tilde{\omega}_c = 0\}$$

is globally asymptotically stable.

Proof. Consider the candidate Lyapunov function $V \,:\, \mathcal{X}_{att} \to \mathbb{R}_{\geq 0}$ given by $V(\tilde{x}) = 2k_p(1 - h\tilde{\eta}) + \frac{1}{2}\tilde{\omega}_c^\top J \tilde{\omega}_c$. It satisfies $V(\mathcal{A}) = 0$, $V(\mathcal{X}_{att} \backslash \mathcal{A}) > 0$, and $\{\tilde{x} \in \mathcal{X}_{att} : V(\tilde{x}) \leq c\}$ is compact for every $c \geq 0$. During flows, since

$$\dot{\tilde{\eta}} \;=\; -\frac{1}{2}\,\tilde{\epsilon}^\top\,\tilde{\omega}_c, \quad \dot{\tilde{\epsilon}} \;=\; \frac{1}{2}\tilde{\eta}\,\tilde{\omega}_c + \frac{1}{2}\,S(\tilde{\epsilon})\,\tilde{\omega}_c$$

and $\tilde{\omega}_c^\top S(J\tilde{\omega}_c)\tilde{\omega}_c = 0$ we have

$$\langle \nabla V(\tilde{x}), F'_{\rm nom}(\tilde{x}) \rangle = -k_d\,\tilde{\omega}_c^\top\,\tilde{\omega}_c \leq 0$$

for all $\tilde{x} \in C_{\text{nom}}$. During jumps, using the fact $\tilde{x} \in D_{\text{nom}}$ implies $\text{sgn}\tilde{\eta} \neq \text{sgn}h$ and $h^+ = -h$,

$$\begin{aligned} V(\xi) - V(\tilde{x}) &= -2k_p(-h)\tilde{\eta} + 2k_p h\tilde{\eta} = 4k_p h\tilde{\eta} \\ &\leq -4k_p\delta < 0 \end{aligned}$$

for all $\xi \in G_{\text{nom}}(\tilde{x})$ and for all $\tilde{x} \in D_{\text{nom}}$. Stability of $\mathcal{A}$ follows by applying [103, Theorem 7.6]. To prove asymptotic stability we make use of an invariance principle. Note first of all that system (3.35) satisfies assumptions (A1)-(A3). In fact, C_{nom} and D_{nom} are closed sets. The flow map is continuous while, since $s \mapsto \overline{sgn}(s)$ is an outer-semicontinuous and bounded map, $\bar{G}_{\text{nom}}$ is outer semi-continuous and locally bounded. Now applying [103, Theorem 4.7] with $u_c : C_{\text{nom}} \to \mathbb{R}$ given by $u_c(\tilde{x}) = -k_d\tilde{\omega}_c^\top\tilde{\omega}_c$ and $u_d : D_{\text{nom}} \to \mathbb{R}$ given by $u_d(\tilde{x}) = -4k_p\delta < 0$, it follows that every bounded and complete solution to (3.35) converges to the largest weakly invariant set contained in

$$V^{-1}(r) \cap W \tag{3.36}$$

for some $r \geq 0$ where $W := \{\tilde{x} \in C_{\text{nom}} : \tilde{\omega}_c = 0\}$. By evaluating the dynamics (3.35) along solutions that remain in (3.36), we have that $\tilde{\epsilon} = 0$ and, since $h\tilde{\eta} \geq -\delta$ for all $\tilde{x} \in W$, $\tilde{q} = h\mathbf{1}$. Then, since the only invariant set is for $r = 0$, i.e., $\{\tilde{x} \in \mathcal{X}_{att} : \tilde{q} = h\mathbf{1}, \tilde{\omega}_c = 0, h \in \{-1, 1\}\}$, and (3.36) with $r = 0$ is contained in $\mathcal{A}$, we have that every bounded and complete solution converges to $\mathcal{A}$. Now it remains to prove that every maximal solution is complete and that solutions are bounded. From the fact that $\{\tilde{x} \in \mathcal{X}_{att} : V(\tilde{x}) \leq c\}$ is compact for every $c \geq 0$ and the fact that V is nonincreasing along solutions to (3.35), we have that solutions are bounded. Moreover, since the viability condition (VC) in [45, Proposition 6.10] holds for (3.35), $G_{\text{nom}}(D_{\text{nom}}) \subset (C_{\text{nom}} \cup D_{\text{nom}})$, and the fact that solutions are bounded, by applying [45, Proposition 6.10], it follows that all maximal solutions are complete. Then $\mathcal{A}$ is attractive. Global asymptotic stability follows from compactness of sublevel sets and from the fact that V is positive definite. □

Now consider system $\mathcal{H}_{\text{nom}}$. The latter is a hybrid system affected by the exogenous input ω_c. As a consequence of Lemma 3.2, we have that the trajectories ω_c can be thought of solutions to the exosystem

$$\dot{\omega}_{\text{e}} \in \mathcal{B}^3_{\bar{\Omega}_{c2}}, \quad \omega_{\text{e}} \in \mathcal{B}^3_{\bar{\Omega}_{c1}} \tag{3.37}$$

in which $\bar{\Omega}_{c1}$, $\bar{\Omega}_{c2}$ are the positive values given in Lemma 3.2. In particular, we have that there exists a solution ω_{e} to (3.37) such that $\omega_{\text{e}}(t) \equiv \omega_c(t)$ for all $t \geq 0$. Consider now the

autonomous hybrid system given by

$$\mathcal{H}_{\mathrm{e}} \begin{cases} \dot{x}_{\mathrm{e}} \in F_{\mathrm{e}}(x_{\mathrm{e}}) & x_{\mathrm{e}} \in C_{\mathrm{e}} \\ \dot{x}_{\mathrm{e}} \in G_{\mathrm{e}}(x_{\mathrm{e}}) & x_{\mathrm{e}} \in D_{\mathrm{e}} \end{cases} \tag{3.38}$$

where $x_{\mathrm{e}} := [\omega_{\mathrm{e}}^{\top},\, \tilde{x}^{\top}]^{\top}$,

$$F_{\mathrm{e}}(x_{\mathrm{e}}) := \begin{bmatrix} \mathcal{B}^3_{\bar{\Omega}_{c2}} \\ F_{\mathrm{nom}}(\tilde{x}, \omega_{\mathrm{e}}) \end{bmatrix},\; G_{\mathrm{e}}(x_{\mathrm{e}}) := \begin{bmatrix} \omega_{\mathrm{e}} \\ G_{\mathrm{nom}}(\tilde{x}) \end{bmatrix}$$

$C_{\mathrm{e}} := \{x_{\mathrm{e}} \;:\; \omega_{\mathrm{e}} \in \mathcal{B}^3_{\bar{\Omega}_{c1}},\, \tilde{x} \in C_{\mathrm{nom}}\}$ and $D_{\mathrm{e}} := \{x_{\mathrm{e}} \;:\; \omega_{\mathrm{e}} \in \mathcal{B}^3_{\bar{\Omega}_{c1}},\, \tilde{x} \in D_{\mathrm{nom}}\}$. For system (3.38), the following stability result holds.

Proposition 3.3. *Consider the hybrid system $\mathcal{H}_{\mathrm{e}}$ in (3.38). For all $k_p > 0$, $k_d > 0$, and $\delta \in (0,\, 1)$, the compact set*

$$\mathcal{A} = \left\{ x_{\mathrm{e}} \in \mathbb{R}^3 \times \mathcal{X}_{att} \;:\; \omega_{\mathrm{e}} \in \mathcal{B}^3_{\bar{\Omega}_{c1}},\; \tilde{q} = h\mathbf{1},\; \tilde{\omega}_c = 0 \right\}$$

is globally asymptotically stable.

Proof. Similarly to the proof of Proposition 3.2, we consider the candidate Lyapunov function $V \;:\; \mathcal{B}^3_{\bar{\Omega}_{c1}} \times \mathcal{X}_{att} \to \mathbb{R}_{\geq 0}$ given by $V(x_{\mathrm{e}}) = 2k_p(1 - h\tilde{\eta}) + \frac{1}{2}\tilde{\omega}_c J \tilde{\omega}_c$. It satisfies $V(\mathcal{A}) = 0$ and $V(\{\mathcal{B}^3_{\bar{\Omega}_{c1}} \times \mathcal{X}_{att}\} \setminus \mathcal{A}) > 0$. Moreover, since $\mathcal{B}^3_{\bar{\Omega}_{c1}}$ is compact, $\{x_{\mathrm{e}} \in \mathcal{B}^3_{\bar{\Omega}_{c1}} \times \mathcal{X}_{att} : V(x_{\mathrm{e}}) \leq c\}$ is compact for every $c \geq 0$.

During flows, since Σ is a skew-symmetric matrix (which implies $\tilde{\omega}_c^{\top}\Sigma(\tilde{\omega}_c, \omega_{\mathrm{e}})\tilde{\omega}_c = 0$), it turns out that the Lyapunov function is nonincreasing, i.e.,

$$\langle \nabla V(x_{\mathrm{e}}), F_{\mathrm{e}}(x_{\mathrm{e}}) \rangle = -k_d\, \tilde{\omega}_c^{\top}\, \tilde{\omega}_c \leq 0$$

for all $x_{\mathrm{e}} \in C_{\mathrm{e}}$. During jumps, following the same arguments as in the proof of Proposition 3.2,

$$\begin{aligned} V(\xi) - V(x_{\mathrm{e}}) &= -2k_p h^{+}\tilde{\eta} + 2k_p h\tilde{\eta} = 4k_p h\tilde{\eta} \\ &\leq -4k_p\delta < 0 \end{aligned}$$

for all $\xi \in G_{\mathrm{e}}(x_{\mathrm{e}})$ and for all $x_{\mathrm{e}} \in D_{\mathrm{e}}$. Stability of $\mathcal{A}$ follows from [103, Theorem 7.6]. Asymptotic stability can be proved by applying an invariance principle as for the proof of Proposition 3.2. Note first of all that system (3.38) satisfies assumptions (A1)-(A3). In fact C_{e} and D_{e} are closed sets. The flow equation is continuous while $\bar{G}_{\mathrm{e}}$ is outer semi-continuous and locally bounded. Now applying [103, Theorem 4.7] with $u_c \;:\; C_{\mathrm{e}} \to \mathbb{R}$ given by $u_c(x_{\mathrm{e}}) = -k_d\, \tilde{\omega}_c^{\top}\, \tilde{\omega}_c$ and $u_d \;:\; D_{\mathrm{e}} \to \mathbb{R}$ given

by $u_d(x_e) = -4k_p\delta < 0$, it follows that every bounded and complete solution to (3.38) converges to the largest weakly invariant set contained in $V^{-1}(r) \cap W$ with $W := \{x_e \in C_e : \tilde{\omega}_c = 0\}$ and for some $r \geq 0$. Following the same arguments as in the proof of Proposition 3.2, this implies that every bounded and complete solution converges to $\mathcal{A}$. Global asymptotic stability of $\mathcal{A}$ then follows as in the proof of Proposition 3.2. □

Stability Properties of the Combined Position and Attitude Closed-Loops in the Nominal Case

By combining the ISS properties of the position error system in Proposition 3.1 with the global asymptotic stability of the attitude error subsystem given in Propositions 3.2 and 3.3, the desired global tracking objective for the whole system in the *nominal case* can be accomplished.

Proposition 3.4. *Consider the closed-loop system given by the position error dynamics (3.20), with $d_f \equiv 0$, controlled by (3.22), with λ_i and k_i, $i = 1, 2$, chosen as in (3.23), (3.24) and $\lambda_i^\star$, $k_i^\star$ chosen as in Proposition 3.1, and the error attitude dynamics (3.27) controlled by (3.28) or, respectively, (3.29), with $k_p > 0$, $k_d > 0$, $\delta \in (0, 1)$ arbitrarily chosen. Then, the compact set*

$$\mathcal{A}^\star = \{(\tilde{\mathbf{p}}, \tilde{x}) \in \mathbb{R}^6 \times \mathcal{X}_{att} \ : \ \tilde{\mathbf{p}} = 0 \, , \ \tilde{q} = h\mathbf{1} \, , \ \tilde{\omega}_c = 0\}$$

is globally asymptotically stable.

Proof. First, consider (3.27) controlled by (3.28). From Lemma 3.2, in the nominal case, the trajectories ω_c can be obtained as solutions to (3.37) and then Proposition 3.3 holds. Since the hybrid system (3.38) satisfies (A1)-(A3), from [45, Theorem 6.8] the hybrid system (3.38) is nominally well-posed and then [45, Lemma 7.8] implies that the compact set $\mathcal{A}$ in Proposition 3.3 is uniformly attractive from every compact set of the state space. Since $\mathcal{A}$ is stable, uniformly attractive from compact subsets of the state space, and, due to global asymptotic stability and by applying [45, Lemma 6.16], the reachable set from every given compact set is bounded, then [45, Lemma 7.11] implies that $\mathcal{A}$ is also $\mathcal{KL}$ asymptotically stable. As a consequence, for each maximal solution to (3.34), given $\Delta > 0$ there exists $T_\Delta > 0$ such that $\Theta(\mathcal{R}(\tilde{q}(t, j))) \leq \Delta$ for all $t + j \geq T_\Delta$, $(t, j) \in \operatorname{dom} \tilde{x}$. Hence, in finite time, $\Theta(\mathcal{R}(\tilde{q}))$ is arbitrarily small and, asymptotically, $\tilde{q} = h\mathbf{1}$ and $\tilde{\omega}_c = 0$. Now consider (3.27) controlled by (3.29). From Proposition 3.2, the same result holds following the same arguments employed above. By focusing now on the position error dynamics (3.20), it follows that, by choosing Δ sufficiently small, restrictions R_Δ on

the exogenous input $\Gamma(R_c, R)$ given Proposition 3.1 are fulfilled in finite time and $\Gamma(R_c, R)$ approaches zero asymptotically. Then, since system (3.20) is a continuous-time system and it has no finite escape time, the result follows applying cascade control arguments as in [54, Corollary B.3.3]. In particular, for all $0 \leq t \leq T_\Delta$ solutions to (3.20) are defined and, since for all $t \geq T_\Delta$ the restrictions on the inputs are satisfied, $|\tilde{\mathbf{p}}|_a \leq \gamma_p |\Gamma|_a$ for some class-$\mathcal{K}$ function γ_p, which, since $|\Gamma_a| = 0$, implies that $\tilde{\mathbf{p}}$ converges to zero. □

Remark 3.3. Note that the fact that the compact set $\mathcal{A}^\star$ in Proposition 3.4 is globally asymptotically stable implies that the position and the attitude of system (3.1) converge to the desired references (3.5), globally with respect to the initial conditions. This result, which overcomes the topological obstruction of globally stabilizing systems on manifolds via continuous feedback laws [12], takes advantage from the methodology proposed in [83] in which a globally stabilizing attitude controller is proposed. △

Remark 3.4. The result in Proposition 3.4 for the overall closed-loop system does not require additional conditions on the tuning of the position and attitude controllers other than to the ones given in Propositions 3.1, 3.2 and 3.3. This useful property is obtained by designing the attitude controller in (3.28) (and (3.29)) so as to decouple (partially decouple) the inner attitude loop from the outer position loop. As a main limitation, the resulting attitude controller relies on the perfect knowledge of the system dynamics (in particular, see the feedforward terms (3.30) and (3.31)) and then it may not be robust to large uncertainties or exogenous disturbances. △

Attitude Stabilization: the Robust Case

Let us focus now on the general case in which the exogenous disturbances d_f and d_τ affect system (3.1) and just the nominal value J_0 of the inertia matrix J is available for feedback.

In this case it is possible to define the new attitude error coordinates

$$\tilde{q} := q_c^{-1} \otimes q, \quad \tilde{\omega}^* := \omega - \bar{\omega}^* \tag{3.39}$$

with $\bar{\omega}^* = R(\tilde{q})^\top \omega^*$. By bearing in mind (3.1) and (3.4), the choice (3.39) leads to an attitude error dynamics of the form

$$\begin{aligned} \dot{\tilde{q}} &= \frac{1}{2}\tilde{q} \otimes \begin{bmatrix} 0 \\ \tilde{\omega}^* \end{bmatrix} - \frac{1}{2}\tilde{q} \otimes \begin{bmatrix} 0 \\ R(\tilde{q})^\top y_\zeta \end{bmatrix} \\ J\dot{\tilde{\omega}}^* &= \Sigma(\tilde{\omega}^*, \bar{\omega}^*)\,\tilde{\omega}^* + S(J\bar{\omega}^*)\bar{\omega}^* - JS(\mathcal{R}(\tilde{q})^\top y_\zeta)\,\bar{\omega}^* - J\mathcal{R}(\tilde{q})^\top \dot{\omega}^* + u_\tau + d_\tau \end{aligned} \tag{3.40}$$

in which $y_\zeta := \omega_c - \omega^*$ and where $\Sigma(\tilde{\omega}^*, \bar{\omega}^*)$ is the skew-symmetric matrix defined just

after (3.27) with $(\tilde{\omega}_c,\, \bar{\omega}_c)$ replaced by $(\tilde{\omega}^*,\, \bar{\omega}^*)$. As above, $\tilde{\eta}$ and $\tilde{\epsilon}$ denote respectively the scalar and vector part of the error quaternion $\tilde{q}$. Regarding the term y_ζ in (3.40) note that, by using the properties in Lemma 3.1, the following holds

$$y_\zeta := \Omega_1(\tilde{\mathbf{p}}, t) - \omega^*(t) + \Omega_2(\tilde{\mathbf{p}}, t)\, \Gamma(R, R_c) + \Omega_2(\tilde{\mathbf{p}}, t)\, d_f(t)$$

where $\Omega_1(\tilde{\mathbf{p}}, t) - \omega^*(t)$ and $\Omega_2(\tilde{\mathbf{p}}, t)\, \Gamma(R, R_c)$ are bounded smooth functions vanishing respectively with $\tilde{\mathbf{p}}$ and $\tilde{\epsilon}$. Namely, $y_\zeta = y_\zeta(\tilde{\mathbf{p}}, \tilde{\epsilon}, d_f, t)$ with $y_\zeta(0, 0, 0, t) = 0$. Furthermore, with the expression of $u^*_{\tau 0}$ in (3.10) at hand, note that the term $S(J\bar{\omega}^*)\bar{\omega}^* - J\mathcal{R}(\tilde{q})^\top\, \dot{\omega}^*$ in the last equation of (3.40) can be expressed as

$$S(J\bar{\omega}^*)\bar{\omega}^* - J\mathcal{R}(\tilde{q})^\top\, \dot{\omega}^* = -u^*_{\tau 0} + \Delta_{1\omega}(\tilde{q}, t)\, \tilde{\epsilon} + \Delta_{2\omega}(\tilde{q}, t)\, (J - J_0)$$

where $\Delta_{1\omega}$ and $\Delta_{2\omega}$ are smooth functions of appropriate dimension such that

$$\begin{aligned} \Delta_{1\omega}(\tilde{q}, t)\tilde{\epsilon} &= J_0(I_3 - \mathcal{R}(\tilde{q})^\top)\dot{\omega}^* + S(J_0\bar{\omega}^*)\bar{\omega}^* + \\ &- S(J_0\omega^\star)\omega^*, \\ \Delta_{2\omega}(\tilde{q}, t)(J - J_0) &= (S(J\bar{\omega}^*) - S(J_0\bar{\omega}^*))\bar{\omega}^* + \\ &+ (J_0 - J)\mathcal{R}(\tilde{q})^\top\dot{\omega}^*. \end{aligned}$$

Note that $\Delta_{i\omega}$, $i \in \{1,\, 2\}$, are uniformly bounded, namely there exists a constant $\bar{\Delta}_\omega$ such that for all $\tilde{q} \in \mathbf{S}_3$ and $t \geq 0$

$$|\Delta_{i\omega}(\tilde{q}, t)| \leq \bar{\Delta}_\omega\,, \quad i \in \{1,\, 2\}\,.$$

Furthermore, note that

$$\omega^* \equiv 0\,,\ \dot{\omega}^* \equiv 0 \ \Rightarrow\ \Delta_{2\omega}(\tilde{q}, t) \equiv 0\,. \tag{3.41}$$

System (3.40) is thus a system driven by control input u_τ and affected by the exogenous disturbances $(d_f, d_\tau, \tilde{\mathbf{p}}, (J_0 - J)[\omega^*,\, \dot{\omega}^*])$. For this system, the attitude controller is selected as

$$u_\tau = u^*_{\tau 0} + u_{\tau,FB}(\tilde{q}, \tilde{\omega}^*, h) \tag{3.42}$$

where the feedback law $u_{\tau,FB}$ is chosen as

$$u_{\tau,FB}(\tilde{q}, \tilde{\omega}^*, h) \;=\; -k_p\, h\, \tilde{\epsilon} - k_p\, k_d\, \tilde{\omega}^* \tag{3.43}$$

in which k_p, k_d are positive gains and where $h \in \{-1,\, 1\}$ is governed by the following

hybrid dynamics

$$\begin{cases} \dot{h} = 0 & h\,\tilde{\eta} \geq -\delta \ \text{ or } \ \tilde{\epsilon}^\top J^U \tilde{\epsilon} + \tilde{\omega}^{*\top} J^U \tilde{\omega}^* \geq 2k_d\,\delta \\ h^+ \in \overline{\text{sgn}}(\tilde{\eta}) & h\,\tilde{\eta} \leq -\delta, \ \tilde{\epsilon}^\top J^U \tilde{\epsilon} + \tilde{\omega}^{*\top} J^U \tilde{\omega}^* \leq 2k_d\,\delta \end{cases} \tag{3.44}$$

with $\delta \in (0,1)$ and with the function $\overline{\text{sgn}}(\cdot)$ defined as Section 3.1.2. Note that, compared to (3.33), jumps occur only for sufficiently small values of the angular position and velocity errors. The whole attitude error system is thus a hybrid system of the form

$$\mathcal{H}_{\text{rob}} \begin{cases} \dot{\tilde{x}} &= F_{\text{rob}}(\tilde{x}, \mathbf{d}) & \tilde{x} \in C_{\text{rob}} \\ \tilde{x}^+ &\in G_{\text{rob}}(\tilde{x}) & \tilde{x} \in D_{\text{rob}} \end{cases} \tag{3.45}$$

with $\tilde{x} = \text{col}(\tilde{q}, \tilde{\omega}^*, h)$, $\mathbf{d} = \text{col}(d_f, d_\tau, \tilde{\mathbf{p}}, (J_0 - J)\omega^\star, (J_0 - J)\dot{\omega}^\star)$, $F_{\text{rob}}(\cdot)$ is the vector given by the right-hand side of (3.40), $G_{\text{rob}}(\cdot)$ is defined as G_{nom} in the previous section, and

$$\begin{aligned} C_{\text{rob}} &= \{\tilde{x} \in \mathcal{X}_{att} \ : \ h\,\tilde{\eta} \geq -\delta\,\} \cup \\ &\quad \{\tilde{x} \in \mathcal{X}_{att} \ : \ \tilde{\epsilon}^\top J^U \tilde{\epsilon} + \tilde{\omega}^{*\top} J^U \tilde{\omega}^* \geq 2k_d\,\delta\} \\ D_{\text{rob}} &= \{\tilde{x} \in \mathcal{X}_{att} \ : \ h\,\tilde{\eta} \leq -\delta\,, \\ &\quad \tilde{\epsilon}^\top J^U \tilde{\epsilon} + \tilde{\omega}^{*\top} J^U \tilde{\omega}^* \leq 2k_d\,\delta\} \end{aligned}$$

It turns out that, if the inputs $(d_f, d_\tau, (J_0 - J)\omega^*, (J_0 - J)\dot{\omega}^*)$ are bounded, the design parameters can be tuned so as, after a finite amount of jumps, system (3.40) flows only and the resulting continuous-time system is characterized by an arbitrarily small asymptotic gain with respect to the input $\mathbf{d}$. This fact is formalized in the next Theorem.

Theorem 3.1. *Consider the hybrid system $\mathcal{H}_{\text{rob}}$ in (3.45). Let $\delta \in (0,1)$ and*

$$|d_f|_\infty \leq R_f, \quad |d_\tau|_\infty \leq R_\tau, \quad |(J - J_0)[\omega^*, \dot{\omega}^*]|_\infty \leq R_W \tag{3.46}$$

for some constants $R_f > 0$, $R_\tau > 0$ and $R_W > 0$. For any $\gamma_{\text{att}} > 0$, $c > 0$, there exists $k_d^\star > 0$ and for all $k_d \leq k_d^\star$ there exists $k_p^\star > 0$ such that for all $k_p \geq k_p^\star$ and for all $\tilde{x}(0,0) \in C_{\text{rob}} \cup D_{\text{rob}}$ each maximal solution $\tilde{x}$ is complete and such that

- *there exists $T^\star > 0$ such that $\tilde{x}(t,j) \in C_{\text{rob}}$ and $|\tilde{\epsilon}(t,j)| \leq c$ for all $(t,j) \in \text{dom}\,\tilde{x}$ such that $t + j \geq T^\star$;*
- $$\limsup_{t \to \infty} |\tilde{x}(t, j^\star)| \ \leq \ \gamma_{\text{att}}\ \max\{|\tilde{\mathbf{p}}|_a\,,\ |d_f|_a\,,\ |d_\tau|_a\,, |J - J_0||[\omega^*, \dot{\omega}^*]|_a\}$$

  where $j^\star = \sup\{j \ : \ (t,j) \in \text{dom}\,\tilde{x}\}$.

Proof. By the expression of ω_c and the properties of $\Omega_1(\cdot)$ and $\Omega_2(\cdot)$ in Lemma 3.1, by the fact that $\Gamma(R_c, R)$ is a smooth function vanishing at $\tilde{\epsilon} = 0$, and by the definition of y_ζ, it turns out that there exist smooth matrices $\Delta_{i\eta}(\tilde{\mathbf{p}}, \tilde{q}, t)$ and $\Delta_{i\epsilon}(\tilde{\mathbf{p}}, \tilde{q}, t)$, $i = 1, 2, 3$, of appropriate dimension such that

$$\begin{aligned}\frac{1}{2}\tilde{q} \otimes \begin{bmatrix} 0 \\ R(\tilde{q})^\top y_\zeta \end{bmatrix} &= \begin{bmatrix} \tilde{\epsilon}^\top \Delta_{1\eta}(\tilde{\mathbf{p}}, \tilde{q}, t) \\ \Delta_{1\epsilon}(\tilde{\mathbf{p}}, \tilde{q}, t) \end{bmatrix} \tilde{\epsilon} \\ &+ \begin{bmatrix} \tilde{\epsilon}^\top \Delta_{2\eta}(\tilde{\mathbf{p}}, \tilde{q}, t) \\ \Delta_{2\epsilon}(\tilde{\mathbf{p}}, \tilde{q}, t) \end{bmatrix} + \begin{bmatrix} \tilde{\epsilon}^\top \Delta_{3\eta}(\tilde{\mathbf{p}}, \tilde{q}, t) \\ \Delta_{3\epsilon}(\tilde{\mathbf{p}}, \tilde{q}, t) \end{bmatrix} d_f\end{aligned}$$

and

$$|\Delta_{i\eta}(\tilde{\mathbf{p}}, \tilde{q}, t)| \leq \bar{\Delta}_q\,, \quad |\Delta_{i\epsilon}(\tilde{\mathbf{p}}, \tilde{q}, t)| \leq \bar{\Delta}_q\,, \quad i = 1, 2, 3$$

for all $\tilde{\mathbf{p}} \in \mathbb{R}^6$, $\tilde{q} \in \mathbf{S}_3$, $t \geq 0$, where $\bar{\Delta}_q$ is a positive constant. Furthermore,

$$\Delta_{2\eta}(0, \tilde{q}, t) = 0\,, \quad \Delta_{2\epsilon}(0, \tilde{q}, t) = 0 \quad \forall\, \tilde{q} \in \mathbf{S}_3\,,\ t \geq 0\,.$$

Moreover, there exist smooth matrices $\Delta_{i\omega}(\tilde{\mathbf{p}}, \tilde{q}, t)$, $i = 3, 4, 5$, of appropriate dimension such that

$$JS(\mathcal{R}(\tilde{q})^\top y_\zeta)\bar{\omega}^* = \Delta_{3\omega}(\tilde{\mathbf{p}}, \tilde{q}, t)\tilde{\epsilon} + \Delta_{4\omega}(\tilde{\mathbf{p}}, \tilde{q}, t) + \Delta_{5\omega}(\tilde{\mathbf{p}}, \tilde{q}, t)d_f$$

and $|\Delta_{i\omega}(\tilde{\mathbf{p}}, \tilde{q}, t)| \leq \bar{\Delta}_\omega$, $i = 3, 4, 5$, for all $\tilde{\mathbf{p}} \in \mathbb{R}^6$, $\tilde{q} \in \mathbf{S}_3$, $t \geq 0$, ω^* bounded, where $\bar{\Delta}_\omega$ is a positive constant. Furthermore, $\Delta_{4\omega}(0, \tilde{q}, t) = 0 \quad \forall\ \tilde{q} \in \mathbf{S}_3\,,\ t \geq 0$. By putting all the previous expressions in (3.40), along flows, the error attitude dynamics can be more explicitly rewritten as

$$\begin{aligned}\dot{\tilde{\eta}} &= -\frac{1}{2}\tilde{\epsilon}^\top \tilde{\omega}^* + \tilde{\epsilon}^\top \Delta_{1\eta}(\cdot)\tilde{\epsilon} + \tilde{\epsilon}^\top d_\eta \\ \dot{\tilde{\epsilon}} &= \frac{1}{2}(\tilde{\eta}\tilde{\omega}^* + S(\tilde{\epsilon})\tilde{\omega}^*) + \Delta_{1\epsilon}(\cdot)\tilde{\epsilon} + d_\epsilon \\ J\dot{\tilde{\omega}}^* &= \Sigma(\cdot)\,\tilde{\omega}^* - u^\star_{\tau 0} + (\Delta_{1\omega}(\cdot) + \Delta_{3\omega}(\cdot))\tilde{\epsilon} + \Delta_{4\omega}(\cdot) \\ &\quad + \Delta_{5\omega}(\cdot)d_f + \Delta_{2\omega}(\tilde{q}, \omega^*, \dot{\omega}^*)(J - J_0) + u_\tau + d_\tau\end{aligned} \tag{3.47}$$

where

$$\begin{aligned}d_\eta &= \Delta_{2\eta}(\tilde{\mathbf{p}}, \tilde{q}, t) + \Delta_{3\eta}(\tilde{\mathbf{p}}, \tilde{q}, t)d_f \\ d_\epsilon &= \Delta_{2\epsilon}(\tilde{\mathbf{p}}, \tilde{q}, t) + \Delta_{3\epsilon}(\tilde{\mathbf{p}}, \tilde{q}, t)d_f\,.\end{aligned}$$

Let us define now the backstepping-like change of variable

$$\tilde{\omega}^* \;\mapsto\; \tilde{z} := \tilde{\omega}^* + \frac{1}{k_d} h\, \tilde{\epsilon}$$

that transforms system (3.47) into

$$\begin{aligned}
\dot{\tilde{\eta}} &= -\frac{1}{2}\tilde{\epsilon}^\top \left(\tilde{z} - \frac{1}{k_d} h\, \tilde{\epsilon}\right) + \tilde{\epsilon}^\top \Delta_{1\eta}(\cdot)\tilde{\epsilon} + \tilde{\epsilon}^\top d_\eta \\
\dot{\tilde{\epsilon}} &= \frac{1}{2}(\tilde{\eta} I_3 + S(\tilde{\epsilon})) \left(\tilde{z} - \frac{1}{k_d} h\, \tilde{\epsilon}\right) + \Delta_{1\epsilon}(\cdot)\tilde{\epsilon} + d_\epsilon \\
J\dot{\tilde{z}} &= \left(-k_p k_d I_3 + \Sigma'_{k_d}(\tilde{z}, \tilde{q}, h)\right) \tilde{z} + \Sigma''_{k_d}(\tilde{z}, \tilde{q}, h)\tilde{\epsilon} + d_z
\end{aligned} \tag{3.48}$$

where

$$\begin{aligned}
\Sigma'_{k_d}(\cdot) &:= \Sigma(\cdot) + \frac{1}{2k_d} h\, (\tilde{\eta} I_3 + S(\tilde{\epsilon})) \\
\Sigma''_{k_d}(\cdot)\tilde{\epsilon} &:= -\Sigma(\cdot)\frac{1}{k_d}\tilde{\epsilon} + (\Delta_{1\omega}(\cdot) + \Delta_{3\omega}(\cdot))\tilde{\epsilon} + \\
&+ \frac{1}{2k_d} h S(\tilde{\epsilon}) \left(\tilde{z} - \frac{1}{k_d} h\tilde{\epsilon}\right) + \frac{1}{k_d} h \Delta_{1\epsilon}\tilde{\epsilon} \\
d_z &:= \left(\frac{J}{k_d} h \Delta_{2\epsilon}(\cdot) + \Delta_{4\omega}(\cdot)\right) + \\
&+ \left(\frac{J}{k_d} h \Delta_{3\epsilon}(\cdot) + \Delta_{5\omega}(\cdot)\right) d_f + \\
&+ \Delta_{2\omega}(\tilde{q}, \omega^*, \dot{\omega}^*)(J - J_0) + d_\tau\,.
\end{aligned}$$

In order to study the asymptotic properties of system (3.45), consider the following candidate Lyapunov function

$$V(\tilde{x}) = 2(1 - h\tilde{\eta}) + \frac{1}{2}\tilde{z}^\top J \tilde{z}\,.$$

During flows, the time derivative of V reads as

$$\begin{aligned}
&\langle \nabla V(\tilde{x}), F_{\text{rob}}(\tilde{x}, \mathbf{d})\rangle = \\
&= -\frac{1}{k_d}\, \tilde{\epsilon}^\top\, \tilde{\epsilon} - \tilde{z}^\top \left(k_p\, k_d I_3 - \Sigma'_{k_d}(\tilde{z}, \tilde{q}, h)\right) \tilde{z} + \\
&+ \tilde{\epsilon}^\top \Delta_{1\eta}(\tilde{\mathbf{p}}, \tilde{q}, t)\tilde{\epsilon} + h\tilde{\epsilon}^\top \tilde{z} + \tilde{z}^\top \Sigma''_{k_d}(\tilde{z}, \tilde{q}, h)\tilde{\epsilon} + \\
&+ \tilde{z}^\top d_z + d_\epsilon\,.
\end{aligned}$$

Note that $|(d_\eta, d_\epsilon, d_z)|_\infty \leq \bar{d}$ for some positive constant $\bar{d}$ dependent on R_f, R_τ, R_W. As a consequence, standard high-gain arguments for continuous time systems (see in particular [53, Chapter 10]) can be used to claim that for any $\ell_1 > 0$ there exist

$k_d^\star > 0$ and, for all $k_d \leq k_d^\star$, $k_p^\star > 0$ such that for all $k_p \geq k_p^\star$ the following holds

$$\left.\begin{array}{rcl} |(\tilde{\epsilon}, \tilde{z})| & \geq & \ell_1 |(d_\eta, d_\epsilon, d_z)| \\ \tilde{x} & \in & C_{\rm rob} \end{array}\right\} \Rightarrow \langle \nabla V(\tilde{x}), F_{\rm rob}(\tilde{x}, \mathbf{d}) \rangle \leq -\gamma V(\tilde{x}) \tag{3.49}$$

for some positive constant γ.

Let us now analyze the behavior of V during jumps. By definition of $D_{\rm rob}$, during jumps $\mathrm{sgn} h \neq \mathrm{sgn} \tilde{\eta}$ which implies that $h^+ = -h$. Hence, $\tilde{z}^+ = \tilde{\omega}^{*+} + (1/k_d) h^+ \tilde{\epsilon}^+ = \tilde{\omega}^* - (1/k_d) h \tilde{\epsilon}$ by which, recalling also (3.3), the following holds

$$\begin{aligned} & V(\xi) - V(\hat{x}) = (1 - h^+ \tilde{\eta}) + \frac{1}{2} \tilde{z}^{+\top} J \tilde{z}^+ - 2(1 - h\tilde{\eta}) - \frac{1}{2} \tilde{z}^\top J \tilde{z} \\ & = 4h\tilde{\eta} + \frac{1}{2} \left(\tilde{\omega}^* - \frac{1}{k_d} h\tilde{\epsilon} \right)^\top J \left(\tilde{\omega}^* - \frac{1}{k_d} h\tilde{\epsilon} \right) - \frac{1}{2} \left(\tilde{\omega}^* + \frac{1}{k_d} h\tilde{\epsilon} \right)^\top J \left(\tilde{\omega}^* + \frac{1}{k_d} h\tilde{\epsilon} \right) \\ & = 4h\tilde{\eta} - \frac{2}{k_d} h \tilde{\omega}^{*\top} J \tilde{\epsilon} \\ & \leq -4\delta + \frac{2}{k_d} |\tilde{\omega}^{*\top} J \tilde{\epsilon}| \leq -4\delta + \frac{1}{k_d} \left(\tilde{\omega}^{*\top} J^U \tilde{\omega}^* + \tilde{\epsilon}^\top J^U \tilde{\epsilon} \right) \\ & \leq -2\delta \end{aligned}$$

for all $\xi \in G_{\rm rob}(\tilde{x})$. Now let $c_1 \leq c$ be a positive constant such that $|\tilde{\epsilon}| \leq c_1 \Rightarrow |\tilde{\eta}| > \delta$ and let $\ell_2 > 0$ be such that

$$V(\tilde{x}) \leq \ell_2 \quad \Rightarrow \quad |(\tilde{\epsilon}, \tilde{z})| \leq c_1 \,.$$

Furthermore, by bearing in mind (3.49), fix $\ell_1 > 0$ (and $k_d^\star$ and $k_p^\star$ accordingly) and $0 < \ell_3 < \ell_2$ so that

$$|(\tilde{\epsilon}, \tilde{z})| \leq \ell_1 \bar{d} \quad \Rightarrow \quad V(\tilde{x}) \leq \ell_3 \,.$$

In summary, for all d_f, d_τ, J, $t \mapsto p^*(t)$, $t \mapsto \omega_z^*(t)$ such that $|d_f|_\infty \leq R_f$, $|d_\tau|_\infty \leq R_\tau$, $|J - J_0||[\omega^*, \dot{\omega}^*]|_\infty \leq R_W$, and for all $\tilde{\mathbf{p}} \in \mathbb{R}^6$, we have that (3.49) holds and V is strictly decreasing during jumps. Accordingly, every complete solutions to $\mathcal{H}_{\rm rob}$ converge in finite time to the set $\mathcal{P}_{\ell 2} := \{\tilde{x} \,:\, V(\tilde{x}) \leq \ell_2\}$.

We show that every maximal solution to (3.45) is complete. From the fact that $\{\tilde{x} \,:\, V(\tilde{x}) \leq c'\}$ is compact for every $c' > 0$, and, for all $\tilde{x}$ such that $V(\tilde{x}) \geq \ell_2$, V is non increasing along solutions to (3.45), we have that solutions are bounded. Moreover the viability condition (VC) in [45, Proposition 6.10] holds for the hybrid system (3.45) with zero input, i.e., $\mathbf{d} \equiv 0$. Then, since $G_{\rm rob}(D_{\rm rob}) \subset (C_{\rm rob} \cup D_{\rm rob})$ and, as shown above, from the fact that solutions are bounded, by applying [45, Proposition 6.10] it follows that all maximal solutions to (3.45) with zero input are complete. Since $G_{\rm rob}(D_{\rm rob}) \cap D_{\rm rob} = \emptyset$ and the inputs d_f, d_τ, d_z in (3.48) are bounded,

there is a finite (nonzero) amount of flows among jumps. This shows that every maximal solution to (3.45) in the presence of the input $\mathbf{d}$ satisfying (3.46) are complete. As a consequence, from (3.49), there exists a $T^\star > 0$ such that $V(\tilde{x}(t,j)) \leq \ell_2$ for all (t,j) in the hybrid time domain of the solution such that $t + j \geq T^\star$. Hence, by definition of c_1, $|\tilde{\epsilon}(t,j)| \leq |(\tilde{\epsilon}(t,j), \tilde{z}(t,j))| \leq c_1 < c$ for all (t,j) such that $t + j \geq T^\star$. This in particular implies that $\tilde{x}(t,j) \in C_{\text{rob}}$ for all (t,j) such that $t + j \geq T^\star$, from which item 1 holds true. To prove item 2, note that for all $(t,j) \in \operatorname{dom} \tilde{x}$ such that $t + j \geq T^\star$ system (3.45) evolves as a continuous time system, namely no jumps occur. Accordingly, for all maximal solutions to (3.45), there exists $j^\star > 0$ such that $(t, j^\star)$ belong to $\operatorname{dom} \tilde{x}$ for all $t \geq T^\star$. The second item then follows by considering the continuous time system (3.48), the Lyapunov function (3.49), and by applying standard ISS arguments for continuous time systems [53, Chapter 10]. In particular the asymptotic bound involving $(J - J_0)[\omega^*, \dot{\omega}^*]$ follows from the property of $\Delta_{2\omega}(\cdot)$ in (3.41). □

Remark 3.5. From the model inversion computed in Section 3.1.1 it follows that ω^*, $\dot{\omega}^*$ are functions of the position references $p^{*(3)}$, $p^{*(4)}$ and the attitude references ω_z^*, $\dot{\omega}_z^*$. Hence, the value of R_W in Theorem 3.1 depends on the desired position and attitude references and on the uncertainties on the inertia matrix J. △

Remark 3.6. Theorem 3.1 shows how the attitude controller (3.42)-(3.44) is actually able to robustly globally stabilize the attitude error (3.39) in the presence of bounded exogenous disturbances and parametric uncertainties. This robustness property represents the main contribution of the proposed attitude control design with respect to the one proposed in [83]. Robust attitude controllers have appeared also in the space control literature (see [25] among others). The approach proposed in this work, however, is also able to overcome the topological obstruction using hybrid feedback control techniques so as to obtain a global property. △

Stability Properties of the Combined Position and Attitude Closed Loops in the Robust Case

By combining the claim of Theorem 3.1 and the one of Proposition 3.1, small gain arguments for continuous time systems can be used to conclude the asymptotic properties of the whole closed-loop system in the *robust case*. In fact, the following proposition holds true. [1]

[1] Hereafter, for the hybrid system $\mathcal{H}_{\text{rob}}$, we compactly denote $|\phi|_a = \limsup_{(t,j) \in \operatorname{dom} \phi,\, t+j \to \infty} |\phi(t,j)|$.

Proposition 3.5. *Consider the whole closed-loop system given by the position error dynamics (3.20) controlled by (3.22), with λ_i and k_i, $i = 1, 2$, chosen as in (3.23), (3.24) and $\lambda_i^\star$, $k_i^\star$ chosen as in Proposition 3.1, and the error attitude dynamics (3.40) controlled by (3.42)-(3.44) with $\delta \in (0,\, 1)$. Let*

$$|d_f|_\infty \leq R_{d_f},\ |d_\tau|_\infty \leq R_{d_\tau},\ |(J - J_0)[\omega^*,\, \dot{\omega}^*]|_\infty \leq R_W$$

with R_{d_f} fixed by Proposition 3.1 and R_{d_τ}, R_W arbitrarily large positive numbers. Then there exists $k_d^\star > 0$ and, for all $k_d < k_d^\star$, $k_p^\star(k_d) > 0$ such that for all $k_p \geq k_p^\star(k_d)$ the following asymptotic bound holds true

$$|(\tilde{\mathbf{p}}, \tilde{\epsilon}, \tilde{\omega})|_a \leq \gamma \max\{|d_f|_a\,,\ \frac{1}{k_p}|d_\tau|_a, \frac{1}{k_p}|(J - J_0)[\omega^*,\, \dot{\omega}^*]|_a\}$$

for some $\gamma > 0$.

Proof. As a first step, note that there exists a positive constant $\bar{\Gamma}$ such that the $\Gamma(R_c, R)$ can be bounded as $|\Gamma(R_c, R)| \leq \bar{\Gamma}|\tilde{\epsilon}|$. With an eye to the statements of Proposition 3.1 and Theorem 3.1 , let $c > 0$ and $\gamma_{\text{att}} > 0$ be fixed so that $\bar{\Gamma}\, c \leq R_\Gamma$ and

$$\gamma_{\text{att}}\, \gamma_{\text{pos}}\, \bar{\Gamma} < 1 \tag{3.50}$$

and let $k_d^\star$ and $k_p^\star$ fixed accordingly so that the properties of Theorem 3.1 are fulfilled with the restrictions on the inputs d_f, d_τ and $(J_0 - J)[\omega^*,\, \dot{\omega}^*]$ given by R_{d_f}, R_{d_τ} and R_W. Then, by Theorem 3.1, the restriction R_Γ on the input Γ of system (3.20) controlled by (3.22) is fulfilled in finite time. Moreover, there exists a time $T^\star$ such that for all $t + j \geq T^\star$ the closed-loop system flows only. At this point, asymptotic stability properties of the closed-loop system can be inferred by considering stability arguments for continuous-time systems. In particular, note that the small gain condition between the position and attitude subsystems is fulfilled by virtue of (3.50). The claimed asymptotic bound then follows from gain composition using the fact that γ_{att} in Theorem 3.1 can be arbitrarily decreased by increasing k_p. □

By considering a possible employment in real-world applications, note that the robust attitude control strategy derived in Section 3.1.2 presents a number of advantages with respect to the nominal one proposed in Section 3.1.2. On the other hand, from Theorem 3.1 the robust attitude controller can be tuned to deal explicitly with model uncertainties and exogenous disturbances. The importance of such robustness will be also emphasized in Section 6.1, in which the problem of controlling a prototype of quadrotor aerial vehicle is considered. Moreover, the robust attitude controller relies only on

a simple feedback law (3.43) which depends on the actual state of the system and on the nominal model inversion computed in Section 3.1.1. All of the above features thus suggest that the robust control strategy can be successfully employed in the design of an autopilot for VTOL aerial vehicles. As far as the stability of the overall position and attitude closed-loop system is concerned, note that the robust control strategy requires the attitude controller to be sufficiently fast (see the conditions on the attitude control parameters given in Proposition 3.5). This is a consequence of the fact that, in the *robust case*, it is not possible to decouple the position and the attitude dynamics trough a suitable design of the control torque as it has been done in the two solutions proposed for the *nominal case*.

Remark 3.7. When a hybrid controller is subjected to measurement noise, multiple jumps or chattering may occur [45, Chapter 4]. This phenomenon may happen when jumps map the state back to the jump domain, namely $G(D) \cap D \neq \emptyset$. With an eye on (3.34) and (3.35), since $\delta > 0$, it follows that $G_{\mathrm{nom}}(D_{\mathrm{nom}}) \cap D_{\mathrm{nom}} = \emptyset$ (see [83, Subsection IV.E]). Moreover, with an eye on (3.45), since $G_{\mathrm{rob}} = G_{\mathrm{nom}}$ and $D_{\mathrm{rob}} \subset D_{\mathrm{nom}}$, it also holds that $G_{\mathrm{rob}}(D_{\mathrm{rob}}) \cap D_{\mathrm{rob}} = \emptyset$. As shown in [83, Subsection IV.E], the parameter δ can be also tuned to ensure robustness to possibly large measurement noise. △

Remark 3.8. The robustness with respect to uncertainties and exogenous disturbances shown in Proposition 3.5 represents one of the main advantage of the inner-outer loop controller developed in this work with respect to the globally stabilizing control strategy proposed in [22]. △

Simulations and real experiments on the proposed control law are presented in chapter 6.

Conclusions

Control strategies to let the dynamics of a class of VTOL aerial vehicles tracking a desired position and attitude trajectory globally with respect to the initial conditions have been presented. The proposed feedback controllers are based on a hierarchical control paradigm in which the attitude, which is governed by means of a hybrid controller to overcome the well-known topological constraint, is employed as a virtual input to stabilize the aircraft position. Two main approaches have been proposed and compared. The first one is based on the idea of decoupling the attitude from the position dynamics by taking into account perfect knowledge of the dynamical model of the system. This approach has the advantage that the attitude and position subsystems can be tuned separately without affecting the stability properties of the overall system. The second approach on the other hand aims at obtaining a controller which is robust to the pres-

ence of uncertainties and exogenous disturbances, such as wind gusts. Since the dynamical model of the system is not perfectly known, the position and attitude dynamics cannot be decoupled. Hence the stability analysis requires to deal with the feedback interconnection between the hybrid attitude and the continuous-time position closed-loop subsystems. The resulting controller is characterized by a very simple structure, i.e., by a linear error feedback term driven by the logic required to overcome the topological obstruction and a feedforward term deriving from the references to be tracked. Simulations obtained considering a prototype of quadrotor aerial vehicle finally show how the robust controller can be effectively employed in practical applications. Experiments on real prototype are presented in the last chapter of this thesis.

3.2 Differential Wheel Robot Control

In this section we want to provide a suitable control law for the differential wheeled robot, compatible to the model proposed in the previous chapter. The goal of the control law is to generate velocity inputs to track a desired trajectory or to reach a position in space. Many authors proposed control laws for unicycles or car-like robots [73], [95], [86]. The proposed control uses the novel concept of vectored speed (similar to vectored thrust in VTOL control) and includes intrinsically a saturation on the linear speed. A rigorous Lyapunov analysis proves the semi-globally asymptotic stability of the origin of the error system, meaning that the robot will asymptotically track the desired trajectory starting from an arbitrary initial state (error). Moreover LES (Local Exponential Stability) is proved and will be useful for the next chapters on motion planning.

Consider the differential wheel robot model in (2.14). The low level control problem consists in building a control law for v and ω to achieve the tracking of a desired position trajectory $x_R(t) = [p_{x,R}(t), \dot{p_{x,R}}(t),\ p_{y,R}(t), \dot{p_{y,R}}(t)]^T$. The desired trajectory contains the position and velocity trajectories.

By defining the error coordinates $\tilde{p}_x := p_x - p_{x,R}$, $\tilde{p}_y := p_y - p_{y,R}$, we design the following control law

$$\begin{aligned} v &= \left\| \begin{bmatrix} sat_\lambda(-K_P\tilde{p}_x) \\ sat_\lambda(-K_P\tilde{p}_y) \end{bmatrix} + \begin{bmatrix} \dot{p}_{x,R} \\ \dot{p}_{y,R} \end{bmatrix} \right\|, \\ \omega &= \omega_C - K_\theta\tilde{\theta} \end{aligned} \tag{3.51}$$

having defined $\tilde{\theta} := \theta - \theta_C$,

$$\begin{aligned} \theta_C &:= atan\left(\frac{\dot{p}_{y,R} + sat_\lambda\left(-K_P\tilde{p}_y\right)}{\dot{p}_{x,R} + sat_\lambda\left(-K_P\tilde{p}_x\right)}\right), \\ \omega_C &:= \dot{\theta}_C \end{aligned}$$

with $sat_\lambda(x) := \lambda\frac{x}{\sqrt{1+x^2}}$ and $K_P, K_\theta \in \mathbb{R}_{>0}$.

The control law consists in a proportional feedback and a feed-forward term for the dynamics of θ, where θ_C is a virtual input for the attitude dynamic, and in a *vectored speed* control for the position dynamics, respectively. The idea is in fact to orient the robot linear velocity vector with the control velocity vector $v[\cos(\theta_C), \sin(\theta_C)]^T$ generated by the position loop. As a consequence, a desired orientation θ_C and a desired linear velocity v are obtained. The desired orientation becomes a virtual input for the attitude control loop.

From (2.14) and (3.51) we obtain the following closed-loop error system

$$\begin{cases} \begin{bmatrix} \dot{\tilde{p}}_x \\ \dot{\tilde{p}}_y \end{bmatrix} &= \begin{bmatrix} sat_\lambda\left(-K_P\tilde{p}_x\right) \\ sat_\lambda\left(-K_P\tilde{p}_y\right) \end{bmatrix} + v \begin{bmatrix} \cos\theta - \cos\theta_) \\ \sin\theta - \sin\theta_C \end{bmatrix} \\ \dot{\tilde{\theta}} &= -K_\theta\tilde{\theta} \end{cases} \tag{3.52}$$

For the above system the following result holds.

Proposition 3.6. *Let us consider the closed-loop system (3.52). Let* $[\tilde{p}_x(0),\, \tilde{p}_y(0),\, \tilde{\theta}(0)] \in \mathbb{X}$*, with* $\mathbb{X} \subset \mathbb{R}^3$ *an arbitrarily large compact set. Then there exist* $K_P^\star > 0$*,* $K_\theta^\star > 0$ *such that for all* $K_P > K_P^\star$*,* $K_\theta > K_\theta^\star$

- $V := \frac{1}{2}(\tilde{p}_x^2 + \tilde{p}_y^2 + \tilde{\theta}^2)$ *is a Lyapunov function for system (3.52) satisfying* $\dot{V} \leq -\gamma V$ *for some* $\gamma > 0$ *and for all* $t \geq 0$*;*
- $\lim_{t\to\infty}[\tilde{p}_x,\, \tilde{p}_y,\, \tilde{\theta}] = [0,\, 0,\, 0]$.

Proof. Let us consider the Lyapunov function V. The derivative of V along system (3.52) is given by:

$$\begin{aligned} \dot{V} &= \tilde{p}_x\left(sat_\lambda(-K_P\tilde{p}_x) + v(\cos\theta - \cos(\theta_C))\right) + \\ &+ \tilde{p}_y\left(sat_\lambda(-K_P\tilde{p}_y) + v(\cos\theta - \cos(\theta_C))\right) - K_\theta\tilde{\theta}^2 \end{aligned}$$

Being v the module of the velocity control vector which is limited in norm for a given position trajectory one can write:

$$v = \left\| \begin{bmatrix} sat_\lambda(-K_P\tilde{p}_x) \\ sat_\lambda(-K_P\tilde{p}_y) \end{bmatrix} + \begin{bmatrix} \dot{p}_{x,\,R} \\ \dot{p}_{y,\,R} \end{bmatrix} \right\| \leq L_v$$

with $L_v \in \mathbb{R}^+$ so as:

$$\begin{aligned} v(\cos\theta - \cos(\theta_C)) &\leq L_v \left|\tilde{\theta}\right| \\ v(\sin\theta - \sin(\theta_C)) &\leq L_v \left|\tilde{\theta}\right| \end{aligned}$$

obtaining

$$\begin{aligned} \dot{V} &\leq -\frac{\lambda K_p\tilde{p}_x}{\sqrt{1+(K_p\tilde{p}_x)^2}}\tilde{p}_x + L_v\left|\tilde{p}_x\right|\left|\tilde{\theta}\right| \\ &- \frac{\lambda K_p\tilde{p}_y}{\sqrt{1+(K_p\tilde{p}_p y)^2}}\tilde{p}_y + L_v\left|\tilde{p}_y\right|\left|\tilde{\theta}\right| - K_\theta\tilde{\theta}^2 \end{aligned} \tag{3.53}$$

Taking into account Young's inequality:

$$|x|\,|y| \leq \frac{\epsilon^2}{2}|x|^2 + \frac{1}{2\epsilon^2}|y|^2$$

with ϵ an arbitrary positive number, we can rewrite (3.53) as:

$$\begin{aligned} \dot{V} \quad \leq \quad & -\frac{\lambda K_p \tilde{p}_x}{\sqrt{1+(K_p\tilde{p}_x)^2}}\tilde{p}_x + \frac{L_v}{2\epsilon^2}\left|\tilde{p}_x\right|^2 + \frac{L_v\epsilon^2}{2}\left|\tilde{\theta}\right|^2 \\ - \quad & \frac{\lambda K_p \tilde{p}_y}{\sqrt{1+(K_p\tilde{p}_y)^2}}\tilde{p}_y + \frac{L_v}{2\epsilon^2}\left|\tilde{p}_y\right|^2 + \frac{L_v}{2}\epsilon^2\left|\tilde{\theta}\right|^2 - K_\theta\tilde{\theta}^2 \end{aligned} \tag{3.54}$$

By defining

$$\begin{aligned} \dot{V}_1 \quad &:= \quad -\frac{\lambda K_p \tilde{p}_x}{\sqrt{1+(K_p\tilde{p}_x)^2}}\tilde{p}_x + \frac{L_v}{2\epsilon^2}\left|\tilde{p}_x\right|^2 \\ \dot{V}_2 \quad &:= \quad -\frac{\lambda K_p \tilde{p}_y}{\sqrt{1+(K_p\tilde{p}_y)^2}}\tilde{p}_y + \frac{L_v}{2\epsilon^2}\left|\tilde{p}_y\right|^2 \\ \dot{V}_3 \quad &:= \quad L_v\epsilon^2\left|\tilde{\theta}\right|^2 - K_\theta\tilde{\theta}^2 \end{aligned}$$

we can rewrite (3.54) as

$$\dot{V} \leq \dot{V}_1 + \dot{V}_2 + \dot{V}_3.$$

We have that:

$$\dot{V}_1 \leq 0 \Leftrightarrow |\tilde{p}_x| \leq \frac{\sqrt{-L_v^2 + 4\lambda^2 K_p^2 \epsilon^4}}{L_v K_p} \triangleq \tilde{x}_{lim} \geq 0 \tag{3.55}$$

and

$$\dot{V}_2 \leq 0 \Leftrightarrow |\tilde{p}_y| \leq \frac{\sqrt{-L_v^2 + 4\lambda^2 K_p^2 \epsilon^4}}{L_v K_p} \triangleq \tilde{y}_{lim} \geq 0 \tag{3.56}$$

and

$$\dot{V}_3 \leq 0 \Leftrightarrow K_\theta \geq L_v\epsilon^2, \ \ \forall\theta \in \mathbb{R} \tag{3.57}$$

For (3.55) and (3.56) to hold, we have to chose the control parameter K_p such that:

$$K_p \geq \frac{L_v}{2\lambda\epsilon^2} \tag{3.58}$$

Note that, by fixing ϵ sufficiently large and accordingly, from (3.57) and (3.58), K_θ and K_p sufficiently large, equalities (3.55)-(3.56) hold from arbitrarily large compact sets. □

3.2.1 Bound with quadratic function

With an eye at the proof of Proposition 3.6, note that it is possible to bound the derivative of the Lyapunov function as:

$$\exists\, \gamma_1 > 0, \tilde{x}^* : \tilde{x}^* < \tilde{x}_{lim} \Rightarrow \dot{V}_1 \leq -\gamma_1\tilde{x}^2 \ \ \forall\, |\tilde{p}_x| \leq \tilde{x}^* \tag{3.59}$$

$$\exists\, \gamma_1 > 0, \tilde{y}^* : \tilde{y}^* < \tilde{y}_{lim} \Rightarrow \dot{V}_2 \leq -\gamma_1\tilde{y}^2 \ \ \forall\, |\tilde{p}_y| \leq \tilde{y}^* \tag{3.60}$$

$$\exists\, \gamma_2 > 0 : \dot{V}_3 \leq -\gamma_2 \tilde{\theta}^2 \tag{3.61}$$

obtaining

$$\dot{V} \leq -\gamma_1 \tilde{x}^2 - \gamma_1 \tilde{y}^2 - \gamma_2 \tilde{\theta}^2 \;;\; \forall\, |\tilde{p}_x| \leq \tilde{x}^* \,, |\tilde{p}_y| \leq \tilde{y}^* \,, \tilde{\theta} \tag{3.62}$$

Then, by defining:

$$\gamma' = min\left\{\gamma_1, \gamma_2\right\} \in \mathbb{R} \tag{3.63}$$

we have:

$$\dot{V} \leq -\gamma' \left(\tilde{p}_x^2 + \tilde{p}_y^2 + \tilde{\theta}^2\right) \;;\; \forall\, |\tilde{p}_x| \leq \tilde{p}_x^* \,, |\tilde{p}_y| \leq \tilde{y}^* \,, \tilde{\theta} \tag{3.64}$$

3.2.2 Closed Form Parameters

Given λ, L_v, x_{lim} an easy solution to find the control parameters to achieve the desired invariant is choosing:

$$K_p = \frac{L_v}{2\lambda} \tag{3.65}$$

$$\epsilon = \sqrt[4]{\frac{4\lambda^2 + \tilde{x}_{lim}^2 L_v^2}{4\lambda^2}} \tag{3.66}$$

and K_θ according to (3.57). The parameters gamma could be find as following:

$$\gamma_1 = \frac{\tilde{x}^* \left(K_p \lambda \epsilon^2 - L_v \sqrt{1 + K_p^2 \tilde{x}^{*2}}\right)}{\sqrt{1 + K_p^2 \tilde{x}^{*2}}\, \epsilon^2} \tag{3.67}$$

$$\gamma_2 = K_\theta - L_v \epsilon^2 \tag{3.68}$$

We now have at hand the conditions for the planning, since:

$$\begin{gathered} \underline{\alpha}(\|e\|) = \bar{\alpha}(\|e\|) = \tfrac{1}{2}\|e\|^2 \\ \gamma(\|e\|) = \gamma' \|e\|^2 \end{gathered} \tag{3.69}$$

Remark 3.9. The result in Proposition 3.6 allow to employ Lemma 4.2 by considering $\underline{\alpha} = \bar{\alpha} = \frac{1}{2}$ and the value $\gamma > 0$ which depends on the tuning of the control parameters K_P, K_θ, λ. △

3.2.3 Simulations

Finally we show some simulations of the proposed control law in action. We propose two simulations: in the first a point-to-point motion is required, while in the second the differential wheel robot needs to perform a trajectory tracking of a spiral-like trajectory.

For both the simulations, the parameters of the controller are tuned as follows: $K_P =$

4.6, $K_\theta = 4.4$, $\lambda = 1.9$. The initial state of the robot is: $[p_x, p_y, \theta] = [0, 0, \pi]$.

In the first simulation, the target $x_R(t) = [10, 0, 5, 0]$, i.e. the final point to reach is $(10, 5)$. We can see the results in Figure 3.5.

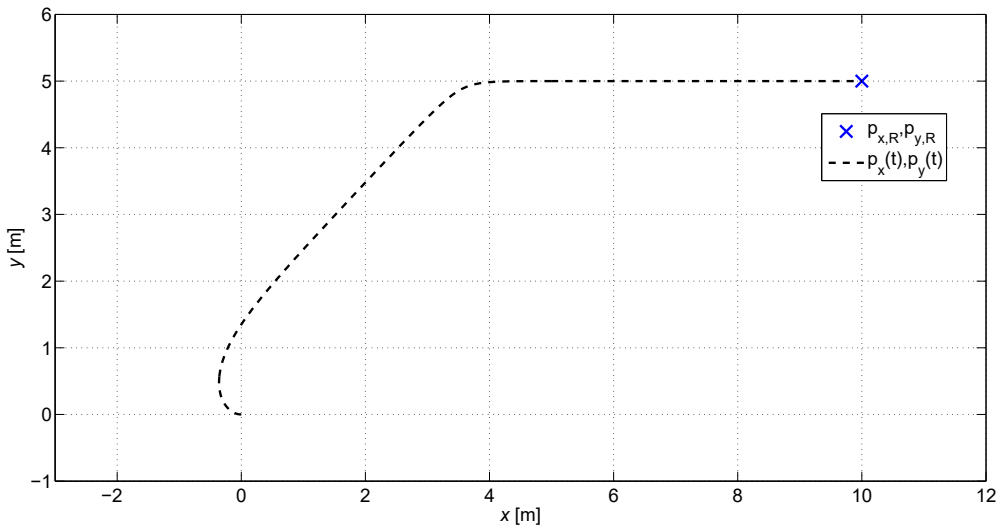

Figure 3.5: Point-to-point motion: the differential wheel robot moves from the initial position to the reference point in [10,5] m.

In second simulation, the initial state is the same, but the reference trajectory is a spiral-like trajectory. In particular, the trajectory is defined as:

$$\begin{aligned} p_{x,R}(t) &= \tfrac{A}{\frac{t}{10}+1}\sin(\omega t) \\ \dot{p_{x,R}}(t) &= -\tfrac{A}{\left(\frac{t}{10}+1\right)^2 10}\sin(\omega t) + \tfrac{A}{\left(\frac{t}{10}+1\right)}\omega\cos(\omega t) \\ p_{y,R}(t) &= \tfrac{A}{\frac{t}{10}+1}\cos(\omega t) \\ \dot{p_{y,R}}(t) &= -\tfrac{A}{\left(\frac{t}{10}+1\right)^2 10}\cos(\omega t) - \tfrac{A}{\left(\frac{t}{10}+1\right)}\omega\sin(\omega t) \end{aligned} \tag{3.70}$$

where A is the initial amplitude, chosen as $A = 5$ in the simulation, ω is the "frequency" of the spiral, chosen as $\omega = 0.1$ in the simulation.

The result of the simulation can be seen in Figure 3.6, where a $x-y$ plane is depicted with both the trajectory and the position of the robot. We can see how smoothly the robot converges and asymptotically tracks the trajectory.

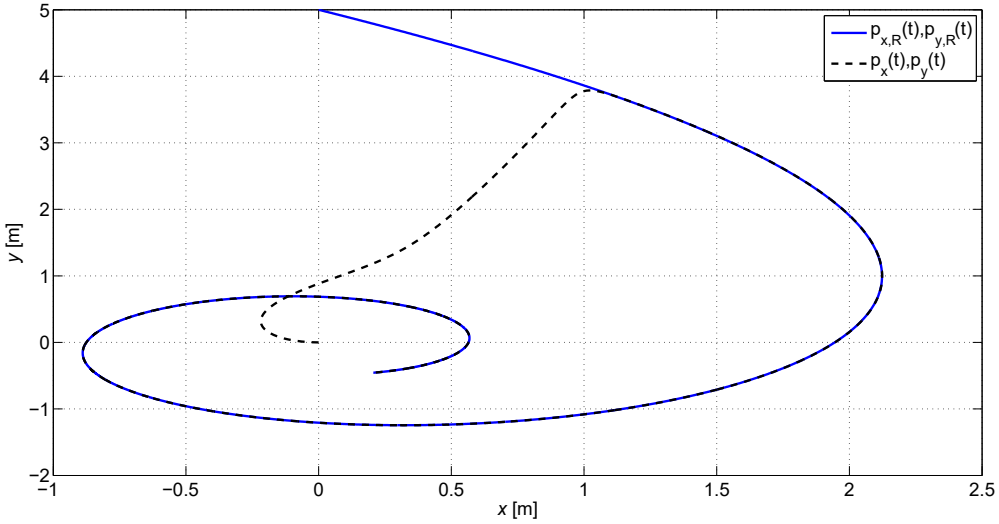

Figure 3.6: Trajectory tracking motion: the differential wheel robot reaches and tracks the spiral-like trajectory.

If you fail to plan, you are planning to fail!

Benjamin Franklin

4

Path Planning Strategies

Motion planning for mobile robots is part of the fundamental three pillars of vehicles and robots movement: guidance, navigation and control. Motion/-path planning for mobile robots falls under guidance, and despite decades of research in this field there are yet many open topics and challenges. Still many robots use actual techniques in real applications and rely on state of the art algorithms despite their deficiencies. The goal of motion planners is to generate a suitable path or trajectory for the robot, given a particular mission, taking into accounts knowledge of the state of the robot and of the environment. Some of the critical challenges the motion planning problem has to face are: the capability to take into account the dynamic differential constraints of the robot, the ability to run in real-time with the on-board computation as a real application requires, the ability to take into account possible disturbances in the environment and knowledge failures. A general solution to the problem with differentially constrained robots was proven to be very challenging and unfeasible for real application due to its high computational cost [20]. Since the general problem is unfeasible, the literature is full of strategies that try to solve one or many of the problems presented. The most recent strategies for real planning applications include randomized algorithms [67] [9] [58] and lattice-based algorithms [99] [98] [71]. In this thesis, the contribution on motion planning is about trying to solve some of the issues of the general planning problem. In this chapter we'll focus in particular on a strategy to handle kinodynamic

planning problems, relying on simple and low computational cost trajectory generation that guarantees kinodynamical feasibility of the problem not directly handling differential constraints. In the next chapter on the other hand we'll focus on a new planner approach based on Discrete Event System theory, that can potentially solve many of the planning challenges.

4.1 Introduction

When we refer to mobile robots path planning we need to first understand the notion of state of the robot. The state $X \subset X_{TOT}$ can be considered as a set of variables of interest to describe the model of the robot in the environment. X can be composed by state-space variables, configuration space, internal states or whatever variables of interest that models the problem. X_{TOT} is the complete set of possible state configurations. Of course standard approaches uses standard configuration, for example the position, orientation and velocity in a 2D or 3D dimensional euclidean space. But it can be the state-space of joints for a manipulator, or the configurations in SO(3) for a rigid body which interest in only the rotation in 3D. We call initial state $X_i = X(t = 0) \subset X_{TOT}$ the set of variables describing the initial state of the robot. We call final state $X_f = X(t = t_f) \subset X_{TOT}$ the set of variables describing the final state of the robot. The final state is usually the goal for the particular mission, where we want to bring the robot's state. We call the allowed state $X_a \subset X_{TOT}$ the set of variables the robot can traverse and with forbidden space $X_{fb} \subset X_{TOT}$ the set of state the robot shouldn't traverse. Forbidden space is usually defined by the environment, or by design to avoid unwanted situations or positions of the robot. The classic example is that X_{fb} includes the obstacles present in the environment, or states that are dangerous such as high speeds or risky areas. $X_a \bigcup X_{fb} = X_{TOT}$. The general problem of robots path planning consists in generating a state time-trajectory that brings the state X of the system from X_i to X_f only traversing X_a. The general problem can then be seen as the problem of computing $X(t)$ such that:

$$\begin{aligned} &X(0) = X_i \\ &X(t_f) = X_f \\ &X(t) \subset X_a, \ \forall t \geq 0 \end{aligned} \tag{4.1}$$

Many are the challenges to build the perfect planning algorithm and some of the most important are included in the following.

4.1.1 Feasibility

Feasibility refers to the ability of the algorithm to generate a feasible trajectory for the dynamic model of the system. Consider a general non-linear model describing the dynamic of a mobile robot:

$$\dot{x} = f(x(t), u(t)) \tag{4.2}$$

with $x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$, $f : \mathbb{R}^{n \times m} \to \mathbb{R}^n$ a locally Lipschitz function.

Consider $x_R(t) \in \mathbb{R}^n$ a state-trajectory reference and $u_R(t) \in \mathbb{R}^m$ the input reference.

The reference trajectory $x_R(t), u_R(t)$ is said to be functionally-controllable or dynam-

ically feasible if:

$$\dot{x}_R(t) = f(x_R(t), u_R(t)) \tag{4.3}$$

such that if the input $u(t) = u_R(t)$, the state-trajectory reference $x_R(t)$ evolves as a possible trajectory of the system (4.2).

This means that a good algorithm should generate a trajectory that respects all the dynamic constraints of (4.2). By not respecting feasibility, it occurs that the real robot will be incapable of exactly following the generated trajectory, leading to possible crash with obstacles. As an example we can consider a double integrator system in two dimension $(x - y)$:

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = u_x \\ \dot{y}_1 = y_2 \\ \dot{y}_2 = u_y \end{cases} \tag{4.4}$$

.

If the trajectory generated by the planner consists of connections of straight lines with constant velocity, it can result in an instantaneously change in velocity. In this scenario the system would require an infinite input (force), thus making it impossible to track. The result can be seen in Figure 4.1 where we can see how the closed-loop system (with a PD controller plus feedforward term) tries to follow the piece-wise lines, but due to the tracking error it collides with the obstacle.

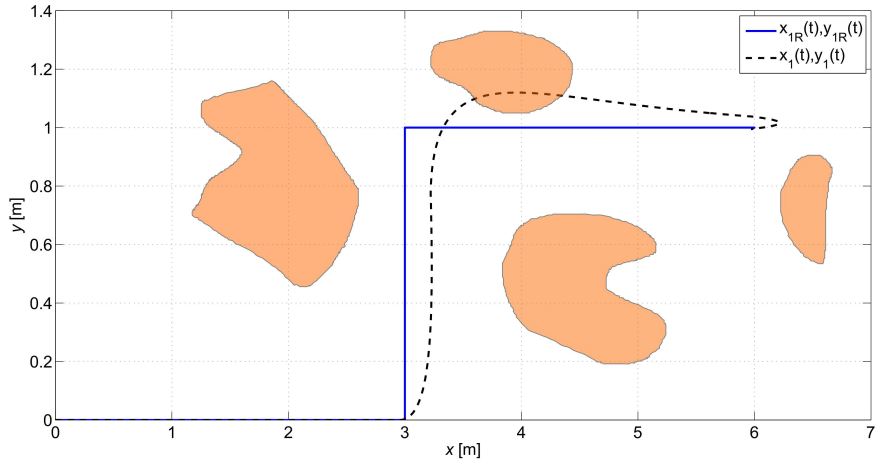

Figure 4.1: Double integrator model tracking unfeasible path. We can see how the closed loop system cannot track perfectly the trajectory, leading to possible crash with obstacles.

The feasibility condition, considered in all kino-dynamic planners [99] [10] [50] [67]

[30] [42], is usually one of the most computationally expensive part in planners, since it has to possibly deal with differential, non-linear and complex equations.

The planning problem with feasibility constraints becomes a Boundary Value Problem where the state trajectory x_R should be computed such that:

$$\begin{aligned} &\dot{x}_R = f(x_R(t), u_R(t)) \\ &x_R(0) = X_i \\ &x_R(t_f) = X_f \\ &x_R(t) \in X_a, \ \forall t \geq 0 \end{aligned} \tag{4.5}$$

4.1.2 Optimality

We talk about optimality when we want to solve the planning problem with an objective we want to optimize. The objective to optimize is usually giver in terms of objective function to maximize or minimize. The definition of the objective function gives us a way to obtain solutions that have a good quality with respect to what we are interested in.

The planning problem with feasibility constraints and optimality becomes an Optimization Problem with differential constraints, where the state trajectory x_R should be computed such that:

$$\begin{aligned} \min_{x_R} / \max_{x_R} \quad & J(v) \\ \text{subject to :} \quad & \dot{x}_R = f(x_R(t), u_R(t)) \\ & x_R(0) = X_i \\ & x_R(t_f) = X_f \\ & x_R(t) \in X_a, \ \forall t \geq 0 \end{aligned} \tag{4.6}$$

with $J(\cdot) \in \mathbb{R}$ the scalar objective function we want to maximize or minimize and v a vector of variables which the objective function can depend on. Some example the function J and the variables v can include, among others, time minimization, such that $J = t_f$, distance minimization, such that $J = \int_0^{t_f} |x_R(t)|dt$ or input minimization, such that $J = \int_0^{t_f} |u_R(t)|dt$.

The problem of computing optimal motion planning was proven to be very challenging and computationally expensive [19], [20].

4.1.3 Runtime

Runtime represents the CPU time, or in general the computational time to evaluate the trajectory with the planning algorithm. This is strictly related to the ability of the robot to react fast to changes in the environment or in the final state target, and the runtime

should be as low as possible if we want to be able to run the algorithm in real-time. This is a very important requirement for real world application in path-planning of mobile robots, since the fast reactivity can avoid unpleasant and dangerous situations for the robot. Usually the high computations comes from many factors of the algorithm: for discretized and graph-search algorithms the complexity comes for the fine discretization that leads to high dimension graphs or lattices, for randomized algorithms the complexity comes to the points connector which needs to take into account the kinodynamic constraints, while for almost every algorithm, the collision checking module requires high computation.

4.1.4 Completeness

An algorithm is said to be complete if it returns a solution to the problem in finite time if it exists, or it fails in finite time otherwise. This is an hard requirements that almost every known and used algorithm doesn't have. Other completness concept as *probabilistic-completeness* or *resolution-completeness* are given in literature for special algorithms. Probabilistic completeness [67] [58] is related to sampling-based algorithms for which the completeness probability goes to 1 as soon as the sampling number increase. Resolution-completeness [66] on the other hand is related to discretized algorithms, where the completeness is guaranteed if the resolution of the algorithm increases the precision. Most of the discretized algorithm fail to have even resolution completeness properties.

4.1.5 Globality

Globality refers to the set of states in which the planning algorithm is computed and valid. It is said to be *global* if the algorithms runs on X_{TOT} while is said to be *local* if it runs on a subset $X_{PAR} \subset X_{TOT}$. Of course the best algorithms should be global, but the drawback is the increase in computational time given by the increased state space. Local algorithms are for example run on the subspace of state included in the range of exteroceptive sensors of the robot. Many planning strategies [113][71] use a combined local and global planner to try keep both the advantages of local and global planner. In some grid-based or lattice-based algorithms for example a fine discretization is used around robot's location, while a rough discretization is used outside. Other strategies may use a kinodynamic planning close to the robot and a simple non-kinodynamic planner outside [99]. There is usually a compromise between runtime performances and globality, where a global planner usually takes more computation to perform. In real world application in unknown environment, the globality is not always useful, since the motion planner is evaluating only local information, making the plans outside the sensor range useless.

4.1.6 Robustness

Robustness is a feature that is very often omitted from the planning algorithms. Designing a robust algorithm means to take into account for generic disturbances that can model: dynamic model uncertainties such as un-modeled dynamics or wrong parameter estimation, state knowledge uncertainties such as localization uncertainties due to sensors noise, environment disturbances such as unknown forces acting on the robots (wind for example). Most of the works in literature that takes into account the problem, proposes solution based on stochastic modeling of uncertainties and embed those limitations into the planner [16] [15] [64]. In [78], a robust planner is developed using Lyapunov invariant funnels and interconnecting motion primitives taking into account disturbances and un-modeled effects.

4.1.7 Replanning

The environment for robots can be inherently uncertain and dynamic. Obstacles can move or be freshly detected by the onboard sensors, making the previous computed path non feasible on non-optimal due to changes in X_a. The replanning is the capacity of the algorithm to re-plan the path after the initial computation. It is strictly linked to runtime capabilities since the new trajectory should be computed in a very limited time, in particular if we are considering the robot dynamics and we are already executing the previous path. For example a robot moving very fast in a forest and detecting a new tree, requires a new trajectory in milliseconds otherwise at high speed it would move far away before a new trajectory is computed, leading to a possible crash. The replanning ability is not only linked to the runtime, but on the nature itself of the algorithm. Some algorithms as the discretization-based or grid-based requires that the state of the robot is in one of the discretized state to compute a trajectory, so the new plan cannot be computed at any time.

4.1.8 Incremental

Incremental capability is strictly linked to replanning. It means that the new plan can be computed based on the differences in the constraints or environment and based on the previously computed path. Starting from the latest path, it is repaired to take into account the new forbidden states and possible cost changes. This is often many times more efficient than re computing the whole path from scratch. Some incremental algorithms for planning on graphs are D* and its variants [62] [107].

4.2 Piece-Wise Controllable Trajectories and Practical Tracking

As widely known, kinodynamic planning is the most difficult task when developing motion planners, and generating feasible paths is usually one of the most computationally expensive operation, because the planner has to deal with dynamic constraints given generally in the form of non-linear differential constraints. What we want to study in this chapter is the effect of generating quasi-feasible (controllable) trajectories, composed as sequences of piece-wise feasible trajectory. The goal is to analyze the interconnection of the low-level control law for trajectory tracking with the planner that generates such trajectories and what's the impact of this choice in the planner. What we obtain is a practical tracking (since asymptotic tracking is impossible due to non-feasibility of the path), where we want to derive bounds on tracking error of such trajectories [38] [41] [40]. The advantages of this analysis is that the planner can be very simple and generate simple non-feasible trajectories, while still guaranteeing bounded tracking error. This information can be used to plan safe and robust paths without directly dealing with differential constraints. As we'll see later in this chapter, piece-wise controllable trajectories for many mobile robots can mean simple straight lines, which are very easy to concatenate with existing algorithms and with the proposed algorithms in this thesis. This reduces by many times the runtime of the planner, making it suitable for fast replanning but still considering the dynamic of the vehicles. Finally we study the same conditions taking into account uncertainties and disturbances on the system. Using ISS (Input to State Stability) Lyapunov analysis we derive bounds on tracking error of piece-wise controllable trajectories in the presence of disturbances on the system, allowing for robust and fast kinodynamic planning algorithms.

4.2.1 Piece-wise Continuous Reference and Integrator Chain System

As first case we want to analyze the interconnection of a chain of integrators dynamic system with state-feedback closed loop and a planner that generates sequences of piece-wise continuous trajectories. Let us consider the following continuous-time single-input time-invariant linear system:

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{4.7}$$

where $x = \begin{bmatrix} x_1, & x_2, & \cdots, & x_n \end{bmatrix}^T \in \mathbb{R}^n$, $u \in \mathbb{R}$, $A \in \mathbb{R}^{n \times n}$ given by

$$A = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \ddots & 0 \\ 0 & 0 & 0 & \ddots & 0 \\ \vdots & \vdots & \vdots & \cdots & 1 \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix} \tag{4.8}$$

and $B \in \mathbb{R}^{n\times 1}$ defined as

$$B = \begin{bmatrix} 0, & \cdots, & 0, & 1 \end{bmatrix}^T \tag{4.9}$$

By construction, the above system represents a chain of integrators of dimension n.

Assume now that the goal of the control design is to track a desired time reference trajectory given by

$$x_R(t) = \begin{bmatrix} x_{1R}(t), & x_{2R}(t), & \cdots, & x_{(n+1)R}(t) \end{bmatrix}^T \tag{4.10}$$

with $x_{iR}(t) = \dot{x}_{(i-1)R}(t),\ \forall i = 2, 3, ..., n+1$. For sake of convenience, we compactly denote as $\overline{x}_R(t)$ the vector which corresponds to the desired trajectory together with the derivatives up to the $(n-1)$-th so as:

$$\overline{x}_R(t) = \begin{bmatrix} x_{1R}(t), & x_{2R}(t), & \cdots, & x_{nR}(t) \end{bmatrix}^T. \tag{4.11}$$

The desired trajectory is assumed to be a piece-wise continuous function of time with arbitrary switch time and piece-wise class C^n on the piece intervals. We define t_j the time when a trajectory switch occurs, namely when the reference is switching from the trajectory piece $j-th$ to trajectory piece $(j+1)-th$, with t_j^- the time just before the j-th reference switch and with t_j^+ the time just after the j-th reference switch, with $j \in \mathbb{N}$. The j-th trajectory piece is continuous and class C^n on the interval $\begin{bmatrix} t_{j-1}^+, & t_j^- \end{bmatrix}$. We suppose that $\left| x_{iR}(t_j^+) - x_{iR}(t_j^-) \right| \leq d_i\ \forall i = 1, 2, ..., n,\ \forall j$, with $d_i \in \mathbb{R}^+$ (i.e. there are bounds on the difference of trajectory and its derivatives after a switch occurs). Finally, we define $\hat{d} = \sqrt{d_1^2 + d_2^2 + ... + d_n^2}$. One can now write the error dynamics of the system by defining $e(t) \triangleq x(t) - \overline{x}_R(t)$, and accordingly:

$$\begin{aligned} e_1(t) &\triangleq x_1(t) - x_{1R}(t) \\ e_2(t) &\triangleq x_2(t) - x_{2R}(t) \\ &... \\ e_n(t) &\triangleq x_n(t) - x_{nR}(t) \end{aligned} \tag{4.12}$$

obtaining, from (4.7),

$$\dot{e}(t) = Ae(t) + B(u(t) - x_{(n+1)R}(t)) \tag{4.13}$$

The error system results in a time-dependent switching system due to the presence of the piece-wise continuous reference $x_R(t)$. In fact, at every switch in the reference signal it corresponds a jump in the state e of the error system.

The goal of the control law is to track the desired reference trajectory and to achieve

$$\|e(t)\| \leq q, \ \ \forall t \geq 0 \tag{4.14}$$

with $q > \hat{d}$, namely to maintain the tracking error bounded despite jumps in the reference signal.

Inspired by [70], we introduce a dwell time τ, i.e. a minimum time between two switches in the reference trajectory $x_R(t)$, so as (4.14) can be achieved. More formally, the following result can be stated.

Lemma 4.1. *With the control input $u(t) = B^T Pe(t) + x_{(n+1)R}(t)$, with P solution of the Riccati equation $A^T P - 2PBB^T P + PA = -aI_n$, and with a Dwell time $\tau \geq -\frac{\bar{\lambda}_P}{a} log\left(\frac{q^2-\hat{d}^2}{q^2}\right)$, with $\bar{\lambda}_P$ the largest eigenvalue of the matrix P, the overall error system (4.13) is stable in the sense of Lyapunov (see [70]) and $\|e(t)\| \leq q, \ \ \forall t \geq 0$.*

Proof: Choosing as common Lyapunov function the quadratic function $V = e^T Pe$, the control law $u(t) = B^T Pe(t) + x_{(n+1)R}(t)$ with P solution to the Riccati equation given in the statement of the lemma makes the matrix $(A + BK)$ Hurwitz and ensures that

$$\dot{V}(t) \ \ = \ \ -a \|e\|^2 (t) \tag{4.15}$$

By applying the *comparison lemma* [61] it also holds

$$\underline{\lambda}_P \|e(t)\|^2 \leq \ \ V(t) \ \ \leq \bar{\lambda}_P \|e(t)\|^2 \tag{4.16}$$

From (4.15) and (4.16) we have:

$$-a\frac{V(t)}{\underline{\lambda}_P} \leq \dot{V}(t) \leq -a\frac{V(t)}{\bar{\lambda}_P} \tag{4.17}$$

The time between two consecutive switches, namely the dwell time, is given by $\tau = t_j^- - t_{j-1}^+$. We have that, because of the switch:

$$\left\|e(t_j^+)\right\|^2 \leq \left\|e(t_j^-)\right\|^2 + \hat{d}^2 \tag{4.18}$$

and then to achieve (4.14), a necessary condition is given by:

$$\left\| e(t_j^+) \right\|^2 \leq q^2 \tag{4.19}$$

which, by considering the Lyapunov function (4.15) and the inequalities in (4.16), (4.18) and (4.19) could be rewritten as:

$$V(t_j^-) \leq \overline{\lambda}_P(q^2 - \hat{d}^2) \tag{4.20}$$

From (4.20), it is possible to write the evolution of the Lyapunov function based on the bounds (4.17) on its derivatives:

$$V(t_j^-) = V(t_{j-1}^+)e^{-\frac{a}{\overline{\lambda}_P}\tau} \leq \overline{\lambda}_P(q^2 - d^2) \tag{4.21}$$

Then, by considering the worst case of $V(t_{j-1}^+) = \overline{\lambda}_P q^2$, note that the inequality (4.21) holds true when the dwell-time is chosen as in the statement of the lemma.

□

Remark: for the sake of simplicity, it is possible to define

$$d^* = max\{d_1, d_2, ..., d_n\}$$

and to take $\hat{d} = \sqrt{n}d^*$, obtaining a more conservative but easier choice for the parameter d.

Example and Application

Let us consider a mobile robot modeled as a fully-actuated double integrator system defined over a 2-dimensional Euclidean space

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = u_1 \\ \dot{y}_1 = y_2 \\ \dot{y}_2 = u_2 \end{cases} \tag{4.22}$$

in which x_1 represents the longitudinal position in a 2D plane and y_1 the lateral position. The above model can be employed to model, for instance, the kinematics of a holonomic vehicle including the lateral / longitudinal dynamics of VTOL aerial vehicle [34].

The trajectory tracking problem consists of allowing the vehicle to track a desired trajectory $x_R(t)$, $y_R(t)$ maintaining a bounded tracking error despite the presence of jumps or asymptotically track the desired reference when no jump occurs. In this case

$x_R(t)$ represents a position trajectory in x direction and not the full state reference. The trajectory tracking problem for (4.22) can be defined as follows. Given the system (4.22), a trajectory $x_{1R}(t)$, $y_{1R}(t)$, and its derivatives up to the 2nd order, build a control law $u_s = \begin{bmatrix} u_1 & u_2 \end{bmatrix}^T$ such that the error system

$$\dot{e}_s(t) = A_s e_s(t) + B_s u_s(t) + \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix}^T \ddot{x}_{1R}(t) + \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T \ddot{y}_{1R}(t) \tag{4.23}$$

is asymptotically stable in the special case no jump in the reference occurs. The error system is derived from (4.22) by defining:

$$e_s(t) = \begin{bmatrix} e_{s1}(t) \\ e_{s2}(t) \\ e_{s3}(t) \\ e_{s4}(t) \end{bmatrix} \triangleq \begin{bmatrix} x_1(t) - x_{1R}(t) \\ x_2(t) - \dot{x}_{1R}(t) \\ y_1(t) - y_{1R}(t) \\ y_2(t) - \dot{y}_{2R}(t) \end{bmatrix} \tag{4.24}$$

so as

$$A_s = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad B_s = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \tag{4.25}$$

The goal could be achieved by a control law:

$$u_s(t) = K_s e_s(t) + \begin{bmatrix} 1 \\ 0 \end{bmatrix} \ddot{x}_{1R}(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \ddot{y}_{1R}(t) \tag{4.26}$$

with $K_s = B_s^T P_s$, with P_s solution of the Riccati equation $A_s^T P_s - 2P_s B_s B_s^T P_s + P_s A_s = -aI_4$.

Remark: The above result follows as in the proof of Lemma 4.1 by considering system (4.22) as two disjoint single-input systems, namely the lateral and the longitudinal dynamics. □

Given the control law, by choosing an appropriate a it is possible to fix the state-feedback gains as well as the matrix P_s. Once the maximum allowed error q and the trajectory discontinuity d are fixed, it is possible to compute the Dwell TIme τ following lemma 4.1. A more detailed are realistic application, with explicit calculation of τ will be given in next chapter.

4.2.2 Piece-wise Controllable Reference and Non-Linear System

We now want to extend the previous case to a more general non-linear dynamic system, to model any kind of robot. In this case we suppose that the reference trajectory is composed of a sequence of piece-wise feasible trajectories, called *primitives*.

To model a more general class of ground / aerial vehicles, we consider the nonlinear system

$$\dot{x} = f(x(t), u(t)) \tag{4.27}$$

with $x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$, $f : \mathbb{R}^{n \times m} \to \mathbb{R}^n$ a locally Lipschitz function.

The desired state and input reference signals are composed of a sequence of *primitives*, i.e.

$$x_R(t) := \bigcup_{i \in \mathbb{N}} x_R^{i,i+1}(t), \;\; u_R(t) := \bigcup_{i \in \mathbb{N}} u_R^{i,i+1}(t) \tag{4.28}$$

with $i \in \mathbb{N}$ and

$$x_R^{i,i+1} \; : \; [t_i, \, t_{i+1}) \to \mathbb{R}^n, \;\; u_R^{i,i+1} \; : \; [t_i, \, t_{i+1}) \to \mathbb{R}^m \, . \tag{4.29}$$

Each i-th primitive is defined over the compact time interval $[t_i, \, t_{i+1})$, with $t_{i+1} > t_i$, and it is such that

$$\dot{x}_R(t) = f(x_R(t), u_R(t)) \quad \forall t \in [t_i, \, t_{i+1}) \, , \tag{4.30}$$

namely it is functionally-controllable in the interval $[t_i, \, t_{i+1})$. By construction, note that the state reference $x_R(t)$ is a piece-wise continuous function of time: in fact discontinuities may occur at times t_i, $i \in \mathbb{N}$. We suppose that, for all $i \in \mathbb{N}$,

$$\left\| x_R^{i,i+1}(t_{i+1}^-) - x_R^{i+1,i+2}(t_{i+1}) \right\| \leq d' \, , \tag{4.31}$$

for some $d' \geq 0$, i.e., the discontinuities of the state trajectory at the switching times are bounded.

For some vehicles, as the case of differential wheel car as seen later, only a subset of the state vector can be of interest for the tracking task and at the same time, to stabilize the system, other states can be taken into account. Accordingly, we assume that the state x can be written as $x := [y, \, z]^T$, with $y \in \mathbb{R}^p$ the outputs to be controlled, and $z \in \mathbb{R}^{n-p}$ and the equivalent $x_R := [y_R, \, z_R]^T$, with $y_R \in \mathbb{R}^p$ and $z_R \in \mathbb{R}^{n-p}$.

Example 4.1. Let us consider the linear system (double integrator)

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = u \end{cases} \tag{4.32}$$

with state $x = [x_1, \, x_2]^T \in \mathbb{R}^2$, input $u \in \mathbb{R}$. The above system can be employed to model

the dynamics of a holonomic robot moving along a line [111]. We assume that the output of interest is given by $y := x_1$, i.e. the position. Following the above construction, given a constant $v \in \mathbb{R}$, a reference state and input can be obtained by means of the following two primitives

$$\begin{aligned} x_R^{0,1}(t) &= [x_{1R}(t_0) + vt,\, v]^T \\ u_R^{0,1}(t) &= 0 \end{aligned} \tag{4.33}$$

which is defined over the time interval $[t_0,\, t_1)$, and

$$\begin{aligned} x_R^{1,2}(t) &= [x_{1R}(t_1) - vt,\, -v]^T \\ u_R^{1,2}(t) &= 0 \end{aligned} \tag{4.34}$$

defined over the time interval $[t_1,\, t_2)$. Note that for the above two primitives we can compute the discontinuity at the switching time as in (4.31) obtaining $d' = \sqrt{2}v$. △

By defining the tracking errors as:

$$\tilde{y}(t) := y(t) - y_R(t)\,, \tag{4.35}$$

the control input $u(t)$ is designed in order to achieve practical tracking of the outputs of interest, i.e.

$$\|\tilde{y}(t)\| \leq q \quad \forall t > 0 \tag{4.36}$$

for some given $q > d$, with $d \in \mathbb{R}$. Moreover we suppose that the set of states for the stability of the system is extended to ξ defined as:

$$\xi = \begin{bmatrix} \tilde{y} \\ \tilde{\eta} \end{bmatrix} \tag{4.37}$$

with $\tilde{\eta} \in \mathbb{R}^r$ a set of auxiliary states. For construction $\|\xi(t)\| \leq q \quad \forall t > 0 \quad \Rightarrow \|\tilde{y}(t)\| \leq q \quad \forall t > 0$. The control input u is designed as function of state ξ and reference x_R as $u = \kappa(\xi, x_R)$. The extended error system is such that:

$$\dot{\xi}(t) = f'(\xi(t)) \tag{4.38}$$

with $f' : \mathbb{R}^{p \times r} \to \mathbb{R}^{p \times r}$ a locally Lipschitz function. Interestingly enough, the error system 4.38 can be considered as a switching system [70], since x_R may switch at t_i to a different value.

The discontinuity d is the equivalent of d', but considering the extended state ξ such that for all $i \in \mathbb{N}$:

$$\left\| \xi^{i,i+1}(t_{i+1}^-) - \xi^{i+1,i+2}(t_{i+1}) \right\| \leq d\,, \tag{4.39}$$

Note that q represents the maximum norm of the tracking error that we want to guarantee for the desired outputs. By construction, the lower bound for this parameter is the discontinuity bound d: the bigger is the discontinuity, the bigger will be the tracking error, with the limit case of theoretically perfect tracking only if $d = 0$ (i.e. the trajectory is continuous).

To achieve target (4.36), the control law u will be designed in order to have, for all $\xi : \left\|\tilde{\xi}\right\| < q$,

$$\begin{aligned} \underline{\alpha}(\|\xi\|) \leq V(\xi) \leq \bar{\alpha}(\|\xi\|) \\ \dot{V} \leq -\gamma(\|\xi\|) \;\; \forall t \in (t_i, t_{i+1}), \forall i \in \mathbb{N} \end{aligned} \tag{4.40}$$

for some class $\mathcal{K}$ functions $\underline{\alpha}(\cdot), \bar{\alpha}(\cdot)$ defined on $[0, q)$, and for a class $\mathcal{K}$ function $\gamma(\cdot)$ defined on $[0, q)$, and where $V : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ is a common Lyapunov function.

Accordingly, along the interval of time $[t_i,\ t_{i+1})$, the Lyapunov function V decreases, while, at times t_i, t_{i+1}, .., the Lyapunov function could increase due to the discontinuous references.

Remark 4.1. The stability properties on the extended error system ξ are present on the subset of states of interest $\tilde{y}$, such that if the origin $\xi = 0$ is stable, even the states of interest for tracking $\tilde{y}$ are stable. △

To guarantee that the tracking error does not exceed the desired maximum value q, the following result, which makes use of the notion of dwell-time defined in [70], can be employed.

Lemma 4.2. *Let us define $V_{max} := \underline{\alpha}(q)$, $V_{min} := \underline{\alpha}(\bar{\alpha}^{-1}(\underline{\alpha}(q)) - d)$ and $e_{max} := \bar{\alpha}^{-1}(\underline{\alpha}(q))$. If, $\forall i \in \mathbb{N}$, $t_{i+1} - t_i \geq \tau$ with τ solution to the boundary value problem*

$$\begin{aligned} V(t_i) &= V_{max} \\ V(t_i + \tau) &= V_{min} \\ \dot{V} &\leq -\gamma\left(\bar{\alpha}^{-1}(V)\right) \end{aligned} \tag{4.41}$$

and the initial state for the error system (4.38) is such that:

$$\|\xi(t_0)\| \leq e_{max} \tag{4.42}$$

then the constraint (4.36) is satisfied for all $t > 0$.

Proof. From (4.40), the goal (4.36) can be satisfied if the Lyapunov function V does not surpass the value V_{max} defined in the statement of the lemma. This requires the error $\xi(t)$ to be such that $\|\xi(t_i)\| \leq e_{max}$ with $e_{max} = \bar{\alpha}^{-1}(\underline{\alpha}(q))$ for all times t_i, $i \in \mathbb{N}$, in which a jump to a different primitive occurs (see Figure 4.2 for a graph-

ical intuition). From (4.40), the constraint $\|\xi(t_i)\| \leq e_{max}$ is guaranteed by having $V(t_i^-) \leq V_{min}$, namely by imposing a constraint on the maximum value of V before a jump to a different primitive. Accordingly, τ can be computed as the time required by Lyapunov function to decrease from V_{max} to V_{min}, namely by solving (4.41). □

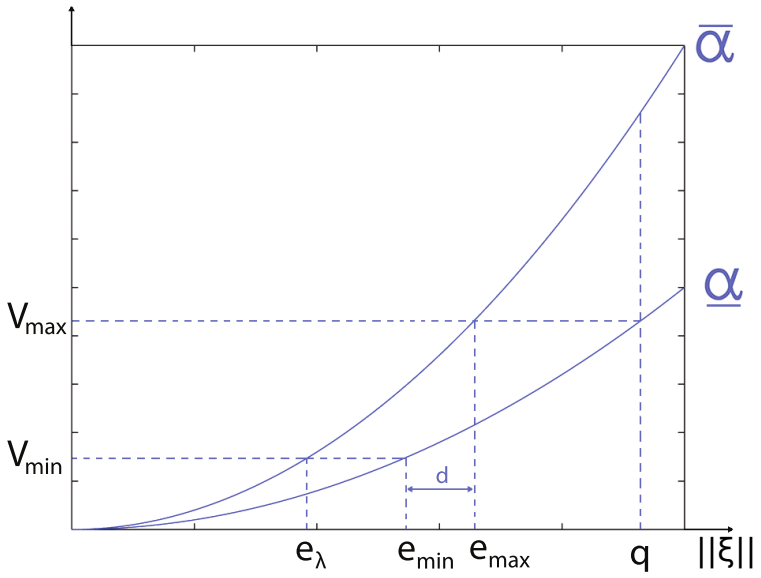

Figure 4.2: Visual representation of the Lemma conditions

Remark 4.2. By using the comparison lemma in [61], the time τ can be upper-bounded by considering in (4.41) the solution to the ordinary linear differential equation $\dot{V} = -\gamma$, in which $\lambda = \gamma\left(\bar{\alpha}^{-1}(V_{min})\right)$, obtaining

$$\tau = \frac{\underline{\alpha}(q) - \underline{\alpha}\left(\bar{\alpha}^{-1}\left(\underline{\alpha}\left(q\right)\right) - d\right)}{\gamma\left(\bar{\alpha}^{-1}\left(\underline{\alpha}\left(\bar{\alpha}^{-1}\left(\underline{\alpha}\left(q\right)\right) - d\right)\right)\right)} \tag{4.43}$$

△

Remark 4.3. The time τ can be upper-bounded by solving a linear ordinary differential equation. In fact there exists $\lambda \in \mathbb{R}$ such that:

$$\dot{V} \leq -\lambda \quad \forall \xi : \|\xi\| \geq h_\lambda \tag{4.44}$$

By choosing $h_\lambda = e_\lambda = \bar{\alpha}^{-1}(V_{min})$ and $\lambda = \gamma(e_\lambda)$ the minimum dwell time τ is chosen such that:

$$V_{max} - \lambda\tau \leq V_{min} \tag{4.45}$$

from which τ could be chosen as:

$$\tau = \frac{\underline{\alpha}(q) - \underline{\alpha}\left(\bar{\alpha}^{-1}\left(\underline{\alpha}\left(q\right)\right) - d\right)}{\gamma\left(\bar{\alpha}^{-1}\left(\underline{\alpha}\left(\bar{\alpha}^{-1}\left(\underline{\alpha}\left(q\right)\right) - d\right)\right)\right)} \tag{4.46}$$

△

4.2.3 Piece-wise Controllable Reference and Non-Linear System with Disturbances

We want to extend the previous subsection considering disturbances in the model of the system. The disturbances can model external inputs, such as wind for UAV, parameter uncertainties, such as mass and inertia of the UAV, or un-modeled effects. This analysis allows us to derive trajectory tracking conditions on the boundedness of the tracking error to generate robust planners. Finally an application to forest navigation of differential-wheel robot and VTOL in presence of disturbances is proposed, where the planner generates simple waypoints but still guarantees dynamic constraints and non-collision of the dynamic system with disturbances.

For this extension we need to recall and manipulate some notion on ISS (Input to state Stability). The following tools for ISS stability were derived from [53], [105]. Considering a system

$$\dot{x} = f(x, w) \tag{4.47}$$

with $x \in \mathbb{R}^n$, $w \in \mathbb{R}^k$, with $f(\cdot)$ locally Lipschitz for all $x : \|x\| \leq x_m \in \mathbb{R} > 0$. The system is said to be Locally Input to State Stable (ISS) if there exist some class $\mathcal{K}$ functions $\overline{\beta}(\cdot), \underline{\beta}(\cdot), \gamma(\cdot), \sigma(\cdot)$ defined on $[0, x_m)$, and a function $V : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ of class C^1 such that:

$$\begin{array}{c} \underline{\beta}(\|x\|) \leq V(x) \leq \overline{\beta}(\|x\|) \\ \dot{V} \leq -\gamma(\|x\|) + \sigma(\|w\|) \end{array} \tag{4.48}$$

From (4.48) we have

$$\|x\| \geq \chi(\|w\|) \Rightarrow \dot{V} \leq -\gamma'(\|x\|) \tag{4.49}$$

with $\chi(\|w\|)$ a class $\mathcal{K}$ function:

$$\chi(\|w\|) = \gamma^{-1}\left(k\sigma(\|w\|)\right) \tag{4.50}$$

,

$k \in \mathbb{R} > 1$ and

$$\gamma'(\|x\|) = \frac{k-1}{k}\gamma(\|x\|) \tag{4.51}$$

Since

$$\dot{V} \leq -\gamma'\left(\underline{\beta}^{-1}(V)\right) \quad \forall V \geq V_1 \tag{4.52}$$

with $V_1 = \overline{\beta}\left(\chi\left(\|w\|_\infty\right)\right)$, there exist a $\bar{\gamma} \in \mathbb{R} > 0$ such that

$$\dot{V} \leq -\overline{\gamma}V \quad \forall V \geq V_1 \tag{4.53}$$

Note: up to V_1 the Lyapunov function value decreases exponentially.

We can define a class $\mathcal{K}$ functions $\Psi(\|w\|)$ that corresponds to the asymptotic gain for the system (4.47). It is defined as:

$$\Psi(\|w\|) = \underline{\beta}^{-1}\left(\overline{\beta}\left(\chi(w)\right)\right) \tag{4.54}$$

such that:

$$\|x(t)\|_\infty \leq \Psi(\|w\|_\infty) \tag{4.55}$$

with $x(t)$ the solution of (4.47).

Recalling the notation of the previous section, we define the closed loop error system with disturbances:

$$\dot{\xi}(t) = f'(\xi(t), w) \tag{4.56}$$

Assumption 1: the closed loop error system 4.56 is ISS such that (4.48) hold, where the state x is the error state ξ. **Assumption 2**: two consecutive trajectories are connected, such that the maximum discontinuity in the error system 4.56 state space is less than $d \in \mathbb{R}$.

Theorem 4.1. *Let Assumption 1 and Assumption 2 hold, let $e_{max} \in \mathbb{R} > 0$. Let:*

$$e_{max} \geq \bar{e} \triangleq \underline{\beta}^{-1}\left(\overline{\beta}\left(\chi\left(\|w\|_\infty\right)\right)\right) \tag{4.57}$$

let:

$$V_{min} \geq V_1 \tag{4.58}$$

let:

$$\xi_0 = \xi(0) : V(\xi_0) \leq V_{max} \tag{4.59}$$

let:

$$V_{max}e^{-\bar{\gamma}\tau} \leq V_{min} \tag{4.60}$$

with $V_{max} = \underline{\beta}(\|e_{max}\|)$, $V_{min} = \underline{\beta}\left(\overline{\beta}^{-1}(V_{max}) - d\right)$.
Then:

$$\|\tilde{y}(t)\| \leq \|\xi\| \leq e_{max} \quad \forall t > 0 \tag{4.61}$$

After fixing the controller parameters we want to find a maximum discontinuity d and a lower bound on the time interval τ, such that if the previous conditions are met we achieve a practical tracking with maximum tracking error of e_{max}. There are many ways to solve the problem in a practical application as we'll see in later, but it can be generalized as an optimization problem that can be non-linear because of functions $\underline{\beta}$ and $\overline{\beta}$.

Proof. Following (4.2): V_{max} is the maximum value of the Lyapunov function such that:

$$V(t) < V_{max} \Leftrightarrow \|\tilde{y}(t)\| \leq \|\xi(t)\| \leq e_{max}, \quad \forall t > 0 \tag{4.62}$$

Every time a new piece-wise trajectory is generated, the discontinuity d increases the Lyapunov function value. For iterability we want that before a new primitive is taken, the Lyapunov value decreases up to the value V_{min}, such that after the discontinuity the Lyapunov function doesn't exceed V_{max}. $V_{min} = \underline{\beta}\left(\overline{\beta}^{-1}(V_{max}) - d\right)$. Equation (4.53) guarantees that:

$$V(t) = V(0)e^{-\bar{\gamma}t} \quad \forall V > V_1 \tag{4.63}$$

Hence (4.60) guarantees that after the discontinuity, the value of Lyapunov function decreases at least to V_{min} in time τ:

Condition (4.58) is required by the ISS condition (4.53).

The minimum tracking error $\bar{e}$ corresponds to the asymptotic gain of the system.

□

We apply the previous theorem to the problem of tracking a particular class of trajectories for some systems of interest. The trajectories we are interested are defined on $\mathbb{R}^2$ (it is possible to extend to $\mathbb{R}^3$) and are piece-wise constant in velocity and continuous in position. This kind of trajectories could be seen as all the trajectories generated connecting waypoints with linear segments with constant velocity along the path. We define the trajectory $p_R(t) \in \mathbb{R}^2$, where $\mathbb{R}^2$ defines the operative space which in our case is the $x - y$ plane (3D extension is trivial). We can define:

$$p_R(t) := \bigcup_{i \in \mathbb{N}} p_R^i(t) \tag{4.64}$$

with $i \in \mathbb{N}$ and

$$p_R^i : [t_i, t_{i+1}) \to \mathbb{R}^2, \tag{4.65}$$

Let's assume that

- The liner velocity along the path is constant and with value v so that $\|\dot{p}_R(t)\| = v$. v is the norm of the velocity in the plane $x - y$.
- The minimum duration of a trajectory piece is T, such that $t_{i+1} - t_i > T$ for all i. Since the velocity is constant the minimum length for the segment or distance between waypoints is given by vT.
- The angle between two segments is at most α_{max}

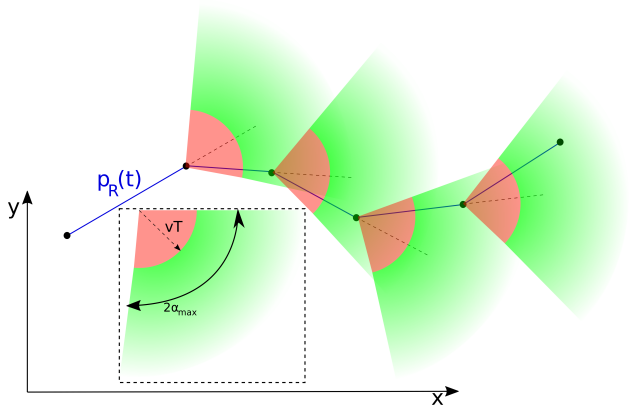

Figure 4.3: Example of Trajectory of interest. The trajectory is composed of segments connecting waypoints with minimum length vT and maximum angle among them of α_{max}

An example of trajectory of interest is given in Figure 4.3. For construction, the trajectory has no discontinuity in terms of position, has maximum discontinuity in velocity of $v\sqrt{2(1 - \cos(\alpha_{max}))}$ and maximum discontinuity as angle $2\alpha_{max}$. In fact, the norm of the difference of two vectors v_1 and v_2 with same norm v whom versors differ α radians, is given by:

$$\|v_1 - v_2\| = \sqrt{v^2 + v^2\cos^2(\alpha) - 2v^2\cos(\alpha) + v^2\sin^2(\alpha)} \tag{4.66}$$

and

$$\|v_1 - v_2\| = \sqrt{v^2 + v^2 - 2v^2\cos(\alpha)} \tag{4.67}$$

The discontinuity value is derived using $\|v_1\| = \|v_2\|$, i.e. the velocity is constant along the path as in the assumptions.

Assumption 1: the closed loop error system is ISS such that (4.48) and (4.53) hold.

Theorem 4.2. *Let Assumption 1 holds, let $e_{max} \in \mathbb{R} > 0$. Let:*

$$\bar{e} \triangleq \underline{\beta}^{-1}\left(\overline{\beta}\left(\chi\left(\|u\|_\infty\right)\right)\right) \tag{4.68}$$

and let

$$v^* \triangleq \frac{\overline{\beta}^{-1}\left(V_{max}\right) - \underline{\beta}^{-1}\left(V_{max}e^{-\bar{\gamma}T}\right)}{\sqrt{2(1-\cos(\alpha_{max}))}} \tag{4.69}$$

with $V_{max} = \underline{\beta}(\|e_{max}\|)$, $\forall\ e_{max} \geq \bar{e}$ and $\forall\ v \leq v^*$ and $\forall\ e_0 = e(0) : V(e_0) \leq V_{max}$ then

$$\|e(t)\| \leq e_{max} \quad \forall t > 0 \tag{4.70}$$

Proof. V_{max} is the maximum value of the Lyapunov function such that:

$$V(t) < V_{max} \Leftrightarrow \|e(t)\| \leq e_{max}, \quad \forall t > 0 \tag{4.71}$$

Every time a new direction is picked (new waypoint), the discontinuity in velocity $\Delta_v = \left(\sqrt{2v^2(1-\cos(\alpha_{max}))}\right)$ increases the Lyapunov function value. For iterability we want that before a new direction is taken, the Lyapunov value decreases up to the value V_{min}, such that after the discontinuity the Lyapunov function doesn't exceed V_{max}. $V_{min} = \underline{\beta}\left(\overline{\beta}^{-1}(V_{max}) - \Delta_v\right)$.
Equation (4.53) guarantees that:

$$V(t) = V(0)e^{-\bar{\gamma}t} \quad \forall V > V_1 \tag{4.72}$$

Hence the following inequality constraint guarantees that after the discontinuity, the value of Lyapunov function decreases at least to V_{min} in time T:

$$V_{max}e^{-\bar{\gamma}T} \leq V_{min} \tag{4.73}$$

This results in:

$$V_{max}e^{-\bar{\gamma}T} \leq \underline{\beta}\left(\overline{\beta}^{-1}(V_{max}) - \sqrt{2v^2(1-cos(\alpha_{max}))}\right) \tag{4.74}$$

which leads to (4.69). A minimum tracking error $\bar{e}$ is due to the disturbance (input u) and the ISS property. The minimum tracking error corresponds to the asymptotic gain of the system.

□

4.2.4 Application to VTOL and Car-like Robot

For the UAV we consider only the position dynamic in a $x-y$ plane from (2.1). The position dynamics is given by:

$$M\ddot{p}(t) = -TRe_3 + Mge_3 \tag{4.75}$$

with state of the system $x = [p \quad \dot{p}]^T \in \mathbb{R}^4$, and where $M \in \mathbb{R} > 0$ is the mass of the vehicle, $p(t) = [x(t) \quad y(t)]^T \in \mathbb{R}^2$ the vector of position in inertial frame $\mathcal{F}_i$, T the thrust input, R the rotation matrix that rotates a vector from body frame $\mathcal{F}_b$ to inertial frame $\mathcal{F}_i$ and g the gravity acceleration.

A position reference trajectory for this model is:

$$p_R(t) = \begin{bmatrix} x_R(t) \\ y_R(t) \end{bmatrix} \tag{4.76}$$

We consider the dynamic model of VTOL in (4.75) disturbed with a force $w = [w_x \quad w_y]^T$ (it can model the non perfect tracking of the attitude, aerodynamic disturbances or wind disturbances) such that $\frac{\|w\|_\infty}{M} \leq \bar{w}$, obtaining:

$$M\ddot{p}(t) = -TRe_3 + Mge_3 + w \tag{4.77}$$

If $-TRe_3$ is consider as an input $u = [u_x \quad u_y]^T$, such that $T = \|u\|$ and $Re_3 = \frac{u}{\|u\|}$ the system (4.77) could be rewritten as:

$$\begin{aligned} M\ddot{x} &= u_x + w_x \\ M\ddot{y} &= u_y + w_y \end{aligned} \tag{4.78}$$

The system is described by two separate dynamics. In particular the dynamics are double integrators with a disturbance w. With a position reference trajectory as in (4.76) one can rewrite the system in error coordinates $e = \begin{bmatrix} e_x & e_y \end{bmatrix}^T = p - p_R$ as:

$$\begin{aligned} \ddot{e}_x &= \frac{u_x}{M} + \frac{w_x}{M} - \ddot{x}_R \\ \ddot{e}_y &= \frac{u_y}{M} + \frac{w_y}{M} - \ddot{y}_R \end{aligned} \tag{4.79}$$

The state vector describing the system is $e = [e_x \quad \dot{e}_x \quad e_y \quad \dot{e}_y]^T$ with input $u = [u_x \quad u_y]^T$ and disturbance $w = [w_x \quad w_y]^T$.

Theorem 4.3. *Let $a \in \mathbb{R} > 0$, let P a symmetric and positive definite matrix solution of the*

Riccati equation $A^TP + 2MPBB^TP + PA = -aI_4$, where:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 \\ \frac{1}{M} & 0 \\ 0 & 0 \\ 0 & \frac{1}{M} \end{bmatrix} \tag{4.80}$$

then the input

$$u = -MB^TPe + M \begin{bmatrix} \ddot{x}_R & \ddot{y}_R \end{bmatrix}^T \tag{4.81}$$

makes the system (4.79) ISS stable with asymptotic gain

$$\Psi(\bar{w}) = \frac{2\bar{P}\bar{w}}{(a-\epsilon)} \sqrt{\frac{\overline{\lambda}_P}{\underline{\lambda}_P}} \tag{4.82}$$

with ν an arbitrary positive scalar, $\bar{P}$ the Euclidean norm of matrix PB and $\overline{\lambda}_P$ and $\underline{\lambda}_P$ the largest and smallest eigenvalue of P respectively.

Proof. Let's consider a Lyapunov function $V = e^TPe$. Its derivative becomes

$$\dot{V} = e^T(A^TP + PA + 2MPBB^TP)e + w^TB^TPe + e^TPBw \tag{4.83}$$

and

$$\dot{V} \leq -a \|e\|^2 + 2\bar{w} \|PB\| \|e\| \tag{4.84}$$

Choosing an $\epsilon \in (0, a)$, we can define

$$\chi(\bar{w}) = \frac{2}{a-\epsilon}\bar{P}\bar{w} \tag{4.85}$$

such that

$$\|e\| \geq \chi(\bar{w}) \quad \Rightarrow \quad \dot{V} \leq -\epsilon \|e\|^2 \tag{4.86}$$

□

Remark 4.4. With the control law at hand we can define equations (4.48) and (4.54) for

this specific system. In particular:

$$\begin{aligned} \overline{\beta}(\|e\|) &= \overline{\lambda}_P \|e\|^2 \\ \underline{\beta}(\|e\|) &= \underline{\lambda}_P \|e\|^2 \\ \Psi(\bar{w}) &= \frac{2\bar{P}\bar{w}}{(a-\epsilon)}\sqrt{\frac{\overline{\lambda}_P}{\underline{\lambda}_P}} \\ \bar{\gamma} &= \frac{\epsilon}{\overline{\lambda}_P} \end{aligned} \tag{4.87}$$

with $\overline{\lambda}_P$ and $\underline{\lambda}_P$ the largest and smallest eigenvalue of P respectively, due to Comparison Lemma [61]. △

For the car-like robots, the results in Chapter 2 are enough to be applied to this problem

As a real application we want to derive a simple strategy to navigate the VTOL and car-like robot in a forest. Let's consider the problem of navigating a robot in a forest environment with a strategy that exploits the tools from previous section. We consider that the navigation is performed by a closed loop controlled robot. In the "forest" environment, the obstacles in $x-y$ plane are represented by "trees" or circular obstacles. The forest has constraints on his geometrical structure, and in particular

Assumption 3:

- obstacles has a maximum radius of R.
- the minimum distance between two obstacle's center is $\bar{p}$.
- $\bar{p}$ is smaller than the range of exteroceptive sensors that detect obstacles.

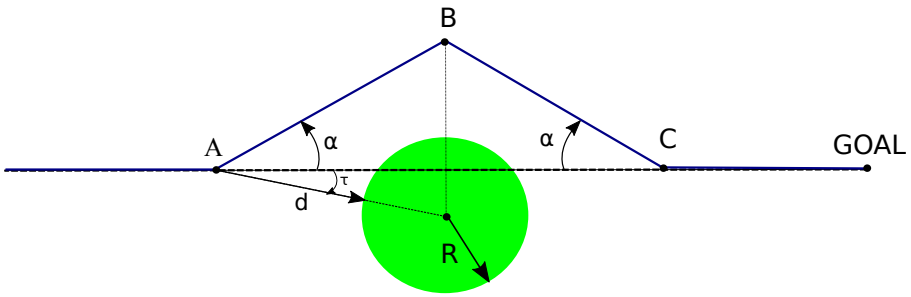

Figure 4.4: Environment and avoid strategy.

We first describe the avoid strategy for a single obstacle (tree) to extend later the problem to the whole forest.

Avoid Strategy Description: Single Obstacle

The environment and the avoid strategy are represented in Figure 4.4. The robot is entering the environment with linear speed v. The obstacle (green circle) can be found in a direction that differs τ rad from the trajectory that connects the robot to the goal. The avoid strategy generates an intermediate waypoint (B) in a direction that differs α rad from the direction connecting the goal. The waypoint is generated when the obstacle is found at distance d from the robot. For sake of clarity, we define a frame centered in waypoint (A) when the the avoid maneuver starts, with axis (x, y), with x axis directed toward the goal, and y axis perpendicular to x axis. The coordinates of waypoint (B) are given by:

$$\begin{bmatrix} x_B \\ y_B \end{bmatrix} = \begin{bmatrix} (d+R)\cos(\tau) \\ (d+R)\cos(\tau)\tan(\alpha) \end{bmatrix} \tag{4.88}$$

Another piece of trajectory connects waypoint (B) to waypoint (C) as in figure and then to the goal (GOAL). The coordinates of waypoint (C) are given by:

$$\begin{bmatrix} x_C \\ y_C \end{bmatrix} = \begin{bmatrix} 2(d+R)\cos(\tau) \\ 0 \end{bmatrix} \tag{4.89}$$

We consider the problem only with $\alpha \geq 0$ and $\tau \geq 0$, since for negative values the problem is symmetric w.r.t x axis. The goal of the sense and avoid strategy is to dodge the obstacle and minimize the traveling time to reach the goal by selecting the parameters α, v, i.e. the deviation angle and the traveling speed in the forest. Meanwhile the strategy needs to guarantee that the closed loop system tracking the generated trajectory does not collide with obstacles. To avoid recomputing optimal parameters for every obstacle found, we suppose to adopt the same strategy for every obstacle, i.e. the same velocity v, the same direction angle α and at same distance d. Because of this assumption:

- The optimization problem needs to be solved only once (offline too).
- The planning strategy becomes very simple and the computation fast (as soon as an obstacle is detected, two waypoints are generated).

Hypothesis:

- We consider the problem with $\tau = 0$ that corresponds to the worst case, since the clearance is smaller.
- Closed Loop conditions: the closed loop error system satisfies (4.48) and (4.53).

Optimization Problem

The problem of navigation in a forest with the avoid strategy described in previous subsection while minimizing the traveling time could be described as an optimization problem.
The optimization problem is the following:

$$\begin{aligned} \min_{\alpha,v} \quad & T(\alpha, v) \\ \text{subject to :} \quad & 0 < v < v'(\alpha, v) \\ & e_{max} > \bar{e} \end{aligned} \tag{4.90}$$

with:

$$v'(\alpha, v) = \frac{\overline{\beta}^{-1}\left(\underline{\beta}(\|e_{max}\|)\right) - \underline{\beta}^{-1}\left(\underline{\beta}(\|e_{max}\|)e^{-\bar{\gamma}T}\right)}{\sqrt{2(1-\cos(2\alpha))}} \tag{4.91}$$

with

$$e_{max} = (d+R)\sin(\alpha) - R \tag{4.92}$$

and

$$T = \frac{d+R}{\cos(\alpha)v} \tag{4.93}$$

following (4.69) and the specific geometry of the problem. T represent the time of the avoiding maneuver, in particular it represents the time to travel the segments $\overline{AB} = \overline{BC}$.

First constraint imposes an upper bound to the traveling velocity v to ensure a maximum tracking error following Theorem 4.2, so that the tracking error is smaller than the clearance and no collision is guaranteed. Second constraint imposes a lower bound on α due to the minimum tracking error because of disturbances.

Combining (4.90) with (4.92), we obtain:

$$\begin{aligned} \max_{\alpha,v} \quad & \cos(\alpha)v \\ \text{subject to :} \quad & \psi(\alpha, v) \geq 0 \\ & \alpha_{MIN} \leq \alpha < \tfrac{\pi}{2} \end{aligned} \tag{4.94}$$

with $\psi(\alpha, v) = v' - v$, and $\alpha_{MIN} = \sin^{-1}\left(\frac{\bar{e}+R}{d+R}\right)$. We imposed an upper bound to α of $\pi/2$ since for $\alpha = \pi/2 \;\Rightarrow\; T = \infty$.

Lemma 4.3. *$\forall\, \alpha \in \left[\alpha_{MIN},\ \frac{\pi}{2}\right)$, there exist $\underline{v} > 0$ such that:*

$$\forall\, v : 0 < v < \underline{v} \;\Rightarrow\; \psi(\alpha, v) > 0 \tag{4.95}$$

Proof.

From (4.91) and (4.93), for $v \to 0$, $\psi(\alpha, v) > 0$, since the exponential of (4.91) tends to zero, leaving v' a positive number.

□

Lemma 4.4. *There exist $\overline{v} > 0$ such that:*

$$\forall\, \alpha \in \left[\alpha_{MIN},\ \frac{\pi}{2}\right),\ \forall\, v : v > \overline{v}\ \Rightarrow\ \psi(\alpha, v) < 0 \tag{4.96}$$

Proof. From (4.91) and (4.93), for $v \to \infty$, $\psi(\alpha, v) < 0$, since v' tends to a finite value while v tends to infinite.

□

First Lemma affirms that a solution to the problem (4.94) always exists, while Lemma 2 gives us a constraint on the maximum value of v. Since objective function and constraints are continuous functions and both variables are bounded, it is possible to find a solution with global optimization. A complete (enumerative) strategy could be used to find a solution with arbitrary tolerance.

Avoid Strategy Description: Forest

With respect to the previous problem of avoiding a single obstacle, we now want to investigate how to robustly avoid every obstacle in the forest. An additional constraint on the distance d needs to be added to ensure that no obstacles are hit during the avoid maneuvers.

Assumption 3: The deviation angle α is fixed to a value $\bar{\alpha}$ and chosen as in the single obstacle case following (4.94) with $d = \bar{p}/2$.

Lemma 4.5. *The problem (4.94) with Assumption 3 holding and with d such that:*

$$d \leq \min\{d_1, d_2, d_3\} \tag{4.97}$$

with

$$\begin{aligned} d_1 &= \tfrac{\bar{p}}{2} - R \\ d_2 &= \tfrac{\bar{p}}{\tan\bar{\alpha}+\sin\bar{\alpha}} - R \\ d_3 &= \tfrac{\bar{p}}{1+\sin\bar{\alpha}} - R \end{aligned} \tag{4.98}$$

guarantees that no obstacles are hit in a forest with characteristics described as in section Assumption 3.

Proof. With respect to the single obstacle problem, we need to guarantee that the robots does not collide with others obstacles while avoiding one obstacle and that the avoidance maneuver is repeatable. For the repeatability, we have to guarantee that the robot reaches waypoint C before a new obstacle is found at distance d. This means that d should be taken such that:

$$2(d + R) \leq \bar{p} \tag{4.99}$$

which leads to the first condition of (4.98). To guarantee that the robot never collides with other obstacles while doing the avoidance maneuver, d should be chosen such that the avoidance maneuver is inside the circle centered in the obstacle center and with radius $\bar{p} - R - e_{max}$ (see Figure 4.5 the grey circle), with e_{max} as in (4.92). This gives two constraint:

$$(d + R)\tan(\bar{\alpha}) \leq \bar{p} - R - e_{max} \tag{4.100}$$

which leads to second condition of (4.98), and:

$$(d + R) \leq \bar{p} - R - e_{max} \tag{4.101}$$

which leads to third condition of (4.98).

□

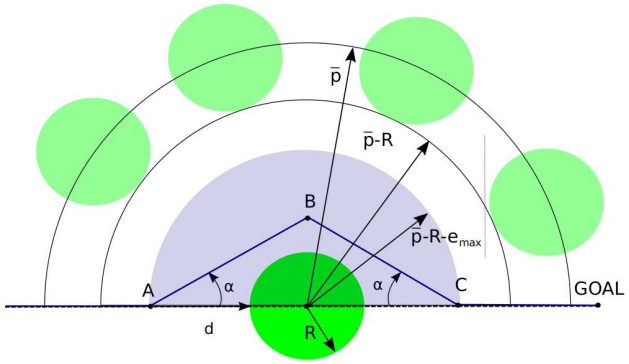

Figure 4.5: Forest geometry. The generated trajectory should stay in the grey circle to be sure that other obstacles are avoided during the maneuver.

Stochastic Forest

In this section, we consider a stochastic forest, and we compute how many obstacle avoidance maneuvers the robot would execute per unit time, on average.

To model the planar forest environment, we utilize a hard-core planar stochastic

point process. Roughly speaking, a stochastic point processes is a collection of randomly-placed points in an Euclidean space. A planar stochastic point process is one where the points are located on the infinite plane ($\mathbb{R}^2$). A hard-core stochastic point process is one where every pair points are spaced at least some distance p apart. Hence, the class of hard-core planar stochastic point process forms a suitable class of models for the planar forest environment which we consider in this paper. In fact, this class of point processes is widely used in the forestry literature [108].

One of the widely-studied hard-core point processes is called the *Matérn process* [27], defined as follows. Let Φ be a Poisson point process on the infinite plane with parameter λ.[1] Delete any two points $x, x' \in \Phi$ of the Euclidean distance between x and x' are smaller than p, *i.e.*, $\|x - x'\| \leq p$. The remaining set of points is called the Matérn point process with parameters λ and p. It is easily verified that any two points from a Matérn point process are at least a distance of p apart from one another.

Our main result of this section stated below exactly quantifies the expected number of maneuvers that the robot will execute while traversing a certain stochastic forest.

Theorem 4.4. *Let us consider a planar forest environment, where the locations of the trees are determined by a Matérn point process with parameters λ and p, and each tree has radius r. Let K_l denote the number of times that the robot has to maneuver if it attempts to travel on a straight path through this forest.*[2] *Then, the following holds:*

$$\mathbb{E}[K_l] = 2\, r\, l\, \lambda\, e^{-\lambda\, \pi\, p^2}.$$

Before providing the proof of this result, we recall an important recent result from the stochastic point process literature.

Theorem 4.5 (See Theorem 3 in [55])**.** *Let Φ be a Poisson point process with parameter λ in the d-dimensional Euclidean space, and delete each point $x \in \Phi$ with probability:*

$$p_0 \prod_{x' \in \Phi, x' \neq x} (1 - f(\|x - x'\|)),$$

where $p_0 \in (0, 1]$ and $f : [0, \infty) \to [0, 1]$ is a measurable function. Then, the intensity of the remaining points is

$$\lambda' = \lambda\, p_0 \exp\left(-\lambda d b_0 \int_0^\infty f(z) z^{d-1} dz\right),$$

[1] The reader is referred to the book by Stoyan et al. [27] for a definition of the Poisson point process. One way to construct a Poisson point process is to place uniformly at random N_A number of points in each bounded partition $A \subset \mathbb{R}^2$, where N_A is a Poisson random variable with parameter equal to the area of A times λ. One important property of the Poisson point process with parameter is that the intensity of the points is λ, hence the expected number of points in a region with area B is $\lambda\, B$.

[2] The starting point and the direction of travel are irrelevant as long as they are both chosen independently of the point process.

where b_0 is the volume of the unit ball in the d-dimensional Euclidean space.

Notice that we recover the Matérn process when $p_0 = 1$ and

$$f(z) = \begin{cases} 1, & \text{if } z \leq p; \\ 0, & \text{otherwise.} \end{cases}$$

Proof (of Theorem 4.4) Suppose the robot attempts to travel the straight path of distance of l. It must maneuver whenever there is a tree on its path, hence the center of the tree is within the rectangle of length l and width $2r$, where r is the tree radius. Plugging in this information inot the formula in Theorem 4.5 gives the result. □

Simulations

Two simulations were run to test the behavior with the proposed method. First simulation was done with the VTOL model, while second simulation with differential wheel robot. Both were simulated with the closed loop control law as defined previously.

VTOL

A forest is randomly generated with parameters chosen as: $\bar{p} = 4.5m$, $R = 0.4m$. A forest of 70×70 meters was simulated, with 200 obstacles generated with uniform distribution. We consider a VTOL with the following parameters: $M = 1Kg$ and $R_{VTOL} = 0.2m$, where R_{VTOL} is the radius of the VTOL, so that the trees are considered expanded of R_{VTOL} to avoid collisions. The parameter d is chosen following (4.97) as $d = 1.65$ while $\bar{\alpha} = 0.6981\ rad$. The control law is chosen with $a = 21$ and $\epsilon = 20$, resulting in a $\bar{\gamma} = 0.83$. The disturbance was chosen as $\bar{w} = 0.1N$ resulting in a $\Psi = 0.13m$. The obstacles were expanded by the value of Ψ to take into account the minimum tracking error given by the disturbance w. From the optimization problem it results a maximum traveling speed of $v = 0.75m/s$. The results could be seen in Figure 4.6. The green solid circles are the obstacles (trees) while the dashed green circles are the obstacles expanded by the radius of the VTOL and Ψ. The red cross is the starting point and the blue cross is the goal. The trajectory in blue is the generated trajectory while in dashed black it is plotted the real position of the VTOL tracking the trajectory. The black squares are the bounding box of the VTOL along the trajectory. As we can notice, the discontinuity in the trajectory causes a non-perfect tracking, but it is still under a threshold that ensure no obstacles are hit. In Figure 4.7 a zoom of the trajectory.

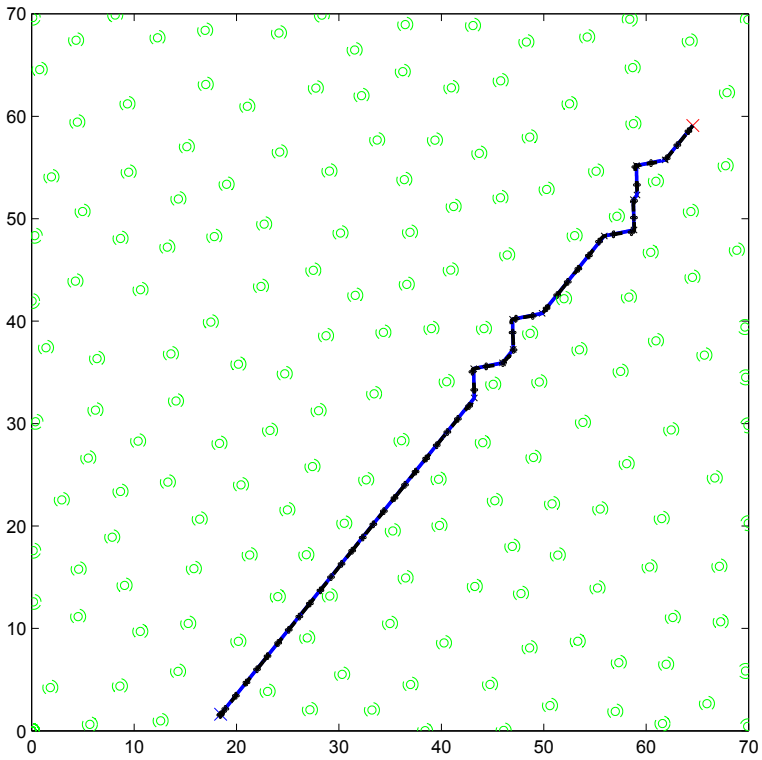

Figure 4.6: Forest simulation with VTOL.

Differential Wheel Robot

A forest is randomly generated with parameters chosen as: $\bar{p} = 4.5m$, $R = 0.2m$. A forest of 70×70 meters was simulated, with 200 obstacles generated with uniform distribution. We consider a VTOL with the following parameters: $R_{DWR} = 0.36m$, where R_{DWR} is the radius of the bounding box of the differential wheel robot considered as a rectangle of side $0.6 \times 0.4\ m$, so that the trees are considered expanded of R_{DWR} to avoid collisions. The parameter d is chosen following (4.97) as $d = 1.59$ while $\bar{\alpha} = 0.619\ rad$. The control law is chosen with $K_P = 4.62$ and $K_\theta = 2.38$, resulting in a $\bar{\gamma} = 1.1$. From the optimization problem it results a maximum traveling speed of $v = 1.3m/s$. The results could be seen in Figure 4.8. The green solid circles are the obstacles (trees) while the dashed green circles are the obstacles expanded by the radius of the robot. The red cross is the starting point and the blue cross is the goal. The trajectory in blue is the generated trajectory while in black it is plotted the real position of the VTOL tracking the trajectory. The black rectangle is the robot, with the triangle indicating the forward direction. In Figure 4.9 a zoom of the trajectory.

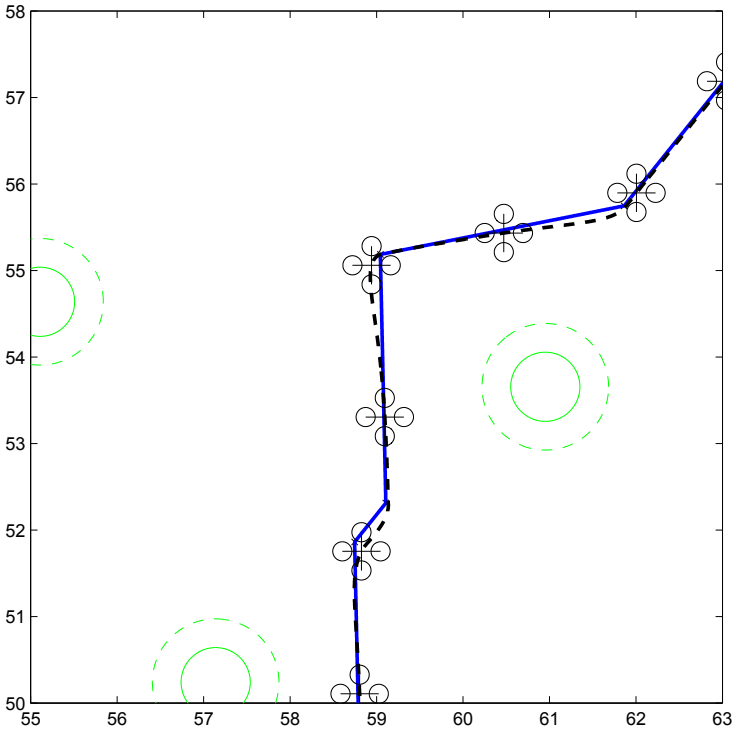

Figure 4.7: Forest simulation with VTOL: zoom.

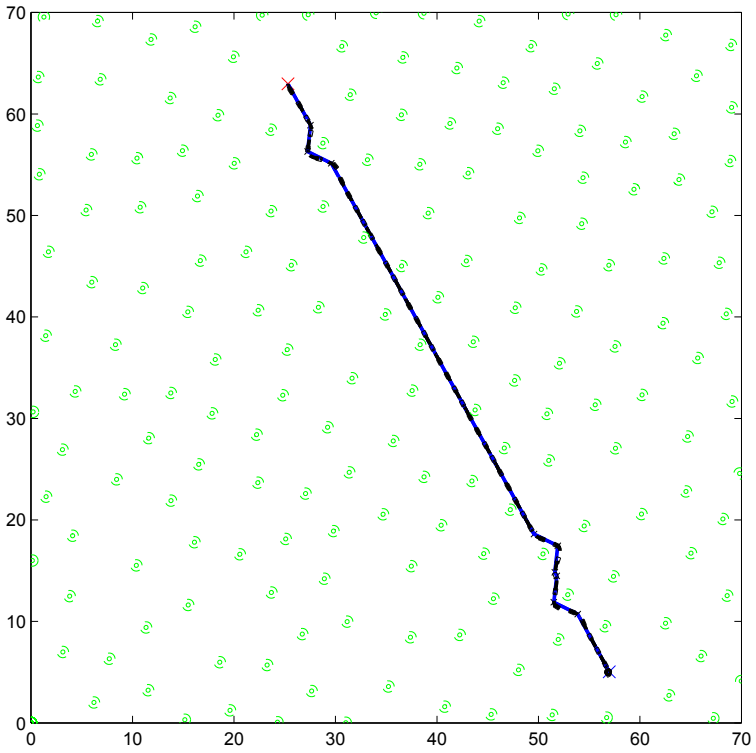

Figure 4.8: Forest simulation with differential wheel robot.

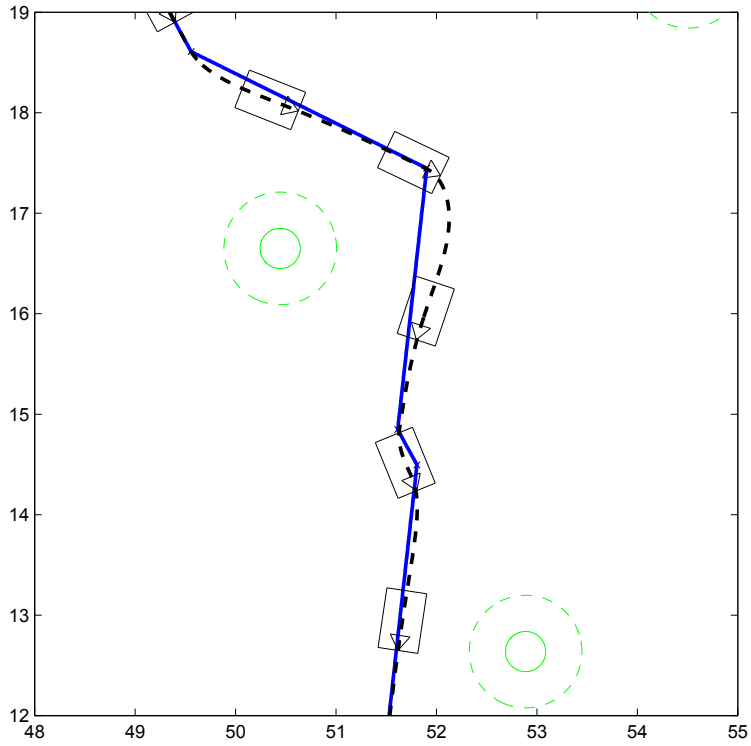

Figure 4.9: Forest simulation with differential wheel robot: zoom.

5

DESP: Discrete Event System Planner

In this chapter we introduce a novel motion planning strategy [39] [40] [41] for heterogeneous mobile robots based on Discrete Event System theory. When real-world populated environments and accurate vehicle's dynamics are taken into account, planning the motion of the vehicle optimally may require to solve a computationally intensive optimization problem [111]. Existing approaches often rely on a discrete description of the environment and of the vehicle dynamics, in order to reduce the admissible region that optimization tools have to explore. In [60] [67] [59], for instance, feasible paths towards a desired target are obtained by sampling a map of the environment. In [11], the planning task has been formulated in terms of languages that can be described formally by means of automata. In [35], the complex nonlinear dynamics of the vehicle is decomposed into a finite number of motion primitives. This approach has also been extended in [102] where stability and robustness of the optimal sequence has been taken into account. In [71] and the path is obtained by connecting a number of dynamically feasible trajectories and fast replanning is allowed by considering incremental search algorithms on lattice. Other lattice-based planner include [98] and [97]. The goal of the proposed approach is to develop a global planner that generates kinodynamic feasible paths for a real-time application. Moreover the algorithm should be capable of

replanning for unknown environments. Taking advantage of a symbolic description of the vehicle dynamics and of the environment, the reference trajectories are generated as sequences of elementary primitives. DES theory is used to model the environment and the dynamics of the robot with automata, and to obtain a so called *Supervisor*. Graph search algorithms are then employed to build the optimal sequence of primitives for the mission.

5.1 DESP: Discrete Event System Planner

In this work DES as well supervisory control theory is utilized to design a motion planner for mobile robotic platforms. To this aim the geographic space (map) is discretized to reduce the computational complexity and the dynamics of the robot are converted in symbolic dynamics using a set of elementary movements (primitives). The primitives are pieces of dynamically-feasible trajectories, that take into account the dynamic of the vehicle. The vehicle is controlled to perform a succession of elementary movements, i.e. the primitives. The single primitives are selected as feasible movements according to the dynamic of the robot, but its interconnection can be kino-dynamically feasible or non-feasible. In the latter case, the theory developed in the previous chapter can be used to derive bounds on the tracking error. The sequence of primitives to be executed is selected in real-time by a high-level supervisor that is designed as a discrete-event system [2]. To decide the optimal sequence of primitives, the supervisor considers an optimization problem in which it takes into account for the current discrete-event model of the environment and of the vehicle. For an introduction to DES and to the tools used in this chapter, the reader can refers to the Appendix section.

The main idea of this approach is to describe both the configuration states (map) for the planning and the dynamics of the robot as automata. The automata are then used to build a *supervisor* for planning. To translate a typically continuous systems into discrete systems as DES some kind of discretizations should be made. Part of the discretization strategies will be discussed in the following.

To define the *supervisor* we need to define 3 automata: Map Automaton, Agent Automaton, Specification Automaton.

5.1.1 Map Automaton

The map automaton is used to describe the "environment" we are interested in. For mobile robots this can be the 2D or 3D space discretized with a certain resolution. For a 2D environment the proposed automaton G_{map} consists of a grid discretization of equal cells with a 4-connectivity grid. The automaton example for a small map discretized by

3×3 states is given in Figure 5.1.

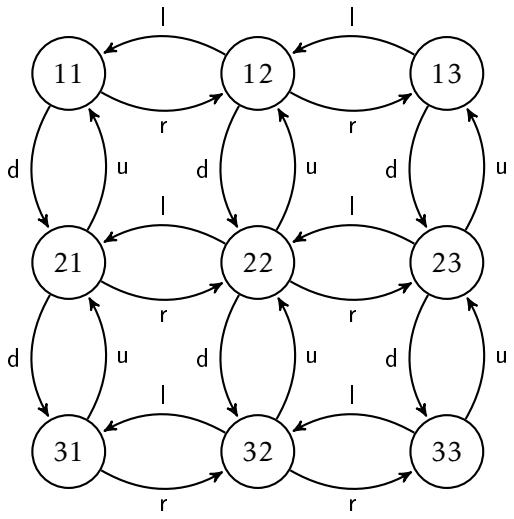

Figure 5.1: Map Automaton Example

The map automaton is defined as: $G_{map} = \{X_{map}, E_{map}, f_{map}, \Gamma_{map}\}$, where the set of states $X_{map} = \{11, 12, 13, 21, ...\}$ represents the quanti of the space with a certain discretization. The set of events $E_{map} = \{r, l, u, d\}$ (right, left, up, down) represents elementary movements on the map and they are not related to robot's motion. At first we suppose that the events E_{map} are controllable. No initial and marked states are present in this automaton. Once the dimension of the map and the discretization level are defined, G_{map} is completely defined and static.

Other kinds of map discretization and connections are possible. However, since not directly related to the robot movements, any other different representation increases the complexity and the number of events for apparently no increased performances, as will be explained later. For example an hexagon discretization could be used, with the cost of having a 6-connection grid and the impossibility to extend it to 3D. Also a 8-connection square grid can be used, but the number of events representing elementary movements would double.

For a natural extension to 3D environments, the same grid can be extended into 3D, with a 6-connection grid and 6 events. Two more events $\{a, b\}$ (above, below) are used to represent elementary movements between planes with different z coordinate as in Figure 5.2.

5.1.2 Specification Automaton

The *Specification Automaton* is a refinement of the *Map Automaton* in which are taken into account the specifications about the goal, the environments constraints and the ini-

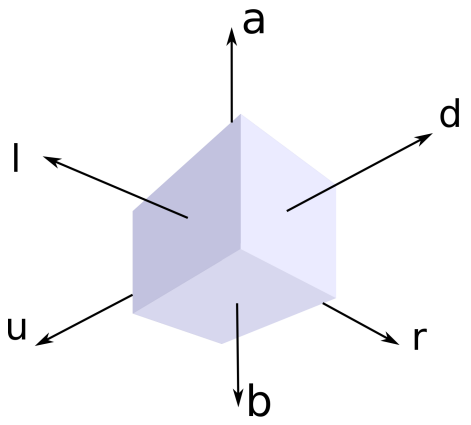

Figure 5.2: 6-connectivity grid for 3D environment.

tial state.
The Specification Automaton is defined as $G_{spec} = \{X_{spec}, E_{spec}, f_{spec}, \Gamma_{spec}, x_0, X_{m,spec}\}$ with $X_{spec} \subseteq X_{map}$, $E_{map} \subseteq E_{spec}$. The state of the Specification Automaton is a subset of the Map Automaton, since the environment constraints X_{fb} are given in terms of forbidden states. The forbidden states (and its connections) are simply removed from the Map Automaton. The initial state x_0 is set based on the real state of the robot (for example it's position on the map). The marked states $X_{m,spec}$ are set as the states of interest we want to reach. They represents the X_f. An example of Specification automaton is given in Figure 5.3.

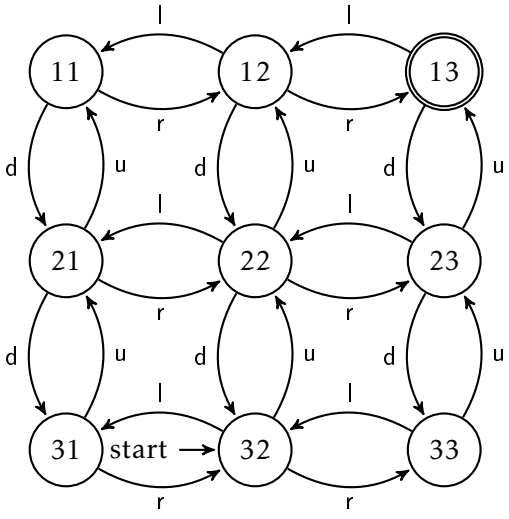

Figure 5.3: Specification Automaton example with no forbidden states.

In the example we can see how the initial state of the robot was set at position 32 and the goal was set at 13. The set of marked states can of course be of any dimension if

we are interested in reaching one of those states indifferently. In the following example in Figure 5.4 we consider two obstacles or forbidden states at 11 and 23. As we can notice, the interested states and its transitions are removed accordingly to embed the new specifications.

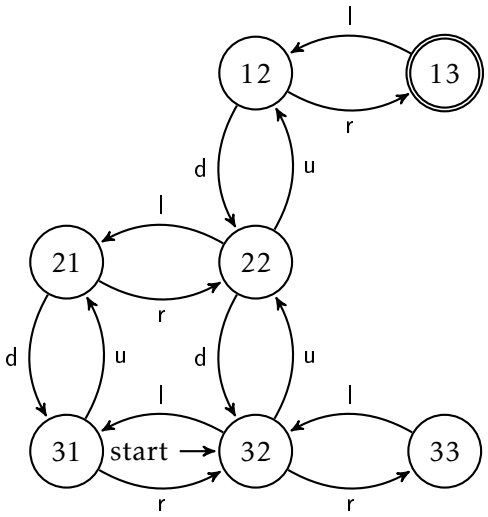

Figure 5.4: Specification Automaton example with forbidden states.

The Specification Automaton is dynamic by nature. In fact, as soon as new information about the environment (such as new obstacles) are gathered or the desired goal(s) changes, the *Specification Automaton* changes its properties.

5.1.3 Agent Automaton

Finally, the *Agent Automaton* is the automaton that describes the set of possible primitives the robot can execute, its logic about how the primitives can be interconnected, and the interaction of the primitives with the map. For instance if the robot we want to model has two primitives $\{go_R, go_U\}$ consistingS of linear trajectories as in Figure 5.5, and the robot can execute those primitives in any order, the agent automaton $G_{ag} = \{X_{ag}, E_{ag}, f_{ag}, \Gamma_{ag}, x_{0,ag}, X_{m,ag}\}$ is given in Figure 5.6.

The event set E_{ag} of the Agent Automaton includes always the event set of the map E_{map} and an event set of the motion primitives of the agent that in this case corresponds to $\{go_U, go_R\}$ obtaining $E_{ag} = \{go_U, go_R, l, r, u, d\}$. What the automaton is describing in addition to the interconnection logic is the effect of those primitives in the map as sequence of elementary movements. We can see in fact how a go_U implies a u elementary movement, while go_R implies a r elementary movement. The marked states $X_{m,ag}$ represent some states of interest. In this simple example the only state of interest is the initial state, in which the robot is ready to execute any primitive. However in a more

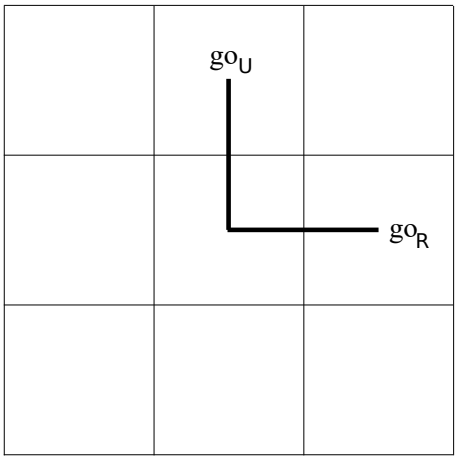

Figure 5.5: Primitives of the Agent Example.

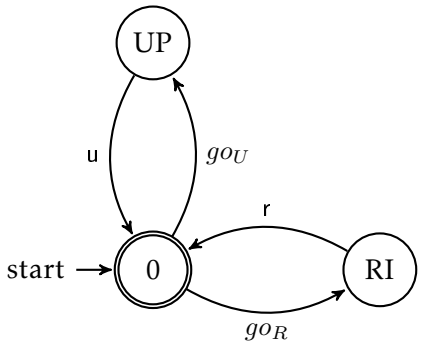

Figure 5.6: Agent Automaton example. The robot can only move up and right with simple linear primitives.

complex model we can consider particular states that represent for example a specific sequence of primitives.

We can decide to model a different agent with the same set of primitives but with a different logic for interconnection. For instance modifying the previous example such that the robot has to execute a go_R primitive after each go_U primitive, the resulting *Agent Automaton* is given by Figure 5.7. A possible valid language becomes for example $\{go_R, go_R, go_U, go_R, ...\}$.

This new automaton embeds the property of the allowed primitives sequence, such that after a go_U, only a go_R is possible, and at the same times it embeds the interaction of the primitive with map's elementary movements. We can see in fact that right after a go_R, only a r event can occur, and after a go_U, only a u event can occur. In this automaton, the marked states are $\{0, UP0\}$, since we want to reach the goal with any sequence of primitives. If instead we want to reach the goal for example with go_U as last primitive, it is enough to mark just the $\{UP0\}$ state.

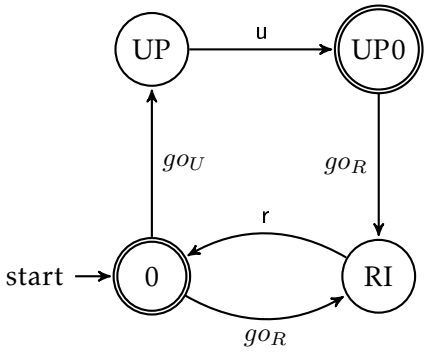

Figure 5.7: Agent Automaton example. The robot can only move up and right with simple linear primitives. After each up movement the robot can only move right.

In most of the cases the *Agent Automaton* can be considered static, but nothing forbids to have a dynamic automaton. For instance, if a fault is detected in a robot's actuator, the controllability of the robot changes the set of executable primitives which can be taken into account in a new *Agent Automaton*.

For more complex behaviors of the robot and if many primitives are present, it may results difficult to design the complete Agent Automaton. But a particular procedure as described below can help the designer.

Agent Primitives Logic and Agent-Map Interactions

By using the parallel composition, one of the most powerful tool for automata, we can compose smaller automata that describe simple logics and languages into bigger automata. The idea is that the Agent Automaton has to include both the primitives interconnection logic and the interaction with the map. Because of this we can think of computing a single automaton called Agent Primitives Logic Automaton (APLA) $G_{apl} = \{X_{apl}, E_{apl}, f_{apl}, \Gamma_{apl}, x_{0,apl}, X_{m,apl}\}$ that embeds the primitives logic and N Agent-Map Interaction Automata (AMIA) $G_{ami,i} = \{X_{ami}, E_{ami}, f_{ami}, \Gamma_{ami}, x_{0,ami}, X_{m,ami}\}$ that embeds how the $i-th$ primitive reflects into base movements on the map. $i \in \mathbb{N}$ is the index of the $i-th$ AMIA and N is the number of primitives. The Agent Automaton is then defined as:

$$G_{ag} = G_{apl}||G_{ami,1}||G_{ami,2}||...||G_{ami,N} \tag{5.1}$$

Both APLA and AMIA should be built such that $E_{apl} = E_{ami,i} = E_{ag}$.

Let's make an example. Suppose that the set of primitives of the robot are defined as in Figure 5.8.

The primitives represent linear segments in 4 directions $\{U, D, R, L\}$ and 8 curves $\{UL, UR, DL, DR, RU, RD, LU, LD\}$. Suppose that to make a smoother path we want

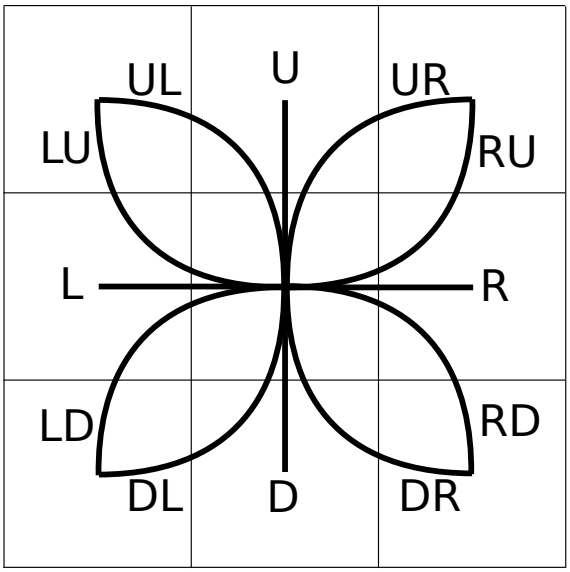

Figure 5.8: Primitives set of the Agent. 12 movements are possible, with 4 straight lines and 8 curves.

that after a primitive, the next primitive should have the initial direction as the last direction of the previous primitive. Some valid sequence of primitives are U, UR, RD, D or UR, R, RD, DR, R as in Figure 5.9.

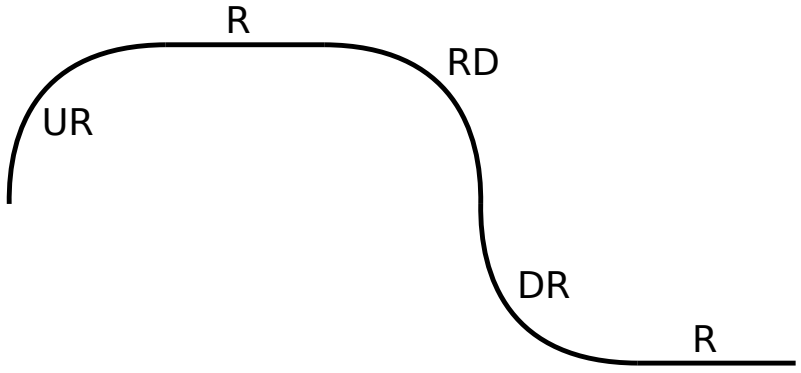

Figure 5.9: Example of valid sequence of primitives for the agent.

So we build first the APLA embedding this language as in Figure 5.10.

Then we can build 12 AMIA, one for each primitive, to model the interaction with map base movements as depicted in Figure 5.11, Figure 5.12 and Figure 5.13. As we can see, the APLA only embeds the logic of how the primitives can be interconnected, while single AMIA embeds how each single primitive reflects its movements onto the map. For example the primitive UL in Figure 5.12 impies the sequence of basic movements u, l in the map.

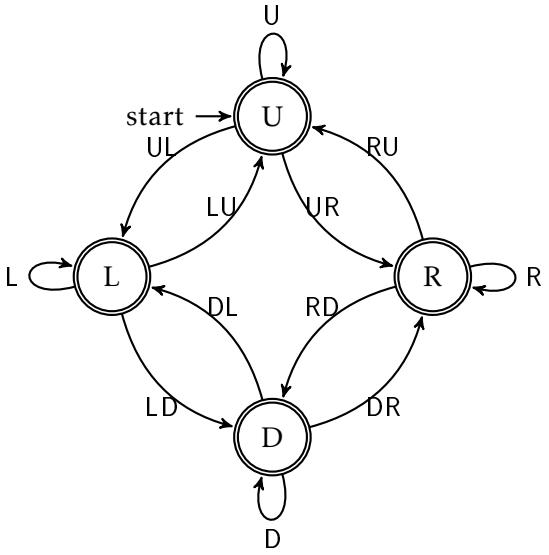

Figure 5.10: APLA example. The final and initial direction of two consecutive primitives is continuous with this logic.

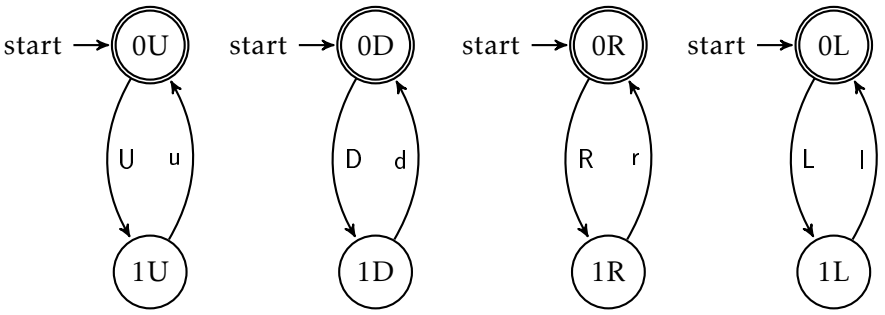

Figure 5.11: Agent Automaton example: First set of 4 AMIA.

5.1.4 Swath and Collision Checking

It is well known that the collision checking algorithms is one of the most computationally expensive task in motion planning, although not often discussed. With the proposed approach the collision checking won't be needed because automatically embedded into the *Supervisor* if the AMIA are build in a certain way. Suppose we have a particular primitive as in Figure 5.14, and the grey cells are the pre-computed trajectory swath, i.e. the cells occupied by the robot's footprint during the maneuver.

If we build the AMIA for the particular trajectory such that it traverses all the cells of the swath, then when the *Supervisor* is built (as described below), it will automatically include the collision information. In the example of Figure 5.14, the particular AMIA can be built such as in Figure 5.16. We can notice how the sequence of elementary movements on the map "covers" all the swath cells. In Figure 5.15 instead we see the swath

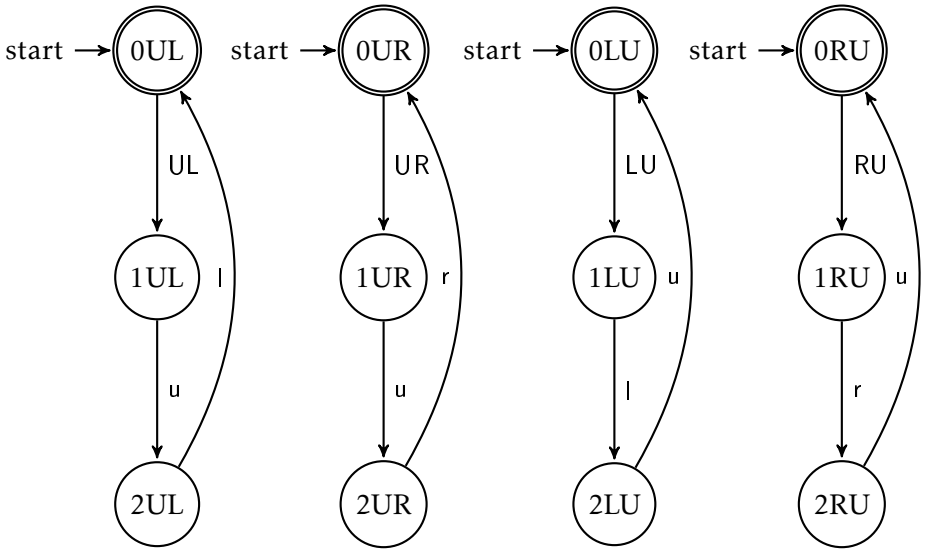

Figure 5.12: Agent Automaton example: Second set of 4 AMIA.

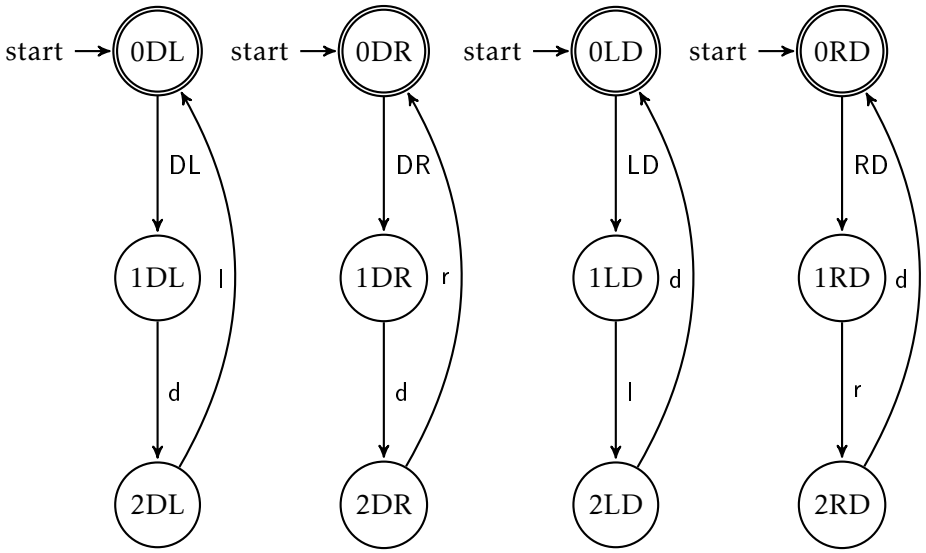

Figure 5.13: Agent Automaton example: Third set of 4 AMIA.

for the robust case, in which we are able to compute a maximum tracking error due to disturbances, uncertainties and discontinuities as discussed in the previous chapter. The number of grey cells is higher because the robot can potentially lie in the dotted area.

This approach is valid for primitive-invariant swath, i.e. for all the primitives that keep the same footprint independently to the previous primitives and state of the system. This is valid for mostly of the mobile robots, but it is not true for example for manipulators, where a linear segment primitive can results in different links configurations in the space hence different swath. In general, this is not valid for multi-body

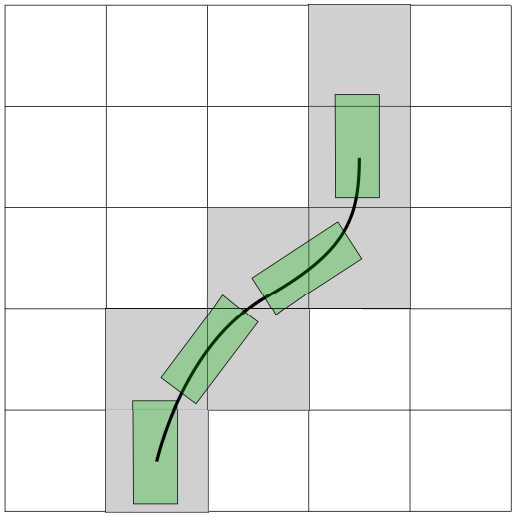

Figure 5.14: Example of swath of a particular primitive.

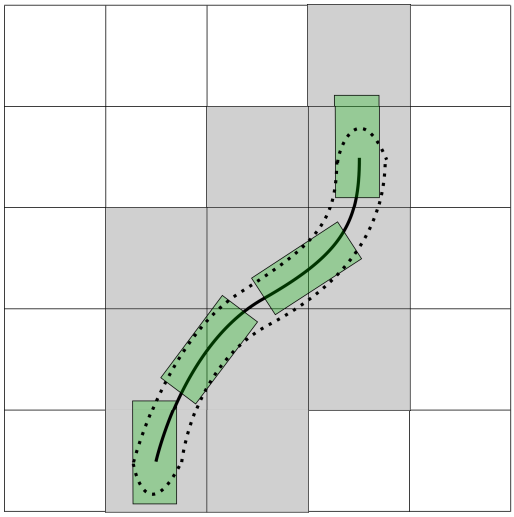

Figure 5.15: Example of swath of a particular primitive with robustness taken into account. The dashed area around the planned trajectory represents the possible position error w.r.t. the trajectory due to disturbances.

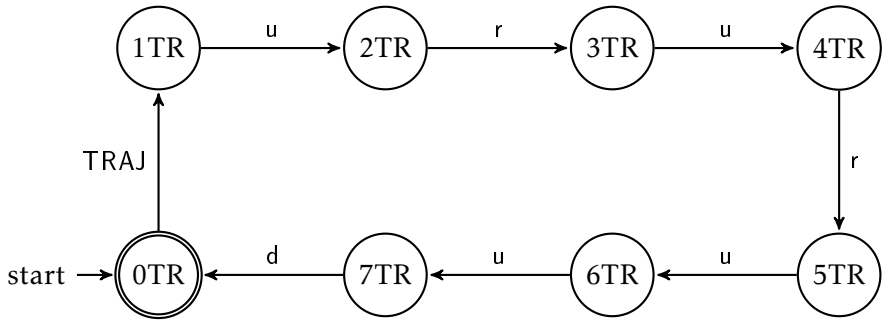

Figure 5.16: AMIA example for a particular primitive. All the elementary movements in the map should cover all the swath cells of the trajectory and at the same time reach the final cell of the trajectory. We can notice the final $u - d$ elementary movements of the chain to "reach" the top grey cell (1st row, 4th column) and then move down to the final cell (2nd row, 4th column).

robots, where a certain primitive can have a different swath based on previous primitives or different states of the system, but with a suitable choices of APLA it can be adapted to certain classes.

5.1.5 Supervisor and Reachability Graph

Finally, with the *Agent Automaton* and the *Specification Automaton* it is possible to compute the *Supervisor*. The supervisor $G_{sup} = \{X_{sup}, E_{sup}, f_{sup}, \Gamma_{sup}, x_{0,sup}, X_{m,sup}\}$ in case of fully observable and controllable events, is defined as:

$$G_{sup} = G_{spec}||G_{ag} \tag{5.2}$$

The G_{sup} obtained represents in practice a reachability graph that embeds both the specifications on the environment (described by the *Specification Automaton*) and the agent constraints (described by the *Agent Automaton*). Because of how the automata are constructed and the properties of parallel composition, the *Supervisor* language will include all the feasible sequence of primitives, excluding the forbidden ones not respecting the agent automaton logic and colliding with obstacles. By analyzing the Supervisor graph, it is possible to check directly whether or not the "mission" is feasible. In fact if $X_{m,sup} = \{\}$, i.e. the marked set of states is empty, there is no sequence of primitives to reach the goal state(s). If $X_{m,sup}$ is non-empty, then one or more possible sequence of primitives can be the solution of the planning problem. In general all the states X_{sup} are reachable with some sequence of primitives, and any possible sequence on the graph won't collide with obstacles. If more than one solution exist to reach the marked states, then some weighted oriented graph search algorithms can be used to find some *optimal* solution [28], [48], [62], [107]. In particular it is possible to add a cost to every event representing a primitive based on some cost function and by assigning a cost of zero to elementary movements $\{l, r, u, d\}$. By continuing the previous example, the *Supervisor* for the Specification Automaton in Figure 5.3 and the *Agent Automaton* in Figure 5.6 can be seen in Figure 5.17.

As we can see, because of the nature of the agent being capable to move only up and right we can reach only states $\{12, 13, 22, 23, 32, 33\}$ in the map. In this example multiple solutions to the problem are possible. In particular the three primitives sequence to reach the marked state are: $\{go_U, go_U, go_R\}$, $\{go_R, go_U, go_U\}$, $\{go_U, go_R, go_U\}$. If we consider the same *Agent Automaton* but the *Specification Automaton* as in Figure 5.4 with the obstacles, the resulting Supervisor is given in Figure 5.18. In this example a unique solution to the problem is given by the primitives sequence $\{go_U, go_U, go_R\}$.

In this case, because of the presence of obstacles, the number of reachable states in the map is smaller. This highlights a very important characteristic in this approach: as the number of obstacles increases, the number of states of the Supervisor graph decreases, making it easier to compute the optimal path with any graph search algorithms. This is the opposite of almost any path planner algorithm, where, with the increasing number of obstacles, the computational complexity to find the feasible and/or optimal path increases.

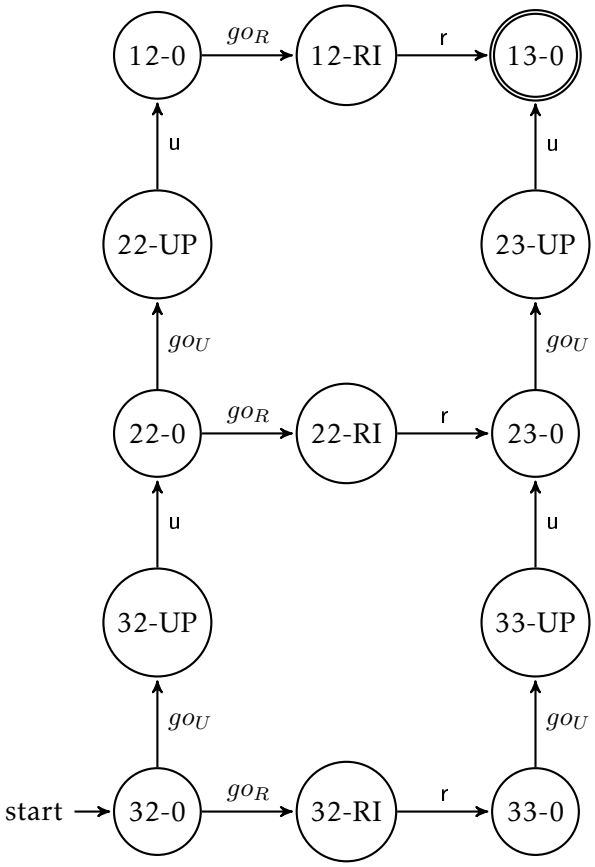

Figure 5.17: Supervisor Automaton example. The reachability graph.

Purge Refinement

The number of states (and transition) in the *Supervisor* can reach a considerable amount. Since the computational requirements of the graph search algorithms depends on the number of states of the graph, it is a good practice trying to reduce the final number of states to reduce the runtime of the algorithm. If the number of states of the *Specification Automaton* is n_{spec}, the number of states of APLA is n_{apla}, and the number of states of all AMIA is n_{amia}, in the worst case scenario, because of the properties of parallel composition, the number of states of the *Supervisor* can be $n_{sup} \leq n_{spec} \cdot n_{apla} \cdot n_{amia}$. In a typical application we have: $n_{spec} >> n_{amia} > n_{apla}$. The parameter n_{spec} depends only on map dimension, discretization and the number of obstacles, so that n_{spec} can be reduced with an increasing number of obstacles or by choosing a rougher discretization. No techniques were studied to reduce the n_{apla}, but the literature presents some methods to reduce the number of primitives [99] for a given agent. The proposed *Purge Refinement* instead focuses on reducing the impact of n_{amia} on the number of states. Because of the nature of AMIA, they are composed by chains of states where events are the primitive,

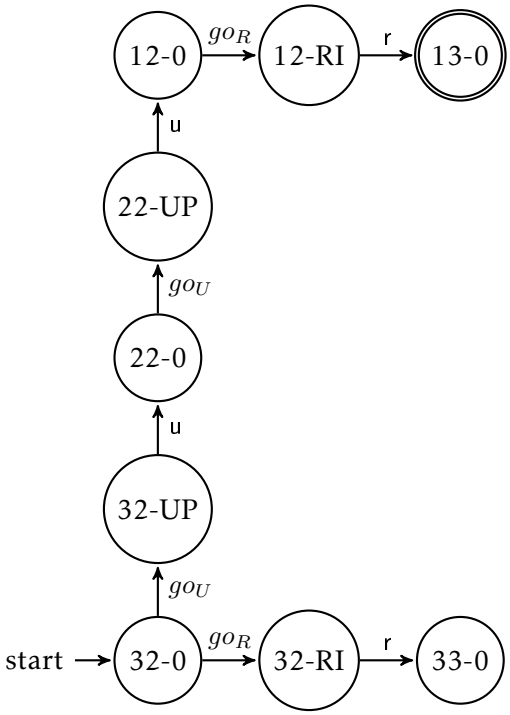

Figure 5.18: Supervisor Automaton example. The reachability graph with obstacles.

followed by a chain of elementary movements. The same structure is then reflected into the *Supervisor*. For construction, all the events representing the elementary movements have a cost of 0, so they don't impact the graph search algorithm optimality, i.e. they can be traversed with zero cost. What the *Purge Refinement* does is to remove from the *Supervisor* all the arcs related to elementary movements, and to collapse the states connected by them. This result in a number of final state for the *Supervisor* $n_{sup} \leq n_{spec} \cdot n_{apla}$. An example of Purging the *Supervisor* in Figure 5.17 can be seen is Figure 5.19.

In the purged Supervisor, only events representing primitives are present, such that the graph-search algorithm doesn't have to iterate in useless states and transitions.

5.1.6 Exploiting DES capabilities

The power of using well developed theory such as DES, allows us to exploit all the capabilities of those systems. In particular we can use states and events to model more complex system than the map and the agent itself. Moreover with controllable and observable events, the modeling capabilities can be extended to more complex scenarios. Some extended capabilities are under investigation, including the use of unobservable events to model sensors limitations, uncontrollable events to model environmental disturbances and the extension with parallel composition with other automata to achieve a decentralized heterogeneous multi-robot coordination for complex tasks. A simple ex-

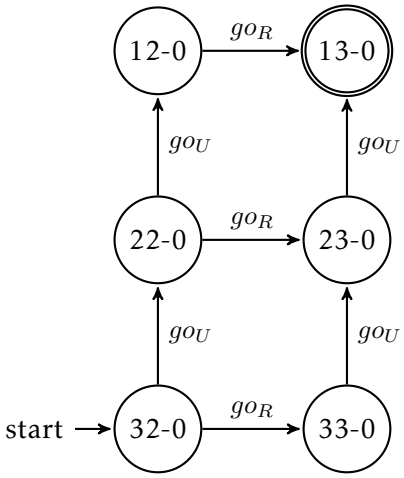

Figure 5.19: Supervisor Automaton example after the purging.

ample on how uncontrollable events are used to model wind disturbances for a UAV on the map is presented in the last section of this Chapter.

5.1.7 Advantages of DESP

We introduce two short lists of general advantages and limitations of DESP, to sum this chapter for the reader.

Advantages:

- Kinodynamic planning: with a suitable choice of the primitives and its interconnection or using the results presented in previous chapter, the generated trajectories respect differential constraints.
- No dimensionality curse: with respect to other "discretized" planners, this approach doesn't suffer the dimensionality curse given by the increased number of states when representing systems with many states in the state space dynamic model. For example using the kinematic of a car-like (state space of dimension 4) or using the dynamic of a car-like with trailer (state space of dimension 9) doesn't increase the number of states of the *Supervisor* if the number of primitives is the same, since the map automaton is the same.
- Complexity and runtime reduced with increasing obstacles: increasing the number of obstacles reduces the *supervisor* graph dimension, hence the graph-search algorithm for optimal path will search in a reduced states set, resulting in a reduced runtime.
- Exploiting DES capabilities: uncontrollable and unobservable events, automata and its operations can be exploited to model complex scenarios.

- Suitable for replanning and incremental algorithms: because of the graph nature, the approach is suitable for replanning and to apply existing incremental graph-search algorithms.
- Suitable for global planning although a variable resolution should be used to reduce the number of states.
- Robust: using the theory developed in previous chapter, it is possible to take into accounts disturbances and uncertainties in the dynamic model, allowing for a robust planning.
- Flexibility on robot behavior: with custom APLA is possible to build complex behaviors for the agent.
- No collision checking: taking into account trajectories swaths and thanks to parallel composition properties, no collision checking module is needed and lots of computation is saved.

Limitations:

- Not complete: Because of both environment and dynamic discretization, the planning solution is not complete in the sense of the ideal planning problem. Some authors are investigating if it exist a minimal resolution in both environment and agent primitives to have a complete solution.
- The optimality is with respect to the chosen discretization of map and agent. Choosing different map resolution or different set of primitives could potentially lead to more optimized solutions. The optimal solution for this approach is in general not the optimal for the generalized problem.

5.2 Applications and Simulations

In this section we provide two examples of the use of DESP, combined with the results of Section 4.2 and using the model and controls developed in Chapters 3 and 2. In the first example the model of a double integrator is used to model the dynamic of a VTOL in $x - y$ plane. Moreover some uncontrollable events are used in the *Map Automaton* to model areas with disturbances (wind). In the second example a differential-wheel model is used and a case of replanning is presented.

5.2.1 Double integrator and Uncontrollable Events

In this example, the considered robot is a VTOL, with only the position dynamic in 2D ($x - y$ plane). The dynamic model of the simplified VTOL can be given by (4.22). We

choose to discretize a 14×14 meters square area with 7×7 states for the *Map Automaton*. Each state represents a 2 square meter block. The discretization of the map is very rough, but it serves only as example. The names of the states are given in terms of discretized x position and discretized y position so as by increasing the first number of the state name represent a positive x movement while a positive increase in the second number of the state name represent a positive y movement. The resulting *Map Automaton* is given in Figure 5.20. All the base movements are omitted for sake of simplicity.

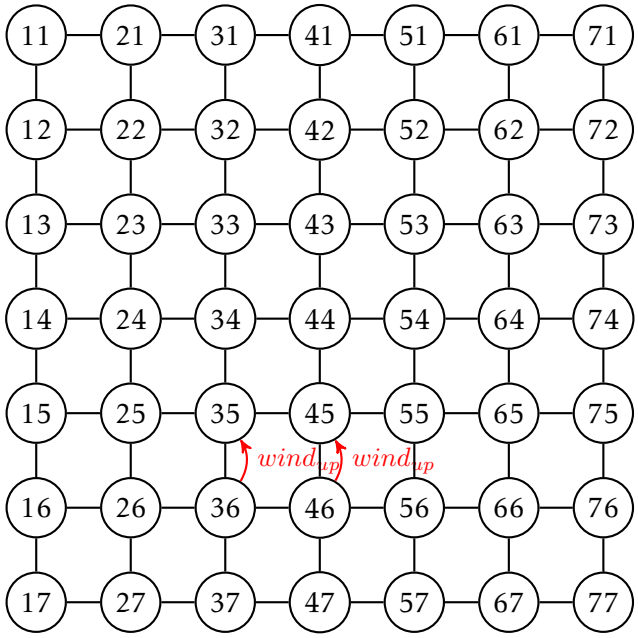

Figure 5.20: Map Automaton Example 7x7

On top of that we add to states $3-6$ and $4-6$ an uncontrollable event *wind_up* which represents a zone in the map in which a big upward wind is present. After the *wind_up* event occurs, the state in the map moves by minus one toward y axis, meaning that the event is modelled as an uncontrollable movement of the agent toward $-y$ direction. The *Specification Automaton* is then built adding forbidden states, initial state, and marked states. We consider this automaton static, such that the specification doesn't change online, but the validity for online planners with dynamically changing specifications is still valid. The forbidden states for this simulation are: 32, 33, 34, 35, 45, the target is set in the state 15 and the initial state (initial position of the robot) is set to 54. The setting and the specification automaton resulting from this specification are depicted in Figure 5.21.

The agent automaton is built to represent a constrained model even if the real model is a fully actuated rigid body. This is to ensure that the final reference is somehow smoothed by the supervisor but it is a matter of control law design. The primitives (possible maneuvers, events) of the agent are 8: *go_north*, *go,_n-w*, *go_n-e*, *go_east*, *go_west*,

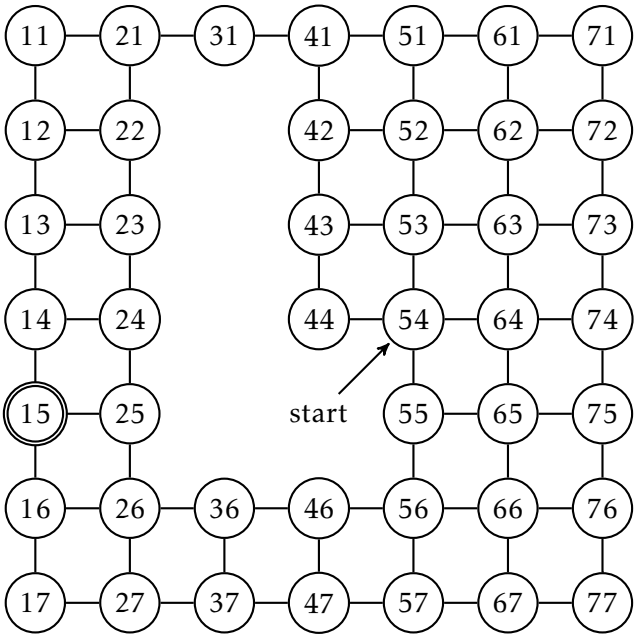

Figure 5.21: Specification Automaton Example 7x7

go_south, *go_s-w*, *go_s-e*. They represent straight lines at 45 degrees each (i.e. the eight directions north, north-west, west, south-west, ...). We assume that after a primitive (straight line in one direction) it is possible to have a primitive which differs only 45 degrees to the previous one (a correct sequence could be *go_north/go_n-w/go_west/go_west*, while a not admissible sequence could be *go_north/go_west/go_s-w/go_south* because *go_north/go_west* primitives differ 90 degrees). This will ensure that the final trajectory will be a broken line with only 45 degrees of difference between consecutive lines. This could be a limitation for a fully actuated model but ensure a smoother trajectory with less constrains on derivative bounds. The resulting APLA can be seen in Figure 5.23.

Every event is the primitive of the eight different movements (eight lines), where north mean a direction toward $-y$, east a direction toward x, n-e (shortened north-east) a direction toward $-y$ and $+x$ and so on. The primitives (maneuvers) are taken as linear motion with constant speed in the eight directions. In particular the speed on every direction is taken as $1.2m/s$. For sake of compactness we use the index $i = 1, 2, ..., 8$ to identify the primitives so as: $go\_north \Rightarrow i = 1$, $go\_n - e \Rightarrow i = 2$, $go\_east \Rightarrow i = 3$, $go\_s - e \Rightarrow i = 4$, $go\_south \Rightarrow i = 5$, $go\_s - w \Rightarrow i = 6$, $go\_west \Rightarrow i = 7$, $go\_n - w \Rightarrow i = 8$ following Figure 5.22. With the discretization of $2m$ and assuming that every maneuver is completed in $\frac{2}{1.2}s$ the position, velocity and acceleration of every time-trajectory mapping the primitive are defined as:

$$\ddot{x}_{1R,i}(t) = 0 \quad \forall i$$

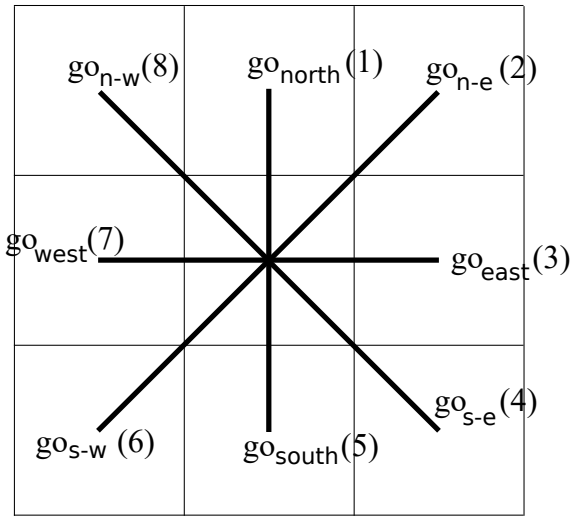

Figure 5.22: The 8 primitives for the robot. They consist of simple straight lines in 8 directions. The interconnection of this primitives is an uncontrollable trajectory for a double integrator model.

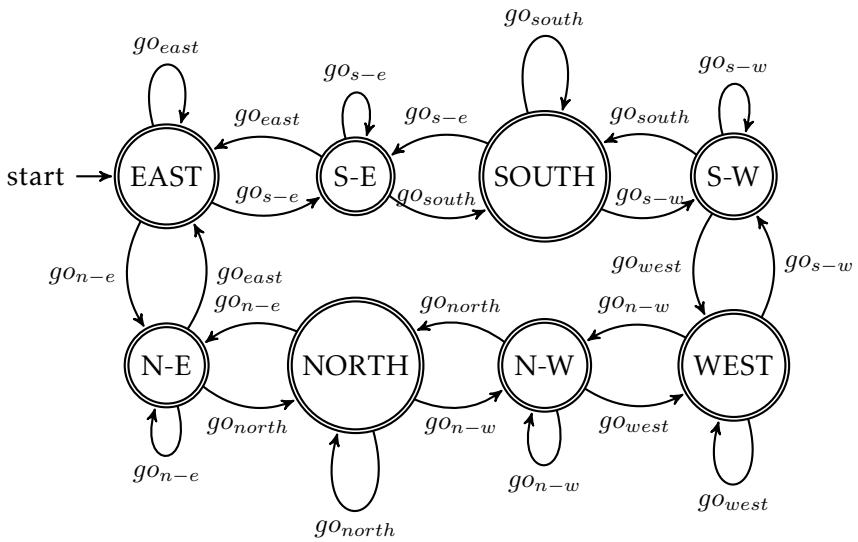

Figure 5.23: APLA automaton for the robot in the example. The string of primitives consist of straight lines that differ only 45 degrees each other.

$$\dot{x}_{1R,i}(t) = \begin{cases} 1.2 & i = 2,3,4 \\ 0 & i = 1,5 \\ -1.2 & i = 6,7,8 \end{cases}$$

$$x_{1R,i}(t) = \dot{x}_{R,i}(t)t + x_{R0} \quad \forall i$$

$$\ddot{y}_{1R,i}(t) = 0 \quad \forall i$$

$$\dot{y}_{1R,i}(t) = \begin{cases} 1.2 & i = 4,5,6 \\ 0 & i = 3,7 \\ -1.2 & i = 1,2,8 \end{cases}$$

$$y_{1R,i}(t) = \dot{y}_{R,i}(t)ty_{R0} \quad \forall i$$

where $x_{1R,i}(t)$ and $y_{1R,i}(t)$ denote the references for the x and y position for the i-th primitive and x_{R0}, y_{R0} represent the position reference before the switch, i.e. the final position of the previous trajectory piece, so as the position reference is continuous.

The initial state is $EAST$ state, meaning that the initial primitive could be only *go_east*, *go_n-e* or *go_s-e*, but it can be chosen indifferently. The resulting trajectory for the system will be piece-wise controllable, since straight lines with constant velocity is a trajectory for the double integrator system, but there are discontinuities at primitives intersection.

The supervisor when dealing with uncontrollable states should be built as $G_{sup} = G_{map}||G_{spec}||G_{ag}$, such that the resulting automaton is the controllable behavior we want IFF the system results controllable (the reader can refer to [24] for controllability notions). In this example the system results controllable, otherwise no solutions exist to the problem.

Low Level Control

The low level control for the simplified dynamic of the VTOL is built following 4.26 in chapter 4. Low level control is defined with matrix K_s. We need to ensure Dwell time condition of Lemma 4.1 for the goal (4.14) and to make the system stable. By having defined the primitives of the supervisor, one can define the parameter d. In particular, by noticing that two consecutive concatenated trajectory will have a difference in position equal to 0 and a maximum difference in velocity equal to 1.2, one can pick $d = 1.2$. Defined d, one can choose $q > d$. In our case we pick $q = 1.4$ to impose the maximum tracking error. By choosing the regulator parameter $a = 40$ we obtain:

$$K_s = \begin{bmatrix} 4.4721 & 4.9469 & 0 & 0 \\ 0 & 0 & 4.4721 & 4.9469 \end{bmatrix}, \quad P_s = \begin{bmatrix} 44.2467 & 4.4721 & 0 & 0 \\ 4.4721 & 4.9469 & 0 & 0 \\ 0 & 0 & 44.2467 & 4.4721 \\ 0 & 0 & 4.4721 & 4.9469 \end{bmatrix}$$

with a minimum dwell time condition $\tau \geq 1.4844$ which is respected by the $\frac{2}{1.2}s$ switch time of the supervisor.

Tradeoff between d, q, τ and a

A few more words need to be spent on the tradeoff for the parameters of control, dwell time and trajectory bounds.

Low level control parameters: a and $\overline{\lambda}_P$.

For a given low level control law, the parameters a and $\overline{\lambda}_P$ are already chosen, and could be inferred by a Lyapunov function of the system. On the other hand, if the law is not chosen and must be decided, one can notice that, by increasing the value a, the final value of τ decreases and the terms of the matrix K increase. This could seem as an easy solution to decrease the dwell time constrain, but in practice a bigger K could lead to problems such as actuators constrains and noise amplification. Those problems are not considered in this work but should be taken into account for any practical implementation.

Trajectory parameter: d.

The parameter d is strictly dependent on the primitives of the supervisor, and on how those primitives are mapped in time-trajectory. The dependence of this parameter is basically related to the discretization of the map, to the time-trajectory chosen (for example a straight line primitive could be mapped in uniformly accelerated trajectory, constant speed trajectory, trajectory with initial and final speed equal to zero, and so on) and to the speed at which the trajectory is executed. This last point is very important; the same geometric path executed with a different speed will lead to different bounds of the derivative, so as at bigger speed, the time needed to complete the trajectory is shorter but the parameter d will result bigger. Reminding that q should be greater than d one can easily understand that for a given primitive, a different execution speed will lead to a shorter execution time but to a bigger tracking error and a bigger constrain on τ. This could inspire a particular policy for choosing the trajectory. In particular one can choose a shorter execution time when the precision doesn't matter (moving through areas without obstacles) leading to a faster but less precise movement, while one can choose a larger execution time when the precision matters (moving through areas full of obstacles, moving through holes, precision operations) leading to a slower but more precise movement.
As final remark one can notice that in the case of piece-wise constant trajectory, the parameter d is given only by the position difference at switching, being all the derivatives equal to zero; in the case of perfect smoother implemented, so as the position and derivatives at switching time are continuous, the parameter d is equal to zero, leading to an ideally perfect tracking and with no constrain on dwell time.

Dwell time: τ.

Dwell time τ represent a constrain which should be fulfilled to guarantee the stability of the system and to guarantee (4.14). If the condition on the minimum dwell time is not respected there are three possible way to solve the problem. The first is to increase a so as increase the values of K but with the problem pointed out in the previous paragraph. Second way is to increase the parameter q with the drawback of a possible larger tracking error. Third way is by changing the time-trajectory which map the primitive, by changing for example the type of motion or decreasing the execution time, so as the parameter d will decrease. This is probably the best solution because by decreasing d, it is even possible to reduce q while fulfilling the dwell time constrain.

Simulations

A simulation was run to test the proposed control and to verify that the conditions for the reference trajectory respect (4.14).

The resulting trajectory is obtained applying the optimization on the supervisor graph, in particular in this case the Dijkstra algorithm ([28]) was applied to obtain the minimum distance trajectory. In particular every state of the Supervisor was handled as a node of the graph and every event as an arc of the graph and weighting every primitive (arc in the graph) with the related distance so as the four primitives representing lateral or vertical movements ($i = 1, 3, 5, 7$) were weighted 1 while the other four primitives (diagonal movements, $i = 2, 4, 6, 8$) were weighted $\sqrt{2}$. The reference generated by the supervisor and the real position of the system are depicted in Figure (5.24). The state $4 - 4$ represents the discretized square centered in the position $(x, y) = (0, 0)$.

The supervisor generated exactly the expected trajectory (Figure 5.25), following the constrains of the agent automaton, choosing the shortest path, avoiding obstacles and avoiding uncontrollable events which could lead to undesired states in the map. The "wind area" was indeed successfully dodged by the trajectory which could have driven the robot to undesired states 35 or 45.

Figure 5.26 and Figure 5.27 depict respectively x and y position and velocity trajectory, how the real system tracked the reference, the norm of the error and the goal bound of (4.14) in terms of parameter q. The goal (4.14) is respected for the whole trajectory.

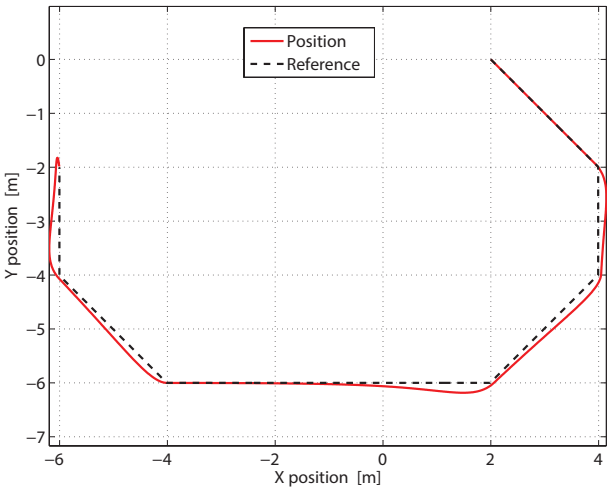

Figure 5.24: Position reference generated by the supervisor and real position of the system in the x-y plane. We can notice how the dynamic system cannot track the generated un-feasible trajectory, but the maximum error is such that the robot doesn't collide.

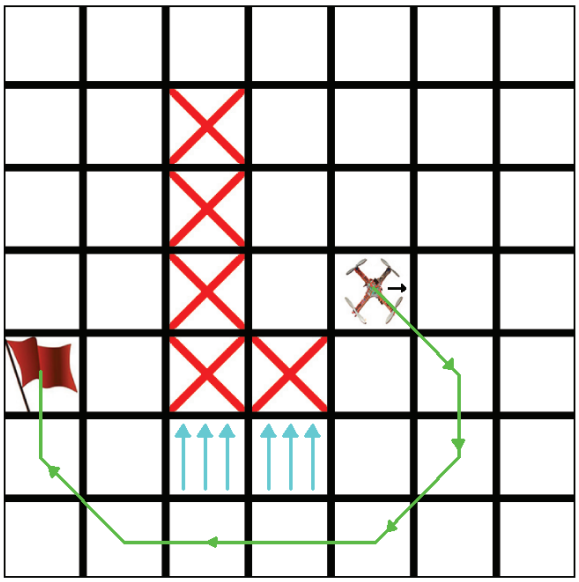

Figure 5.25: Representation of the discretized map, the obstacles, the target and the initial state of the robot. In green we can see the optimal trajectory for the set of primitives.

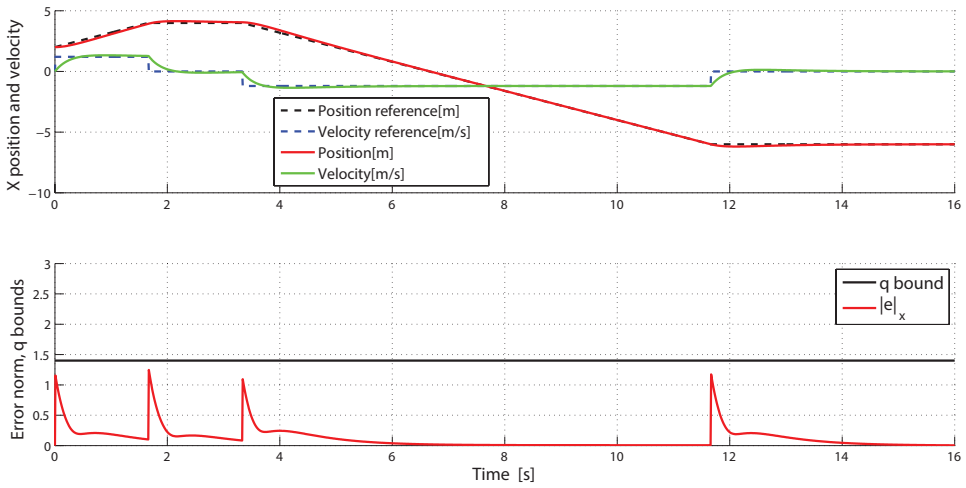

Figure 5.26: First plot includes the x position and velocity reference and the real x position and velocity. Second plot includes the norm of the x error and the bound given by the parameter q: the bound is respected for the whole time.

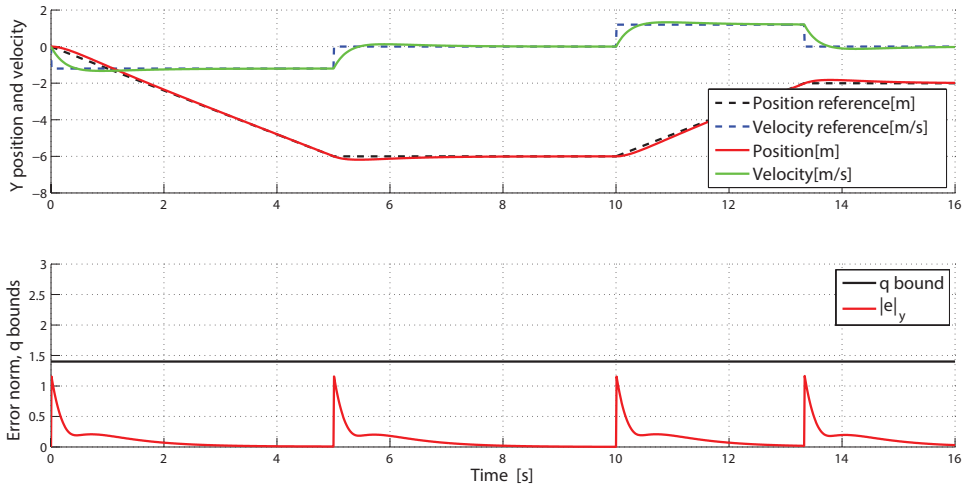

Figure 5.27: First plot includes the y position and velocity reference and the real y position and velocity. Second plot includes the norm of the y error and the bound given by the parameter q: the bound is respected for the whole time.

5.2.2 Differential Wheel Robot and Replanning

In this experiment the differential wheel robot model and control are used to model the *Supervisor*. The robot kinematic model is the one defined in Section 2.2 and the control law the one defined in Section 3.2. With the low level control at hand for our application, the *Supervisor* can be defined. We consider again a map in 2D $x - y$ plane. The environment consists in a map of dimensions: 80m in x direction and 80m in y direction. A simulation was run to test the behavior of the proposed approach in case of replanning, where some obstacles are known a priori, i.e. before the first computation of the trajectory, while another obstacle is detected by the robot only when close to it.

Map automaton:

Since the robot is moving only in $x-y$ plane, the automaton for the map, from which the specification automaton will be derived, consists of the discretization of the $x - y$ space. In particular, the space is partitioned into equal pieces, and every state of the automaton represents a quantum of the space. We consider a rough discretization of squares of 4m for a total of 20x20 space quanti. The resulting automaton is not depicted because of its dimension.

Agent automaton:

The primitives used for this problem are the same of the previous example in Figure 5.22 with the difference that the primitives are longer than in previous example because of different discretization and the linear velocity is different. The corresponding APLA is different from the one in the previous example, as we want to model a different behavior for primitives concatenation. To exaggerate the discontinuities and to show how this approach works with highly non-controllable trajectories, the logic of the *Agent Automaton* is built such that primitives that differ 90 degrees can be interconnected. The resulting APLA can be seen in Figure 5.28.

For this simulation we set $v = 4$. With this choice, the primitives will be executed with a constant linear speed of 4m/s.

Specification automaton:

The *Specification Automaton* is built as usual, by deleting the forbidden states where obstacles are present. We don't depict the particular automaton because of its dimensions.

Let's now consider the parameters for the low-level trajectory tracking law derived in Section 3.2 and in Section 4.2. In term of the angle θ, the maximum discontinuity at

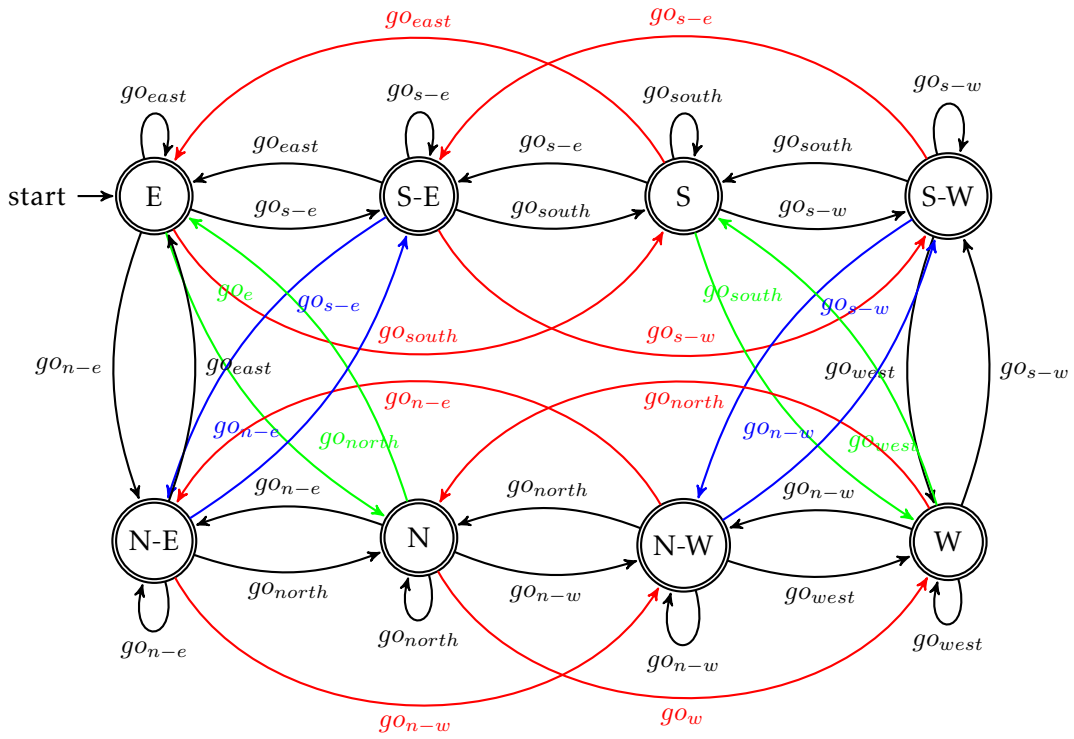

Figure 5.28: APLA automaton for the robot in the example. The string of primitives consist of straight lines that differ only 90 degrees maximum each other.

primitive switch is $d = \pi/2$, in which the worst case of primitives that differ 90 degrees is considered (For example: go_{west}, go_{north}). As far as the position p_x, p_y is concerned, note that the maximum discontinuity is 0 since we connect continuous primitives. From (4.41) it is possible for this particular application calculate the maximum tracking error q, given d and τ, or viceversa it is possible to calculate the minimum dwell time τ with at hand d and q. Since $\underline{\alpha} = \bar{\alpha} = 1/2$ and γ is constant we have

$$\tau = -\frac{\ln\left(\frac{(q-d)^2}{q^2}\right)}{2\gamma}$$

or

$$q = \frac{d\left(1+\sqrt{e^{-2\gamma\tau}}\right)}{(1-e^{-2\gamma\tau})}\,.$$

We chose as maximum norm error $q = 2$, to guarantee that in the worst case, the robot still remains in the 4m box.

The parameters for the low level controller are chosen as:

- $K_P = 3,4$

- $K_\theta = 180$
- $\lambda = 3,99$
- $\tilde{x}^* = \tilde{y}^* = 2$, since $q = 2$

resulting in $\gamma = 1.6$ and $\tau = 0.96$. The dwell time τ is compatible with the smaller time of execution of primitive which is $1s$ for the horizontal and vertical primitives.

In the simulation, the scenario could be seen in Figure 5.29. The obstacles 1 and 2 are the ones known at the first planning, while obstacle 3 is detected when the robot is 6m away from it, forcing a re-plan. In red we can see the path generated before the detection of the new obstacle and in blue the path after the detection. The robot is forced to do a 90 degree turn to avoid the obstacle.

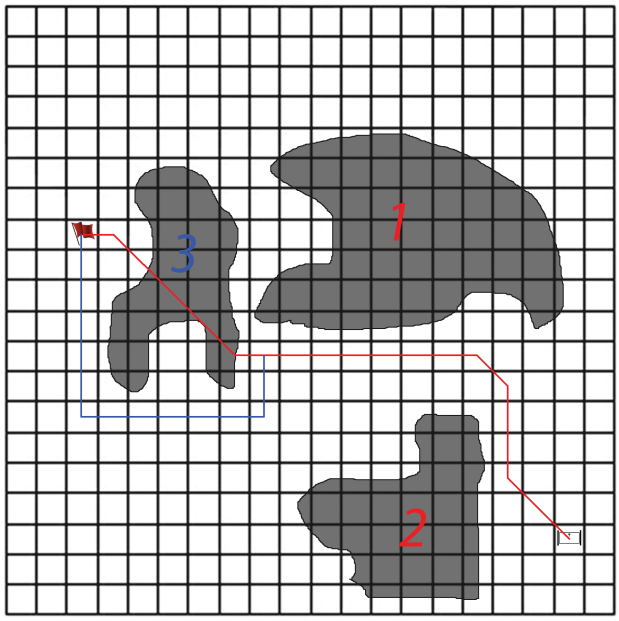

Figure 5.29: Scenario of simulation. In red the path generated to avoid obstacle 1 and 2. In blue the re-planning trajectory after the detection of obstacle 3.

In Figure (5.30) we can see how the trajectory is tracked by the system, piloted by the low level control. As we can see the error never exceed the constraint and the obstacles are avoided. In Figure (5.31) the inputs generated by the low level control.

In Figure (5.32) a section of the Specification Automaton of the simulated scenario. The states without obstacles are kept, while the states with obstacles are deleted. The goal becomes a marked state (double circle).

Setting $v = 8$ and consequently reducing by half the execution time of the primitive and doubling the linear speed, the condition on dwell-time in Lemma 4.2 is violated.

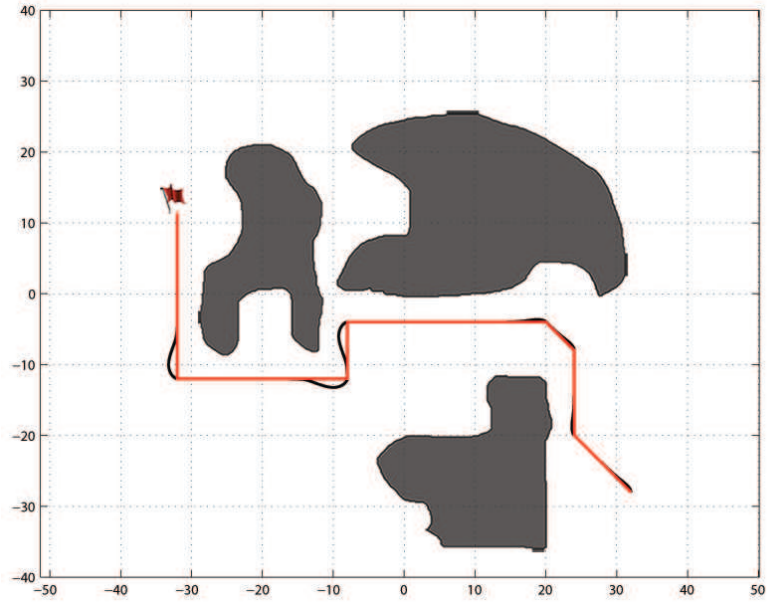

Figure 5.30: Simulation with the dynamical model. In red the generated trajectory and in black the actual trajectory of the robot with closed loop control.

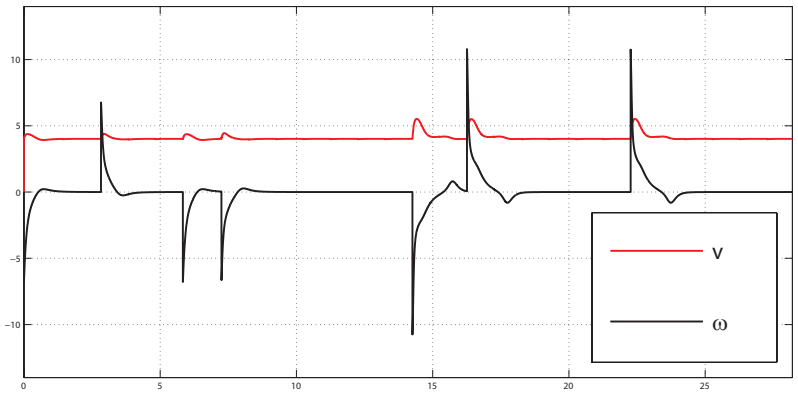

Figure 5.31: The inputs computed by the low level controller. In black the angular velocity and in red the linear velocity.

Hence we can see the behavior of the robot in Figure (5.33) where the tracking error increases, leading to a crash.

This could inspire a particular strategy, in which $v(t)$ is dynamically manipulated to adapt the boundaries conditions and consequently the maximum tracking error to adopt different strategies in different scenarios. Let's imagine for example a scenario in which one area is cleared from obstacles, while another area is cluttered of obstacles or some obstacles can appear. Then a policy could for example use faster v in the first area to cover the trajectory faster, while keeping a low v in the other area, to ensure a more

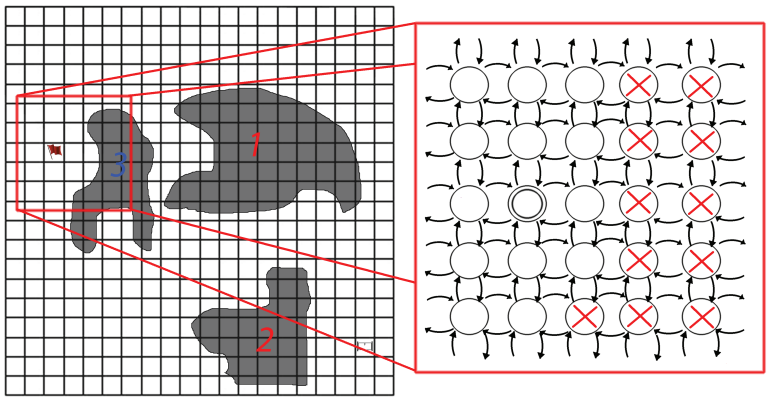

Figure 5.32: A section of the Specification Automaton.

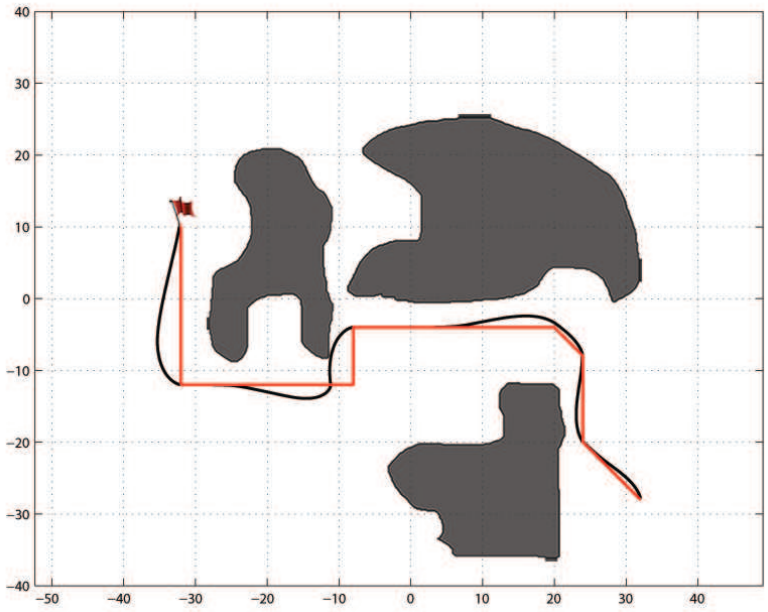

Figure 5.33: Not ensuring the condition on dwell-time, the resulting trajectory could not respect the boundary imposed.

precise and controlled tracking.

6

Experiments and Applications

In this chapter we show some of the applications and experiments related to the contributions of this thesis. While on path planning strategies and DESP only simulations were conducted, many experiments were performed on the proposed control law for the VTOL. In particular we show some basic simulations and experiments on the trajectory tracking control law applied to a quadrotor prototype and how the same control law was used to develop an architecture for nano-quadrotor swarm control and coordination. Finally some references are presented where other application of the proposed control law were implemented and tested.

6.1 Application to the control of a quadrotor aerial vehicle

The control strategy developed in chapter 3 for the *robust case* has been tested on a real quadrotor aerial vehicle.

The selected prototype, which is depicted in Figure 6.1, is based on a carbon fiber tubular airframe designed on a tubular structure to obtain the required mechanical properties, such as stiffness, while maintaining a relative low weight (about 220 grams). The prototype is actuated by four fixed-pitch propellers, each one driven by a brushless DC (BLDC) electric motor. The selected motors (Dualsky XM-400, 130 W of maximum power) and propellers (APC, 8 inches diameter) are able to produce approximately 2 Kg of total thrust when using a $3S$ LiPo (Lithium Polimery) battery as the poser source.

Figure 6.1: Quadrotor prototype

Moreover we included markers to be tracked with Optitrack camera system (for a more precise position estimation). For an indoor flight just Optitrack and IMU are used, while for outdoor flight or not in presence of Optitrack, IMU, optical flow and sonar are used together to evaluate the state.

Following [101], the dynamics of the system can be described by means of (3.1) in which the resultant force and torques can be computed as a function of the four thrusts

T_i, $i = 1, 2, 3, 4$, generated by the four different propellers, namely

$$\begin{bmatrix} u_f \\ u_\tau \end{bmatrix} = \begin{bmatrix} -1 & -1 & -1 & -1 \\ 0 & -d & 0 & d \\ d & 0 & -d & 0 \\ K_{tm} & -K_{tm} & K_{tm} & -K_{tm} \end{bmatrix} \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{bmatrix} \tag{6.1}$$

where d denotes the distance of the propeller spin axis from the center of gravity of the system, K_{tm} is a parameter which relates the thrust of a single motor to the aerodynamic torque produced along the spin axis of the propeller. The thrust T_i produced by each propeller is a function of the angular speed $\omega_{P,i}$ of the motors, namely $T_i = k_T \omega_{P,i}^2$, with $i \in \{1, 2, 3, 4\}$. The parameters of the specific prototype are $M = 1.05\ Kg$, $J_0 = \text{diag}(0.0082, 0.0082, 0.0164)\ Kg\,m^2$, $d = 0.29\ m$, and $K_{tm} = 0.026$. The values of K_{tm}, k_T, d and M have been measured experimentally with the help of a load-cell, while the value of the inertia J_0 has been estimated using a computer aided design software. To take into account possible uncertainties of the mass distribution, the following upper bound has been taken into account $J^U = \text{diag}(0.01, 0.01, 0.02)\ Kg\,m^2$.

With the above prototype at hand, the following two subsections propose respectively simulations and experimental results obtained by considering the robust control strategy defined in Sections 3.1.2 and 3.1.2.

6.1.1 Simulations

Two different simulations are proposed. In the first one, the quadrotor is required to hover at a fixed position $p^\star$ starting from an initial attitude configuration in which the vehicle is overturned (attitude recovery maneuver). To govern the position dynamics, the controller (3.22) has been employed by choosing, according to Proposition 3.1, the control gains as $k_1^\star = 1$, $\lambda_1^\star = 5$, $k_2^\star = 150$, $\lambda_2^\star = 150$ and $\epsilon = 0.06$. For the attitude loop, the controller in (3.42), yielding robust global stability results, has been considered with $k_p = 40$, $k_d = 0.2$ and $\delta = 0.1$. To show the robustness of the proposed control law, realistic parametric uncertainties and disturbances have been considered in the simulations. In particular, the actual inertia J of the vehicle is assumed to be $10\ \%$ larger than the nominal one, while the forces and torques disturbances d_f and d_τ are selected as coloured noise with maximum amplitude of $1.5\ N$ and $0.05\ Nm$, respectively, to model the effects of possible wind impacting propellers. Finally, a white noise with maximum amplitude of $0.04\ rad$ has been added to the attitude position measurement to represent uncertainties in the sensor model.

Figures 6.2, 6.3 and 6.4 show the attitude position, the angular speed and the linear position of the vehicle during the attitude recovery maneuver. Note that, despite the

vehicle in the initial position being overturned, i.e., $[\eta(0), \epsilon(0)^\top]^\top = [0, 1, 0, 0]^\top$, the vehicle converges rapidly to the desired hover configuration by rotating around the body x-axis. This is achieved without undesired switches of the hybrid state h, as depicted in Figure 6.5. This result may not be achieved with continuous time controllers having the above initial condition as equilibrium point, or with discontinuous feedback laws not robust to measurement noise (see also [43]). Finally, the force and torque control inputs applied to the quadrotor during the maneuver are depicted in Figure 6.6.

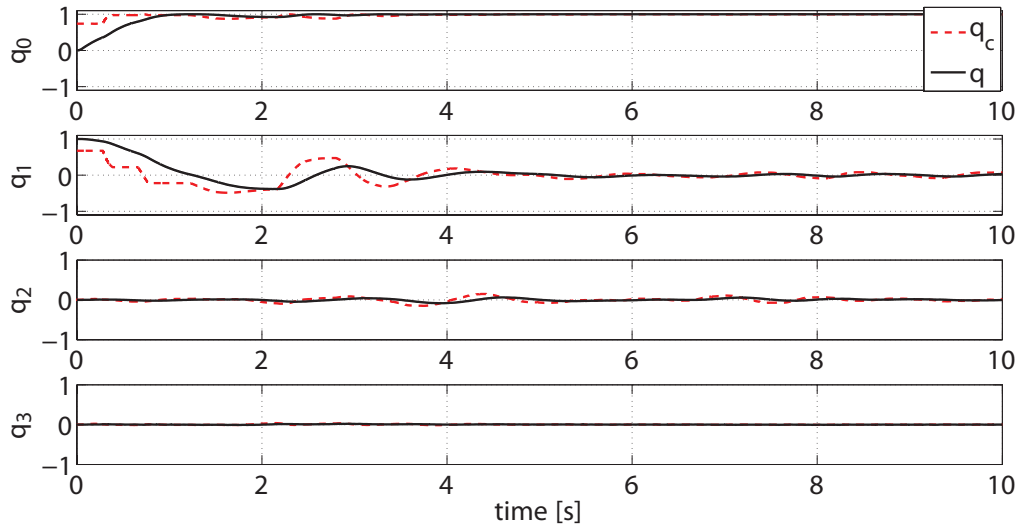

Figure 6.2: The attitude trajectory of the quadrotor during the attitude recovery maneuver.

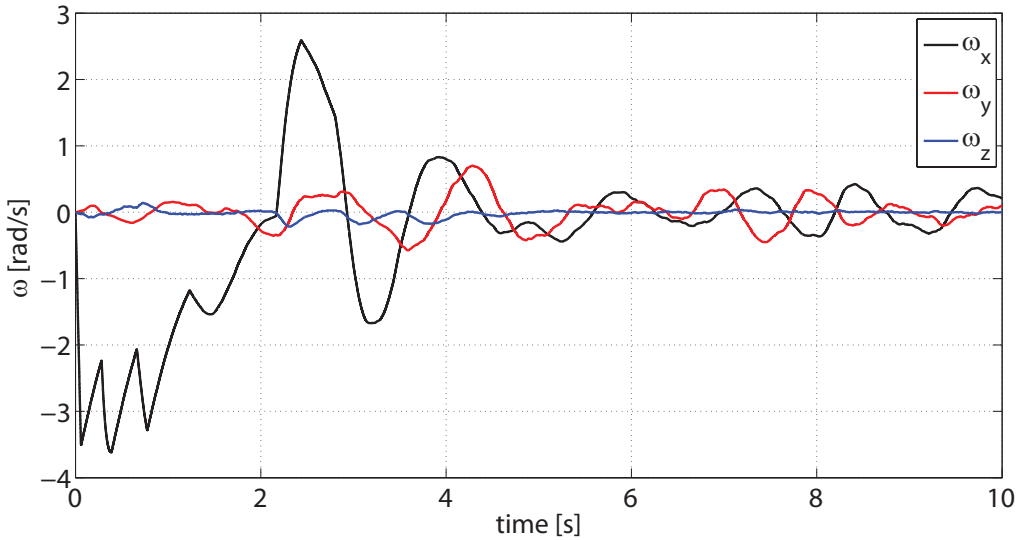

Figure 6.3: The angular speed of the quadrotor during the attitude recovery maneuver.

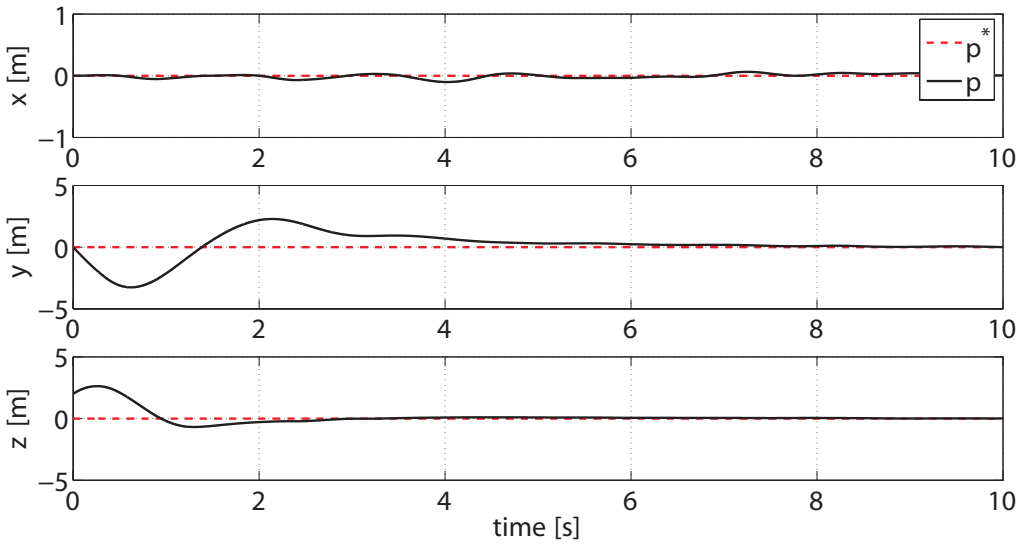

Figure 6.4: The position trajectory of the quadrotor during the attitude recovery maneuver.

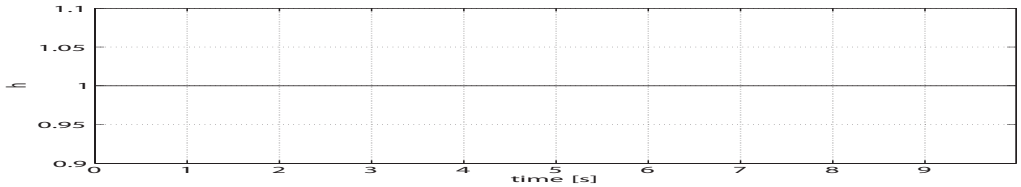

Figure 6.5: The logic state h of the quadrotor during the attitude recovery maneuver.

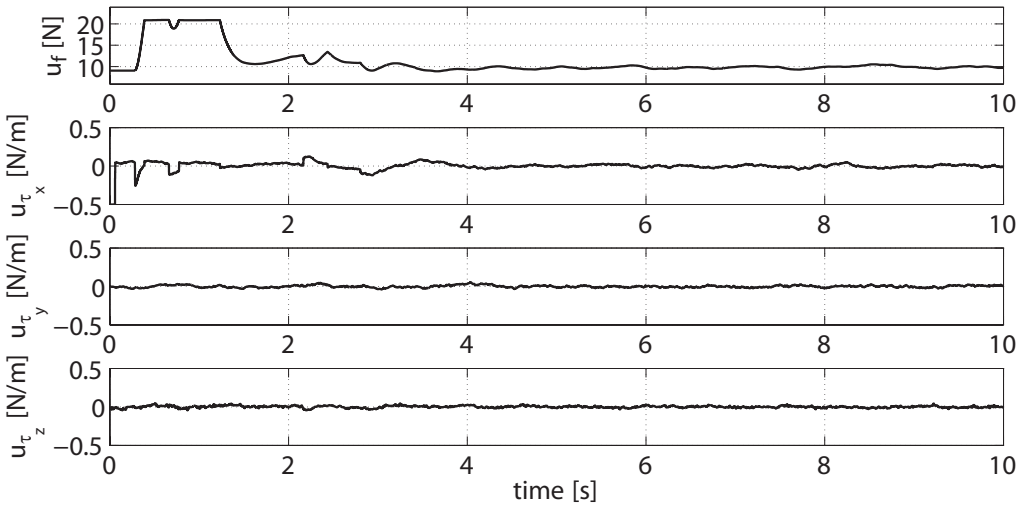

Figure 6.6: The control force and torques applied to the quadrotor during the attitude recovery maneuver.

The second simulation considers an aggressive maneuver (barrel flip) to be accomplished by the vehicle. In particular, the desired time reference signals are given by

$x^*(t) := 0$, $y^*(t) := \cos(\gamma t)$, $z^*(t) := -\sin(\gamma t)$, where $\gamma := 1.4\pi\ rad/s$. In practice the quadrotor is required to follow a circular trajectory along the y and z inertial axis maintaining a constant speed along the path. The reference inputs u_f^* and u_τ^*, required to compute the feed-forward control terms in (3.14) and (3.42), have been computed as in Section 3.1.1 with R^* obtained using the algorithm in Appendix A.1.4. For the above reference trajectory, condition (3.8) holds with $\epsilon \leq 0.1$. Thus the same position and attitude control parameters of the first simulation have been considered. The actual and the reference position trajectories are depicted in Figure 6.9, showing how the system converges to the desired path. Figure 6.7 shows the attitude of the vehicle during the aggressive maneuver and Figure 6.8 shows the angular velocity. Note that, to compensate for the high centrifugal force with the thrust forces generated by the propellers, the quadrotor has to continuously rotate around the body x axis. Finally in Figure 6.10 one can see a sequence of the aggressive maneuver with the proposed control law.

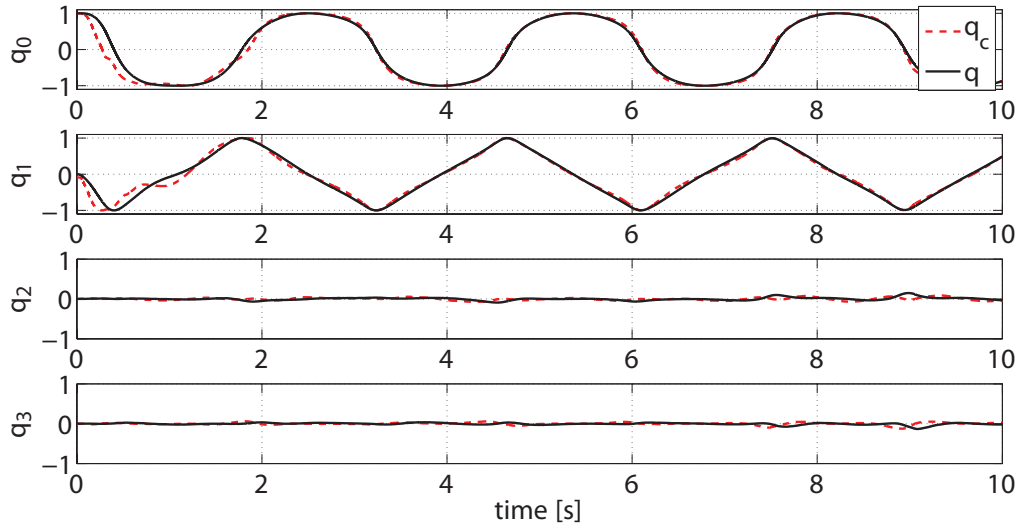

Figure 6.7: The attitude trajectory of the quadrotor during the aggressive maneuver: barrel flip.

6.1.2 Experiments

The goal of this section is to show the performance of the proposed robust inner-outer loop control strategy in a real-world application scenario. In particular, the robust control strategy developed in Sections 3.1.2 and 3.1.2 has been implemented on a real autopilot [84] in order to stabilize the quadrotor prototype described in Section 6.1. To determine the attitude of the vehicle, the selected autopilot includes an Inertial Measurement Unit (IMU) which consists of $3D$ accelerometers, magnetometers and gyros. The low-level sensor information obtained by the IMU is processed by an Attitude and

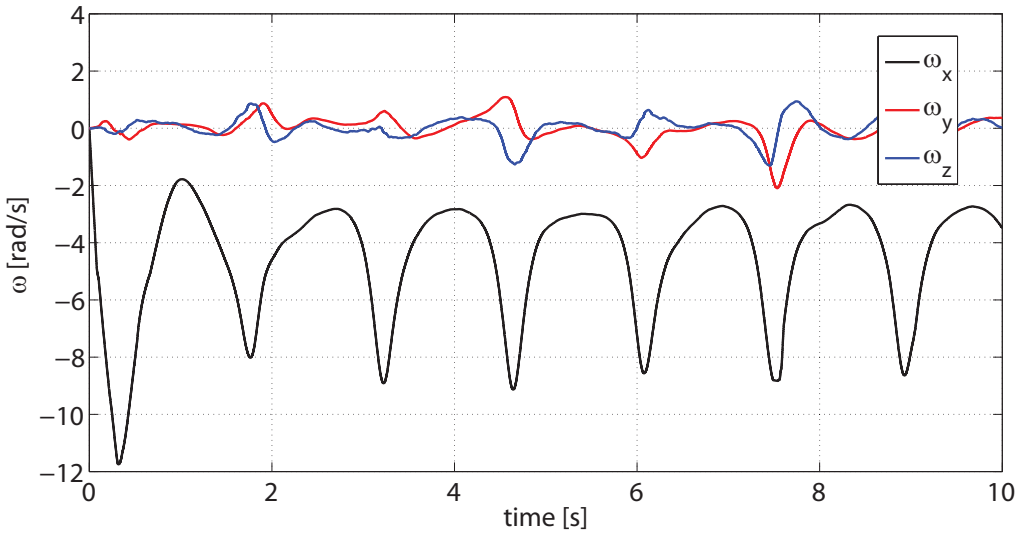

Figure 6.8: The angular speed of the quadrotor during the aggressive maneuver: barrel flip.

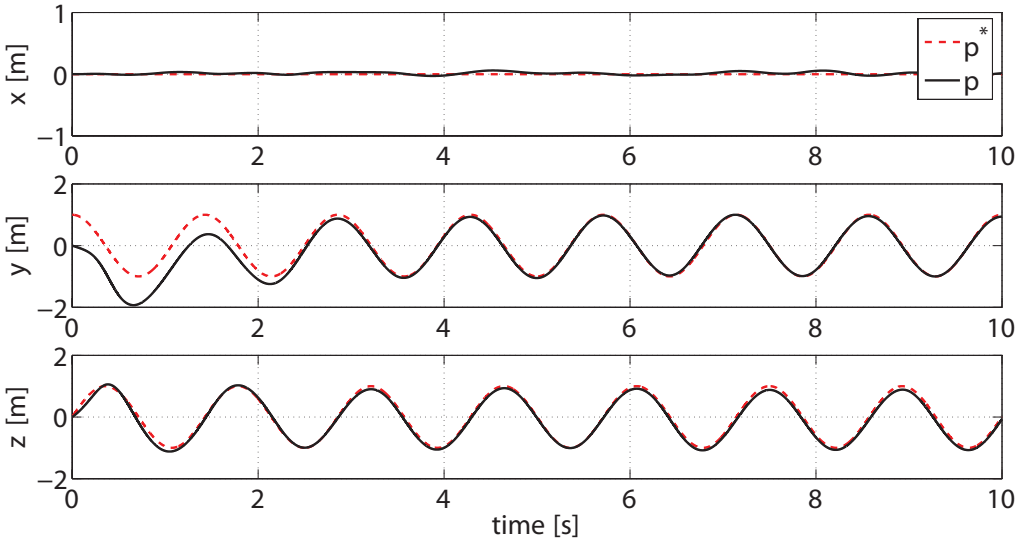

Figure 6.9: The position trajectory of the quadrotor during the aggressive maneuver: barrel flip.

Heading Reference System (AHRS) algorithm, derived following [31], in order to compute the attitude quaternion and the angular speed of the system. An external motion tracking system [100] has then been employed to determine the linear position of the center of gravity of the vehicle. The selected motion tracking system is able to measure the position at 100 Hz with approximately 1 mm accuracy. With this information at hand, a standard high-gain observer [61, Chapter 14.5] has been employed to compute the linear velocities in real-time. All of the control and estimation algorithms run in

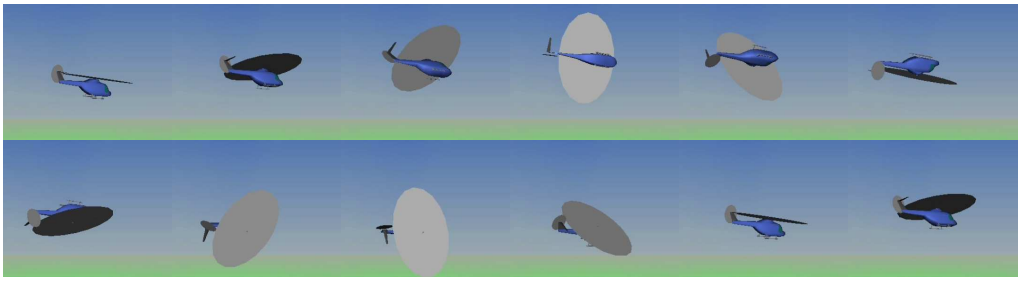

Figure 6.10: Sequence of simulated aggressive maneuver: barrel flip.

real-time at 200 Hz rate on a 32 bit ARM processor. The proposed control law and the high-gain observer were implemented in c++ as APPs for the PX4 stack. The parameters for the controller used in the experiment are: $k_1^\star = 1$, $\lambda_1^\star = 5$, $k_2^\star = 150$, $\lambda_2^\star = 150$, $\epsilon = 0.06$, $k_p = 16$, $k_d = 0.7$ and $\delta = 0.15$. The value of δ has been selected to take into account the measurement noise on attitude estimation of the specific prototype.

The proposed experiment has two goals. On one hand it shows how the proposed control strategy is actually effective in practical applications in which model uncertainties, disturbances and noises are present. On the other hand, it shows how the proposed global stabilizing controller is able to overcome typical limitations of some continuous-time stabilizing controllers, such as in particular the *un-winding* phenomenon [112]. In the experiment, the vehicle is deployed by the hand of a human operator and the controller is required to maintain a constant position and attitude. Due to this particular type of deployment, large initial attitude errors can be introduced since the vehicle may rotate when it is launched by the operator.

Figures[1] 6.12 and 6.13 show the attitude position and angular velocity of the vehicle during the experiment. Note that the vehicle starts close to the unit quaternion $q = [1,\, 0,\, 0,\, 0]^\top$ and then, due to manual deployment procedure, is rotated by approximately 2π around the body z axis. Note that at time $t \approx 3.9$ sec, η is close to zero while ϵ is close to $[0,\, 0,\, -1]^\top$. Due to this large attitude error, the jump domain D_{rob} in (3.45) is entered. Note that, due to the small values of the inertia J^U, the jump condition in (3.44) can be entered even when the vehicle is rotating at relative large angular speed values (up to 3 rad/s for the parameters employed in the experiments). Thus the logic state h, which is depicted in Figure 6.14, switches from 1 to -1 and the unit quaternion $q = [-1,\, 0,\, 0,\, 0]^\top$ is stabilized. This fact prevents the vehicle to perform a complete rotation around its body z axis in order to reach $q = [1,\, 0,\, 0,\, 0]^\top$, which, according to the Rodrigues formula, corresponds to the same orientation. The position of the vehicle during the maneuver has been depicted in Figure 6.11 while the control inputs are given in Figure 6.15. A video showing the experiment is available at `https://www.youtube.`

[1] Measurements are obtained from the onboard autopilot at 10 Hz rate.

com/watch?v=FFW2MchU79A.

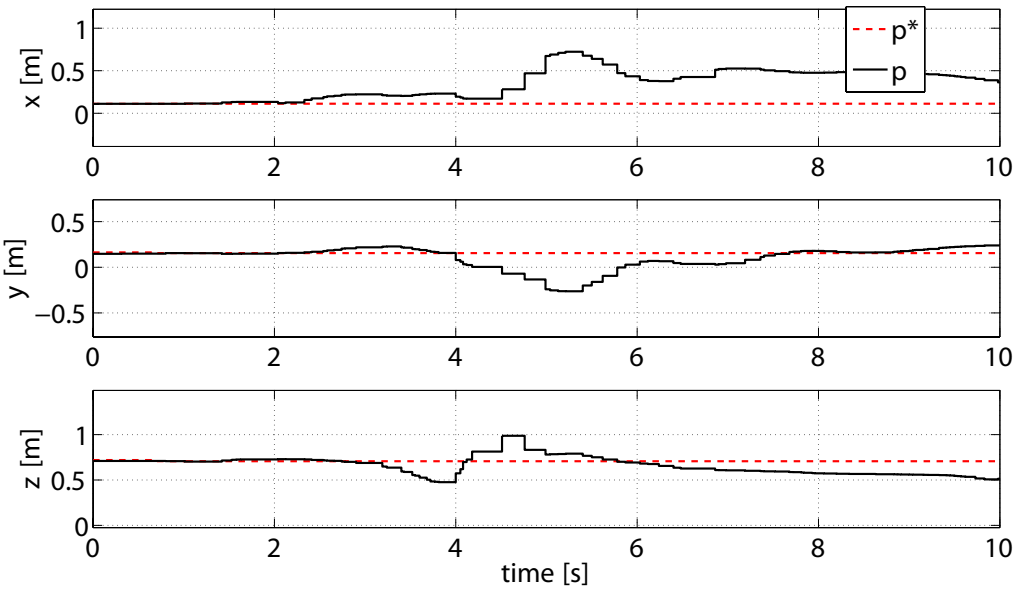

Figure 6.11: Position during the hand deployment maneuver.

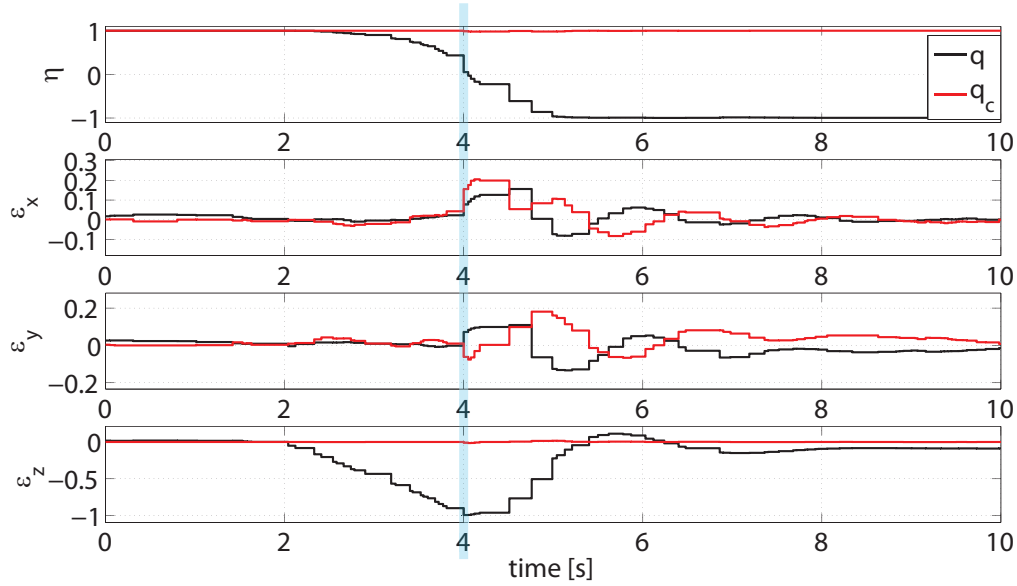

Figure 6.12: Attitude quaternion during the hand deployment maneuver. Between time $t = 2s$ and $t \simeq 4s$ the quadrotor is forced to a rotation around z axis by the human operator. At time $t \simeq 4s$ the quadrotor is deployed.

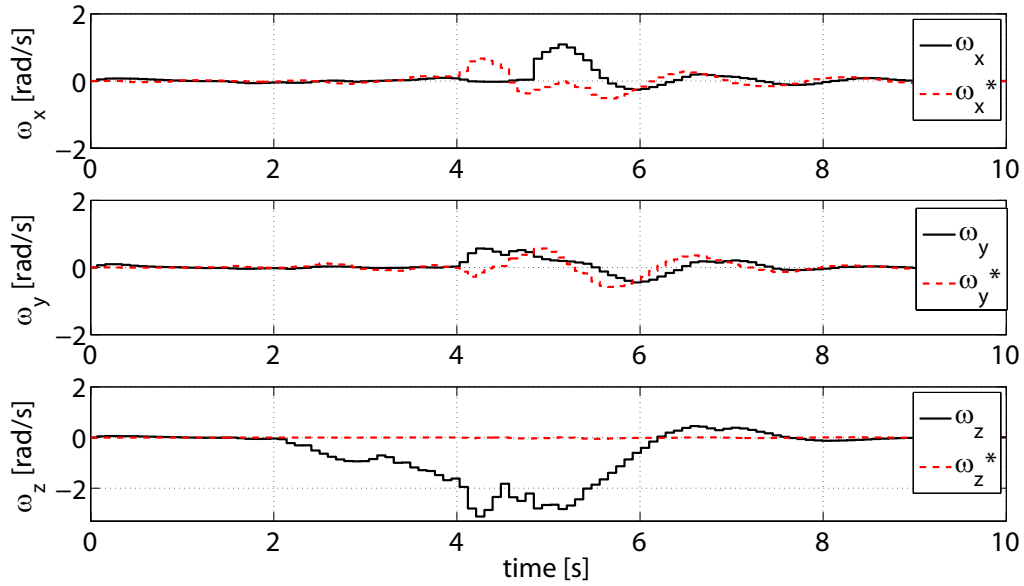

Figure 6.13: Angular speed during the hand deployment maneuver.

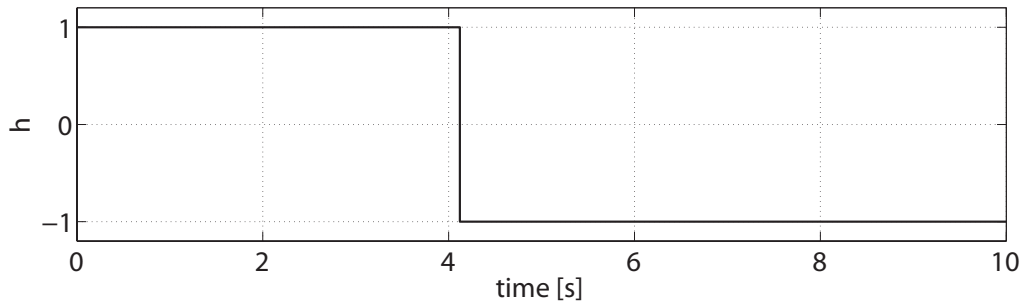

Figure 6.14: The logic state h during the hand deployment maneuver.

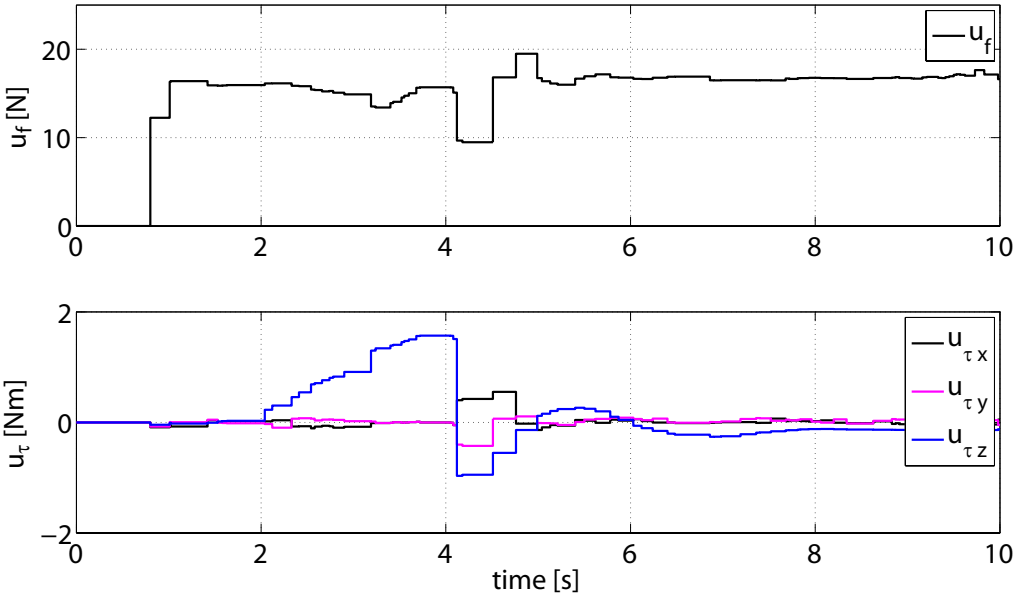

Figure 6.15: The force and torques control inputs applied to the aerial vehicle during the hand deployment maneuver.

6.2 Architecture for Control and Coordination of a Swarm of Micro-Quadrotors

The control law presented in chapter 3 for the VTOL was used in developing an architecture for control and coordination of a swarm of micro-quadrotors.

Several contributions in the literature document the effectiveness of rapid prototyping control frameworks for the development and testing of coordination and control algorithms [74] [65]. Most of these frameworks are based on motion tracking systems in which ad-hoc quadrotor platforms can be employed to perform advanced maneuvers. While the control algorithms are often released by the authors, the overall control architecture may not be directly available to other research groups. More recently, an open-source and open-hardware project, the Crazyflie [14], has proposed a miniature low-cost quadrotor platform. The vehicle can be easily piloted by a human operator though a remote computer by means of a standard joystick interface. However, the currently available software architecture does not include the guidance, control and navigation layers required to test closed-loop control algorithms and to perform advanced operations.

The goal of this section, presented in [37] is to develop an open source control framework to allow the Crazyflie nano quadrotors to be employed as a test bench for the development of advanced control and coordination algorithms. In particular, the guidance, control, and navigation layers have been designed to let a group of nano quadrotors to be controlled simultaneously using a motion tracking system. The control layer is based on the trajectory tracking controller presented in Chapter 3. This feature allows to exploit the agility of the selected nano quadrotors to perform acrobatic maneuvers, as well as to effectively recover the vehicle from an arbitrary initial configuration. The cascade structure of the controller is exploited to distribute the computation on the ground and the on-board embedded processor. More specifically, the attitude loop is implemented on the onboard processor while the outer position loop as well as the overall guidance layer are implemented on a remote ground station, which consists of a PC. The guidance layer is in charge of generating the reference position and orientation to be tracked by the cascade controller and also to coordinate multiple nano quadrotors.

The proposed control architecture is then validated by means of an experiment in which two different vehicles are required to perform a coordinated maneuver. The experiment also shows the capability of the proposed control layer to recover the vehicle from an arbitrary initial configuration, confirming the hypothesis presented in Chapter 3.

6.2.1 Crazyflie Platform

The Crazyflie nano-quadrotor is an open-source, open-hardware project developed by the company Bitcraze AB [14]. The available components [2] include the airframe, the on-board avionics hardware, a wireless communication device and the software packages that allow a human pilot to govern the vehicle by means of a standard joystick connected to a ground PC. The following two subsections describe the available hardware and software components, respectively.

Hardware

- *Crazyflie Quadrotor:* Because of its dimensions - $90\ mm$ from rotor to rotor - and flight time - up to 7 minutes - the Crazyflie falls under the nano-quadrotor category [32]. Its small weight, 19 grams, and the low-cost of all the components make it suitable to safely perform experiments without the risk of damaging expensive hardware. Moreover, the high thrust-to-weight ratio (the maximum thrust is more than 35 grams) and torque-to-inertia ratio make it suitable to perform aerobatic maneuvers. The core of the airframe is given by the Printed Circuit Board (PCB), which includes the microprocessor (an ARM Cortex $M3$), the Inertial Measurement Unit (IMU) sensor - 3-axis gyros and 3-axis accelerometers integrated in a single $MPU6050$ chip - and the power circuit for the motors. Each of the four DC current motors drives a fixed-pitch propeller having a diameter of 1.8 inches.
- *Crazyflie Wireless Communication Device:* The Crazyflie setup includes a wireless radio for bidirectional communication between the quadrotor and a ground station. The radio device, denoted as *Crazyflie Dongle*, is characterized by a frequency of 2.4 GHz and it is based on a Nordic Semiconductor $nRF24LU1$ chip. The wireless communication is based on the *Enhanced ShockBurst* protocol. The wireless device supports approximately 100 different channels simultaneously. Accordingly, the hardware architecture is suitable to employ a large number of different nano quadrotors simultaneously.

Software

The original software released by the company includes a PC application, the *Crazyflie Client*, and the on-board firmware. The two components, which are described hereafter, are specifically designed to let the vehicle be easily piloted by a human operator.

- *Crazyflie Client:* the main goal of the *Crazyflie Client* is to let the Crazyflie nano quadrotor communicate with a ground station PC. From a hardware point of view,

[2] We have considered the 6 DoF version of Crazyflie nano-quadrotor.

the communication is obtained by means of the wireless device described in the previous subsection. The *Crazyflie Client* includes a user interface for manual set-up and operations with the quadrotor. In particular, the user interface presents to the operator a number of useful information (battery status, current motor thrust, etc.) and it allows to change the flight parameters in real-time. In the standard flight mode, the *Crazyflie Client* reads the position of the four axis of a standard joystick device attached to the ground PC in order to produce, as output, the set of commands to be sent to the onboard avionics. In particular, the commands are the attitude roll and pitch angle, the yaw angular speed, and the resultant thrust to be produced by the four propeller.

- *Crazyflie Firmware:* the firmware is the code running on the on-board microcontroller. The firmware runs on a Real-Time Operative System (FreeRTOS) able to handle multiple threads. The most important components include the sensor drivers, the attitude estimation filter, the attitude control law, the communication layer, and the motor control. The attitude estimator, in particular, is based on a complementary filter proposed in [76]. The attitude control law is based on a simple PID control loop that has been implemented to stabilize the three Euler angles, i.e., the roll, the pitch and the yaw, parameterizing the attitude of the vehicle. As a consequence, the available attitude control algorithm suffers from singularities deriving from the attitude parameterization and it is not able to globally stabilize a desired angular position.

6.2.2 Motivation and Control Architecture

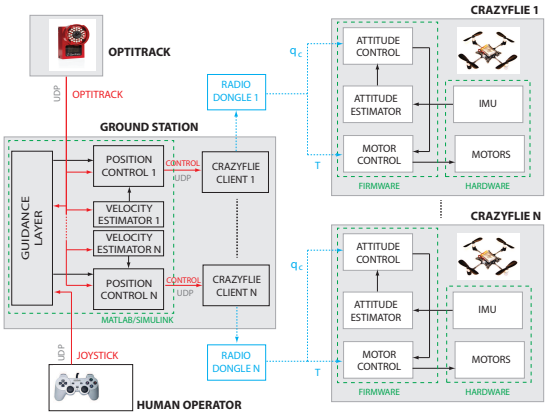

Figure 6.16: Control Framework

The idea behind this work is to define a *distributed layered control architecture*. By taking advantage of the Crazyflie platform described in Section 6.2.1, this work derives

a control architecture suitable for experimental tests of a swarm of quadrotors. The architecture can be divided into these layers:

- Control Layer is designed to achieve:
  - a cascade closed loop control scheme with decentralized computation;
  - a globally asymptotically stable controller, which means that for any possible initial condition the position and attitude error is steered to zero asymptotically;
- Navigation Layer is designed to obtain information about the state of the system: it relies on sensors (IMU, gyroscopes, Optitrack System) and estimation filters (filters for the attitude and velocity estimation);
- Guidance Layer: define a coordination protocol to control a swarm of quadrotors.

The overall control architecture is depicted in Figure 6.16 where the main interactions between hardware and software components, necessary to achieve the proposed goals, are shown. The core elements of this architecture, which are detailed in the following, are given by the Crazyflie quadrotor, the Optitrack System, the human-machine interface and the ground station.

- *Optitrack System:* is an off-the-shelf real-time motion tracking solution based on infrared cameras [100]. The set of camera with the legacy software package Tracking Tools, are able to provide attitude and position of rigid bodies at a frequency of 100 Hz. A set of infra-red reflective markers needs to be pinned to the rigid-body to be tracked (in our setup, 3 markers for each Crazyflie quadrotor). A flight arena, in which 12 infrared cameras are employed, has been used in the experiments. The resulting tracking volume is approximately given by a box of $4 \times 4 \times 2$ meters. Any tracking system could be substituted to Optitrack by sending appropriate UDP packets following the protocol described later;
- *Human-Machine Interface:* the human operator (pilot) can communicate with the ground station through a joystick. By means of the joystick the operator can interact with the guidance layer trajectory, and guarantee a safe flight termination. From a software point of view, the commands generated by the human via the joystick are processed by a control algorithm implemented using Matlab/Simulink;
- *Ground Station:* it consists of a desktop PC running Matlab/Simulink with RTWTK (Real Time Windows Target Kernel) and Crazyflie Clients. The Simulink software is employed to rapidly implement and validate all the control and coordination

algorithms. An open source library has been developed to interpret the joystick commands and data received from the Optitrack System and transmit them, via UDP protocol, to the Simulink controller.

Control Layer

A cascade control structure is chosen to reflect the intrinsic physical connection between attitude and position of the quadrotor model. This strategy allows a simple and intuitive tuning of the attitude and position controllers separately.

The cascade control structure requires frequency separation between the two loops, in which the attitude is referred as the inner loop while the position is the outer loop. Thus, the attitude controller is running on-board to fast compute attitude sensors data, while the position control is running on the ground station PC using Optitrack position data.

The outer loop layer (position) is running on the ground station PC at 100 Hz, while the inner loop layer (attitude) is running on-board on the Crazyflie quadrotor at 250 Hz. The communication between these layers is described in Section 6.2.2, while the modified control law is defined in Section 6.2.3.

Navigation Layer

The Optitrack measured yaw is sent on-board to allow yaw angle control, since the Crazyflies PCB does not embed a magnetometer and thus the on-board estimation diverges. The Optitrack System measures only the position. Hence, a high gain filter for each quadrotor is implemented on Simulink to estimate the linear velocity. The linear velocity estimation is necessary to implement the control law described in Section 6.2.3.

Also, with the aforementioned control law in mind, the original on-board attitude estimation filter is slightly modified to evaluate the attitude in terms of quaternion instead of Euler Angles.

Guidance Layer

All the high level logic (state machines, trajectory generation, swarm algorithms, etc.) can be implemented directly on Simulink, depending on the specific experiment to perform. The *Guidance Layer* is defined as the layer that aims to control the overall behavior of the system, being a single quadrotor or a swarm. In therms of output, the Guidance Layer generates reference trajectories to be tracked by the Control Layer: thus, from a functional point of view, it can be seen as a complex reference generator.

Communication Protocols

On the Ground Station PC, the communication between the layers is obtained by means of UDP protocols while the communication from the Ground Station PC to the the quadrotors is performed by means of the Crazyflie standard radio communication.

"Stream Input" and "Stream Output" Simulink blocks (Real Time Windows Target Toolbox) are used to receive input data from Optitrack and Joystick into Simulink and to send output data from Simulink to the Client, respectively. The Crazyflie standard radio communication protocol is used to send the desired attitude and thrust from the Crazyflie Client to the quadrotors on-board. Three types of messages can be defined (see Figure 6.17):

- Optitrack Message: Used to send UDP data from the Optitrack System to Simulink; a single packet is used for every quadrotor, and the ID of the vehicle is distinguished using different UDP port for every agent;
- Joystick Message: Used to send UDP data from Joystick to Simulink;
- Control Message: Used to send UDP data from Simulink to the Client.

Message Type	FIELDS											
OPTITRACK	LEN	TYPE	**X**	**Y**	**Z**	**Q0**	**Q1**	**Q2**	**Q3**	ERR	T	CRC
JOYSTICK	LEN	TYPE	**A**	**B**	**C**	**D**	**BUT**	T	CRC	-	-	-
CONTROL	LEN	TYPE	**R**	**P**	**Y**	**THR**	**YAW**	T	CRC	-	-	-

Figure 6.17: Messages Protocol Definition

Fields description:

- X, Y, Z: x, y, z positions from optitrack of the tracked quadrotor
- Q0, Q1, Q2, Q3: the four component of the attitude of the tracked quadrotor, expressed with quaternion
- A, B, C, D: the values from the four axis of joystick (from 0 to 1024)
- BUT the value of the button of joystick
- R, P, Y, THR: roll, pitch, yaw and thrust reference sent to the client.
- YAW: the actual yaw measured by the Optitrack System, to be sent to the client
- T: timestamp (data not used in the current architecture)
- ERR: mean tracking error of optitrack system (data not used in the current architecture)

- CRC: cyclic redundancy check for packets integrity check

To allow the use of multiple quadrotors with the proposed architecture, multiple Clients should be used on the Ground Station. Each quadrotor is associated to one Crazyflie Client, each one reading from a different UDP ports and communicating data on-board via a private Dongle. The client is modified to allow the scan for multiple Dongles and to be able to link every dongle to a quadrotor.

A preliminary code example, including the modified onboard firmware, the modified Client and the Simulink control scheme for multiple Crazyflies can be found in [13].

6.2.3 Global Trajectory Tracking Control Law

The position control law is slightly modified from the one in Chapter 3. In particular (3.22) becomes the following

$$\kappa(\cdot) := \lambda_3 \sigma \left(\frac{k_3}{\lambda_3} \left(\dot{\tilde{p}} + \lambda_2 \sigma \left(\frac{k_2}{\lambda_2} \tilde{p} + \lambda_1 \sigma \left(\frac{k_1}{\lambda_1} \eta_p \right) \right) \right) \right) \tag{6.2}$$

with λ_i, k_i, $i \in \{1, 2, 3\}$, control parameters to be tuned, and where η_p denotes the state of the following integrator

$$\dot{\eta}_p = \tilde{p}\,.$$

What differs is the addition of an integral action. The integrator is included to be robust to constant disturbances (as battery discharge) or to non perfect parameters estimation (for example the mass). Adding the integral action, all the stability properties are preserved, since the design of $\kappa(\cdot)$ still follows the theorem in the proof.

6.2.4 Experiments and Results

To validate the proposed distributed control law and to show the effectiveness of the overall control architecture, in this section experimental results are presented. We expect two crazyflie nano-quadrotors to track a desired trajectory, generated by the Guidance Layer. The quadrotors are hand deployed by the human operator. One nano-quadrotor is deployed mid-air with a harsh initial condition (it's deployed overturned, facing downward and with initial high linear speed) and thus has to perform an attitude recovery maneuver, assuring that the proposed control law is globally asymptotically stable. The second quadrotor is then deployed. As soon as the quadrotors are deployed and have recovered the hovering state, the desired trajectory is generated and they are requested to track such a reference.

It is important to stress that the harsh initial conditions from which the quadrotor may start, can impact gravely on a small aerial vehicle without a GAS control law: in our

case, the proposed control law guarantees that, after a small transient, the quadrotors recover the hovering condition globally.

Experimental Setup

Two Crazyflie mini-quadrotors are equipped with a carbon fiber structure, to bear markers for the Optitrack System (Figure 6.18). To avoid limitation in the thrust, the structure is designed to position the markers outside the airflow. The whole structure and markers weight is approx. 3.5 grams. This payload impacts on the total flight time, which drops from 7 to 5 and a half minutes.

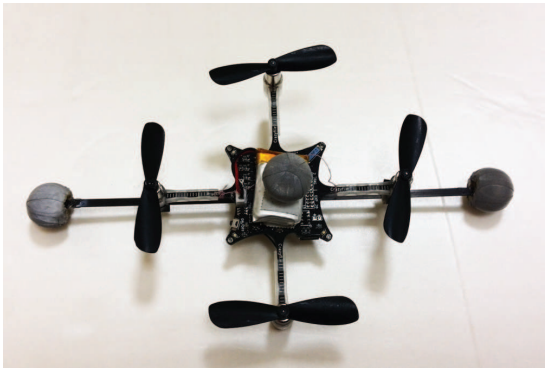

Figure 6.18: Crazyflie quadrotor with structure and markers

Guidance Layer

In this experiment, the Guidance Layer is a simple centralized trajectory generator. As soon as the two quadrotors reach hovering condition, it generates two circular path at constant height, in counter-clockwise direction, with a phase of $\pi \ \ rad$ and radius of $0.85 \ m$.

Parameters

The control parameters employed in the experiments are the following.

Table 6.1: Control parameters for the experiment

$\lambda_1 = 0.1$	$\lambda_2 = 0.15$	$\lambda_3 = 0.24$
$k_1 = 0.01$	$k_2 = 0.1$	$k_3 = 0.11$
$k_{p,x} = 60000$	$k_{p,y} = 30000$	$k_{p,z} = 40000$
$k_{d,x} = 70$	$k_{d,y} = 60$	$k_{d,z} = 130$
$\delta = 0.1$	$K_T = 8.5e^{-7}$	$K_Q = 1.3e^{-8}$

The mass is computed by adding to the weight of a Crazyflie, the weight of markers and structure. The marker's weight is approx. $1\ g$ each, while the weight of the structure is negligible. It results a final mass of $22\ g$.

The inertia tensor is computed using Huygens-Steiner theorem and considering separate masses. In particular, the frame and the marker mounted on the battery are considered as a parallelepiped of dimension 4x4x3 cm and mass $12\ g$. The inertia contribution of this parallelepiped is given by:

$$\begin{aligned} I_{xx,f} &= 13g/cm^2 \\ I_{yy,f} &= 13g/cm^2 \\ I_{zz,f} &= 32g/cm^2 \end{aligned}$$

The motors and propellers are considered as point masses of $2\ g$ at a distance of $4\ cm$ from the center of mass. The markers are considered as point masses centered at $7,5\ cm$ from the center of mass, in the y-axis. Contribution of motors to inertia is given by:

$$\begin{aligned} I_{xx,m} &= 64g/cm^2 \\ I_{yy,m} &= 64g/cm^2 \\ I_{zz,m} &= 128g/cm^2 \end{aligned}$$

Contribution of markers to inertia is given by:

$$\begin{aligned} I_{xx,m} &= 112.5g/cm^2 \\ I_{yy,m} &= 0g/cm^2 \\ I_{zz,m} &= 112.5g/cm^2 \end{aligned}$$

The total inertia is given by:

$$\begin{aligned} I_{xx} &= I_{xx,f} + I_{xx,m} + I_{xx,m} = 189.5g/cm^2 \\ I_{yy} &= I_{yy,f} + I_{yy,m} + I_{yy,m} = 77g/cm^2 \\ I_{zz} &= I_{zz,f} + I_{zz,m} + I_{zz,m} = 272.5g/cm^2 \end{aligned}$$

Results

In Figure 6.19, the attitude (reference and actual) of the first quadtotor is shown: due to the harsh initial condition, it has to perform an acrobatic maneuver (attitude recovery) to reach the desired hovering position before the trajectory starts. Between $7\ sec$ and $9\ sec$, the human operator manually turns the quadrotor downward. At time $9\ sec$, the quadrotor is deployed and in the time interval $9\ sec$ to $12\ sec$, the quadrotor recovers the attitude and reaches the requested position.

We can notice an offset in tracking the 3rd and 4th component of the quaternion dur-

ing the whole experiments, due to the fact that the real attitude is estimated on-board, while the attitude plotted in Figure 6.19 is given by the Optitrack System. A sequence of the deployment and attitude recovery maneuver can be seen in Figure 6.20. The condition (3.33) for switching the hybrid parameter h of the attitude control is fulfilled when the quadrotor is deployed. The change can be seen in Figure 6.21.

In Figure 6.22 the thrust command of first quadrotor is depicted.

In Figure 6.23 and 6.24, the position of the two quadrotors performing trajectory tracking is shown: the real position is the black line and the trajectory reference generated by the Guidance Layer is dashed red.

A video of the experiment can be seen at [87]. Finally in figure 6.25 the 3D plot of the trajectory executed by the two quadrotors. The different flight phases are clear: two bump upward are present in the deployment phase, then the circular path is performed, and a downward slope during the circular path is executed in the end of the experiment to land.

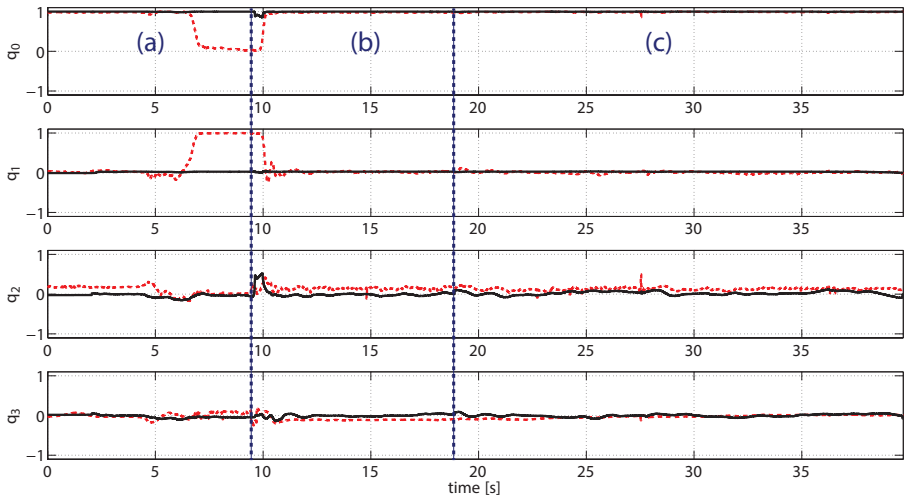

Figure 6.19: Desired (black solid) and actual (red dashed) attitude of Quadrotor n.1. (a): deployment phase. (b): recovery and stabilization phase. (c): trajectory tracking phase.

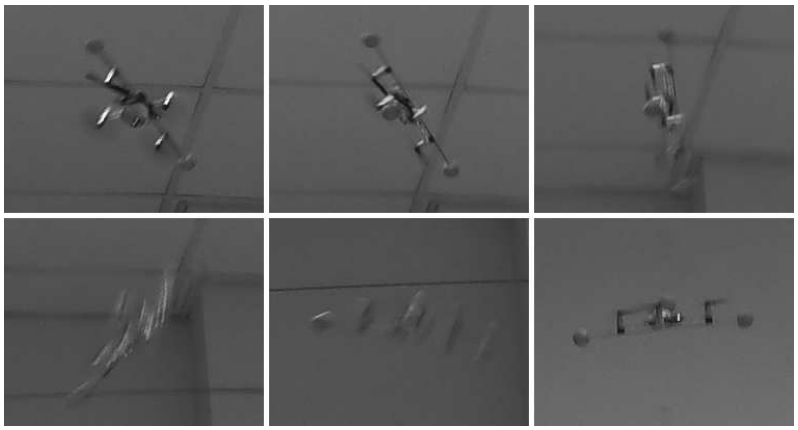

Figure 6.20: Sequence of deployment maneuver and attitude recovery

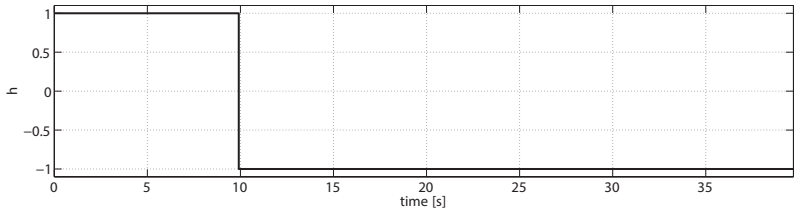

Figure 6.21: Hybrid parameter h of Quadrotor n.1

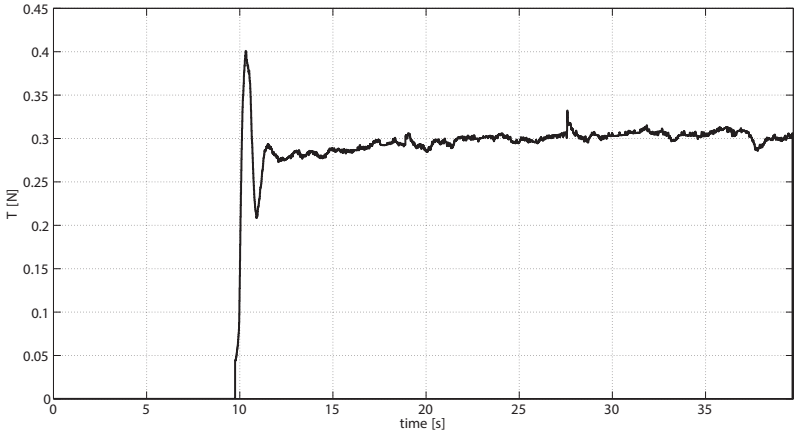

Figure 6.22: Thrust command of Quadrotor n.1

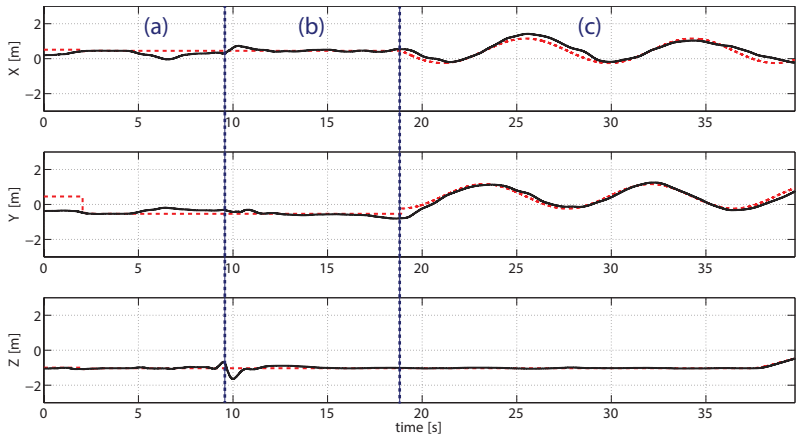

Figure 6.23: Position reference and state of Quadrotor n.1. (a): deployment phase. (b): recovery and stabilization phase. (c): trajectory tracking phase.

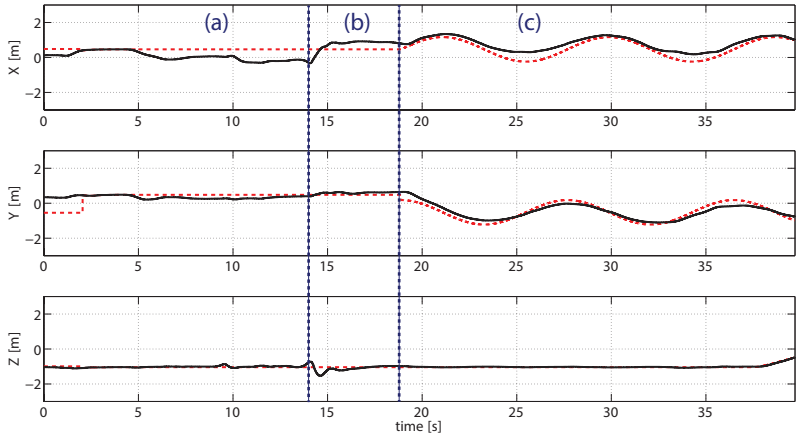

Figure 6.24: Position reference and state of Quadrotor n.2. (a): deployment phase. (b): recovery and stabilization phase. (c): trajectory tracking phase.

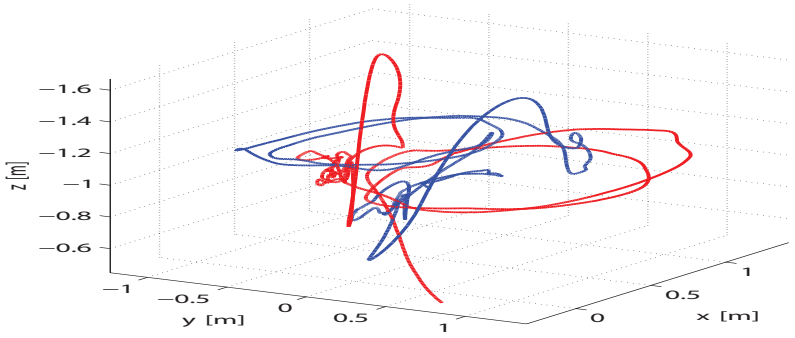

Figure 6.25: 3D plot of trajectories of both quadrotors

6.3 Other applications

The control law for the UAV was also applied to a smarthphone-based autopilot for a quadrotor [5], where an android smartphone was used to develop a complete autopilot using onboard sensors and computation.

While both the control law and the architecture for a swarm of micro-quadrotor was applied in [21], where 3 crazyflie micro-quadrotor were coordinated using consensus theory to achieve a distributed agreement on their trajectories.

The DESP planner was recently developed for parallel computation on GPU and will be soon published in proceedings of *LP-EMS16: 2nd Workshop on design of Low Power EMbedded Systems* under the title *An Energy-Efficient Parallel Algorithm for Real-Time Near-Optimal UAV Path Planning*.

7

Conclusion and Future Works

In this thesis we presented novel approaches for modeling and control of certain classes of mobile robots and novel strategies for motion planning of mobile robots. In particular, a new approach for modeling multi-propeller UAV has been developed to model VTOL UAV as modular systems composed of payloads and actuators. On the control level, two control law for VTOL UAV and differential wheel robots (DWR) were proposed. The first with the goal of globally stabilizing the VTOL, allowing for acrobatic maneuvers such as attitude recovery or barrel flips. The second to semi-globally stabilize the DWR for trajectory tracking or waypoint tracking purpose. On planning level, a new combined control and planning approach is proposed do build simple non-feasible trajectory that still guarantee safety in presence of disturbances and respect the kinodynamic differential constraints. The advantage is a simple strategy suitable for real-time due to low runtime. Moreover a general planning framework based on DES for heterogeneous robots was proposed. The proposed framework allows to do quasi-optimal planning on a symbolic representation of both the robot dynamic and the environment, reducing the planning problem to a graph-search algorithm. The result is a real-time, kinodynamic, quasi-optimal planner suitable to replanning for unknown environments.

7.1 Future Works

Many are the open questions in this thesis. In particular:

- Differential wheel robot model and control: the dynamic can be extended to the full dynamic system instead of the kinematic system, to capture a more realistic behavior. Moreover real saturation on the actuators can be added to reflect the constraints of the system.
- Multi propeller modeling: an extension to a general model where actuators trust can be directed into any direction in the 3D space can be taken into account.
- VTOL control: the control is simple, robust and easy to implement. It was tested on real application with great success, so there are no open problems on this topic.
- Tracking of piece-wise feasible trajectories: some less conservative approaches can be studied to reduce the impact of discontinuities into the tracking error.
- DESP: many open question are related to this topic being the first approach of this kind. In particular:
  - More detail on the computational complexity of the proposed algorithms.
  - More detailed use case and generalization about the use of non-controllable and non-observable events.
  - Extension to non rigid body robots, where the swath of the primitives can change based on the history of the primitives. In general an application to non mobile robots should be studied for application such as: manipulators, humanoid and articulated robots.
  - How to use other automata to coordinate the motion planning problem to other high-level problems such as complex tasks, coordination of multi-agent?
  - How to sample the environment and the dynamic of the agent (primitives) to have a complete problem?
  - How to sample the environment and the dynamic of the agent (primitives) such that the solutions are the optimal solutions of the general planning problem and not optimal for the discretized problem?

A

Appendix

A.1 Robots Control

A.1.1 Hybrid Systems: Definitions and Stability Notions

In this work, we consider hybrid systems $\mathcal{H}$ given by

$$\mathcal{H} \quad \begin{cases} \dot{x} & \in & F(x, v_c) & x \in C \\ x^+ & \in & G(x) & x \in D\,, \end{cases} \tag{A.1}$$

with state $x \in \mathbb{R}^n$ and input $v_c \in \mathbb{R}^m$ acting only on the flows. The sets $C \subset \mathbb{R}^n$ and $D \subset \mathbb{R}^n$ define the flow and jump sets, respectively, while the set-valued mappings $F : \mathbb{R}^n \times \mathbb{R}^m \rightrightarrows \mathbb{R}^n$ and $G : \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ define the flow map and jump map, respectively. For details about hybrid systems, see [45].

In the special case in which $v_c \equiv 0$, the hybrid systems considered in this paper will satisfy the *hybrid basic conditions* (see [45]), namely

(A1) The sets C and D are closed in $\mathbb{R}^n$.

(A2) The set-valued mapping $(x, 0) \mapsto F(x, 0)$ is outer semicontinuous relative to $\mathbb{R}^n \times \{0\}$ and locally bounded, and for all $x \in C$, $F(x, 0)$ is nonempty and convex.

(A3) The set-valued mapping $x \mapsto G(x)$ is outer semicontinuous relative to $\mathbb{R}^n$ and locally bounded, and for all $x \in D$, $G(x)$ is nonempty.

Solutions

Solutions to hybrid systems $\mathcal{H}$ are given by pairs of *hybrid arcs* and *hybrid inputs* defined over extended time domains called *hybrid time domains*. A set $S \subset \mathbb{R}_{\geq 0} \times \mathbb{N}$ is a hybrid time domain if, for all $(T, J) \in S$, the set $S \cap ([0, T] \times \{0, 1, ..., J\})$ can be written as

$$\bigcup_{j=0}^{J-1} ([t_j, t_{j+1}], j)$$

for some finite sequence of times $0 = t_0 \leq t_1 \leq t_2 ... \leq t_J$. A hybrid arc $x : \operatorname{dom} x \to \mathbb{R}^n$ is such that $\operatorname{dom} x$ is a hybrid time domain and, for each j, $t \mapsto x(t, j)$ is absolutely continuous on the interval $\{t : (t, j) \in \operatorname{dom} x\}$. A hybrid arc is parameterized by (t, j), where t is the ordinary-time component and j is the discrete-time component that keeps track of the number of jumps. A hybrid input $v_c : \operatorname{dom} v_c \to \mathbb{R}^m$ is such that $\operatorname{dom} v_c$ is a hybrid time domain and, for each $j \in \mathbb{N}$, the function $t \mapsto v_c(t, j)$ is Lebesgue measurable and locally essentially bounded on the interval $\{t : (t, j) \in \operatorname{dom} v_c\}$. Then, given a hybrid input $v_c : \operatorname{dom} v_c \to \mathbb{R}^m$ and an initial condition ξ, a hybrid arc $\phi : \operatorname{dom} \phi \to \mathbb{R}^n$ defines a *solution pair* (ϕ, v_c) *to the hybrid system* $\mathcal{H}$ in (A.1) if the following conditions hold:

(S0) $\xi \in \overline{C} \cup D$ and $\operatorname{dom} \phi = \operatorname{dom} v_c$ $(= \operatorname{dom}(\phi, v_c))$;

(S1) For each $j \in \mathbb{N}$ such that
$I_j := \{t : (t, j) \in \operatorname{dom}(\phi, v_c)\}$ has nonempty interior $\operatorname{int}(I_j)$, $\phi(t, j) \in C$ for all $t \in \operatorname{int}(I_j)$, and, for almost all $t \in I_j$, $\frac{d}{dt}\phi(t, j) \in F(\phi(t, j), v_c(t, j))$;

(S2) For each $(t, j) \in \operatorname{dom}(\phi, v_c)$ such that $(t, j+1) \in \operatorname{dom}(\phi, v_c)$, $\phi(t, j) \in D$ and $\phi(t, j+1) \in G(\phi(t, j))$.

A solution pair (ϕ, v_c) to $\mathcal{H}$ is said to be *complete* if $\operatorname{dom}(\phi, v_c)$ is unbounded, *maximal* if there does not exist another pair $(\phi, v_c)'$ such that (ϕ, v_c) is a truncation of $(\phi, v_c)'$ to some proper subset of $\operatorname{dom}(\phi, v_c)'$.

Stability Notions

For a hybrid system $\mathcal{H}$ with $v_c \equiv 0$, which will be denoted as $\mathcal{H}_0$, the following definition of stability will be used.

Definition A.1. *A compact set $\mathcal{A} \subset \mathbb{R}^n$ is said to be*

- stable *if for each $\epsilon > 0$ there exists $\delta > 0$ such that each maximal solution ϕ to $\mathcal{H}_0$ with $|\phi(0,0)|_{\mathcal{A}} \leq \delta$ satisfies $|\phi(t,j)|_{\mathcal{A}} \leq \epsilon$ for all $(t,j) \in \operatorname{dom}\phi$;*
- attractive *if there exists $\mu > 0$ such that every maximal solution ϕ to $\mathcal{H}_0$ with $|\phi(0,0)|_{\mathcal{A}} \leq \mu$ is complete and satisfies*

$$\lim_{(t,j)\in\operatorname{dom}\phi, t+j\to\infty} |\phi(t,j)|_{\mathcal{A}} = 0;$$

- asymptotically stable *if it is stable and attractive.*

Asymptotic stability is said to be global when the attractivity property holds for every point in $\overline{C} \cup D$.

A.1.2 Proof of Lemma 3.1

By the definition of R_c in (3.17), (3.19) and using $R_c^\top \dot{R}_c e_3 = S(\omega_c)e_3$ it follows that

$$\omega_c = \bar{G}\, R_c^\top \frac{d}{dt}\frac{v_c(\tilde{\mathbf{p}},t)}{|v_c(\tilde{\mathbf{p}},t)|} + \omega_z^* e_3$$

in which $\bar{G} \in \mathbb{R}^{3\times 3}$ is the matrix with the first, second and third rows given by $[0,\ -1,\ 0]$, $[1,\ 0,\ 0]$ and $[0,\ 0,\ 0]$, respectively, and

$$\frac{d}{dt}\frac{v_c}{|v_c|} = \left(\frac{I_3}{|v_c|} - \frac{v_c v_c^\top}{|v_c|^3}\right)\left(\dot{v}_f^* + \dot{\kappa}(\tilde{\mathbf{p}})\right).$$

By taking advantage from the nested saturation structure of $\kappa(\cdot)$ in [54] it is possible to show that $\dot{\kappa}(\cdot)$ is upper bounded by a value not dependent on $\tilde{\mathbf{p}}$. To this purpose, let $\zeta := \dot{\tilde{p}} + \lambda_1\sigma\left(k_1\tilde{p}/\lambda_1\right)$ so that $\kappa(\tilde{\mathbf{p}}) = \lambda_2\sigma(k_2\zeta/\lambda_2)$ and

$$\begin{aligned}\dot{\kappa}(\cdot) &= \frac{k_2}{M}\sigma'\left(\frac{k_2\zeta}{\lambda_2}\right)\left(-\lambda_2\sigma\left(\frac{k_2\zeta}{\lambda_2}\right) + \right.\\ &+ \left. Mk_1\sigma'\left(\frac{k_1\tilde{p}}{\lambda_1}\right)\dot{\tilde{p}} + \Gamma + d_f\right)\end{aligned}$$

from which the expressions of $\Omega_1(\cdot)$ and $\Omega_2(\cdot)$ immediately follow. The fact that $\Omega_1(0,t) = \omega^\star$ follows from (3.11) and $\sigma(0) = 0$. Finally, the fact that Ω_1 and Ω_2 are uniformly bounded by a constant $\bar{\Omega}$ follows from the fact that v_c, $\sigma(\cdot)$ and $\sigma'(\cdot)$ are bounded functions (by the definition of saturation function and by (3.14)), and that $\sigma'(k_2\zeta/\lambda_2)\dot{\tilde{p}}$ is a bounded term. As a matter of fact note that, in the computation of a bound for $\sigma'(k_2\zeta/\lambda_2)\dot{\tilde{p}}$, it is possible to assume $|\zeta| \leq \lambda_2/k_2$ (otherwise $\sigma'(k_2\zeta/\lambda_2) = 0$ by the definition of saturation function) and thus, by the definition of ζ, $|\dot{\tilde{p}}| \leq \sqrt{3}(\lambda_2 + \lambda_2/k_2)$. This completes the proof of the lemma.

A.1.3 Proof of Lemma 3.2

From Lemma 3.1, since $d_f \equiv 0$ and Γ is bounded, we have that $\omega_c(t)$ is a bounded function of time. By computing the derivative of $\omega_c(t)$ we obtain

$$\dot{\omega}_c = \bar{G}S(\omega_c)^\top R_c^\top \frac{d}{dt}\frac{v_c}{|v_c|} + R_c^\top \frac{d^2}{dt^2}\frac{v_c}{|v_c|} + \dot{\omega}_z^\star e_3$$

where the expression of $(d/dt)(v_c/|v_c|)$ is given in the proof of Lemma 3.1, and it is bounded when $d_f \equiv 0$, while

$$\begin{aligned}\frac{d^2}{dt^2}\frac{v_c}{|v_c|} &= \left(\frac{-v_c v_c^\top}{|v_c|^3} - \frac{\dot{v}_c v_c^\top + v_c \dot{v}_c^\top - 3(v_c v_c^\top)^2}{|v_c|^5}\right)\dot{v}_c + \\ &+ \left(\frac{I_3}{|v_c|} - \frac{v_c v_c^\top}{|v_c|^3}\ddot{v}_c\right)\end{aligned}$$

with $\dot{v}_c = \dot{v}_f^\star + \dot{\kappa}(\cdot)$, $\ddot{v}_c = \ddot{v}_f^\star + \ddot{\kappa}(\cdot)$. From the assumptions on the references given in Section 3.1.1 and by considering the proof of Lemma 3.1 for the special case in which $d_f \equiv 0$, we have that $\ddot{v}_f^\star$ and $\dot{\kappa}(\cdot)$ are bounded functions of time. Then the result follows by showing that, when $d_f \equiv 0$, also $\ddot{\kappa}(\cdot)$ is bounded. Specifically, we have

$$\ddot{\kappa}(\cdot) = \frac{k_2^2}{\lambda_2}\sigma''\left(\frac{k_2\zeta}{\lambda_2}\right)\dot{\zeta}^2 + k_2\sigma'\left(\frac{k_2\zeta}{\lambda_2}\right)\ddot{\zeta}$$

with

$$\begin{aligned}\dot{\zeta} &= \frac{1}{M}\left(-\kappa(\cdot) + \Gamma\right) + k_1\sigma'\left(\frac{k_1\tilde{p}}{\lambda_1}\right)\dot{\tilde{p}} \\ \ddot{\zeta} &= -\frac{1}{M}\dot{\kappa}(\cdot) + \frac{1}{M}\dot{\Gamma} + \frac{k_1^2}{\lambda_1}\sigma''\left(\frac{k_1\tilde{p}}{\lambda_1}\right)\dot{\tilde{p}}^2 + \\ &+ k_1\sigma'\left(\frac{k_1\tilde{p}}{\lambda_1}\right)\ddot{\tilde{p}}.\end{aligned}$$

When $|\zeta| > k_2/\lambda_2$, from the definition of $\sigma(\cdot)$, we have that $\sigma'(k_2\zeta/\lambda_2) = \sigma''(k_2\zeta/\lambda_2) = 0$. In the other case, when $|\zeta| \leq k_2/\lambda_2$, from the definition of ζ given in the proof of Lemma 3.1, we have that $|\dot{\tilde{p}}| \leq \sqrt{3}(\lambda_2 + \lambda_2/k_2)$. Then, by considering also the definition of Γ given in (3.21) and $\ddot{\tilde{p}}$ given by (3.20) with $d_f \equiv 0$, since $\sigma(\cdot)$, $\sigma'(\cdot)$, $\sigma''(\cdot)$ are bounded functions, all terms in the expressions of $\dot{\zeta}$ and $\ddot{\zeta}$ are bounded. This proves the lemma.

A.1.4 Computation of the rotation matrix

A rotation matrix satisfying (3.7) (or equivalently (3.17)) can be obtained parameterizing rotations using Euler angles. In fact, given $\nu = [\nu_x, \nu_y, \nu_z]^\top \in S_2$, a matrix $R' \in SO(3)$

s.t. $R'e_3 = \nu$ can be obtained as (i) $R' = R_x R_y R_z$ or (ii) $R' = R_y R_x R_z$ with

$$
\begin{aligned}
R_x &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}, \\
R_y &= \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}, \\
R_z &= \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix},
\end{aligned}
$$

where ϕ, θ, $\psi \in \mathbb{R}$. Since $R_z e_3 = e_3$ for all $\psi \in \mathbb{R}$, the value of ψ can be considered as a degree of freedom (namely the *heading direction* of the vehicle can be assigned arbitrarily) when R' is computed as in (i) or (ii). For the case (i), if $\nu_z \neq 0$ or $\nu_y \neq 0$, ϕ and θ can be obtained as

$$\phi = \arctan\left(\frac{-\nu_y}{\nu_z}\right), \theta = \text{asin}(\nu_x)\,.$$

When ν_z and ν_y are close to zero, since ν_x is different from zero from the definition of S_2, the expression of R' can be computed as in (ii) with ϕ and θ given as

$$\phi = \text{asin}(-\nu_y),\ \theta = \arctan\left(\frac{\nu_x}{\nu_z}\right).$$

A.2 Path Planning Strategies

A.2.1 Discrete Event Systems and Automata

The following are derived by [24].

Discrete Event Systems (DES), in constrast to standard Continuous-Variable Dynami Systems, obeys to two properties:

- The state space is a discrete set $X = \{s_1, s_2, s_3, ...\}$.
- The state transition mechanism is event-driven.

where events represent the input to the system. Events are instantaneous inputs that belongs to a discrete set $E = \{e_1, e_2, e_3, ...\}$. The behavior of a DES is completely described by its model and by the sequence of events $e_1 e_2 e_3 ....$ If the set of events E is called *alphabet*, this sequence is called a *word* or *string* and a collection of possible words is called the *alphabet*. The set of all finite strings made from E is called E^*. The transition function $f : X \times E \to X$ describes the evolution of the states of the DES based

on the actual state and the input (event), such that $X(i+1) = f(X(i), E(i))$, where $i \in \mathbb{N}$ represents the $i-th$ event and the $i-th$ state reached. An automaton is a "device" capable of representing a language. The easiest way is to represent a language with directed graph representation as in the example in Figure A.1

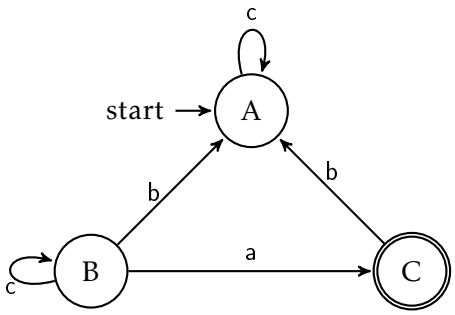

Figure A.1: Automaton Example

In this example $E = \{a, b, c\}$, $X = \{A, B, C\}$ and the transition function is defined as:

$$f(A, c) = A$$
$$f(B, a) = C$$
$$f(B, b) = A$$
$$f(B, c) = B$$
$$f(C, b) = A$$

As we can notice other information are embedded in the graph. Let's try to formalize the definition of automaton.

A Deterministic Automaton, denoted by G, is composed by a six-tuple:

$$G = (X, E, f, \Gamma, x_0, X_m)$$

with X the set of *states*, E the set of *events*, f the *transition function*, $\Gamma : X \to 2^E$ is the *active event function* (the set of all events e for which $f(x, e)$ is defined), x_0 the *initial state* and $X_m \subseteq X$ is the set of *marked states*. The *marked states* represent a set of states with a particular meaning. The language generated by an automaton is given by:

$$\mathcal{L}(\mathcal{G}) = \{\int \in \mathcal{E}^* : \{(\S_\prime, \int)\rangle \int \lceil \rceil \{\rangle \backslash \rceil \lceil\}$$

while the marked language generated by an automaton is given by:

$$\mathcal{L}_{\Updownarrow}(\mathcal{G}) = \left\{\int \in \mathcal{L}\mathcal{G} : \{(\S_\prime, \int) \in \mathcal{X}_{\Updownarrow}\right\}$$

We can make a further specification on the events E of an automaton. Events can be observable or unobservable and controllable or uncontrollable. The observability refers to the fact that an event may be not seen from a sensor for example, or not communicated. Unobservable events can be used to represent particular situation in the system we want to model with DES. Controllability of the events instead refers to the capability of our "system" to control or not that particular event. It can be strictly related to the actuators capabilities of the system we are modeling. The events E can be divided then into E_{uo} and E_o which relate to unobservable and observable events respectively or into E_{uc} and C_c which relate to uncontrollable and controllable events respectively. $E_{uo} \bigcup E_o = E = E_{uc} \bigcup E_c$. Events can be divided even into E

Operations on Automata

We want to present the operations we can perform on automata. Operations on automata changes the transition function, but doesn't alter the event set.

Accessible Part: The accessible part of an automaton G is:

$$\begin{aligned} &Ac(G) = (X_{ac}, E, f_{ac}, x_0, X_{ac,m}) \\ &X_{ac} = \{x \in X : (\exists s \in E^*)[f(x_0, s) = x]\} \\ &X_{ac,m} = X \cap X_{ac} f_{ac} = f|_{X_{ac} \times E \to X_{ac}} \end{aligned}$$

Parallel Composition: The parallel composition of two automata G_1 and G_2 is the automaton:

$$G_3 = G_1 || G_2 = Ac\left(X_1 \times X_2, E_1 \times E_2, f, \Gamma_{1||2}, (x_{01}, x_{02}), X_{m1} \times X_{m2}\right)$$

where

$$f\left((x_1, x_2), e\right) = \begin{cases} (f_1(x_1, e), f_2(x_2, e)) & if\ e \in \Gamma_1(x_1) \cap \Gamma_2(x_2) \\ (f_1(x_1, e), x_2) & if\ e \in \Gamma_1(x_1) \setminus E_2 \\ (x_1, f_2(x_2, e)) & if\ e \in \Gamma_2(x_2) \setminus E_1 \\ undefined & otherwise \end{cases}$$

that means that a "common" event is executed if both the automata can execute is (it's included in both active event functions), while a "private" event can be executed if the automaton associated to that event can execute it (it's icluded in the active event function of the automaton the event belongs to).

Bibliography

[1] Sherpa project website. http://www.sherpa-project.eu/.

[2] Sherpa project YouTube channel. www.youtube.com/user/SherpaProjectEU.

[3] *The Role of Propeller Aerodynamics in the Model of a Quadrotor UAV*, Budapest, 2009.

[4] A. Abdessameud and A. Tayebi. Global trajectory tracking control of VTOL-UAVs without linear velocity measurements. *Automatica*, 46(4):1053–1059, April 2010.

[5] L. Aldrovandi, M. Hayajineh, M. Melega, M. Furci, R. Naldi, and L. Marconi. A smartphone based quadrotor: Attitude and position estimation. In *Proceedings of the International Conference on Unmanned Aircraft Systems (ICUAS), 2015*, 2015.

[6] J. C. Alexander and J. H. Maddocks. On the kinematics of wheeled mobile robots. *Int. J. of Robotics Research*, 8:15–27, 1989.

[7] D. Angeli, Y. Chitour, and L. Marconi. Robust stabilization via saturated feedback. *IEEE Transactions on Automatic Control*, 50(12):1997 – 2014, 2005.

[8] A. Bachrach, R. He, and N. Roy. Autonomous flight in unknown indoor environments. *International Journal of Micro Air Vehicles*, 1(4):217–228, 2009.

[9] J. Barraquand, L. Kavraki, J.C. Latombe, R. Motwani, T. Li, and P. Raghavan. A random sampling scheme for path planning. *International Journal of Robotics Research*, 16(6):759–774, 1997.

[10] K. E. Bekris and L. E. Kavraki. Greedy but safe replanning under kinodynamic constraints. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2007.

[11] C. Belta, A. Bicchi, M. Egerstedt, E. Frazzoli, E. Klavins, and G.j. Pappas. Symbolic planning and control of robot motion: Finding the missing pieces of current methods and ideas. *IEEE Robotics & Automation Magazine*, pages 61–70, March 2007.

[12] S.B. Bhat and D.S. Bernstein. A topological obstruction to continuous global stabilization of rotational motion and the unwinding phenomenon. *System & Control Letters*, 39:63–70, 1999.

[13] Bitcraze. Crazyflie control architecture code. `https://sites.google.com/site/robertonaldi/research-interests/aerial-robotics`.

[14] Bitcraze. The Crazyflie nano quadrotor. `http://www.bitcraze.se/crazyflie/`.

[15] L. Blackmore. Robust path planning and feedback design under stochastic uncertainty. In *Proceedings of AIAA Guidance, Navigation and Control Conference*, 2008.

[16] L. Blackmore, H. Li, and B. C. Williams. A probabilistic approach to optimal robust path planning with obstacles. In *Proceedings of American Control Conference*, 2006.

[17] K.A. Bordignon. *Constrained Control Allocation for Systems with Redundant Control Effectors*. PhD. Thesis, Virginia Polytechnic Institute and State University, 1996.

[18] S. Bouabdallah and R. Siegwart. *Advances in Unmanned Aerial Vehicles*, chapter Chapter 6: Design and Control of a Miniature Quadrotor, pages 171–210. Springer Press, Feb. 2007.

[19] J. Canny and J. Reif. New lower bound techniques for robot motion planning problems. In *Foundations of Computer Science, 1987., 28th Annual Symposium on*, pages 49–60. IEEE, 1987.

[20] J.F. Canny. *The Complexity of Robot Motion Planning*. MIT Press, 1988.

[21] G. Casadei, M. Furci, R. Naldi, and L. Marconi. Quadrotors motion coordination using consensus principle. In *Proceedings of The 2015 American Control Conference (ACC), 2015*, 2015.

[22] P. Casau, R.G. Sanfelice, R. Cunha, D. Cabecinhas, and C. Silvestre. Global trajectory tracking for a class of underactuated vehicles. In *Proceedings of American Control Conference*, pages 419–424, Washington DC, US, 2013.

[23] A. Casavola and E. Garone. Adaptive fault tolerant actuator allocation for overactuated plants. In *Proceedings of the 2007 American Control Conference*, New York City, USA, 2007.

[24] C.G. Cassandras and S. Lafortune. *Introduction to Discrete Event System*. Springer, 2008.

[25] Z. Chen and J. Huang. Attitude tracking and disturbance rejection of rigid spacecraft by adaptive control. *Automatic Control, IEEE Transactions on*, 54(3):600–605, March 2009.

[26] R. Cunha, D. Cabecinhas, and C. Silvestre. Nonlinear trajectory tracking control of a quadrotor vehicle. In *Proc. European Control Conference*, 2009.

[27] J. Mecke D. Stoyan, W. S. Kendall. *Stochastic Geometry and Its Applications*. John Wiley & Sons, 1995.

[28] E.W. Dijkstra. A note on two problems in connexion with graph. *Numerische Mathematik*, 1:269–271, 1959.

[29] P. Doherty and P. Rudol. A UAV search and rescue scenario with human body detection and geolocalization. In Mehmet Orgun and John Thornton, editors, *AI 2007: Advances in Artificial Intelligence*, volume 4830 of *Lecture Notes in Computer Science*, pages 1–13. Springer Berlin / Heidelberg, 2007.

[30] B. Donald, P. Xavier, J. Canny, and J. Reif. Kinodynamic motion planning. *J. ACM*, 40(5):1048–1066, November 1993.

[31] J. Farrell. *Aided Navigation: GPS with High Rate Sensors*. Mc Graw Hill, 2008.

[32] E. Feron and E.N. Johnson. Aerial robotics. In B. Siciliano and O. Khatib, editors, *Springer Handbook of Robotics*, pages 1009–1027. Springer, 2008.

[33] F. Forte, R. Naldi, A. Serrani, and Marconi. Control of modular aerial robots. In *Proceedings of the 51st IEEE Conference on Decision and Control*, Maui, HI, 2012.

[34] E. Frazzoli, M. Dahleh, and E. Feron. Trajectory tracking control design for autonomous helicopters using a backstepping algorithm. *Proceedings of the American Control Conference*, pages 4102–4107, 2000.

[35] E. Frazzoli, M.A. Dahlel, and E. Feron. Maneuver-based motion planning for nonlinear systems with symmetries. *IEEE Transaction on Robotics*, 21(6):1077–1091, december 2005.

[36] M. Fumagalli, R. Naldi, A. Macchelli, R. Carloni, S. Stramigioli, and L. Marconi. Modeling and control of a flying robot for contact inspection. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012.

[37] M. Furci, G. Casadei, R. Naldi, R. Sanfelice, and L. Marconi. An open-source architecture for control and coordination of a swarm of micro-quadrotors. In *Proceedings of the International Conference on Unmanned Aircraft Systems (ICUAS), 2015*, 2015.

[38] M. Furci, R. Naldi, S. Karaman, and L. Marconi. A combined planning and control strategy for mobile robots navigation in populated environments. In *Proceedings of Decision and Control (CDC), 2015 IEEE 53rd Annual Conference on*, 2015.

[39] M. Furci, R. Naldi, and A. Paoli. A supervisory control strategy for robot-assisted search and rescue in hostile environments. *Emerging Technologies & Factory Automation (ETFA), 2013 IEEE 18th Conference on*, September 2013.

[40] M. Furci, R. Naldi, A. Paoli, and L. Marconi. A robust control strategy for mobile robots navigation in dynamic environments. In *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*, pages 698–703, 2014.

[41] M. Furci, R. Naldi, A. Paoli, and L. Marconi. Robust supervisory-based control strategy for mobile robot navigation. In *Proceedings of the 13th International Conference on Intelligent Autonomous Systems (IAS13), 2014*, 2014.

[42] F. Gaillard, M. Soulignac, C. Dinont, and P. Mathieu. Deterministic kinodynamic planning with hardware demonstrations. In *Proceedings of IEEE International Conference on Intelligent Robots and Systems*, 2011.

[43] Emanuele Garone, Roberto Naldi, and Emilio Frazzoli. Switching control laws in the presence of measurement noise. *Systems & Control Letters*, 59(6):353 – 364, 2010.

[44] V. Gavrilets, E. Frazzoli, B. Mettler, M. Piedimonte, and E. Feron. Aggressive maneuvering of small autonomous helicopters: A human-centered approach. *The International Journal of Robotics*, 20(10):795–807, 2001.

[45] R. Goebel, R. G. Sanfelice, and A. R. Teel. *Hybrid Dynamical Systems Modeling, Stability, and Robustness*. Princeton University Press, 2012.

[46] Haomiao H., G.M. Hoffmann, S.L. Waslander, and C.J. Tomlin. Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering. In

Robotics and Automation, 2009. ICRA '09. IEEE International Conference on, pages 3277–3282, 2009.

[47] T. Hamel, R. Mahony, R. Lozano, and J. Ostrowski. In *Dynamic Modelling and Configuration Stabilization for an X4-flyer*, 2002.

[48] P.E. Hart, N.J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4:100–107, 1968-2007.

[49] J. Hauser, S. Sastry, and G. Meyer. Nonlinear control design for slightly non-minimum phase systems: application to V/STOL aircraft. *Automatica*, 28(4):665–679, 1992.

[50] D. Hsu, R. Kindel, J. Latombe, and S. Rock. Randomized kinodynamic motion planning with moving obstacles, 2000.

[51] M. Hua, T. Hamel, P. Morin, and C. Samson. Introduction to feedback control of underactuated VTOL vehicles. *IEEE Control Systems Magazine*, 33(2):61–75, February 2013.

[52] M.D. Hua, T. Hamel, P. Morin, and C. Samson. A control approach for thrust-propelled underactuated vehicles and its applications to VTOL drones. *IEEE Transsactions on Automatic Control*, 54(8):1837–1853, 2009.

[53] A. Isidori. *Nonlinear Control Systems II*. Communication and Control Engineering Series. Springer–Verlag London, 1998.

[54] A. Isidori, L. Marconi, and A. Serrani. *Robust Autonomous Guidance: An Internal Model Approach*. Advances in Industrial Control. Springer-Verlag London, 2003.

[55] K. G. van den Boogaart J. Teichmann, F. Ballani. Generalizations of matérn's hard-core point process. *Spatial Statistics*, 3:33–53, 2013.

[56] Q. Jiang, D. Mellinger, C. Kappeyne, and V. Kumar. Analysis and synthesis of multi-rotor aerial vehicles. *IDETC/CIE*, 2011.

[57] T. A. Johansen and T. I. Fossen. Control allocation: A survey. *Automatica*, 49(5):1087 – 1103, 2013.

[58] Karaman and Frazzoli. Sampling-based algorithms for optimal motion planning. *International Journal of Robotics Research*, 30(7):846–894, June 2011.

[59] S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *International Journal of Robotics Research*, 30(7):846–894, June 2011.

[60] L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transaction on Robotics and Automation*, August 1994.

[61] H. K. Khalil. *Nonlinear systems*. Prentice Hall, 1996.

[62] S. Koenig and M. Likhachev. D*-lite. In *Eighteenth National Conference on Artificial Intelligence*, 2002.

[63] T. Koo and S. Sastry. Output tracking control design of a helicopter model based on approximate linearization. In *Decision and Control, 1998. Proceedings of the 37th IEEE Conference on*, volume 4, pages 3635–3640 vol.4, Dec 1998.

[64] M. Kothari and I. Postlethwaite. A probabilistically robust path planning algorithm for uavs using rapidly-exploring random trees. *Journal of Intelligent & Robotic Systems*, 71:231–253, 2013.

[65] Alex Kushleyev, Daniel Mellinger, Caitlin Powers, and Vijay Kumar. Towards a swarm of agile micro quadrotors. *Autonomous Robots*, 35(4):287–300, 2013.

[66] J.C. Latombe. *Robot Motion Planning*. Springer, 1991.

[67] S.M. LaValle and J. J. Kuffner Jr. Randomized kinodynamic planning. In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, volume 1, pages 473–479, May 1999.

[68] T. Lee, M. Leok, and N.H. McClamroch. Nonlinear robust tracking control of a quadrotor UAV on SE(3). *Asian Journal of Control*, 15(3):1–18, May 2013.

[69] J. G. Leishman. *Principles of Helicopter Aerodynamics*. Cambridge University Press, 2000.

[70] Daniel Liberzon. *Switching in Systems and Control*. Systems and Control: Foundations and Applications. Birkhauser, Boston, MA, June 2003.

[71] M. Likhachev and D. Ferguson. Planning long dynamically-feasible maneuvers for autonomous vehicles. *International Journal of Robotics Research*, pages 933–945, 2009.

[72] W. Liu, Y. Chitour, and E.D. Sontag. On finite gain stabilization of linear systems subject to input saturation. *SIAM J. Control Optim.*, 34:1190 – 1219, 1996.

[73] A. Luca, G. Oriolo, and C. Samson. Feedback control of a nonholonomic car-like robot. In J.-P. Laumond, editor, *Robot Motion Planning and Control*, volume 229 of *Lecture Notes in Control and Information Sciences*, pages 171–253. Springer Berlin Heidelberg, 1998.

[74] S. Lupashin, A. Schollig, M. Hehn, and R. D'Andrea. The flying machine arena as of 2010. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2970–2971, 2011.

[75] R. Mahony and T. Hamel. Robust trajectory tracking for a scale model autonomous helicopter. *Int. J. Robust Nonlinear Control*, 14:1035–1059, 2004.

[76] R. Mahony, T. Hamel, and J.-M. Pfimlin. Nonlinear complementary filters on the special orthogonal group. *IEEE Transactions of Automatic Control*, 53(5):1203–1218, 2008.

[77] R. Mahony, V. Kumar, and P. Corke. Multirotor aerial vehicles. modeling, estimation and control of quadrotors. *Robotics Automation Magazine*, 19(3):20–32, 2012.

[78] A. Majumdar and R. Tedrake. *Algorithmic Foundations of Robotics X*, chapter Robust Online Motion Planning with Regions of Finite Time Invariance, pages 543–558. Springer, 2013.

[79] L. Marconi and R. Naldi. Robust nonlinear full degree of fredom control of an helicopter. *Automatica*, 42:1584–1596, 2007.

[80] L. Marconi and R. Naldi. Control of aerial robots. hybrid force/position feedback for a ducted-fan. *IEEE Control System Magazine*, 32(4):43–65, 2012.

[81] P. Martin, S. Devasia, and B. Paden. A different look at output tracking: control of a VTOL aircraft. *Automatica*, 32(1):101–107, 1996.

[82] C. Mayhew, R. Sanfelice, and A. Teel. On path-lifting mechanisms and unwinding in quaternion-based attitude control. *IEEE Transactions on Automatic Control*, 58(5):1179–1191.

[83] C.G. Mayhew, R.G. Sanfelice, and A.R. Teel. Quaternion-based hybrid controller for robust global attitude tracking. *IEEE Transactions on Automatic Control*, 56(11):2555–2566, November 2011.

[84] L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys. Pixhawk: A system for autonomous flight using onboard computer vision. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2992–2997, May 2011.

[85] D. Mellinger, M. Shomin, N. Michael, and V. Kumar. Cooperative grasping and transport using multiple quadrotors. *Distributed Autonomous Robotic Systems*, 83:545–558, 2013.

[86] A. Micaelli and C. Samson. Trajectory tracking for unicycle-type and two-steering wheels mobile robots. In *INRIA, Sophia-Antipolis*, 1993.

[87] Furci Michele. Video of icuas experiment on sherpa youtube channel. `https://www.youtube.com/watch?v=oreVFqRntXY`.

[88] R.R. Murphy, K.S. Pratt, and J.L. Burke. Crew roles and operational protocols for rotary-wing micro-uavs in close urban environments. In *Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction*, HRI '08, pages 73–80, New York, NY, USA, 2008. ACM.

[89] R. Naldi, F.Forte, and L. Marconi. A class of modular aerial robots. In *Proceedings of the 50th IEEE Conference on Decision and Control*, Orlando, USA, 2011.

[90] R. Naldi, M. Furci, and L. Marconi. Modeling and control of a class of multi-propeller aerial vehicles. In *Proceedings of Workshop on Research, Education and Development of Unmanned Aerial Systems (REDUAS)*, Compiegne, FR, 2013.

[91] R. Naldi, M. Furci, R.G.. Sanfelice, and L. Marconi. Global trajectory tracking for underactuated vtol aerial vehicles using a cascade control paradigm. In *Proceedings of IEEE Conference on Decision and Control*, Florence, IT, 2013.

[92] R. Naldi, L. Gentili, L. Marconi, and A. Sala. Design and experimental validation of a nonlinear control law for a ducted-fan miniature aerial vehicle. *Control Engineering Practice*, 18(7):747–760, 2010.

[93] J. I. Neimark and N. A. Fufaev. *Dynamics of Nonholonomic Systems*. Amer. Mathem. Society, 1972.

[94] R. Oung, F. Bourgault, M. Donovan, and R. Dï£¡Andrea. The distributed flight array. In *Proceedings of the 2010 IEEE International Conference on Robotics and Automation*, Anchorage, Alaska, USA, 2010.

[95] J. J. Park and B. Kuipers. A smooth control law for graceful motion of differential wheeled mobile robots in 2d environment. In *Proceedings of Robotics and Automation (ICRA), 2011 IEEE International Conference on*, 2011.

[96] J.M. Pflimlin, P. Soueres, and T. Hamel. Hovering flight stabilization in wind gusts for ducted fan UAV. *42nd IEEE Conf. on Decision and Control*, 2004.

[97] M. Pivtoraiko and A. Kelly. Kinodynamic motion planning with state lattice motion primitives. In *Proceedings of Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, 2011.

[98] M. Pivtoraiko, R. A. Knepper, and A. Kelly. Differentially constrained mobile robot motion planning in state lattices. *Journal of Field Robotics*, 26:308–333, 2009.

[99] M. N. Pivtoraiko. *Differentially Constrained Motion Planning with State Lattice Motion Primitives*. PhD thesis, The Robotics Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, 2012.

[100] Natural Point. Optitrack motion tracking system. `www.naturalpoint.com/optitrack`.

[101] P. Pounds, R. Mahony, and P. Corke. Modelling and control of a large quadrotor robot. *Control Eng. Pract.*, 18(7):691–699, 2010.

[102] R.G. Sanfelice and E. Frazzoli. A hybrid control framework for robust maneuver-based motion planning. *American Control Conference*, pages 2254–2259, 2008.

[103] R.G. Sanfelice, R. Goebel, and A.R. Teel. Invariance principles for hybrid systems with connections to detectability and asymptotic stability. *IEEE Transactions on Automatic Control*, 52(12):2282–2297, 2007.

[104] M.D. Shuster. A survey of attitude representation. *The Journal of the Astronautical Sciences*, 41(4):439–517, December 1993.

[105] E.D. Sontag and Y. Wang. New characterizations of input-to-state stability. *IEEE Trans. Automat. Control*, 41(9):1283–1294, 1996.

[106] R.F. Stengel. *Flight Dynamics*. Princeton University Press, 2004.

[107] A.y Stentz. The focussed d* algorithm for real-time replanning. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, 1995.

[108] D. Stoyan and A. Penttinen. Recent applications of point process methods in forestry statistics. *Statistical Science*, 15:61–78, 2000.

[109] A. Teel. A nonlinear small gain theorem for the analysis of control systems with saturations. *IEEE Transactions on Automatic Control*, 41:1256–1270, 1996.

[110] B. Thwaites. *Incompressible Aerodynamics*. Oxford University Press, 1960.

[111] S.M. La Valle. *Planning Algorithms*. Cambridge University Press, 2006.

[112] J.T.-Y. Wen and K. Kreutz-Delgado. The attitude control problem. *Automatic Control, IEEE Transactions on*, 36(10):1148–1162, Oct 1991.

[113] H. Zhang, J. Butzke, and M. Likhachev. Combining global and local planning with guarantees on completeness. In *Proceedings of Robotics and Automation (ICRA), 2012 IEEE International Conference on*, 2012.