Nonlinear and Geometric Controllers for Rigid Body Vehicles

by

Adeel Akhtar

A thesis
presented to the University of Waterloo
in fulfilment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Mechanical and Mechatronics Engineering

Waterloo, Ontario, Canada, 2018

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner	Dr. Youmin Zhang Professor
Supervisor(s)	Dr. Steven L. Waslander Associate Professor
Internal Member	Dr. Kaan Erkorkmaz Professor
Internal Member	Dr. Sean Peterson Associate Professor
Internal-external	Dr. Christopher Nielsen Associate Professor

Abstract

In this thesis we investigate the motion control problem for a class of vehicles $\mathcal{C}_V$, which includes satellites, quadrotors, underwater vehicles, and tailsitters. Given a globally represented model of $\mathcal{C}_V$, and a curve, the motion control problem entails following the curve using control inputs. In this thesis the motion control problem is viewed under two settings, 1) as a local path following problem, 2) as a geometric trajectory tracking problem. We provide solutions to both problems by designing controllers based on the concept of feedback linearization.

In the local path following problem, the $\mathcal{C}_V$ class of vehicles is represented by a local chart. The problem is solved in a monolithic control setting, and the path that needs to be followed is treated as a set to be stabilized. The nonlinear model under study is first dynamically extended and then converted into a fully linear form through a coordinate transformation and smooth feedback. This approach achieves path invariance. We also design a fault tolerant local controller that ensure path following and path invariance in the presence of a one rotor failure for a quadrotor.

The second major problem addressed is the geometric trajectory tracking problem, which is treated in an inner-outer loop setting. Specifically, we design a controller class for the attitude dynamics of the $\mathcal{C}_V$ class of vehicles. The novel notion of Lie algebra valued functions are defined on the Special Orthogonal group $\mathsf{SO}(3)$, which constitutes a family of functions. This family of functions induces a novel geometric controller class, which consists of almost globally stable and locally stable controllers. This class is designed using the idea of feedback linearization, and is proven to be asymptotically stable through a Lyapunov-like argument. This allows the system to perform multiple flips. We also design geometric controllers for the position loop, which are demonstrated to work with the attitude controller class through simulations with noisy sensor data.

Acknowledgements

I would like extend my thanks first and foremost to my supervisor, Dr. Steven L. Waslander for all of his support, suggestions, our many insightful conversations, and the amazing opportunity I have had to work with him at the University of Waterloo. I could not have asked for a better arrangement. I would like to thank my committee, especially Dr. Christopher Nielsen for his help and support, and I sincerely appreciate all of his guidance. Furthermore I appreciate the love, patience, and support of my family and my parents. I would be remiss if I did not also acknowledge my friends, and my colleagues at the Waterloo Autonomous Vehicle Laboratory (WAVELab) for their help and support in completing this thesis.

Dedication

To my parents!

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The study of mobile robotics is of central importance because of its vast scale of applications in the commercial, industrial, and defense sectors. Mobile robots, as the name suggests, are capable of moving freely in the environment without being attached to a fixed base. These mobile robots can be classified, broadly, into two groups based on the environment in which they move around. In the first group, the robots' motion is restricted to planar (two-dimensional) movement. In the second group, the robots are capable of moving in three dimensional space. In this thesis we focus on certain mobile robots belonging to the second group that are capable of producing thrust along one of their body axes by a propeller, and of inducing torques about each body axis. Examples of some of the robots that fall under this category are multi-rotor robots, such as quadrotors, hexa-copters; satellites; underwater vehicles, such as submarines; and tailsitters. These mobile robots constitute a large class of vehicles, denoted by $\mathcal{C}_V$ in this thesis, and are presented in detail in Chapter 2.

An important problem in mobile robotics is to move the robot in a desired manner. In most applications, each vehicle belonging to $\mathcal{C}_V$ is required to follow a given path or trajectory in three dimensional space. In the literature, the path following or trajectory tracking problem is sometimes referred to as the motion control problem. The $\mathcal{C}_V$ class of vehicles constitutes a class of under-actuated systems because six degrees of freedom (three positions and three orientations) must be controlled by four inputs (a thrust, and three torques)[1]. This makes the motion control problem challenging.

The difference between the trajectory tracking problem and the path following problem is crucial. In the path following problem, the main task of the controller is to follow a path with no *a priori* time parameterization. In the trajectory tracking problem, the task of the controller is to follow a path with a pre-specified timing law associated with the motion. Thus, tracking can be thought of as a special case of path following. One advantage of path following is that cases exist where the trajectory tracking problem is unsolvable; yet the associated path following problem has a solution [2]. The main advantage of adopting the path following approach is that it is possible to guarantee that the resulting feedback control is invariant with respect to the path. This means that if the mobile robot is initialized on the path with the appropriate orientation, it will stay on the path for all future time. A trajectory tracking controller cannot ensure path invariance [3].

In this thesis we consider the local and geometric motion control problems for the $\mathcal{C}_V$ class of vehicles, and design controllers based on the idea of feedback linearization. We call the control problem local when the dynamics of the system under study are represented by a local chart, and geometric otherwise. Intuitively, feedback linearization is the following: given a nonlinear system, the idea is to "cancel out" all (or some) of the nonlinearities and convert the system into a linear (or partially linear) form, if possible. Mainly we solve two problems for the $\mathcal{C}_V$ class of vehicles: a path following problem when the system dynamics are represented by a local chart, and a trajectory tracking problem when the system dynamics are represented in a coordinate-free way. We call the latter the geometric tracking control problem.

In the first problem, i.e., the local path following problem, given dynamics of the $\mathcal{C}_V$ class of vehicles represented by a local chart (such as Euler angles) and a path, the goal is to follow the

[1] A similar class of vehicles are considered in [1], but we have broadened the vehicle class in this thesis.

path, while satisfying some other secondary tasks such as maintaining a specific speed profile along the path. The problem is treated in a monolithic (or unified) way, i.e., the problem is not divided into two or more stages. Moreover, using feedback linearization, we treat the problem as a set stabilization problem by treating the path to be followed as a set to be stabilized. This setting allows each vehicle in the class $\mathcal{C}_V$ to achieve path invariance. Despite attractive characteristics such as path invariance, the monolithic approach poses some practical challenges. This motivates us to formulate and solve the second major problem in this thesis.

The second problem, i.e., the geometric tracking problem, is solved using the cascade (or inner-outer loop) approach. Intuitively, the outer loop (also called the position loop) controls the system's position and velocity, while assigning a desired attitude and body rates to the inner loop (also called the attitude loop). The task of the inner loop is to "track" the desired attitude and body rates. Because of this inner-outer loop structure, it is natural to treat the inner loop problem as a trajectory tracking problem, and not as a path following problem. The inner-outer loop approach allows us to relax some of the practical limitations caused by the unified approach. The heart of inner-outer loop tracking control is the design of the inner-loop controller. Using the idea of feedback linearization, we design a novel class of geometric tracking controllers that stabilizes the attitude dynamics of the $\mathcal{C}_V$ vehicle class. After designing inner loop, we design both tracking and following controllers for the outer loop, and each of these controllers assign a desired attitude to the inner loop. In the next section we present some of the main approaches to solving motion control problems, and highlight major differences between the approaches followed in this thesis.

1.1 Literature review

Primarily the controller design of this thesis is based on the idea of feedback linearization. We treat two instances of feedback linearization: 1) the transverse feedback linearization, which allows one to solve the path following problem, and 2) the feedback linearization when the system dynamics are defined on a Lie group. First we present a literature review on feedback linearization which focuses on transverse feedback linearization and path following. Then we present some state of the art controller design techniques for one of the vehicles in the $\mathcal{C}_V$ class of vehicles, i.e., a quadrotor. We conclude the section by presenting the existing geometric control methods, and highlight some key differences with ours.

1.1.1 Feedback linearization and transverse feedback linearization

Feedback linearization is a popular technique in the field of nonlinear control, and much work has been done in the last decade in control design using this technique. Conceptually speaking, feedback linearization allows a given nonlinear system, whenever possible, to be converted into a fully linear form using coordinate transformation and feedback. More details on feedback linearization can be found at [4, 5]. In [6], the authors propose a path following method for a class of nonlinear systems which divides the path following problem into two tasks: one geometric and the other dynamic. The former forces the system to converge to the desired path, while the latter involves objectives such as tracking speed or velocity profiles. One of the drawbacks in this work is that the system must be fully feedback linearizable, i.e., differentially flat [7]. Since most mobile robots do not satisfy differential flatness, this work is not directly applicable to most mobile robots.

In exact feedback linearization, a given nonlinear system is converted into a fully linear system using a coordinate and feedback transformation. In [4, 5], the authors present the necessary and sufficient conditions for a system to be feedback linearizable. It is not always possible to fully feedback linearize all nonlinear systems. If that is the case, the system dynamics can be converted into a partially linear form via partial feedback linearization. In [8] the authors use the notion of transverse feedback linearization to linearize the dynamics of a system transverse to a closed orbit in the state space and propose a method of solving the path following problem using transverse feedback linearization.

In [9], the authors show that it is possible to make a desired path invariant through transverse feedback linearization. The authors in [10] provide the necessary and sufficient conditions for the linearization of dynamics transverse to the curve. In [11], the authors solve a path following problem for the planar vertical takeoff and landing of a fixed wing UAV for smooth Jordan curves. The proposed controller enjoys the property of path invariance. In [10], it has been shown that using transverse feedback linearization, path following can be achieved for a maglev positioning system. This control methodology is applied to a five degree-of-freedom maglev positioning system. The authors demonstrate with their experimental results the effectiveness of the control design. Path following controller design for a mechanical system is discussed in [12], where an input-output feedback linearization approach is applied to a planar five-bar linkage robot and an under-actuated five-bar robot with a flexible link. It has been shown experimentally by the authors that the behavior of a path following controller is fundamentally different from that of standard tracking control. In [13], the authors propose a path following controller for the two input kinematic model of a car-like robot. A smooth dynamic feedback control law is designed to make the car's position follow a large class of curves with a desired velocity, which guarantees invariance of the path.

The first problem considered in this thesis, i.e., local path following of $\mathcal{C}_V$, and fault tolerant path following control for quadrotors, is solved using the concept of transverse feedback linearization proposed in [9]. The key difference between our work and the work considered in [9, 12, 10, 11] is the design of dynamic feedback controllers. Moreover, as a first step of this thesis, we implement dynamic feedback controllers on a car-like robot that was proposed in our previous work [13]. To the best of the author's knowledge, these are the first experimental results of implementing a dynamic feedback linearized path following controller on a mobile robotic platform, although static transverse feedback linearized controllers were implemented on a robotic arm and a maglev system in [12] and [10], respectively. The added difficulty in the implementation of the dynamic feedback linearized controller lies in the construction of "virtual" states. Virtual states will be discussed in detail in the following chapters. Furthermore, we consider a quadrotor with a single rotor failure, and design a path following controller using dynamic transverse feedback linearization. Next we present a literature review of one of the most popular vehicles belonging to the vehicle class, the quadrotor.

1.1.2 Local quadrotor control

Broadly speaking, in literature the control design problem of a quadrotor is studied under two different settings: 1) A cascade, or an inner-outer loop setting, 2) A unified, or monolithic setting. As the name suggests, the cascade structure uses two (or more) loops. The main advantages to using this approach are the simplicity of the control design, and ease of tuning of control parameters. For quadrotors, the outer loop deals with position and velocity, and generates a reference signal for the inner loop. Given a reference attitude signal, the main task of the inner loop is to track the given reference attitude signal. Under cascade settings, a large class of linear control techniques have been proposed that rely on a plant model linearized about hover flight conditions. PD, PID or LQR controllers have all been tested on a range of platforms, and have demonstrated reliable and precise performance near hover conditions [14, 15, 16]. In [17] the authors presented PID, and LQR controller for quadrotors using the inner-outer loop approach, and demonstrate successful experimental results. Theoretically speaking, one of the main drawbacks of the inner-outer loop structure is that the stability of each loop does not guarantee stability of the overall system. The stability of the overall system is a typical stability problem of cascade control [18, 19, 20, 21], and is beyond the scope of this thesis, as it has been practically demonstrated by [17, 15] that under the inner-outer loop approach, the controller works reasonably well. In [22] the authors use feedback linearization under the cascade setting for quadrotor control design.

A wide range of nonlinear techniques have also been proposed to address the envelope restrictions inherent in linear controller designs. Backstepping [23], sliding mode [24, 25] and feedback linearization [26, 27] methods have been presented, with varying choices on the controller structure being explored to deal with the under-actuated nature of the quadrotor platform. Many researchers have been attracted to sliding mode controllers because of their robustness properties [28]. Sliding mode controller design requires finding a sliding surface on which the system exhibits desirable behavior. Once such a surface is identified, the goal is to design a feedback controller so that the system trajectories converge to the sliding surface in a finite period of time [25]. In [27], the authors compare the performance of an adaptive sliding mode controller with that of a feedback linearized controller through simulations. The authors claim that the adaptive sliding mode controller performs better than the feedback linearized controller; however, the sliding mode controller consists of a discontinuous feedback control that switches on the desired path. The switching of control occurs at an infinitely high frequency to eliminate deviations from the path. Such switching is both unrealistic and undesirable since practical systems are not infinite bandwidth and rapidly oscillating actuator commands may actually cause damage to the actuators. Several techniques have been introduced in the literature to deal with the chattering issue [29]. However, to the best of the author's knowledge, no one has claimed reliable path following experimental results on a quadrotor system using sliding mode control. Further, sliding mode path following controllers are not able to guarantee path invariance. Under a unified control scheme, most of the quadrotor controllers are trajectory tracking controllers. In [30] the authors present a path following controller for quadrotors to follow splines in the output space. The control scheme is based on the cascaded design scheme and the path is represented using Frenet-Serret (FS) frames. The resulting controller achieves path invariance.

In [31] the authors present a path following controller for a quadrotor, similar to our work presented in Chapter 4 and [32]. However, we propose the following three extensions. First, the controller allows the quadrotor to move along the path in any desired manner, including stopping along the curve as well as changing the direction of traversal along the path. Second, we fully linearize the system, and the dimension of the zero dynamics is therefore zero. Finally, both closed and non-closed curves can be used for the path definition, resulting in a more general class of paths that can be followed. One of the major limitations is that this gives controllers that suffer the so-called gimbal lock. In [33] the authors present a path following methodology combined with a obstacle avoidance scheme for quadrotors capable of working in cluttered and hazardous environments. A new cross-track error prediction based mechanism is proposed for the path following scheme. The authors show in simulation the effectiveness of the path following controller in cluttered scenarios.

1.1.3 Geometric quadrotor control

Most of the geometric control methods for quadrotors are designed based on the inner outer loop structure. The dynamics of the inner loop (or the attitude loop) is defined on the Special Orthogonal group $\mathsf{SO}(3)$, and this poses a challenge as standard nonlinear tools cannot be used for controller design. Compared to the inner loop, the controller design of the outer loop (position loop) is simpler. Under reasonable assumptions, any stable outer loop controller can work fairly well with a stable inner loop controller, while maintaining overall stability of the closed loop system. In this thesis we focus on designing novel geometric inner loop controllers. Theoretically speaking, for a general cascade control system, given each subsystem is stable, the overall stability of the system is of central importance. However, in the case of quadrotors (or $\mathcal{C}_V$ class of vehicles), this is relatively less crucial as the attitude dynamics are "faster" compared to the translational dynamics, and from a practical viewpoint the separation principle holds. Roughly speaking, the separation principle holds when the inner loop dynamics are at least three to five times faster than the outer loop dynamics.

Attitude dynamics can be represented by various parameterizations that can be Euclidean or non-Euclidean [34]. An example of a Euclidean parameterization is the famous set of Euler angles which lie in $\mathbb{R}^3$. An example of a non-Euclidean parameterization is the unit quaternion, which lie on a three sphere $\mathbb{S}^3$. As outlined in [34], regardless of the choice of parameterization, a rigid body attitude cannot be represented either globally or uniquely. As discussed earlier, a local representation, although unique, suffers singularities such as gimbal lock. On the other hand, quaternions are not unique. In other words, quaternions do not have singularities but double-covers $\mathsf{SO}(3)$, i.e., one attitude may be represented by two antipodal points. This ambiguity should be carefully resolved in quaternion-based attitude control systems. Otherwise, they may exhibit unwinding, in which a rigid body unnecessarily rotates through a large angle even if the initial attitude error is small [35]. The focus of our work presented in Chapter 6 is control design in a coordinate-free setting, i.e., when the attitude is defined on $\mathsf{SO}(3)$.

The shortcomings of attitude control using local parameterization can be overcome by nonlinear geometric control techniques. In [34] the authors consider the problem of attitude stabilization in great detail and underscore the fact that no continuous time invariant feedback controller can globally asymptotically stabilize to a desired equilibrium point. In addition, the authors present controllers for a rigid body represented in terms of both reduced attitude and full attitude. A reduced attitude control problem deals with the configuration of a rigid body defined on a two-sphere $\mathbb{S}^2$. In this case, the objective is to point a body-fixed object, such as an antenna, in a specified direction in the body fixed reference frame, where the rotation about that body fixed axis is irrelevant [34]. In our work, we focus on the full attitude control problem.

Local attitude control methods, although nonlinear, can only be applicable in a small neighborhood around the desired attitude. For unmanned aerial vehicles, such as a multi-rotor system, with a restricted flight envelope or near hover flight conditions, these local control methods are sufficient, as long as the attitude of the system is in the neighborhood of the desired attitude for all time. Local controllers fail to handle sufficiently large initial errors (such as a quadrotor initialized upside down) or large attitude errors during flight. Therefore, almost global attitude controllers are required for applications involving acrobatic maneuverability of a rigid body, such as a UAV performing single or multiple flips. In this work, we propose a class of controllers that leads to both local and almost global attitude controllers, and depending on the application, a suitable controller can be selected.

In [36] the authors investigate the Almost Global Attitude Stabilization (AGAS) problem of a rigid body on a Lie group. The authors construct potential functions $V : \mathsf{SO}(3) \to \mathbb{R}$, that satisfy certain properties to find controllers to solve the AGAS problem. A similar approach is used in [37], in which the authors present error functions for UAV control. In [38] the authors propose AGAS of a rigid satellite in a circular orbit by using Lyapunov methods to analyze closed loop almost global stability on $\mathsf{SO}(3)$. A global attitude stabilization of a rigid aircraft with unknown actuator time delay is considered in [39]. Based on Lyapunov theory, it has been proven that the controller proposed by the authors forces the trajectories of the closed loop system to converge to a small neighborhood of the origin. In [40] the authors analyze AGAS on Lie groups. They consider an optimal control problem on $\mathsf{SO}(3)$ of minimizing the distance traveled by the reduced attitude while stabilizing the full attitude. They give a two-step solution, to first stabilize the reduced attitude, and then to align the remaining two vectors by means of a planar rotation. Their work is based on potential functions, which leads to a class of error functions. In [41], the authors design a geometric tracking control of a quadrotor UAV on the Special Euclidean group, $\mathsf{SE}(3)$. The analysis is based on a globally defined model of the UAV. Through simulations, it has been shown that the quadrotor can recover from an initial almost upside down position. A quadrotor model defined using Euler angles cannot perform similarly as the UAV has to pass through a singularity point in the rotation parameterization. Similar to this work, in [42, 35, 43, 44], the authors design geometric tracking controllers using an inner-outer loop approach. Each of this work is inspired by the definition of one particular real-valued error-looking function on $\mathsf{SO}(3)$. Using this error function, controllers are designed that range from simple PID controllers to adaptive and robust geometric controllers. This is unlike our work presented in Chapter 6, in which our main contribution is the generalization of a class of error-looking functions. Moreover, our function family is not real-valued, but Lie algebra valued functions.

Primarily, the results presented in Chapter 6 are motivated by the work of Bullo and Murray [45]. In their work, the authors present a general framework for the control of fully actuated Lagrangian systems. The authors propose a geometric design algorithm for the tracking control of mechanical systems, and give a notion of an error function. The main contribution of our work is the nontrivial expansion of the class of error functions introduced in [45]. This has been achieved by relaxing some of the restrictions on the class of error functions. In particular we define a broader family of functions, such that any error function chosen from this family leads to a controller that stabilizes a rigid body. Our broader class of controllers, depending on the choice of error function, can almost globally or locally stabilize the inner loop on $\mathsf{SO}(3)$. For attitude dynamics, we highlight that in terms of the region of convergence, almost global stability is the strongest possible result.

1.2 Thesis organization and contributions

This thesis is organized as follows. After presenting a literature review and a statement of contributions in Chapter 1, we present a brief overview of global kinematic and dynamic modeling of a rigid body on $\mathsf{SO}(3)$ in Chapter 2. Then we give a summary of some of the most famous local methods of representing an element of $\mathsf{SO}(3)$, which can be used to formulate local kinematic and dynamic models of a rigid body. We conclude Chapter 2 by presenting a class of vehicles denoted by $\mathcal{C}_V$. The class $\mathcal{C}_V$ includes satellites, quadrotors, underwater vehicles, and tail-sitters. We show that each vehicle in this class can be represented by the rigid body model, and an input map. This chapter underscores that the rigid body dynamics are the central part of the mathematical description of each system in $\mathcal{C}_V$. Specifically, once a controller is designed for a rigid body, it can be easily applied to any system contained in the class $\mathcal{C}_V$.

In Chapter 3, we present an example of a path following control design for a car-like robot. This example is taken from the author's master's work and is presented in this thesis only for illustrative purposes and to support the experimental results. This chapter introduces the path following problem in a somewhat less formal and conceptual way, and highlights the fact that path following controllers can be used to achieve path invariance through the easy example of a car-like robot. This path following controller gives the desired performance in simulated environments in the absence of sensor noise. This raises a natural and important question: how will this controller perform on an actual platform? We answer this crucial question by implementing the path following controller on a Chameleon R100 robot. The controller is implemented in ROS, and the system is tested under an indoor positioning system. We conclude this chapter by presenting experimental results that demonstrate accurate path following performance on both closed and non-closed paths. This chapter highlights that these path following controllers which are based on the concept of feedback linearization can practically perform well irrespective of the need of dynamic extension[2], which require computation of virtual states in some cases. Moreover, the controller provides path invariance which is an attractive feature from the motion control point of view.

In Chapter 4, we consider the motion control problem for the class of vehicles $\mathcal{C}_V$. A local representation of the rigid body dynamics is selected for controller design. Since path following controllers are demonstrated to work well on a car-like robot, a controller, based on a similar idea, is designed for the class of vehicles $\mathcal{C}_V$. We show by a diffeomorphism that the nonlinear dynamics of the rigid body, expressed in terms of Euler angles, can be transformed into a fully linear system, which allows us to easily design a controller in a linear domain. This local controller gives each system belonging to the class $\mathcal{C}_V$ the capability to follow both closed and non-closed curves, and achieve path invariance. The controller is tested thoroughly, through simulations, on a quadrotor platform by considering almost all practical aspects of a real quadrotor platform (the *AscTec* Pelican), including sensor noise. The controller gives satisfactory performance in the presence of noise on each state. We conclude this chapter by highlighting the fact that after adding sensor noise on the inputs, the noise on one of the augmented states gets amplified, which

[2]Dynamic extension is discussed in detail in Chapter 4

degrades the performance of the controller significantly. This makes the controller infeasible for practical implementations.

In Chapter 5, we consider the fault tolerant path following control problem. We specifically consider a quadrotor in this chapter and study the case when one of the rotors of the quadrotor fails. The control design task is challenging as it is required to control the under-actuated six degree of freedom system with only three inputs. It has been shown that, without sensor noise, the quadrotor still performs the task of following the desired path with the desired speed, but is unable to follow a desired yaw profile. However, the internal dynamics of the quadrotor in the three rotor case remain bounded. This controller also requires dynamic extension, and can suffer degraded performance in the face of large input noise levels.

The aim of Chapter 6 is to select a control scheme for the class of systems that does not require dynamic extension, and computation of derivatives of the inputs. This can be achieved by adopting an inner-outer loop control approach. From the controller design prospective, the inner loop, also called the attitude loop, is the heart of the control design process, and is presented in this chapter. We propose a family of functions $\mathcal{F}_R$ that induces a class of geometric controllers $\mathcal{C}_R$ that contains both almost-global and local controllers. Since the controllers are geometric, they are designed directly on the manifold $\mathsf{SO}(3)$. This chapter provides detailed derivations of some of the controllers from this class. We prove the asymptotic stability of the whole class using Lyapunov theory, and conclude the chapter by presenting simulation results in the presence of noise. These almost-global geometric attitude controllers allow the $\mathcal{C}_V$ class of vehicles to perform multiple flips.

In Chapter 7, we design two geometric outer loop controllers: a geometric trajectory tracking outer loop controller, and a geometric path following outer loop controller. Practically speaking, under certain limitations, the modular structure allows one to pick any stable outer loop controller ranging from a simple PID controller, to nonlinear controllers, or geometric controllers to work with the controller class $\mathcal{C}_R$ of the inner loop. We discuss asymptotic stability of these outer loop controllers and test it in simulation with an inner-outer loop controller belonging to the controller class $\mathcal{C}_R$. We conclude the thesis with some future directions in Chapter 8.

1.2.1 Statement of contributions

The following is the list of original contributions made in this thesis:

1. Experimental implementation of a path following controller on a car-like robot (published in the IEEE Transactions on Robotics [46]). This involves showing:
   - Control implementation in ROS, and practically demonstrating the path following controller working on a Chameleon R100 (a car-like) robot in the presence of sensor noise, and modeling inaccuracies, for the first time.
   - A series of experiments showing the repeatability and accuracy of a path following controller that allows the robot to follow a given path with a small path following error of about 1 cm.
2. Path following for a quadrotor using dynamic extension and transverse feedback linearization (published in the IEEE Conference on Decision and Control [32]). This involves showing:
   - The $\mathcal{C}_V$ class of vehicles fail to have a well defined vector relative degree anywhere in the state space, Lemma 4.3.1.
   - The $\mathcal{C}_V$ class of vehicles can achieve a well defined vector relative degree in the neighborhood of a point via dynamic extension, Lemma 4.4.3.
   - A diffeomorphism that transforms the extended system into a fully linear system, Corollary 4.4.4.
3. Fault tolerant path following for a quadrotor (published in the IEEE Conference on Decision and Control [47]). This involves showing:

- The quadrotor system, with one fully broken rotor, has a well defined vector relative degree, Lemma 5.5.1.
- A diffeomorphism that transforms the quadrotor, with one broken rotor, into a partial linear system, Corollary 5.5.2.
- For bounded inputs, all internal states are bounded, Lemma 5.6.3.

4. A class of geometric attitude controllers of rigid bodies (to be submitted to the IEEE Transactions on Automatic Control). This involves showing:

   - Derivatives of certain log functions of product of elements of $\mathsf{SO}(3)$, Proposition 6.3.10. To the best of the author's knowledge these closed form expressions are derived for the first time in this thesis.
   - A novel notion of skew-symmetric valued error-like functions that form a family of functions $\mathcal{F}_R$, see Definition 6.4.1.
   - The family $\mathcal{F}_R$ induces a novel controller class $\mathcal{C}_R$. This novel controller class is geometric, and contains both almost-global and local controllers.
   - Two detailed step by step controller derivations (see Example 6.4.2, and Example 6.4.3) from the controller class $\mathcal{C}_R$. One controller leads to almost global results, while the other controller gives local results.
   - The main result: Lyapunov stability of the whole controller class $\mathcal{C}_R$, Theorem 6.4.5.
   - A detailed simulation study of both of these controllers from the class $\mathcal{C}_R$, that shows desired attitude tracking even in the presence of noise. Since these controllers are geometric, the system can perform multiple flips, as shown in the simulations.

1.3 Notation

In this thesis $\mathbb{R}$ denote the set of real numbers, $\mathbb{N}$, the set of natural numbers, and $\mathbb{Z}$ the set of integers. The symbol $:=$ is used to represent equal by definition. Let $\mathbb{R}^n$, where $n \in \mathbb{N}$ denotes the n–fold Cartesian product. An element $x \in \mathbb{R}^n$ is considered as an n-tuple of real numbers. Moreover, we consider $x \in \mathbb{R}^n$ denote a column vector by $\mathrm{col}(x_i, \ldots, x_n) := \begin{bmatrix} x_i & \cdots & x_n \end{bmatrix}^\top$ where $^\top$ denotes transpose.

A point-to-set distance from a point $x \in \mathbb{R}^m$ to a set $\Gamma \subset \mathbb{R}^n$ is denoted by $\|x\|_\Gamma := \inf_{p\in\Gamma} \|x - p\|$ where $\|\cdot\|$ is the Euclidean norm. Given two vectors $x, y \in \mathbb{R}^n$ the inner product is denoted by $\langle x, y\rangle$ and, when $n = 3$, the vector (cross) product is denoted by $x \times y$. Trigonometric functions are abbreviated as $S_i := \sin(x_i)$, $C_i := \cos(x_i)$ and $T_i := \tan(x_i)$. Given a C^1 map $f : \mathbb{R}^n \to \mathbb{R}^m$ and a point $p \in \mathbb{R}^n$, the Jacobian of f evaluated at p is denoted df_p. If $f, g : \mathbb{R}^n \to \mathbb{R}^n$ and $\lambda : \mathbb{R}^n \to \mathbb{R}$ are smooth, we use the following standard notation for iterated Lie derivatives $L_g^0\lambda := \lambda$, $L_g^k\lambda := L_g(L_g^{k-1}\lambda)$, $L_gL_f\lambda := L_g(L_f\lambda)$. Let $\sigma : \mathbb{R} \to \mathbb{R}^n$ represents a parameterized curve. We use the symbol $\mathbb{D}$ to denote the domain of σ. For non-closed curves $\mathbb{D} = \mathbb{R}$. For closed curves with finite length L, this means that $\mathbb{D} = \mathbb{R} \bmod L$ and σ is L-periodic, i.e., for any $\lambda \in \mathbb{D}$, $\sigma(\lambda + L) = \sigma(L)$.

Given two manifolds $\mathcal{M}$, and $\mathcal{N}$, and a function $f : \mathcal{M} \to \mathcal{N}$, for $\chi \in \mathcal{M}$, the differential of f with respect to χ is represented as $\mathrm{d}_\chi f$. The Euclidean norm of a vector $v \in \mathbb{R}^n$ is represented as $||v||$. For a matrix $A \in \mathbb{R}^{n\times n}$, the 2-norm and Frobenius norm are represented as $||A||_2$, and $||A||_F$, respectively. Let 0_n represent a column vector of dimension n where all of the elements are zero. Let I_n represent an n by n identity matrix. When the dimension is 3, we often drop the subscript for notational compactness. The trace of a matrix $A \in \mathbb{R}^{n\times n}$ is represented by $\mathrm{trace}(A)$. For any natural number $n \in \mathbb{N}$, an $n - sphere$ of radius r is defined as

$$\mathbb{S}^n := \{x \in \mathbb{R}^{n+1} : \|x\| = r\}.$$

A circle is an example of a $1-$sphere, and is denoted by $\mathbb{S}^1$.

The Special Orthogonal group $\mathsf{SO}(3)$ is the set of all orthonormal three by three matrices, with unit determinant,

$$\mathsf{SO}(3) = \left\{ R \in \mathbb{R}^{3\times 3} : R^\top R = RR^\top = I, \det(R) = 1 \right\}.$$

The manifold $\mathsf{SO}(3)$ is a matrix Lie group and its associated Lie algebra is the real vector space of 3×3 skew symmetric matrices

$$\mathfrak{so}(3) = \left\{ A \in \mathbb{R}^{3\times 3} : A = -A^\top \right\},$$

equipped with the matrix commutator $[A, B] := AB - BA$ as its binary operation, for all $A, B \in \mathfrak{so}(3)$. As a vector space, $\mathfrak{so}(3)$ is isomorphic to $\mathbb{R}^3$. The isomorphism is denoted by[3] $\widehat{\cdot} : \mathbb{R}^3 \to \mathfrak{so}(3)$ and its inverse is denoted $(\cdot)^\vee : \mathfrak{so}(3) \to \mathbb{R}^3$. For $v \in \mathbb{R}^3$, and $A \in \mathfrak{so}(3)$, we write $(\widehat{v})^\vee = ((v)^\wedge)^\vee = v$, and $((A)^\vee)^\wedge = A$. Assuming the natural basis $\{e_1, e_2, e_3\}$ for $\mathbb{R}^3$, then $\{\widehat{e}_1, \widehat{e}_2, \widehat{e}_3\}$ is a basis for $\mathfrak{so}(3)$ and, in these bases, with $v = (v_1, v_2, v_3) \in \mathbb{R}^3$ we have

$$\widehat{v} = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix}, \quad \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix}^\vee = v.$$

Routine calculations verify that for any $v, w \in \mathbb{R}^3$, $\widehat{v}w = v \times w$ where $\times$ is the usual cross product in $\mathbb{R}^3$. For $x \in \mathbb{R}^3$, and $A \in \mathbb{R}^{3\times 3}$, the following property holds [44],

$$\widehat{x}A + A^\top \widehat{x} = (\{\operatorname{trace}(A)I - A\}x)^\wedge. \tag{1.1}$$

[3]Given $v \in \mathbb{R}^3$, we write $\widehat{v}$ or equivalently $(v)^\wedge$ for the corresponding element of $\mathfrak{so}(3)$.

Chapter 2

Mathematical Modeling

In this chapter we consider motion of a rigid body in three-dimensional space, and give a brief introduction of kinematic and dynamic modeling of a rigid body. The purpose of this modeling section is not to present the most complete picture of mathematical modeling of rigid bodies, but rather an introductory background for the chapters to follow. We enlist some of the most famous ways of representing orientation of a rigid body, and highlight the advantages of common methods over the other. We conclude this chapter by presenting a class of vehicles $\mathcal{C}_V$, which include quadrotors, underwater vehicles, satellites, tail-sitter, and mono-rotors. Each system in this class can fit under the umbrella of rigid bodies, and we underscore why studying the kinematic and dynamic models of a rigid body rotating in three-dimensional space is crucial in robotics. For details about modeling of a rigid body, see [48, 49, 50].

2.1 Rigid body

A body is considered rigid if the distance between all points in the body remains fixed for all time, and under all motions (transforms). The rigid body motion can be divided into two parts, rotation, and translation. Consider a rigid body moving in free space as shown in Figure 2.1. Let

$\mathcal{I} := \{i_1, i_2, i_3\}$ denote a fixed reference frame. To specify the position and orientation of the rigid body let $\mathcal{B} := \{b_1, b_2, b_3\}$ be a frame attached to its center of mass which shares the same orientation as $\mathcal{I}$. First we look at the kinematic and dynamics of a rigid body under pure rotation.

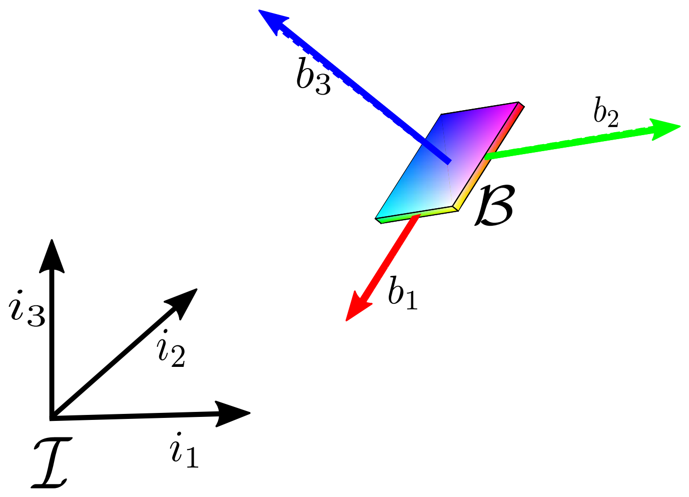


Figure 2.1: Rigid body, with inertial frame $\mathcal{I}$, and body frame $\mathcal{B}$ attached to the rigid body

2.1.1 Rotation of a rigid body

We first consider rotation (without translation) of a rigid body in space. In other words, we first present the motion of a rigid body such that there is a point in the body that stays fixed during motion. Each orientation or attitude of the rigid body is an element of the set of all orthonormal frames in the three-dimensional space with positive orientation. As shown in [51], this set can be identified with Special Orthogonal group $\mathsf{SO}(3)$. The definition of $\mathsf{SO}(3)$ readily implies that $R^{-1} = R^\top$ for each $R \in \mathsf{SO}(3)$. Throughout this thesis, we assume that the rotational matrix $R \in \mathsf{SO}(3)$ is time dependent, i.e., $R(t) \in \mathsf{SO}(3)$, unless stated otherwise. For simplification purposes, and a slight abuse of notation, we drop the time dependent argument t to represent a time varying rotation matrix by $R \in \mathsf{SO}(3)$. However, to avoid confusion in certain sections, we add the argument t with the rotation matrix, whenever necessary. It is well-known that $\mathsf{SO}(3)$ is a three-dimensional, compact, connected, embedded submanifold of $\mathbb{R}^{3\times 3}$. In other words, $\mathsf{SO}(3)$ is not only a matrix group, but also a differentiable manifold, and is called a Lie group. More detail about matrix groups, Lie groups, and differentiable manifolds can be found in [50, 5, 49], and other standard text books on differential geometry, and smooth manifolds. The matrix exponential, denoted by $\exp$, is an analytic diffeomorphism between

$$U_{\mathfrak{so}(3)} := \left\{ \widehat{\omega} \in \mathfrak{so}(3) : \omega \in \mathbb{R}^3, \|\omega\|_2 < \pi \right\}$$

and

$$U_{\mathsf{SO}(3)} := \left\{ R \in \mathsf{SO}(3) : \mathrm{trace}\,(R) \neq -1 \right\}.$$

The inverse map from $U_{\mathsf{SO}(3)} \to U_{\mathfrak{so}(3)}$ is the principle matrix logarithm and is denoted Log.

For $R(t) \in \mathsf{SO}(3)$, the first constraint of the set $\mathsf{SO}(3)$ can be written in the form of the following two equations.

$$R^\top(t)R(t) = I. \tag{2.1}$$

$$R(t)R^\top(t) = I. \tag{2.2}$$

To formulate an ordinary differential equation representing the kinematics of a rotating rigid body, we take the time derivative of both (2.1), and (2.2). By taking derivative of both sides of (2.1)

$$\begin{aligned} \frac{d}{dt}(R^\top(t)R(t)) &= \frac{d}{dt}(I) \\ \dot{R}^\top(t)R(t) + R^\top(t)\dot{R}(t) &= 0 \\ R^\top(t)\dot{R}(t) &= -\dot{R}^\top(t)R(t) \\ R^\top(t)\dot{R}(t) &= -(R^\top(t)\dot{R}(t))^\top \end{aligned}$$

It is easy to see that $R^\top(t)\dot{R}(t) \in \mathfrak{so}(3)$, or in other words, a skew symmetric matrix. This means, there exists a vector $\Omega(t) := \mathrm{col}(\Omega_1(t), \Omega_2(t), \Omega_3(t))$ such that,

$$R^\top(t)\dot{R}(t) = \widehat{\Omega}(t),$$

where $\widehat{\Omega}(t)$ represents the body angular velocity. Left multiplying both sides of the above equation by $R(t)$, we get,

$$\dot{R}(t) = R(t)\widehat{\Omega}(t). \tag{2.3}$$

The above system (2.3) represents a kinematic model of a rigid body rotation in free space, sometimes also called "left-handed" system, because of the fact that the rotation matrix acts from the left in (2.3). Similarly, by taking the derivative of (2.2)

$$\begin{aligned}
\frac{d}{dt}(R(t)R^\top(t)) &= \frac{d}{dt}(I) \\
\dot{R}(t)R^\top(t) + R(t)\dot{R}^\top(t) &= 0 \\
\dot{R}(t)R^\top(t) &= -R(t)\dot{R}^\top(t) \\
\dot{R}(t)R^\top(t) &= -(\dot{R}(t)R^\top(t))^\top \\
\dot{R}(t)R^\top(t) &= \widehat{\omega}(t),
\end{aligned}$$

for a vector $\omega(t) := \mathrm{col}(\omega_1(t), \omega_2(t), \omega_3(t))$, called spatial angular velocities. Right multiplying both sides of the above equation by $R(t)$, we get,

$$\dot{R}(t) = \widehat{\omega}(t)R(t). \tag{2.4}$$

Again, for simplification of notation we drop the time dependent argument t of $\Omega(t)$, and $\omega(t)$ to represent time-varying rates, in rest of the thesis. The above system (2.4) represents the kinematic model of a rigid body rotating in free space, sometimes also called as "right-handed" system, because of the fact that the rotation matrix acts from the right in (2.4). It should be noted that in the kinematic model (2.3), $\widehat{\Omega}(t)$ is the body angular velocity. The kinematic model (2.4) represents the same rigid body; however, the angular velocities $\widehat{\omega}$ are given in the spatial frame.

Most practical robotic systems are equipped with an Inertial Measurement Unit (IMU), which gives body angular velocities $\widehat{\Omega}(t)$. Therefore, for the rest of the thesis we consider "left-handed" kinematic model (2.3) of the rigid body. From the point of view of analysis and controller design the "right-handed" is not any different. Moreover, throughout the thesis $\widehat{\Omega}(t) \in \mathfrak{so}(3)$ represents body angular velocities, while, $\widehat{\omega}(t) \in \mathfrak{so}(3)$ represents spatial angular velocities. Body angular velocity $\widehat{\Omega}(t)$, and spatial angular velocity $\widehat{\omega}(t)$ represent the same physical quantity expressed in different reference frames. It is easy to see from (2.3), and (2.4) that the body angular velocities and spatial angular velocities are related by,

$$\widehat{\omega}(t) = R(t)\widehat{\Omega}(t)R^{\top}(t),$$

or more formally it can be written in terms of adjoint map. Given a rigid body with orientation $R(t) \in \mathsf{SO}(3)$, the transformation from body to spatial reference frames is called the adjoint map and is given by[1]

$$\begin{aligned} \mathrm{Adj}_R : \mathfrak{so}(3) &\to \mathfrak{so}(3) \\ \left(R, \widehat{\Omega}\right) &\mapsto R\widehat{\Omega}R^{\top}. \end{aligned} \tag{2.5}$$

The dynamic model of a rotating rigid body is a well studied topic, and can be derived either using energy-based approach i.e., Euler-Lagrange, or Newton-Euler method. i.e., by balancing torques. For details on dynamic modeling, the readers are referred to [50, 49, 52, 15, 14, 41, 16, 34]. Let $J \in \mathbb{R}^{3\times 3}$ represent the inertia matrix of the rigid body in the body fixed frame. Let $\tau := (\tau_p, \tau_q, \tau_r)$ denote the total moments about the body axes. Then the angular accelerations of the rigid body evolve according to

$$J\dot{\Omega}(t) = \tau(t) - (\Omega(t) \times J\Omega(t)). \tag{2.6}$$

Together equations (2.3) and (2.6) constitute a model of a rigid body rotating in free space as shown by a block diagram in Figure 2.2. It is easy to see that the states of the kinematic model

[1]Note, the argument t has been dropped for simplification purposes

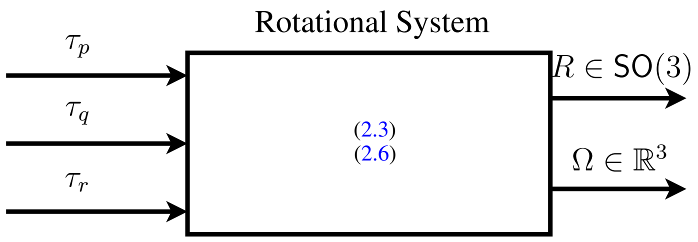


Figure 2.2: Kinematic and dynamic model of a rigid body with inputs and outputs

of the rigid body (2.3) are on a manifold, i.e., $R \in \mathsf{SO}(3)$, while the states of the dynamic model of the rigid body (2.6) are on Euclidean space, i.e., $\Omega \in \mathbb{R}^3$.

2.2 Representation of attitudes

The kinematic model of the rigid body (2.3) is an ordinary differential equation, and to control the attitude of a given rigid body one needs to work with the system given by (2.3), and (2.6). The kinematic equation (2.3) is defined on a manifold, i.e., $\mathsf{SO}(3)$. This makes analysis challenging, because we cannot directly use classical analysis tools, and standard non-linear control tool, as these tools deal with systems defined on $\mathbb{R}^n$. To over come this difficulty, it is natural to "represent" the rotation matrix R in some other from, such that in new representation of (2.3) the states are on $\mathbb{R}^n$, or a structure (manifold) simpler than $\mathsf{SO}(3)$. Ideally, we want the representation to be both global, and unique, but is it even possible? We summarize this section by answering this question, and enlisting advantages and disadvantages of representing R in a different form. For a complete overview of attitude representation see [53, 34].

It is well know that the Special Orthogonal group $\mathsf{SO}(3)$ has a group structure which is also a differentiable manifold. It is a compact three-dimensional subgroup of the 9-dimensional group $\mathrm{GL}(3, \mathbb{R})$. Each column vector of R poses three constraints, as each have unit length. Moreover, these three column vectors are orthogonal to each other, which adds three more constraints. This results in $9 - 3 - 3 = 3$ degrees of freedom. In other words, locally, a rotation matrix can be described by three independent parameters. Informally, we seek to find a map that defines some portion of $\mathsf{SO}(3)$, if not all, by three independent coordinates. Such a map is called a coordinate map, or a coordinate chart, which is an invertible map (or more precisely a homeomorphism) between an open subset of a given manifold and an open subset of $\mathbb{R}^n$.

Euler first showed that the group of rotations $\mathsf{SO}(3)$ is a three dimensional manifold. We know that by definition for a real n-dimensional manifold $\mathcal{M}$, every point of the manifold has a neighborhood homeomorphic to $\mathbb{R}^n$. Let $\mathcal{M}$ be a manifold, $\mathcal{U} \subset \mathcal{M}$, and $\mathcal{V} \subset \mathbb{R}^n$ be open sets. A homeomorphism $\Phi : \mathcal{U} \to \mathcal{V}$ is called a coordinate system on $\mathcal{U}$. The pair $(\mathcal{U}, \Phi)$ is called a chart on $\mathcal{M}$. The inverse map Φ^{-1} is a parameterization of $\mathcal{U}$. A differentiable manifold has the property that it can be covered by a collection of charts such that every point of the differentiable manifold must be represented in at least one chart. Ideally, one seek to find a single chart such that it covers the whole manifold under study, however it is not often the case. For $\mathsf{SO}(3)$, as indicated in [54], at least four charts are needed to cover $\mathsf{SO}(3)$ completely. From the point of view of controller design, if four or more charts are selected to cover the whole $\mathsf{SO}(3)$, the advantage would be a controller can be designed for every possible attitude on $\mathsf{SO}(3)$, i.e., the controller can be global. However, the main disadvantage of using a collection of charts is that each chart requires a different controller, and a switching scheme is needed to switch controllers as the chart changes. We give a brief summary of some of widely used coordinate charts in literature. Properties of different representations are summarized in Table 2.1.

2.2.1 Cardan angles

A rotation matrix can be constructed by three rotations about any three body axis such that two successive rotations are not about the same axis. This can be divided into two groups: 1) performing rotation about all three body axes, 2) performing rotation about only two body axis. The first group is often called Cardan angles, or Tait-Bryan angles, while the second group is

Table 2.1: Attitude Representation

Attitude Representation	Global	Unique
Cardan Angles	×	×
Geodesic polar coordinates	×	×
Cayley parameters	×	×
Angle-axis representation	✓	×
Unit quaternions	✓	×
Rotation Matrices	✓	✓

called Proper Euler angles. Cardan angles can be constructed by performing rotation about three different axis, and this can be achieved in six different ways $x-y-z, y-z-x, z-x-y, x-z-y, z-y-x, y-x-z$. Moreover, Proper Euler angles can also be constructed by performing rotation about two axis, and this can be achieved in six different ways $z-x-z, x-y-x, y-z-y, z-y-z, x-z-x, y-x-y$. Together, there are twelve different ways of representing a rotation matrix in terms of three different angles. An abuse of terminology is quite common in literature, and both Cardan angles and Proper Euler angles are called Euler angles. In the rest of the thesis, by Euler angles we refer to $z-y-x$ sequence of rotation. Here we present one such chart by following $z-y-x$ sequence of rotation. We define roll-pitch-yaw $(\phi, \theta, \psi) \in \mathbb{R}^3$ angles. Generally, roll is a rotation about x-axis R^x_ϕ, pitch is a rotation about y-axis R^y_θ, and yaw is a rotation about z-axis R^z_ψ. We pick a chart $\Phi_c : \mathcal{V} \subset \mathbb{R}^3 \to \mathsf{SO}(3)$

$$\Phi_c(\phi, \theta, \psi) = R^z_\psi R^y_\theta R^x_\phi$$

$$\Phi_c(\phi, \theta, \psi) = \begin{bmatrix} C_\psi & -S_\psi & 0 \\ S_\psi & C_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} C_\theta & 0 & S_\theta \\ 0 & 1 & 0 \\ -S_\theta & 0 & C_\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & C_\phi & -S_\phi \\ 1 & S_\phi & C_\phi \end{bmatrix}$$

$$R = R^z_\psi R^y_\theta R^x_\phi = \begin{bmatrix} C_\theta C_\psi & C_\psi S_\theta S_\phi - C_\phi S_\psi & C_\phi C_\psi S_\theta + S_\phi S_\psi \\ C_\theta S_\psi & S_\psi S_\theta S_\phi + C_\phi C_\psi & C_\phi S_\psi S_\theta - S_\phi C_\psi \\ -S_\theta & C_\theta S_\phi & C_\theta S_\phi \end{bmatrix},$$

where $C_i := \cos(i)$, and $S_i := \sin(i)$, for $i = \{\phi, \theta, \psi\}$. The map Φ_c is surjective since it generate the entire $\mathsf{SO}(3)$. It is easy to see that the map is not unique because it generates the same rotation matrix for each 2π multiple of roll, pitch and yaw angles. This map is 1-to-many map, for example $\Phi_c(0,0,0) = \Phi_c(2n\pi, 0, 0)$, for $\{n = 1, 2, 3, \ldots\}$. The map Φ_c becomes injective, in fact diffeomorphic [1], by setting the domain $\mathcal{V}$ to the following open set,

$$\mathcal{V} = \{(-\pi,\pi) \times (-\pi/2, \pi/2) \times (-\pi, \pi)\}.$$

A differential equation representing kinematics of a rigid body by Cardan angles representation can be found in [32, 34, 55], and will be presented in Chapter 4.

2.2.2 Cayley parameters

Similar to the Cardan angles, we seek to represent $\mathsf{SO}(3)$ by three parameters. This can be achieved by exploiting the fact that $\mathfrak{so}(3)$ is isomorphic to $\mathbb{R}^3$ via the "hat" and "inv-hat" operators. By Cayley transform, which is a mapping between skew symmetric matrices $\mathfrak{so}(3)$ and special orthogonal matrices $\mathsf{SO}(3)$, i.e.,

$$\begin{aligned} C\colon\ \mathfrak{so}(3) &\to \mathsf{SO}(3) \\ \widehat{\omega} &\mapsto (I + \widehat{\omega})(I - \widehat{\omega})^{-1}, \end{aligned}$$

for $\widehat{\omega} \in \mathfrak{so}(3)$. It is easy to check the $(I + \widehat{\omega})(I - \widehat{\omega})^{-1}$ is always orthogonal. The inverse Cayley transform C^{-1} is only defined for rotation matrices such that $\operatorname{trace}(R) \neq -1$, and is given by

$$\begin{aligned} C^{-1}\colon\ \mathsf{SO}(3) \setminus C(\mathfrak{so}(3)) &\to \mathfrak{so}(3) \\ R &\mapsto (R + I)^{-1}(R - I), \end{aligned}$$

for $R \in \mathsf{SO}(3)$. Cayley parameters are not global nor unique [54]. A differential equation representing kinematics of rigid body using Cayley parameters can be found in [55].

2.2.3 Geodesic polar coordinates

Geodesic polar coordinates is another way to map $\mathsf{SO}(3)$ to $\mathfrak{so}(3)$. It uses the idea of matrix exponential. For $t \in \mathbb{R}$, the matrix exponential is given by,

$$\begin{aligned}\exp\colon U_{\mathfrak{so}(3)} &\to \mathsf{SO}(3)\\ (t\widehat{\omega}) &\mapsto \sum_{n\geq 0} \frac{1}{n!}(t\widehat{\omega})^n.\end{aligned}$$

The inverse of matrix exponential is given by matrix log,

$$\begin{aligned}\exp^{-1} := \mathrm{Log}\colon U_{\mathsf{SO}(3)} &\to \mathfrak{so}(3)\\ R &\mapsto \sum_{n\geq 0} \frac{-1^{n+1}}{n}(R-I)^n.\end{aligned}$$

Similar to the Cayley parameters, by geodesic polar coordinates a rotation matrix can be represented by three parameters. Geodesic polar coordinates are not global nor unique [54]. A differential equation representing kinematics of a rigid body using geodesic polar coordinates can be found in [55].

2.2.4 Angle-axis representation

Another way to represent the attitude of a rigid body is by angle-axis representation. The motivation comes from the so called Euler theorem, which informally means that any rigid body undergoing a rotation can be represented by an axis (or vector) and a rotation by some angle about that axis. Let the axis be $v \in \mathbb{R}^3$, and the angle be $\theta \in \mathbb{R}$. We call the pair (v, θ) be angle-axis representation, or the exponential coordinates, i.e., $R = \exp(\widehat{v}\theta)$. It is easy to see that $R = \exp(\widehat{v}\theta) = \exp\left((-v)^\wedge(-\theta)\right)$, i.e., the map is surjective, but not injective. In other words, the representation is global but not unique. For more details on angle-axis and representing dynamics of rigid body in this form, the readers are referred to [53, 1, 34].

2.2.5 Unit quaternions

Quaternions, which can be interpreted as a vector of four elements, can be used to encode the angle-axis representation by four parameters. Clearly, this looks like an over parameterization of $\mathsf{SO}(3)$. In fact, quaternions is a double covering of $\mathsf{SO}(3)$ in a sense that each attitude corresponds to two different quaternions. Rotation matrix representation by quaternions is a well studied topic, for more details see [34, 31, 56, 53, 1]. Quaternions provide a global but non-unique parameterization of $\mathsf{SO}(3)$. Global representation of quaternions seem nice from the point of view of controller design, but the failure of uniqueness may result in unwinding effects if controllers are not designed carefully [34]. A differential equation representing kinematics of rigid body using quaternions can be found in [55, 34, 1].

Remark 2.2.1. *As seen from Table 2.1 the only way to represent the attitude of a rigid body both globally and uniquely is by rotation matrices. Therefore, from the point of view of analysis or controller design a natural choice is to represent the rigid body attitude using rotation matrices as the resulting representation is both global and unique. However, by doing so the analysis becomes challenging as the system states are not on a Euclidean space. Controller design using rotation matrices are discussed in Chapter 6. Some practical situations do not require global properties of a controller, such as a quadrotor during a near hover flight. In these situation one can represent kinematics of a rigid body using a non-global representation such as Euler angles, and design a controller. A controller design using Euler Angles is presented in Chapter 4, and 5.*

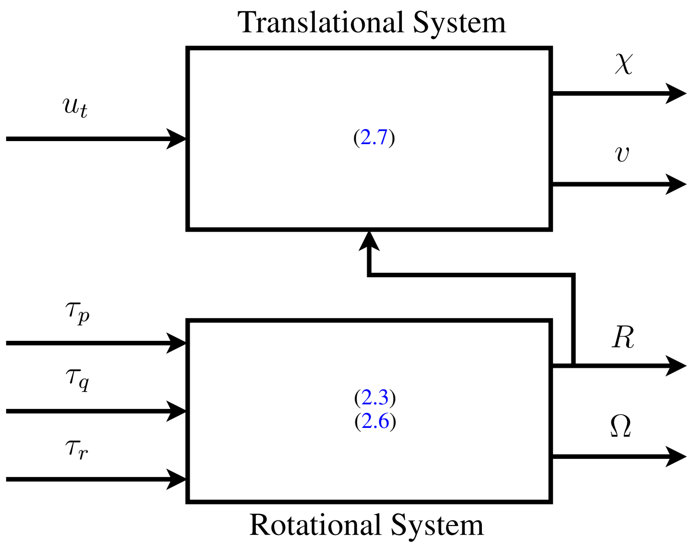


Figure 2.3: Block diagram representing a rigid body moving in free space

2.2.6 Translation of a rigid body

In this section we consider the rigid body shown in Figure 2.1, such that the body goes under some translation as well as rotation. We suppose, without loss of generality that b_3 is in the direction of the magnitude of the thrust vector, and the magnitude of thrust can be controlled externally. The thrust input is represented by u_t. It is further assumed that the vehicle is equipped with some mechanism that provides body torques about three body axis, which can be externally controlled. The input body torques τ along the body axis b_1, b_2, and b_3, are represented by τ_p, τ_q, τ_r, respectively. Let $\chi := \mathrm{col}(x_I, y_I, z_I) \in \mathbb{R}^3$ and $\dot{\chi} := \mathrm{col}(\dot{x}_I, \dot{y}_I, \dot{z}_I) \in \mathbb{R}^3$ represent the position and velocity of the vehicle in the frame $\mathcal{I}$, respectively. Let m be the mass, and g be the acceleration due to gravity. Using Newton's law, the translational model of the vehicle is given by,

$$\begin{aligned} \dot{\chi} &= v \\ m\dot{v} &= mgb_3 - u_t R b_3. \end{aligned} \tag{2.7}$$

The states of the translational model are on Euclidean space, i.e., $(\chi, v) \in \mathbb{R}^6$. It can be easily seen that (2.3), (2.6), and (2.7) are coupled differential equations, and represent a rigid body moving in free space as shown by the block diagram in Figure 2.3. A broad class of vehicles $\mathcal{C}_V$ can be represented under this description.

2.3 Class of vehicles $\mathcal{C}_V$

As shown previously, a rigid body capable of moving (i.e., rotation and translation) in free space, can be modeled globally by (2.7), (2.3), and (2.6). Let $U_{RB} := (u_t, \tau) \in \mathbb{R}^4$ be the input of the rigid body. A large class of unmanned vehicles, with four inputs, fits in this class, such as space vehicles or satellite, unmanned aerial vehicles or quadrotors, underwater vehicles, and flying wing tailsitter vehicles. Let U_i for $i = \{S, Q, UW, T\}$ be the input of a vehicle of this class, i.e., U_S, U_Q, U_{UW} and U_T be the inputs of the satellite vehicle, quadrotor, underwater vehicle, and tailsitter vehicle, respectively. We show that each vehicles of this class can fit under the system represented by (2.7), (2.3), and (2.6) by mapping the inputs of the class of vehicles $U_i \in \mathbb{R}^4$ to the inputs of the rigid body moving in free space, $U_{RB} \in \mathbb{R}^4$,

$$M_i : U_i \to U_{RB},$$

for $i = \{S, Q, UW, T\}$, such that this map is bijective. The overall scheme is shown in Figure 2.4. Next we present a short description of each vehicle and give M_i.

2.3.1 Space vehicle or satellite

A typical satellite or space vehicle, as shown in Figure 2.5, is a rigid body attached with some mechanism that is capable of providing torques τ_x, τ_y, and τ_z about all three body axis. In a typical satellite application only attitude control is required, but in some cases position control is also required. For such cases, a satellite is attached with a mechanism of providing thrust along

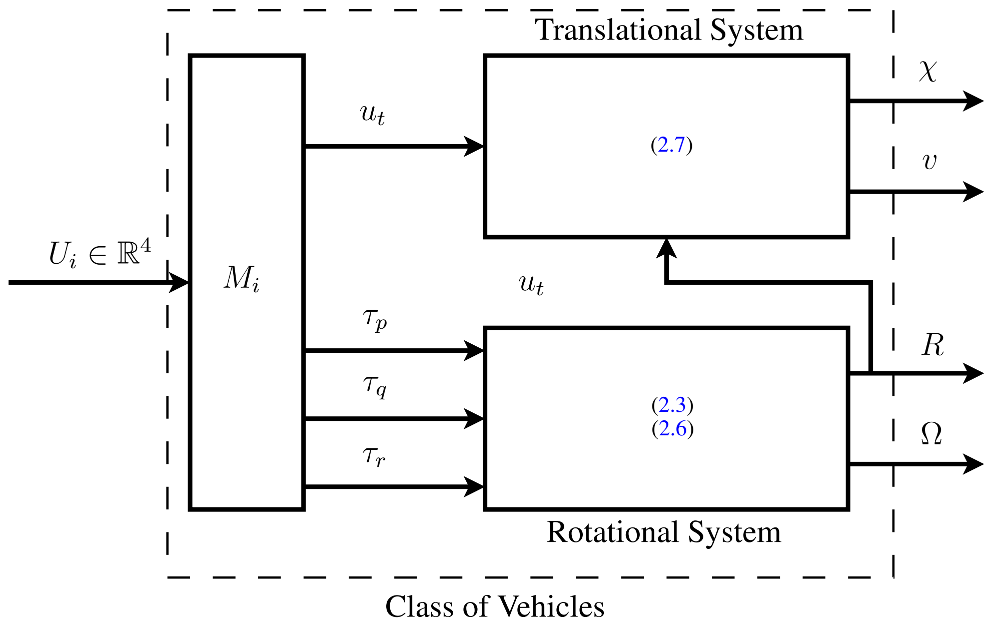


Figure 2.4: Block diagram representing class of vehicles

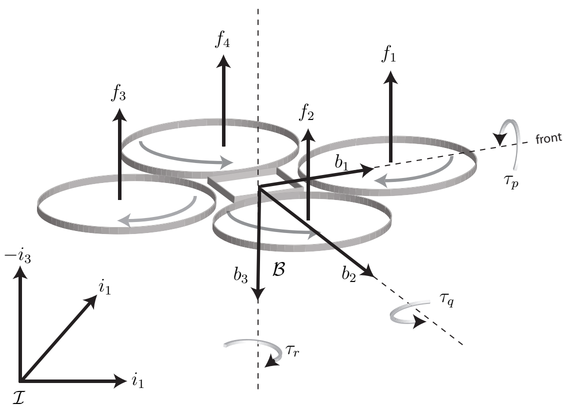
one of the body axis. Without the loss of generality, it is assumed that this thrust u_{th} is provided along z-axis of the satellite body, which together with three body torques constitute all four inputs of the satellite, i.e., $U_S = \text{col}(u_{th}, \tau_x, \tau_y, \tau_z) \in \mathbb{R}^4$. The satellite inputs are related to the inputs of the rigid body by $M_S(U_S) = U_{RB}$, which can be equivalently written as $U_{RB} = N_S U_S$, where N_S is a four by four identity matrix. Trivially, this map is invertible. With this definition of U_{RB}, the satellite is globally modeled by (2.7), (2.3), and (2.6), and therefore belong to the class of vehicles.

2.3.2 Unmanned Aerial Vehicle (UAV) or quadrotor

A quadrotor is a mechatronic system with four propellers attached with four motors which are setup in a cross configuration, as shown in Figure 2.6a, and the schematic diagram is represented in Figure 2.6b. It is a nonlinear system consisting of four inputs (the thrust provided by each pro-

Figure 2.5: A typical Satellite

(a) *AscTec Pelican*



(b) Quadrotor Model

Figure 2.6: Unmanned Aerial Vehicle, quadrotor

peller) and six degrees-of-freedom (the motion in three translational and three rotational DOFs), and is therefore an under-actuated system. Let $f_i, i \in \{1, \cdots 4\}$ represents the thrust provided by the i^{th} motor $m_i, i \in \{1, \cdots 4\}$. These four thrust forces f_i provided by each propeller constitute the input control vector $U_Q := (f_1, f_2, f_3, f_4) \in \mathbb{R}^4$ of the quadrotor system. Two diagonally opposed motors, m_1 and m_3, rotate in one direction while the other two, m_2 and m_4, rotate in the opposite direction. Because of this configuration the gyroscopic effects and aerodynamic effects tend to balance each other. Moreover, it provides an advantage over a conventional helicopter in terms of mechanical complexity, because no swashplate is needed and fixed pitch blades can be used. A positive (negative) roll moment is obtained by increasing (reducing) the speed of m_2 and reducing (increasing) the speed of m_4. This allows the quadrotor to move in the y direction. Similarly, a positive (negative) pitch movement is obtained by increasing (reducing) the speed of m_3 and reducing (increasing) the speed of m_1. This allows the quadrotor to move in the x direction. Yaw movement is obtained by increasing (decreasing) the speed of the diagonal motors. The quadrotor inputs are related to the inputs of the rigid body by $M_Q(U_Q) = U_{RB}$, which can be equivalently written as $U_{RB} = N_Q U_Q$, where

$$N_Q = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & -l & 0 & l \\ -l & 0 & l & 0 \\ d & -d & d & -d \end{bmatrix}, \tag{2.8}$$

and, l is the distance from the center of mass to the rotors, d is the ratio between the drag and the thrust coefficients of the blade. With the above definition of U_{RB}, the quadrotor is globally modeled by (2.7), (2.3), and (2.6), and fits under the class of rigid bodies. It is easy to verify that the map M_Q is invertible. For more details about quadrotor modeling the readers are referred to [14, 52, 15, 44, 16, 57].

A quadrotor system is shown in 2.6a called *AscTec*[2] Pelican[3]. The vehicle dimension is 651 x 651 x 188 mm. It is equipped with a real-time autopilot board that has the capability to communicate with an onboard Intel Core i7 embedded computer called mastermind.[4] The mastermind computer is capable of running a standard Linux distribution and Robot Operating System (ROS)[5], and communicates with autopilot board via a UART connection [58, 59]. The autopilot consists of two ARM micro-controllers. One is called a closed source Low Level Processor (LLP), and the other is called a open source High Level Processor (HLP). The pelican

[2] Ascending Technologies, is a part of Intel.

[3] `http://www.asctec.de/en/uav-uas-drones-rpas-roav/asctec-pelican/`

[4] The onboard computer is AscTec Mastermind.

[5] ROS

system is equipped with standard sensors such as Inertial Measurement Unit (IMU), that is capable of giving attitude information of the pelican. The IMU data, i.e., R, Ω, is first processed and filtered at LLP and then transmitted to HLP at 1 kHz for control and other purposes, which can be further accessed by mastermind in ROS. The sensors and overall communication structure is shown in Figure 2.7[6]. The position and velocity of the pelican system is measured by a precise

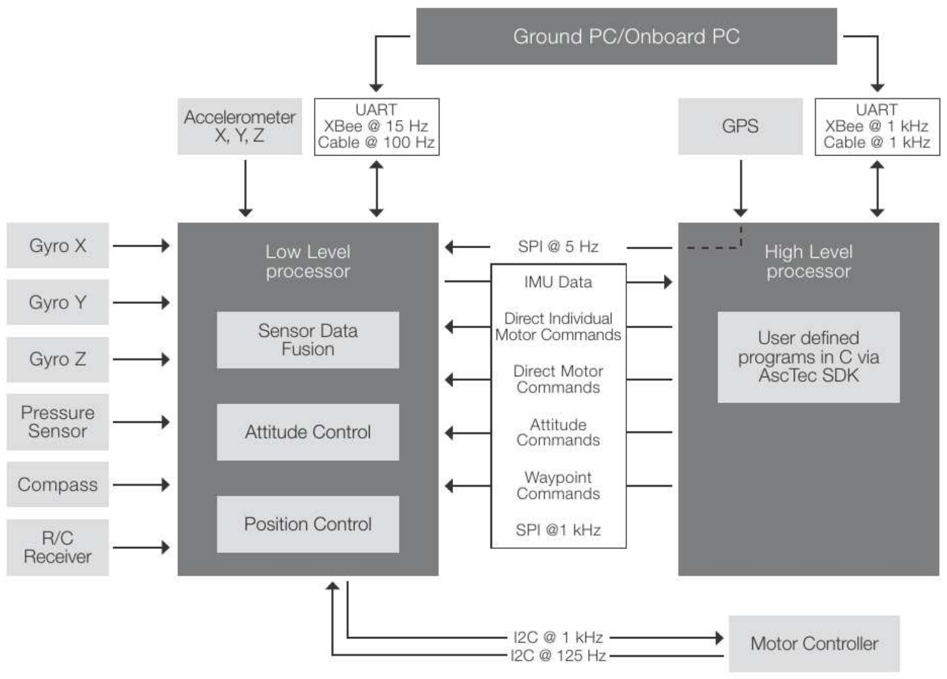
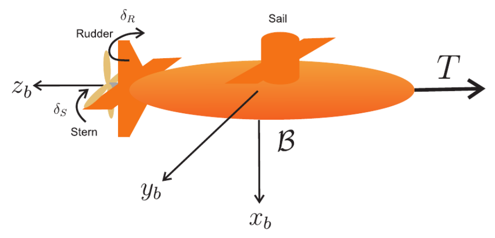


Figure 2.7: *AscTec Pelican's* sensors and computers

Indoor Positioning System (IPS) at 100 Hz using a set of 16 Vantage camera system and Vicon motion capture system. The position and velocity data is sent to the onboard mastermind via a wireless router. In summary, all the state information of the pelican system is accessible to the mastermind in ROS. We will consider practical aspect of the *AscTec* pelican such as sensor update rates, and sensor noise for simulations in the chapters to follow. The *AscTec* pelican system is attached with four motors, which can be controlled by a Pulse Width Modulated (pwm) signal

[6]The picture is taken from AscTec documentation

(a) Bluefin SandShark Autonomous Underwater Vehicle by General Dynamics



(b) Under Water Vehicle [1]

Figure 2.8: An Under Water Vehicle (UWV) or submarine

ranging between 0 to 200. The output of each motor is motor speed ω_{m_i}, for $i = \{1, \cdots, 4\}$. We observed by performing the standard system identification of the motor that the motor dynamics are fast enough to be ignored for all practical purposes. Motor speeds ω_{m_i} are measured by hall effect sensors on the pelican system, and the motor speed data is available at HLP and mastermind at an update rate of up to 1 kHz. It should be noted that the inputs of the quadrotor system are forces f_i generated by motor m_i, however, the inputs of *AscTec* pelican is pwm. Experimentally, we determine a relationship between input of each motor pwm and the resulting force generated by each motor by attaching the pelican with a heavy (10 kg) weight. The pelican attached with the weight was placed on a precision scale and the pelican was given pwm values from 0 to 200 at a step of 5 units. Reading from the precision scale and the input values give a relation between input, i.e., pwm and force generated by each motor,

$$f_i = 0.053 + 0.000030153 pwm_i^2. \tag{2.9}$$

The controllers designed for rigid bodies in the chapters to follow will be tested on a quadrotor system in simulation by considering actuator limitations, sensor noise, and sensor update rate.

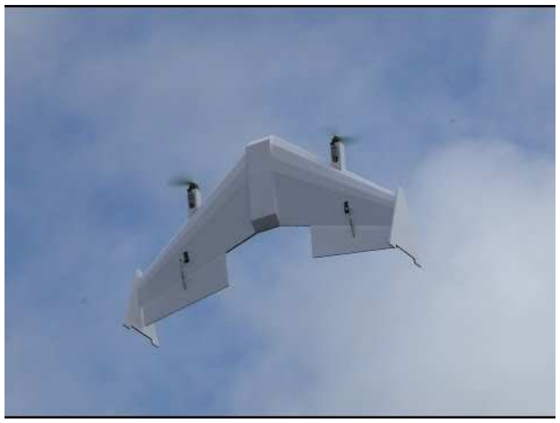
2.3.3 Under Water Vehicle (UWV) or submarine

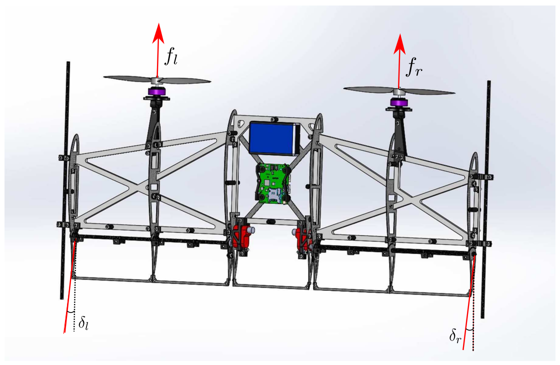
A submarine or a UWV is illustrated in Figure 2.8b, and Figure 2.8a. Generally, a submarine system such as Bluefin SandShark, is equipped with a pumping system that fills water in a water tank attached to the vehicle, which makes the vehicle go down in the water. Similarly, the vehicle can be brought back to the surface of the water by pumping the water out of the water tank. As seen in the Figure 2.8a a propeller is attached at the back of the vehicle that produces a thrust $u_{thrust} \in \mathbb{R}$ along the z-axis. The rear end of the vehicle is attached with a rudder and stern system which is capable of applying torques about the body axis. Let the axial position of the rudder be x_R, the density of water be ρ, the rudder lift coefficient be c_L, the rudder platform area be S_R, the rudder angle be δ_R, and the effective rudder speed be v_R. Similarly, the axial position of the stern is x_S, the stern platform area is S_S, the stern angle is δ_S, and the effective stern speed is v_S. As shown in [1] the torque induced about x_b-axis, and y_b-axis are given by $\tau_x = \frac{1}{2}x_R\rho c_L S_R\delta_R v_R^2$, and $\tau_y = \frac{1}{2}x_S\rho c_L S_S\delta_S v_S^2$, respectively. For the vehicle under study, the torque is not applied about the z-axis (i.e., $\tau_z = 0$). Instead, the vehicle remains upright by having more weight at the bottom. The vehicle is balanced by the stern and sail planes [1]. Similar to the previous case, the inputs of the under water vehicle can be mapped to the inputs of the rigid body, i.e., (2.7), (2.3), and (2.6) by

$$
\begin{aligned}
u_t &= u_{thrust} \\
\tau_p &= \tau_x \\
\tau_q &= \tau_y \\
\tau_r &= \tau_z = 0.
\end{aligned}
$$

2.3.4 Flying wing tailsitter vehicle

A flying wing tail-sitter is equipped with two rotors on the top left and top right part of the vehicle as shown in Figure 2.9a, and 2.9b. More details can be found in [60]. We assume zero


(a) Unmanned tail-sitter during a flight



(b) Schematic model

Figure 2.9: Tail-sitter

free stream velocity. Moreover, we assume that the left propeller rotates counter-clockwise, and the right propeller clockwise. Each rotor produces a force f_r, and f_l which combined together produces a thrust force along the body z-axis b_3. The tail-sitter has two flaps at the bottom part which produces an aerodynamic force and torque by deflecting the airflow behind the wings. The left and right flap angles are represented by δ_r, and δ_l, respectively. The four input of the tail-sitter are $U_T := (f_r, f_l, \delta_r, \delta_l) \in \mathbb{R}^4$. Let $\tau = (\tau_p, \tau_q, \tau_r) \in \mathbb{R}^3$ be the total applied torque on the tail-sitter, with τ_p, τ_q, and τ_r are the torques about body axis b_1, b_2, and b_3, respectively. This torque can be divided into two aerodynamic torque τ_{aero}, and propeller torque τ_{prop},

$$\tau = \tau_{aero} + \tau_{prop}. \tag{2.10}$$

Let v_{bx},v_{by}, and v_{bz} represent body velocity in each direction, then the angle of attack α, and reference air speed can be given by,

$$\alpha = \arctan(v_{bx}, v_{by}),$$

$$v = \sqrt{(v_{bx})^2 + (v_{by})^2}.$$

Let $s \in \{l, r\}$ represent left and right side of the vehicle. As shown in [60], for the vehicle flying forward, for each side s, the reference airspeed over the flap is give by,

$$v_{s,bz} = \sqrt{\frac{2f_s}{\rho d} + \max(0, v_{bz})^2},$$

where ρ denotes the density of air, and d is the propeller disk area. The total reference airspeed over the flap is given by

$$v_s = \sqrt{(v_{by})^2 + (v_{s,bz})^2}.$$

Following [60], the aero dynamic torque can be written as,

$$\tau_{aero} = \begin{bmatrix} (k_x + p_x\delta_l)v_l^2 + (k_x + p_x\delta_l)v_r^2 \\ k_y v_l^2 - k_y v_r^2 \\ (k_z + p_z\delta_l)v_l^2 - (k_z + p_z\delta_l)v_r^2 \end{bmatrix}, \tag{2.11}$$

where k_x, k_y, k_z, p_x,and p_z are wing and flap constants. The propeller torques τ_p is given by,

$$\tau_{prop} = \begin{bmatrix} 0 \\ l(f_r - f_l) \\ \kappa(f_r - f_l) \end{bmatrix}, \tag{2.12}$$

where κ denotes the torque-to-thrust ratio of the propellers [60].

By assuming negligible drag and lift forces, the forces produced by two propellers can be written as,

$$u_t = f_l + f_r. \tag{2.13}$$

In summary, by using (2.11), (2.12), and (2.13), the relation between the control inputs of the tail-sitter $U_T \in \mathbb{R}^4$ and the control inputs of the rigid body U_{RB} can be written as,

$$U_{RB} = \begin{bmatrix} u_t \\ \tau \end{bmatrix} = \begin{bmatrix} f_l + f_r \\ \tau_{aero} + \tau_{prop} \end{bmatrix}. \tag{2.14}$$

With the above definition of U_{RB}, the tailsitter is globally modeled by (2.7), (2.3), and (2.6).

This completes the derivation of four example systems that are all members of the vehicle class $\mathcal{C}_V$. The focus of the rest of the thesis is on controller design of rigid bodies, with detailed application on *AscTec* pelican system. However, the rigid body controller can be implemented on any vehicle in this class by applying the input map $M_i : U_i \to U_{RB}$.

Chapter 3

Path Following Control Implementation on a Car-like Robot

In this chapter we present a simple system, i.e., a car-like robot, and illustrate the idea of path following, and a procedure of path following controller design. Since the idea of control design in this thesis is based on feedback linearization, the resulting controller may raise questions about the viability of implementation on a real platform, as mostly the controllers based on feedback linearization have a reputation of degraded performance in the face of sensor noise, and modeling uncertainties. The car-like robot controller was originally proposed as part of the author's masters research. However, the implementation on a ground vehicle under IPS system and ROS is part of the author's PhD work.

3.1 Example of path following control for a car-like robot

This section presents an example of a path following controller for a car-like robot on a circular path [61]. Consider the kinematic model of a car-like robot, Figure 3.1,

$$
\begin{aligned}
\dot{x} &= f(x) + g_1(x)v + g_2(x)\omega \\
&= \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} \cos x_3 \\ \sin x_3 \\ \frac{1}{\ell}\tan x_4 \\ 0 \end{bmatrix} v + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \omega
\end{aligned} \tag{3.1}
$$

where $x \in \mathbb{R}^4$ is the state, the input $v \in \mathbb{R}$ is the translational speed and $\omega \in \mathbb{R}$ is the angular velocity of the steering angle. We take the car's position in the plane as the output of (3.1)

$$
y = h(x) = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^\top . \tag{3.2}
$$

Suppose the desired path is a unit circle and is given as a regular parameterized curve. It should

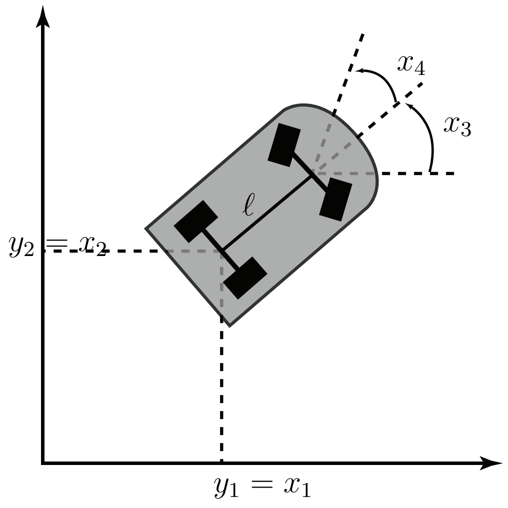


Figure 3.1: The kinematic model of the car-like robot.

be noted that the path is parameterized by a path parameter λ, instead of by time t, so that evolution of the vehicle motion along the path need not proceed at a predefined rate.

$$\begin{aligned}\sigma : \mathbb{S} &\to \mathbb{R}^2 \\ \lambda &\mapsto \begin{bmatrix} \cos\lambda \\ \sin\lambda \end{bmatrix},\end{aligned} \tag{3.3}$$

We desire to solve the following path following problem,

- For each initial condition in the neighborhood of the path, the system asymptotically converges to the path.
- Once the robot is initialized on the path (or reaches the path) with the heading vector tangent to the path, the system will stay on the path for all future times.
- On the path, the mobile robot tracks a desired velocity or acceleration profile.

In order to solve the path following problem a procedure similar to feedback linearization is adopted. Roughly speaking, in feedback linearization we differentiate the outputs of the given system until the control input appears and then use that control input to cancel out the nonlinearities to express the system in fully or partial linear form. Unlike feedback linearization, we select functions of outputs and take derivatives of these functions until the control input appears. However, it should be noted that the output functions are not just arbitrary functions rather these functions are selected from the given path. In this example one function is selected from the zero-level set representation of the path, while the other is selected from the parametric representation of the path.

The path given in this example is a *regular* curve,

$$(\forall\, \lambda \in \mathbb{S})\ \ \|\sigma'(\lambda)\| = \sqrt{\cos^2\lambda + \sin^2\lambda} = 1.$$

It can be seen that there exists a smooth map $s : \mathbb{R}^2 \to \mathbb{R}^1$ such that 0 is a *regular value* of s and $\sigma(\mathbb{D}) = s^{-1}(0)$, where $\mathbb{D}$ is the domain. Let $\gamma := s^{-1}(0)$,

$$\gamma = \{y \in \mathbb{R}^2 : y_1^2 + y_2^2 - 1 = 0\}.$$

Therefore, γ represents the zero level set representation of the path (3.3). The map $h : \mathbb{R}^4 \rightarrow \mathbb{R}^2$ is transversal [62] to γ and the path γ when expressed in state space takes the following form,

$$\Gamma := (s \circ h)^{-1}(0) = \left\{x \in \mathbb{R}^4 : s(h(x)) = x_1^2 + x_2^2 - 1 = 0\right\}$$

Define $\alpha(x) := s \circ h(x) = x_1^2 + x_2^2 - 1$. Intuitively, making $x \rightarrow \Gamma$ is equivalent to making $y \rightarrow \gamma$. The path following *manifold*, denoted by $\Gamma^\star$, associated with the curve γ is the maximal controlled *invariant* subset of the manifold Γ. Physically it consists of all those motions of the car-like robot (3.1) for which the output signal (3.2) can be made to remain on the curve γ by suitable choice of control signal [10]. The path following manifold is the key object that allows one to treat the path following problem as a set stabilization problem. The first two path following objectives can be achieved by making the path following manifold attractive and controlled invariant. In order to identify the largest controlled invariant manifold of Γ, we start taking the derivatives of the function α until the control input appears,

$$\begin{aligned}
\dot{\alpha} &= \frac{\partial \alpha(x)}{\partial x}\dot{x} \\
&= \frac{\partial \alpha(x)}{\partial x} f(x) + \frac{\partial \alpha(x)}{\partial x} g_1(x)v + \frac{\partial \alpha(x)}{\partial x} g_2(x)\omega \\
&= L_f\alpha(x) + L_{g_1}\alpha(x)v + L_{g_1}\alpha(x)\omega \\
&= v(2x_1\cos x_3 + 2x_2 \sin x_3).
\end{aligned}$$

It can be seen that $\Gamma^\star = \Gamma$. This is because one can trivially make the entire set Γ controlled invariant by setting the translational speed v to zero. From the point of view of mobile robots, this is not a useful characterization because such a controller causes the system to stop and hence not traverse the curve.

Instead, the problem can be solved by dynamic extension [13]. Let $v = \mathrm{v} + \zeta_1$, where ζ_1 is the first state of our dynamics controller. In general [4] we are free to choose any dynamics for $\dot{\zeta}_1$ but we take the simplest possible structure for the control law (4.8) and let $\dot{\zeta}_1 = \zeta_2$. In order to finish defining the control law we let $\dot{\zeta}_2 = u_1$ where u_1 is a new, auxiliary input that we will use to indirectly change the translational velocity v. The structure of the control law so far is

$$\begin{aligned} \dot{\zeta}_1 &= \zeta_2 \\ \dot{\zeta}_2 &= u_1 \\ v &= \mathrm{v} + \zeta_1 \\ \omega &= u_2. \end{aligned} \tag{3.4}$$

To simplify notation we will no longer distinguish between states of the system (x_1, x_2, x_3, x_4) and states of the controller (ζ_1, ζ_2). Let $x_5 := \zeta_1$, $x_6 := \zeta_2$. Therefore the system we study is given by

$$\begin{aligned} \dot{x} &= f(x) + g_1(x)u_1 + g_2(x)u_2 \\ &= \begin{bmatrix} (\mathrm{v}+x_5)\cos x_3 \\ (\mathrm{v}+x_5)\sin x_3 \\ \frac{(\mathrm{v}+x_5)}{\ell}\tan x_4 \\ 0 \\ x_6 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u_1 + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} u_2. \end{aligned} \tag{3.5}$$

For the extended system the path following manifold is given by

$$\Gamma^\star = \left\{ x \in \mathbb{R}^6 : \alpha(x) = \dot{\alpha}(x) = \ddot{\alpha}(x) = 0 \right\}$$

To this end we select outputs functions equal to the number of the inputs to check the *vector relative degree* of the system. Let $\varpi = \tan^{-1}(y_2/y_1)$, $\pi(x) = \varpi \circ h(x)$, and $\alpha = s \circ h(x)$. We define a "virtual" output function,

$$\hat{y} = \begin{bmatrix} \pi(x) \\ \alpha(x) \end{bmatrix} = \begin{bmatrix} \tan^{-1}\left(\frac{x_2}{x_1}\right) \\ x_1^2 + x_2^2 - 1 = 0 \end{bmatrix}. \tag{3.6}$$

We check the vector relative degree of the car-like robot (3.5) with output (3.6). Consider an arbitrary point $x^\star \in \Gamma$. Direct calculations show that

$$L_{g_1} L_f^i \pi(x) = L_{g_2} L_f^i \pi(x) = L_{g_1} L_f^i \alpha(x) = L_{g_2} L_f^i \alpha(x) = 0.$$

Moreover, the determinant of the decoupling matrix simplifies to,

$$\det(D(x^\star)) = \begin{bmatrix} L_{g_1} L_f^2 \pi(x^\star) & L_{g_2} L_f^2 \pi(x^\star) \\ L_{g_1} L_f^2 \alpha(x^\star) & L_{g_2} L_f^2 \alpha(x^\star) \end{bmatrix} = \frac{2(\mathrm{v} + x_5)^2}{\ell \cos^2 x_4}. \tag{3.7}$$

The determinant goes to zero if either $v + x_5$ goes to zero, or $\ell \cos^2 x_4$ goes to infinity. ℓ is a finite constant. Since x_4 represents wheel angle and for typical cars the wheel angle can never goes to $\pm 90^0$. So it is assumed that $x_4 \in (-\pi/2, \pi/2)$. The decoupling matrix loses rank if $x_5 = -\mathrm{v}$. Physically this condition means stopping the robot along the path. In other words, with this controller the robot cannot achieve point stabilization, but still it allows the robot to maneuver along the path and follow speed profiles bounded away from zero. We can now define a coordinate transformation. Let $x^\star \in \Gamma \backslash \{x \in \mathbb{R}^6 : x_5 + \mathrm{v} = 0\}$. There exists a neighbourhood $U \subset \mathbb{R}^6$ containing $x^\star$ such that the *mapping* $T : U \subset \mathbb{R}^6 \to T(U) \subset \mathbb{R}^6$, defined by

$$\begin{bmatrix} \eta_1 \\ \eta_2 \\ \eta_3 \\ \xi_1 \\ \xi_2 \\ \xi_3 \end{bmatrix} = T(x) = \begin{bmatrix} \pi(x) \\ L_f \pi(x) \\ L_f^2 \pi(x) \\ \alpha(x) \\ L_f \alpha(x) \\ L_f^2 \alpha(x) \end{bmatrix} \tag{3.8}$$

is a *diffeomorphism onto* its image. Using the coordinate transformation T from (3.8), in a neighbourhood of any point $x^\star \in \Gamma$, the system (3.5) in (η, ξ) coordinates reads

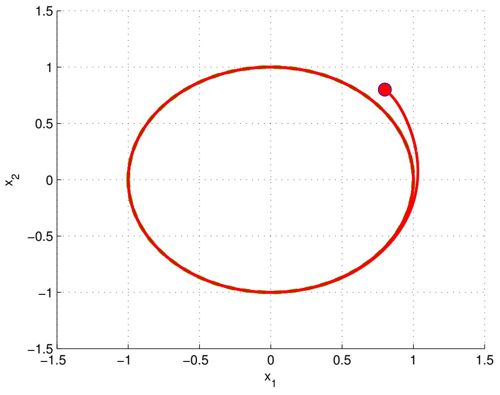
$$\begin{aligned}
\dot{\eta}_1 &= \eta_2 \\
\dot{\eta}_2 &= \eta_3 \\
\dot{\eta}_3 &= L_f^3\pi + L_{g_1}L_f^2\pi u_1 + L_{g_2}L_f^2\pi u_2\big|_{x=T^{-1}(\eta,\xi)} \\
\dot{\xi}_1 &= \xi_2 \\
\dot{\xi}_2 &= \xi_3 \\
\dot{\xi}_3 &= L_f^3\alpha + L_{g_1}L_f^2\alpha u_1 + L_{g_2}L_f^2\alpha u_2\big|_{x=T^{-1}(\eta,\xi)}
\end{aligned} \tag{3.9}$$

Stabilizing $\xi = 0$ stabilizes the path following manifold $\Gamma^\star$, because $\alpha(x)$, $\dot{\alpha}(x)$ and $\ddot{\alpha}(x)$ converge to zero. This implies the car will converge onto the desired path with heading velocity tangent to the path. On the path following manifold the motion of the car-like robot on the path is governed by the η-dynamics. When the robot is on the path following manifold, i.e., $\xi = 0$ then η_1 determines the position of the robot on the path, η_2 represent velocity of the robot along the path and η_3 represent acceleration of the robot along the path. Consider the regular feedback transformation,

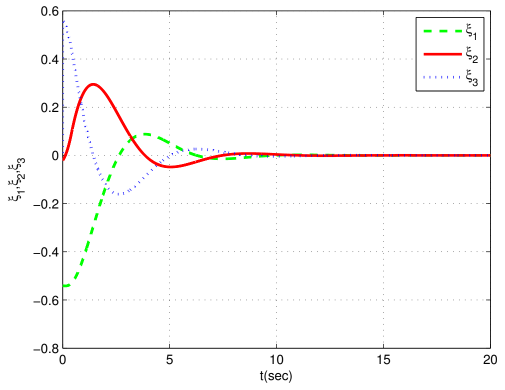
$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} := D^{-1}(x)\left(\begin{bmatrix} -L_f^3\pi \\ -L_f^3\alpha \end{bmatrix} + \begin{bmatrix} v^{\parallel} \\ v^{\pitchfork} \end{bmatrix},\right) \tag{3.10}$$

where $(v^{\parallel}, v^{\pitchfork}) \in \mathbb{R}^2$ are auxiliary control inputs. The controller is well defined in a neighbourhood of every $x^\star \in \Gamma \backslash \{x \in \mathbb{R}^6 : x_5 + \mathrm{v} = 0\}$. Thus in a neighbourhood of $x^\star$, the closed-loop system becomes

$$\begin{aligned}
\dot{\eta}_1 &= \eta_2 \\
\dot{\eta}_2 &= \eta_3 \\
\dot{\eta}_3 &= v^{\parallel} \\
\dot{\xi}_1 &= \xi_2 \\
\dot{\xi}_2 &= \xi_3 \\
\dot{\xi}_3 &= v^{\pitchfork}
\end{aligned} \tag{3.11}$$



(a) Car-like robot following a circle



(b) ξ-states converging to zero.

Figure 3.2: Convergence of system's output and transformed states

Where $v^{\pitchfork}$ and $v^{\parallel}$ are the transversal and tangential input. The control law (3.10) has decoupled the transversal and tangential subsystems which makes designing $(v^{\parallel}, v^{\pitchfork})$ to solve the path following problem particularly easy. In summary, dynamic extension and transverse feedback linearization allows us to represent the system as a linear time-invariant system (LTI) and use LTI controller design techniques to design the controller for system (3.11). In this simulation the car-like robot is following a unit circular path as shown in Figure 3.2a. It can be seen in Figure 3.2b that all the ξ-states converge to zero. While the robot follows the circular path a desired speed profile is achieved as shown in Figure 3.3. We now investigate the performance of this, apparently well behaved, controller on an experimental platform. The control implementation is challenging because of two reasons. First, although, the controller poses some robustness properties because of feedback design, it is not obvious how the controller will perform in the face on sensor noise. Second, the control implementation entails computation of ζ_1, and ζ_2 states. The augmented state ζ_2 which needs to be computed by taking a numeric derivative of ζ_1 have a relatively larger level of noise compared to other states. Despite of these challenges, we demonstrate that path following performance and convergence to the path are reliably achieved in practice.

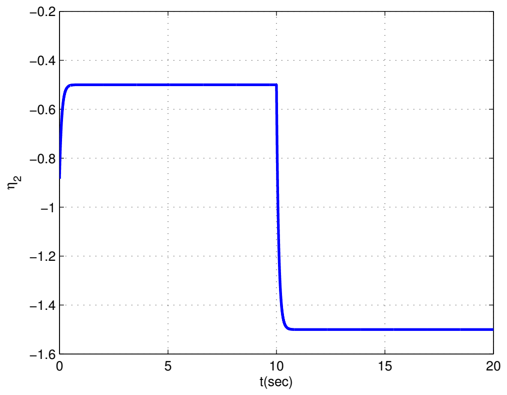


Figure 3.3: Robot following a speed profile.

3.2 Linear control design in transformed coordinates

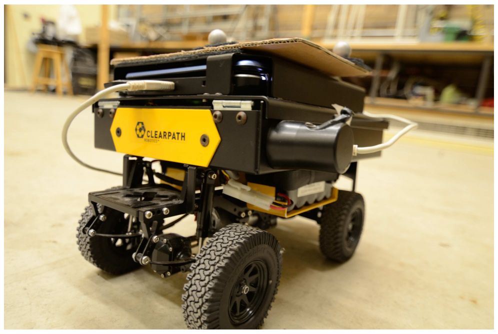
As shown in the previous section that given a circular path the nonlinear model of a car-like can be transformed into a fully linear system (3.11). The linear system has two parts: the transversal subsystem and the tangential subsystem. Since the system is linear, the controller can be designed using the well known theory of Linear Time Invariant (LTI) systems. For the transversal subsystem a controller can be chosen that stabilizes the origin of the transversal subsystem.

$$v^{\pitchfork}(\xi) = k_1\xi_1 + k_2\xi_2 + k_3\xi_3, \tag{3.12}$$

with $k_i < 0$, $i \in \{1, 2, 3\}$. It is easy to observe that the controller (3.12) exponentially stabilizes the LTI transversal subsystem and forces all the ξ-states go to zero. Physically, since $\xi = 0$ is an equilibrium of the closed-loop transversal subsystem, if the robot is initialized on the path with the initial velocity tangent to the path, then it will remain on the path for all future time. Hence, the property of path invariance is achieved.

In order to achieve the goal of controlling the speed of the robot on the curve, the following controller is used,

$$v^{\parallel}(\eta) = k_4(\eta_1 - \eta_1^{ref}) + k_5(\eta_2 - \eta_2^{ref}) + k_6\eta_3, \tag{3.13}$$


(a) The Chameleon R100 robot.

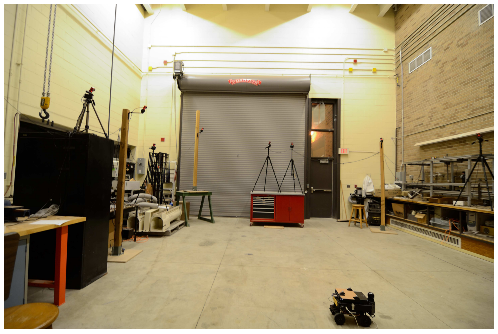

(b) Experimental Setup.

Figure 3.4: The Chameleon R100 robot and the experimental setup with IPS

where $k_i < 0$, $i \in \{4, 5, 6\}$. The parameter η_1^{ref} is the desired reference position on the path and η_2^{ref} is a desired reference velocity profile. Note however that whenever $x_5 = -\mathrm{v}$, the robot has no translational velocity. In that case, the decoupling matrix loses rank and the control law (4.23) is not well defined. Hence, we cannot stabilize a particular point on the curve using this control law and henceforth we set $k_4 = 0$.

3.3 Experimental implementation

In this section we present experimental verification of the effectiveness of the proposed controllers.

3.3.1 Experimental platform and setup

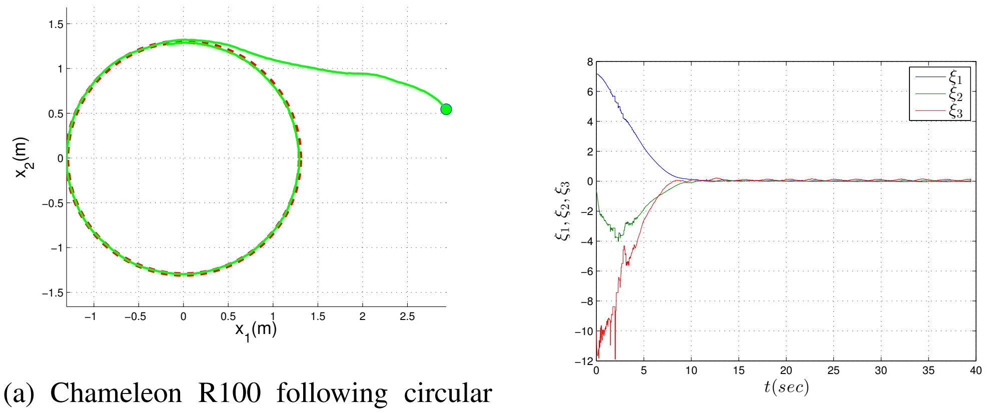
The Chameleon R100 built by Clearpath Robotics Inc., see Figure 3.4a, is a low cost car-like robot for testing control and estimation algorithms. A DC motor is attached to the rear axle of the robot. A servo motor is used to control the steering angle of the robot. The maximum steering angle is approximately 27 degrees. The wheels of the robot provide sufficient friction with the ground to make the rolling without slipping assumption implicitly made in (3.1) hold. However, the steering linkage to front wheels permits up to $\pm$ 7 degrees of error. This error source is not captured by the mathematical model (3.1) used for control design. The chassis of the robot measures $30 \times 22 \times 20$ cm (l/w/h) and is controlled from an Intel Atom Notebook. Onboard electronics provide low-level commands to the motors while the proposed control algorithm is implemented on the notebook, hereafter called the control computer, running the Robot Operating System (ROS) in Linux.

To implement the path following controller, all of the robot's states are needed. To this end, an Indoor Positioning System (IPS) is employed using the NaturalPoint OptiTrack local positioning system. The IPS uses sixteen near-infra red cameras. Infra red (IR) reflectors are attached to the robot's chassis to make the position (x_1, x_2) and orientation x_3 available for feedback, via the IPS, over WiFi. The control computer uses multithreaded Publish/Subscribe model to read the position and orientation of the robot at 100Hz from the IPS.

In many car-like robot platforms, the steering angle can be directly measured using a potentiometer or an absolute optical encoder; however the Chameleon R100 lacks this feature. Furthermore, the steering angle of the robot cannot be measured by the IPS. Therefore a standard Extended Kalman Filter (EKF) is used to obtain estimate $(x_1, x_2, x_3, x_4, x_5, x_6)$ from the measurements (x_1, x_2, x_3) and the control inputs (u_1, u_2). The control inputs of the Chameleon are its steering angle and translational speed. However, the control inputs of (3.1) are the rate of change of the steering angle and translational speed. The steering control input can be computed from the rate of change of steering angle by integration. The signals from the proposed controller (rate of change of steering angle and translational speed) are used to compute the steering angle, which is the control input of the experimental platform. The controller gains used in all the experiments are given in Table 3.1.

Table 3.1: Controller gains used in Section 3.3

Description	Symbols	Values
Transversal gains (3.12)	$\{k_1, k_2, k_3\}$	$\{30, 20, 10\}$
Tangential gains (3.13)	$\{k_4, k_5, k_6\}$	$\{0, 30, 20\}$



(a) Chameleon R100 following circular path

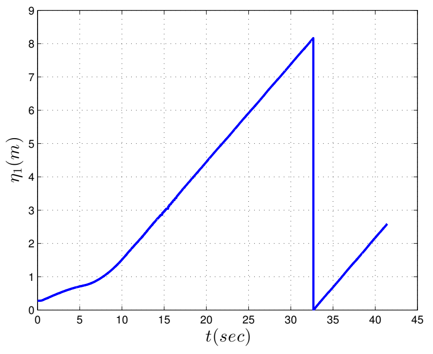
(b) Convergence of ξ_1, ξ_2, ξ_3 states.

Figure 3.5: Chameleon R100 robot following the circular curve $\sigma : [0, 2.6\pi) \to \mathbb{R}^2$, $\lambda \mapsto \mathrm{col}(1.3\sin(\lambda/1.3), 1.3\cos(\lambda/1.3))$.

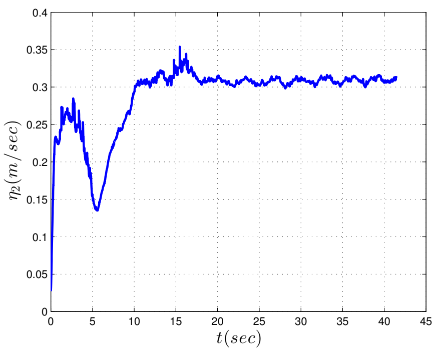
3.3.2 Experimental results

In the first experiment the Chameleon R100 robot is asked to follow a circular path of radius $r = 1.3$ meters while maintaining a constant speed of $\eta_2^{\mathrm{ref}} = 0.3$ m/sec along the path. The robot's initial position is indicated by a solid green dot in Figure 3.5a. The desired circle is represented by a dotted line in the figure.

The position of the robot along the path is given by the transformed state $\eta_1 \in [0, 2.6\pi)$. In this example the path is closed and has arc-length 2.6π; therefore $\mathbb{D} = [0, 2\pi r) = [0, 2.6\pi)$ and η_1 remains bounded between 0 to $2\pi r$ as shown in Figure 3.6a. The transformed state η_2 equals



(a) Position η_1 of the robot along the path.



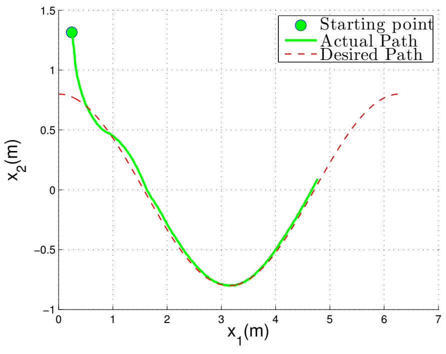
(b) Chameleon R100 maintaining a desired speed of 0.3 m/sec along the path.

Figure 3.6: Velocity and position of the Chameleon R100 robot while following the circular curve $\sigma : [0, 2.6\pi) \to \mathbb{R}^2$, $\lambda \mapsto \text{col}(1.3\sin(\lambda/1.3), 1.3\cos(\lambda/1.3))$.

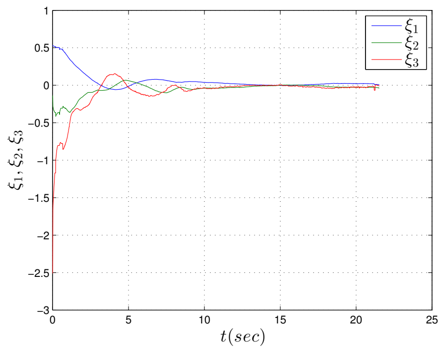
the rate of change of the arc-length at the robot's position on the circle. It is shown in Figure 3.6b.

In the second experiment the Chameleon R100 robot is made to follow a non-closed sinusoidal path. Figure 3.7a shows that the robot first converges to the desired path and follows it. Due to limited lab space the robot is asked to follow only a small portion of the sinusoidal path. All the transversal states (the ξ states) converge to zero, as required, as shown in Figure 3.7b. As the robot follows the sinusoid path a desired speed of 0.3 m/sec is achieved as shown in Figure 3.8.

In the third experiment the performance of the proposed path following controller is rigorously tested on a circular path of radius 1.3 meters. The experiment is repeated six times and the convergence of the path following error is analyzed. In each test the robot converges to the desired path starting from an initial point away from the path as shown in Figure 3.9a. The path following error $e^{\text{PF}} := \sqrt{x_1^2 + x_2^2} - 1.3$, is shown in Figure 3.9b. The initial pose (position and orientation) and steady-state path following error $|e^{\text{PF}}_{\text{ss}}| := \lim_{t\to\infty} \sup |e^{\text{PF}}|$ of the robot in each run is presented in Table 3.2. Figure 3.10 gives a zoomed in view of the path following error. We see that the path following error in each run remains within $\pm 0.015m$ or, in other words, less



(a) Chameleon R100 following a sinusoid path.



(b) Convergence of ξ_1, ξ_2, ξ_3 states.

Figure 3.7: Chameleon R100 robot following the non-closed, non-unit speed, sinusoidal path $\tilde{\sigma}(\lambda) = \text{col}(\lambda, 0.8\cos(\lambda))$.

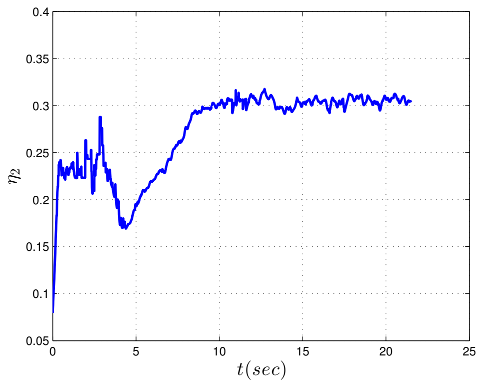
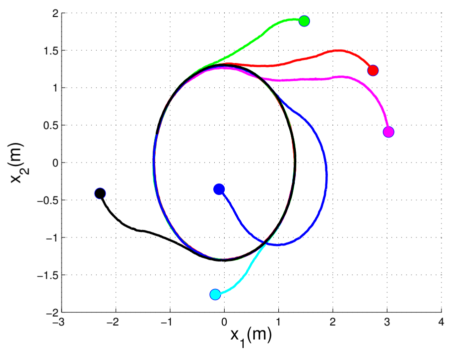


Figure 3.8: The Chameleon R100 maintaining a desired speed of 0.3 m/sec along the sinusoidal path.

than 1.15%. It can be concluded that path following controller gives fairly accurate and reliable results as the mean path following error of the six runs is 1.0689cm with a standard deviation of 0.154cm.


(a) Convergence of robot's position to the circle

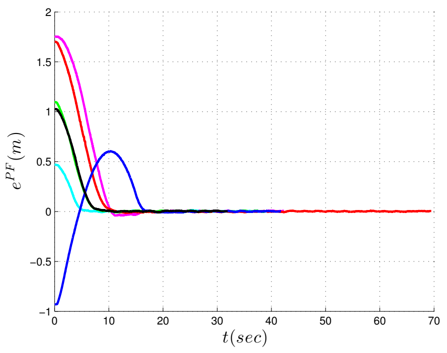

(b) Path following error.

Figure 3.9: Multiple experiments following circular path $\sigma : [0, 2.6\pi) \to \mathbb{R}^2$, $\lambda \mapsto \mathrm{col}(1.3\sin(\lambda/1.3), 1.3\cos(\lambda/1.3))$.

Table 3.2: Steady-state path following error. The initial conditions $(x_1(0), x_2(0))$ and $x_3(0)$ are given in metres and radians, respectively. The path following error is given in centimeters.

Test	$(x_1(0), x_2(0))$	$x_3(0)$	$\lvert e^{\mathrm{PF}}_{\mathrm{ss}}\rvert$
1	$(3.0267, 0.4083)$	1.8153	1.0580
2	$(-0.1675, -1.7628)$	0.1440	1.3766
3	$(2.7383, 1.2309)$	2.3205	0.9556
4	$(1.4719, 1.8907)$	2.9793	1.0089
5	$(-0.0971, -0.3565)$	-0.6987	1.0148
6	$(-2.2894, -0.4131)$	-1.0454	0.9992

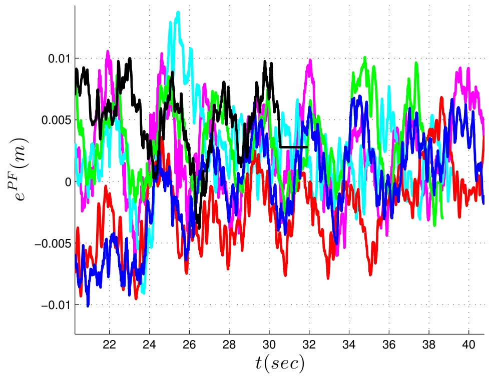


Figure 3.10: Zoomed view of the path following error after the convergence of the robot to the desired path.

During experimentation, it was observed that the closed-loop performance is very sensitive to IPS calibration errors. In particular, a small misalignment between the origin of the coordinate frame defined by the IR reflectors, and center of the rear axle, i.e., (x_1, x_2), is reflected in the path following error. Moreover, we observed that the error is reduced by a few centimeters if an EKF is used, as described above, on all six states of the system. An adaptive path following controller may perform better in the face of calibration errors.

Despite having dynamically extended states, the path following controller investigated in this chapter, not only demonstrates path invariance, but also performs well in the presence of real world disturbances and robot performance limitations. This motivates us to adopt a similar controller design scheme for the class of vehicles $\mathcal{C}_V$. In the next chapter we design path following controller for $\mathcal{C}_V$, which is a non-trivial extension of the path following controller investigated in this chapter.

Chapter 4

Path Following for the $\mathcal{C}_V$ Class of Vehicles

This chapter presents a path following controller for the $\mathcal{C}_V$ class of vehicles modeled in Chapter 2. A smooth, dynamic, feedback controller is designed that allows the $\mathcal{C}_V$ class of vehicles to follow both closed and non-closed embedded curves while maintaining a desired speed, a desired acceleration or while stabilizing desired points along the curves. The system dynamics are transformed into a linear system via a coordinate and feedback transformation. Once transformed, a path following controller is designed that guarantees invariance of the path while enforcing the desired motion along the path. The controller designed for $\mathcal{C}_V$ class of vehicles is applied to one of the systems, a quadrotor, belonging to this class.

4.1 Local representation of rigid body model

As shown in Chapter 2, the dynamics of the $\mathcal{C}_V$ class of vehicles under rotation and translation is given by (2.3), (2.6), and (2.7). A convenient choice is to pick a local chart to represent the dynamics of the rigid body. Obviously, a local representation of rigid body dynamics will lead to a local controller, but in most practical scenarios these local controllers are sufficient for most practical applications. Using Euler angles, a rotation matrix can be represented by three angles.

Let these three Euler angles be $\Phi = \mathrm{col}(\phi, \theta, \psi)$. We use the following compact notation to represent trigonometric functions; $S_i := \sin(i), C_i := \cos(i), T_i := \tan(i)$ for $i = \{\phi, \theta, \psi\}$. The kinematics equation (2.3) can be represented in local coordinates as [14, 52, 15],

$$\dot{\Phi} = M(\Phi)\Omega, \tag{4.1}$$

where M is given by,

$$M = \begin{bmatrix} 1 & S_\phi T_\theta & C_\phi T_\theta \\ 0 & C_\phi & -S_\phi \\ 0 & S_\phi \sec_\theta & C_\phi \sec_\theta \end{bmatrix}. \tag{4.2}$$

In summary, the dynamics of $\mathcal{C}_V$ class of vehicles, i.e., (2.3), (2.6), and (2.7), can be represented as,

$$\begin{aligned} \dot{\Phi} &= M(\Phi)\Omega \\ J\dot{\Omega} &= \tau - (\Omega \times J\Omega) \\ \dot{\chi} &= v \\ \dot{v} &= mgb_3 - u_t R b_3, \end{aligned} \tag{4.3}$$

where R can be represented in terms of Euler angles as,

$$R = \begin{bmatrix} C_\theta C_\psi & C_\psi S_\theta S_\phi - C_\phi S_\psi & C_\phi C_\psi S_\theta + S_\phi S_\psi \\ C_\theta S_\psi & S_\psi S_\theta S_\phi + C_\phi C_\psi & C_\phi S_\psi S_\theta - S_\phi C_\psi \\ -S_\theta & C_\theta S_\phi & C_\theta S_\phi \end{bmatrix}. \tag{4.4}$$

Let the state vector of the rigid body be $x := \mathrm{col}(x_1, \ldots, x_{12}) = \mathrm{col}(\Phi, \Omega, \chi, \dot{\chi}) \in \mathbb{R}^{12}$. With a slight abuse of notation, we represent $S_i := \sin(x_i), C_i := \cos(x_i), T_i := \tan(x_i)$. By direct substitution of (4.2), and (4.4), in (4.3) dynamics can be written as,

$$
\begin{aligned}
\dot{x}_1 &= x_4 + x_5 S_1 T_2 + x_6 C_1 T_2 \\
\dot{x}_2 &= x_5 C_1 - x_6 S_1 \\
\dot{x}_3 &= \sec(x_2)(x_5 S_1 + x_6 C_1) \\
\dot{x}_4 &= -((J_z - J_y)/J_x) x_5 x_6 + (1/J_x)\tau_p \\
\dot{x}_5 &= -((J_x - J_z)/J_y) x_4 x_6 + (1/J_y)\tau_q \\
\dot{x}_6 &= -((J_y - J_x)/J_z) x_4 x_5 + (1/J_z)\tau_r \\
\dot{x}_7 &= x_{10} \\
\dot{x}_8 &= x_{11} \\
\dot{x}_9 &= x_{12} \\
\dot{x}_{10} &= (1/m)(C_1 S_2 C_3 + S_1 S_3) u_f \\
\dot{x}_{11} &= (1/m)(C_1 S_2 S_3 - S_1 C_3) u_f \\
\dot{x}_{12} &= g - (1/m)(C_1 C_2) u_f,
\end{aligned}
\tag{4.5}
$$

The control inputs are τ_p, τ_q, τ_r, and u_t. The model (4.5) can be written compactly in control affine form as $\dot{x} = f(x) + g(x)u$. The output of the system is the position of the center of gravity of the rigid body in the inertial frame

$$
y(x) = h(\chi) = \begin{bmatrix} x_7 & x_8 & x_9 \end{bmatrix}^\top. \tag{4.6}
$$

4.2 Problem formulation

Informally, path following entails making the output of the system approach and move along a given path with no pre-specified timing law associated with the motion along the path. Before analyzing the path following problem, we present a formal definition of a “path” in this context.

Definition 4.2.1. *A path is a smooth parameterized curve in* $\mathbb{R}^3$

$$\begin{aligned}\sigma : \mathbb{D} &\to \mathbb{R}^3 \\ \lambda &\mapsto \begin{bmatrix} \sigma_1(\lambda) \\ \sigma_2(\lambda) \\ \sigma_3(\lambda) \end{bmatrix},\end{aligned} \tag{4.7}$$

such that it satisfies the following assumptions.

1. *The desired path σ is regular which implies it can be parameterized by its arc length.*
2. *The set $\sigma(\mathbb{D})$ is assumed to be an embedded submanifold of $\mathbb{R}^3$. There exists a smooth map $s : \mathbb{R}^3 \to \mathbb{R}^2$ so that $\sigma(\mathbb{D}) = s^{-1}(0)$ with 0 as regular value of s. This condition implies that* $\operatorname{rank}(ds_y) = 2$ *for all $y \in \gamma$.*

Since σ is a regular curve, therefore, without loss of generality, we henceforth assume that σ is parameterized by its arc length, i.e., $\|\sigma'\| \equiv 1$. If $\sigma(\mathbb{D})$ is an embedded submanifold of $\mathbb{R}^3$, then it is always possible to locally represent the curve as the zero level set of a function. The assumption ensures that the *entire* path can be represented as the zero level set of a smooth function. Let $\gamma := s^{-1}(0)$, then, in the case of the system belonging to the class of vehicles (4.5), the path is represented in the output space as

$$\gamma := \left\{ y \in \mathbb{R}^3 : s_1(y) = s_2(y) = 0 \right\}.$$

Similar to the previous work [10], the lift of the path γ to $\mathbb{R}^{12}$ is defined as

$$\Gamma := \left\{ x \in \mathbb{R}^{12} : s_1(h(x)) = s_2(h(x)) = 0 \right\}.$$

The control objective is to make the output y of the system (4.5) asymptotically converge and then follow the path. Making $y \to \gamma$ is equivalent to making $x \to \Gamma$. However, we will see that in general Γ can not be made invariant and hence we will try to stabilize a subset of Γ.

4.2.1 Problem statement

Given a path presented in Definition 4.2.1, we seek a smooth dynamic feedback law

$$\begin{aligned}\dot{\zeta} &= \mathcal{A}(x,\zeta) + \mathcal{B}(x,\zeta)u \\ \overline{u} &= \mathcal{C}(x,\zeta) + \mathcal{D}(x,\zeta)u,\end{aligned} \tag{4.8}$$

with[1] $\zeta \in \mathbb{R}^{\widetilde{q}}$, $u = (u_1, u_2, u_3, u_4) \in \mathbb{R}^4$ and an open subset of initial conditions $U \times V \in \mathbb{R}^{12} \times \mathbb{R}^{\tilde{q}}$ with $\gamma \subset h(U)$, such that for any initial condition $(x(0), \zeta(0)) \in U \times V$ the corresponding solution $x(t)$ for the closed-loop system is defined for all $t \geq 0$ and

PF1 The rigid body approaches the path, $\|h(x(t))\|_\gamma \to 0$ as $t \to \infty$.

PF2 The level set $s(y)$ is output invariant, i.e., if the rigid body is initialized on the path with output velocity tangent to the path, it remains on the path for all $t \geq 0$.

PF3 On the path, the system meets additional application specific requirements such as

- Stabilizing a desired point along the path
- Tracking a desired speed and/or acceleration profile along the curve.
- Tracking a desired yaw angle value or reference profile along the curve.

4.3 Dynamic extension

To solve the path following problem we seek to find the largest controlled invariant subset of Γ. As discussed in [10, 63, 13], the largest controlled invariant submanifold $\Gamma^\star$ contained in Γ is called the path following manifold. It consists of all those trajectories of the system whose associated output signal can be made to remain on the desired path by a suitable choice of the

[1]The dimension $\widetilde{q}$ of the controller state ζ is not fixed *a priori*.

control input. The path following manifold plays a key role in designing path following controllers because if $\Gamma^\star$ can be made attractive then **PF1** and **PF2** are achieved. In order to find $\Gamma^\star$ we first define

$$\alpha := s \circ h(x) = \begin{bmatrix} s_1 \circ h(x) \\ s_2 \circ h(x) \end{bmatrix} = \begin{bmatrix} \alpha_1(x) \\ \alpha_2(x) \end{bmatrix}. \tag{4.9}$$

With this definition we have that $\Gamma = \alpha^{-1}(0)$ and as a result, we can apply the zero dynamics algorithm to the function α to obtain a local characterization of $\Gamma^\star$.

Given that the system has four inputs, it is natural to augment the function (4.9) with two additional "virtual outputs" and then check if this resulting virtual output has a well-defined vector relative degree. To this end let $\pi_1(\chi) : \mathbb{R}^3 \to \mathbb{R}$ be a map defined in the output space of the rigid body. A refined definition of π_1 will be presented in the following section. We choose $\pi_2(x_3, \chi) : \mathbb{R}^4 \to \mathbb{R}$. This choice is motivated by the fact that, according to **PF3** , we would like the yaw angle $\psi = x_3$ to be virtually constrained by the position of the rigid body along the path, which is completely specified by the values of χ.

Assumption 1. *The function $\pi_2(x_3, \chi) : \mathbb{R}^4 \to \mathbb{R}$ satisfies*

$$\frac{\partial \pi_2}{\partial x_3} \neq 0$$

for every $x_3 \in [0, 2\pi)$ and every $\chi \in \mathbb{R}^3$ such that $h(x) \in \gamma$.

Assumption 1 ensures that, at each point along the path, we can apply the implicit function theorem on $\overline{y}_4 = \pi_2(x_3, \chi)$ and express x_3 as a function of χ and $\overline{y}_4$. In other words, this ensures that π_2 represents a valid, positionally dependent, constraint on the yaw angle along the path.

In summary, motivated by the path following problem, we define the "virtual" output function by

$$\overline{y} = \begin{bmatrix} \alpha(\chi) \\ \pi_1(\chi) \\ \pi_2(x_3, \chi) \end{bmatrix}. \tag{4.10}$$

The output function is intuitively appealing for the purposes of meeting **PF1**, **PF2** and **PF3**, the following result shows that it fails to yield a well-defined relative degree.

Lemma 4.3.1. *System* (4.5) *with output* (4.10) *does not have a well-defined vector relative degree at any* $x \in \mathbb{R}^{12}$.

Proof. Since α and π_1 are functions of χ and π_2 is a function of x_3, χ it is easy to check, in light of the model (4.5) that

$$L_{g_i}\pi_1(x) = L_{g_i}\pi_2(x) = L_{g_i}\alpha_1(x) = L_{g_i}\alpha_1(x) \equiv 0$$

for $i \in \{1,2,3,4\}$. Direct calculations also reveal that $L_{g_i}L_f\pi_1(x) = L_{g_i}\alpha_1(x) = L_{g_i}\alpha_2(x) \equiv 0$ for $i \in \{2,3,4\}$ while $L_{g_1}L_f\pi_1(x)$, $L_{g_1}\alpha_1(x)$ and $L_{g_1}\alpha_2(x)$ are not identically equal to zero in any open set of $\mathbb{R}^{12}$. In other words, the only control input that is not always multiplied by zero in the second derivative of the functions α_1, α_2, π_1 is τ_f.

Similar calculations show that $L_{g_2}L_f\pi_2(x) \equiv 0$ while $L_{g_i}L_f\pi_2(x)$, $i \in \{1,3,4\}$ are not all identically zero in any open subset of $\mathbb{R}^{12}$. From these calculations we deduce that the decoupling matrix for the quadrotor (4.5) with output (4.10) has, at each point of $\mathbb{R}^{12}$, the form

$$D(x) = \begin{bmatrix} L_{g_1}L_f\alpha_1(x) & 0 & 0 & 0 \\ L_{g_1}L_f\alpha_2(x) & 0 & 0 & 0 \\ L_{g_1}L_f\pi_1(x) & 0 & 0 & 0 \\ L_{g_1}L_f\pi_2(x) & 0 & L_{g_3}L_f^2\pi_2 & L_{g_4}L_f^2\pi_2 \end{bmatrix}. \tag{4.11}$$

Clearly $D(x)$ is rank deficient for all x in $\mathbb{R}^{12}$. □

One possible interpretation of Lemma 4.3.1 is that the decoupling matrix loses rank because the control input τ_p does not appear.This problem is overcome by delaying the appearance of the control input u_t with the help of two integrators, which are included through two additional states $\zeta = (\zeta_1, \zeta_2)$. Let $x_{13} := \tau_f$ and $x_{14} := \dot{\tau}_f$. This dynamic extension generates the dynamic control law

$$\begin{aligned} \dot{\zeta}_1 &= \zeta_2 \\ \dot{\zeta}_2 &= u_d \\ \tau_f &= \zeta_1. \end{aligned} \tag{4.12}$$

To simplify notation, we no longer distinguish between the quadrotor's states $(x_1, \cdots, x_{12})$ and the controller states (ζ_1, ζ_2). Let $x_{13} := \zeta_1$, $x_{14} := \zeta_2$, $u_d = \ddot{u}_t$, $u := \operatorname{col}(u_1, u_2, u_3, u_4) = \operatorname{col}(u_d, u_p, u_q, u_r) \in \mathbb{R}^4$, where $u_r := \tau_r$, $u_p := \tau_p$ and $u_q := \tau_q$. The model of the rigid body after dynamic extension gets the form,

$$\begin{aligned} \dot{x}_1 &= x_4 + x_5 S_1 T_2 + x_6 C_1 T_2 \\ \dot{x}_2 &= x_5 C_1 - x_6 S_1 \\ \dot{x}_3 &= \sec(x_2)(x_5 S_1 + x_6 C_1) \\ \dot{x}_4 &= -((J_z - J_y)/J_x) x_5 x_6 + (1/J_x) u_p \\ \dot{x}_5 &= -((J_x - J_z)/J_y) x_4 x_6 + (1/J_y) u_q \\ \dot{x}_6 &= -((J_y - J_x)/J_z) x_4 x_5 + (1/J_z) u_r \\ \dot{x}_7 &= x_{10} \\ \dot{x}_8 &= x_{11} \\ \dot{x}_9 &= x_{12} \\ \dot{x}_{10} &= (1/m)(C_1 S_2 C_3 + S_1 S_3) x_{13} \\ \dot{x}_{11} &= (1/m)(C_1 S_2 S_3 - S_1 C_3) x_{13} \\ \dot{x}_{12} &= g - (1/m)(C_1 C_2) x_{13} \\ \dot{x}_{13} &= x_{14} \\ \dot{x}_{14} &= u_d. \end{aligned} \tag{4.13}$$

With a slight abuse of notation we write the extended model compactly as

$$\dot{x} = f(x) + \sum_{i=1}^{4} g_i(x) u_i.$$

By applying the zero dynamics algorithm to the output (4.9), the extended system yields the path following manifold

$$\Gamma^\star = \left\{ x \in \mathbb{R}^{14} : L_f^i \alpha(x) = 0, i = 0, 1, 2, 3, 4 \right\}. \tag{4.14}$$

4.4 Path following controller design

In this work the path following problem is treated as an instance of the set stabilization problem and the general approach for solving path following problem is applied to a rigid bodies [64, 13, 10]. In contrast to the differential flatness based controller which involves finding an output such that the resulting feedback linearized system is fully linear, we have chosen flat outputs that are physically meaningful for the path following problem. We now refine the definition of π_1 in the virtual output (4.10) by choosing a specific function. A mapping is introduced that associates to a point y in the output space of the rigid body system, sufficiently close to the path, a number in the domain $\mathbb{D}$ that minimizes the distance from the path γ. This mapping was used in [63] for curves in $\mathbb{R}^2$. Let $\gamma_\epsilon \subset \mathbb{R}^3$ be a tubular neighbourhood of the path γ and define the map

$$\begin{aligned} \varpi :& \gamma_\epsilon \to \mathbb{D} \\ & y \mapsto \arg \inf_{\lambda \in \mathbb{D}} \| y - \sigma(\lambda) \|. \end{aligned} \tag{4.15}$$

The above function is smooth so long as γ_ϵ is a sufficiently small "tube" around the curve γ. With these definitions and Assumption 4.2.1, we refine the "virtual output" function $\hat{y}$ to be

$$\hat{y} = \begin{bmatrix} \alpha_1(\chi) \\ \alpha_2(\chi) \\ \pi_1(\chi) \\ \pi_2(x_3, \chi) \end{bmatrix} = \begin{bmatrix} s_1 \circ h(\chi) \\ s_2 \circ h(\chi) \\ \varpi \circ h(\chi) \\ \pi_2(x_3, x_7, x_8, x_9) \end{bmatrix}. \tag{4.16}$$

The next two results are presented to support our claim that the extended system has a well-defined vector relative degree at each point on the path following manifold. The first is presented without proof as it is the well known triple product result from linear algebra.

Lemma 4.4.1 ([65])**.** *If v_1,v_2,v_3 are linearly independent vectors in $\mathbb{R}^3$ then $\langle v_1, (v_2 \times v_3)\rangle \neq 0$.*

Let

$$d_\chi \alpha_i := \text{col}\left(\frac{\partial \alpha_i}{\partial x_7}, \frac{\partial \alpha_i}{\partial x_8}, \frac{\partial \alpha_i}{\partial x_9}\right)$$

for $i = \{1, 2\}$, and

$$\sigma' := \text{col}\left(\frac{\partial \sigma}{\partial \lambda}, \frac{\partial \sigma}{\partial \lambda}, \frac{\partial \sigma}{\partial \lambda}\right)$$

Lemma 4.4.2. *Let α_1 and α_2 be as defined in* (4.9)*. Then, for all $\chi \in \gamma$,*

$$\text{span}\{d_\chi \alpha_1, d_\chi \alpha_2, \sigma'\} = \mathbb{R}^3.$$

Proof. We first show that each of the vectors $d_\chi \alpha_1, d_\chi \alpha_2, \sigma'$ are non zero. By assumption, σ is regular which means $\sigma' \neq 0$. Also by definition 4.2.1, at each $y \in \gamma$, ds_y has rank two. Since $dh_x = I$ this shows, using the chain rule, that at each $\chi^\star \in \Gamma$, $d_\chi \alpha$ has rank two. Since σ' is a tangent vector and $d_\chi \alpha_1, d_\chi \alpha_2$ are non-zero gradient vectors, we conclude that $\text{span}\{d_\chi \alpha_1, d_\chi \alpha_2, \sigma'\} = \mathbb{R}^3$ as claimed. □

It is claimed that state x_{13}, which represents thrust u_t, can not be zero. In fact, x_{13} is zero if and only if thrust applied by rigid bodies is zero. For system belonging to the class of vehicles such as quadrotors, tailsitter, and satellite, this means all the rotors of the system stop spinning at the same time. Therefore, for almost all practical purposes $x_{13} \neq 0$ is a valid assumption. Moreover, we assume that the rigid body does not encounter gimbal lock situation[2], i.e., $\phi = \theta \neq \pm 90^0$.

[2]The singularity associated with gimbal lock is due to Euler angle parameterization on $\mathsf{SO}(3)$.

Lemma 4.4.3. *The extended model of the rigid body* (4.13) *with output* (4.16) *yields a well-defined vector relative degree of* $\{4, 4, 4, 2\}$ *at each point on* $\Gamma^\star \cap \{x \in \mathbb{R}^{14} : x_{13} \neq 0, \phi = \theta \neq \pm 90^0\}$.

Proof. Let $x^\star \in \Gamma^\star \cap \{x \in \mathbb{R}^{14} : \phi = \theta \neq \pm 90^0, x_{13} \neq 0\}$ be arbitrary. By definition of Γ, and since $\Gamma^\star \subseteq \Gamma$, the output $h(x^\star)$ is on the path γ. Let $\lambda^\star \in \mathbb{D}$ be such that $h(x^\star) = \sigma(\lambda^\star)$. By the definition of vector relative degree we must show that

$$L_{g_i} L_f^j \pi_1(x) = L_{g_i} \pi_2(x) = L_{g_i} L_f^j \alpha_k(x) \equiv 0 \tag{4.17}$$

for $i \in \{1, 2, 3, 4\}$, $j \in \{0, 1, 2\}$ in a neighbourhood of $x^\star$ and that the 4×4 decoupling matrix

$$D(x^\star) = \begin{bmatrix} L_g L_f^3 \alpha_1(x^\star) \\ L_g L_f^3 \alpha_2(x^\star) \\ L_g L_f^3 \pi_1(x^\star) \\ L_g L_f \pi_2(x^\star) \end{bmatrix}, \tag{4.18}$$

is non-singular. It is easy to check, by direct computations, that the first condition holds and

$$\det(D(x)) = \left(\frac{l^2 (x_{13})^2 \cos(x_1)}{I_x I_y I_z m^3 \cos(x_2)} \right) \left(\frac{\partial}{\partial x_3} \pi_2 \right) \langle d_\chi \alpha_1, (d_\chi \alpha_2 \times \sigma') \rangle . \tag{4.19}$$

The determinant goes to zero if and only if any term in the numerator of (4.19) is zero or any term in the denominator is infinity. The terms I_x, I_y, I_z, l and m are finite constants. By assumption, at $x^\star \in \Gamma^\star \cap \{x \in \mathbb{R}^{14} : \phi = \theta \neq \pm 90^0, x_{13} \neq 0\}$, the combined thrust $x_{13} \neq 0$. By Lemma 4.4.2, $\text{span}\{d_\chi \alpha_1, d_\chi \alpha_2, \sigma'\}(x^\star) = \mathbb{R}^3$ and therefore, by Lemma 4.4.1 $\langle d_\chi \alpha_1, (d_\chi \alpha_2 \times \sigma') \rangle \neq 0$ at $x^\star$. By Assumption 1 $\partial \pi_2 / \partial x_3 \neq 0$. It is further assumed that the system does not encounter gimbal lock situation, therefore $\cos x_i \neq 0$ for $i = \{1, 2\}$. Thus we have shown that for any $x^\star \in \Gamma^\star \cap \{x \in \mathbb{R}^{14} : \phi = \theta \neq \pm 90^0, x_{13} \neq 0\}$, $\det(D(x^\star)) \neq 0$, therefore the extended system has a well defined vector relative degree at $x^\star$. $\square$

The singularities at $x_1 = x_2 = \theta = \pm 90^\circ$ are not intrinsic to the physics of rigid body system, and arise due to singularities in the local chart i.e., Euler angle representation. These singularities could potentially be eliminated by using rotation matrices. In other words by designing a controller on $\mathsf{SO}(3)$ directly, such singularities can be avoided, and one can achieve global results. In Chapter 6 we eliminate such singularities and present global controllers. Since the extended system (4.13) has a well defined vector relative degree of $\{4, 4, 4, 2\}$, this implies that the dimension of the zero dynamics is zero. In other words, we can fully linearize the extended system of the quadrotor. This leads to the definition of a local coordinate transformation.

Corollary 4.4.4. *Let $x^\star \in \Gamma^\star \cap \{x \in \mathbb{R}^{14} : \phi = \theta \neq \pm 90^0, x_{13} \neq 0\}$. There exists a neighbourhood $U \subset \mathbb{R}^{14}$ containing $x^\star$ such that the mapping $T : U \subset \mathbb{R}^{14} \to T(U) \subset \mathbb{R}^{14}$, defined by*

$$\begin{bmatrix} \xi_{ji} \\ \eta_{1i} \\ \eta_{2k} \end{bmatrix} = T(x) = \begin{bmatrix} L_f^{i-1}\alpha_j(x) \\ L_f^{i-1}\pi_1(x) \\ L_f^{k-1}\pi_2(x) \end{bmatrix}, \tag{4.20}$$

for $i \in \{1, 2, 3, 4\}$, $j \in \{1, 2\}$ and $k \in \{1, 2\}$ is a diffeomorphism.

Proof. Let $x^\star \in \Gamma^\star \cap \{x \in \mathbb{R}^{14} : \phi = \theta \neq \pm 90^0, x_{13} \neq 0\}$. By Lemma 4.4.3, system (4.13) with output (4.16) yields a well-defined vector relative degree of $\{4, 4, 4, 2\}$ at $x^\star$. By [4, Lemma 5.2.1] the row vectors

$$\begin{gathered} d\alpha_1(x^\star), dL_f\alpha_1(x^\star), dL_f^2\alpha_1(x^\star, dL_f^3\alpha_1(x^\star) \\ d\alpha_2(x^\star), dL_f\alpha_2(x^\star), dL_f^2\alpha_2(x^\star, dL_f^3\alpha_2(x^\star) \\ d\pi_1(x^\star), dL_f\pi_1(x^\star), dL_f^2\pi_1(x^\star, dL_f^3\pi_1(x^\star) \\ d\pi_2(x^\star), dL_f\pi_2(x^\star), \end{gathered} \tag{4.21}$$

are linearly independent. These are the rows of the 14×14 Jacobian matrix $dT_{x^\star}$ which implies that $dT_{x^\star}$ is non-singular. By the inverse function theorem [66, Theorem 5.23] T is a diffeomorphism onto its image. $\square$

Using the coordinate transformation T from Corollary 4.4.4, the system is transformed in (η, ξ) coordinates

$$\begin{aligned}
\dot{\xi}_{j1} &= \xi_{j2} \\
\dot{\xi}_{j2} &= \xi_{j3} \\
\dot{\xi}_{j3} &= \xi_{j4} \\
\dot{\xi}_{j4} &= L_f^4 \alpha_j + \sum_{i=1}^{4} L_{g_i} L_f^3 \alpha_j u_j \Big|_{x=T^{-1}(\eta,\xi)} \\
\dot{\eta}_{11} &= \eta_{12} \\
\dot{\eta}_{12} &= \eta_{13} \\
\dot{\eta}_{13} &= \eta_{14} \\
\dot{\eta}_{14} &= L_f^4 \pi_1 + \sum_{i=1}^{4} L_{g_i} L_f^3 \pi_1 u_3 \Big|_{x=T^{-1}(\eta,\xi)} \\
\dot{\eta}_{21} &= \eta_{22} \\
\dot{\eta}_{22} &= L_f^2 \pi_2 + \sum_{i=1}^{4} L_{g_i} L_f \pi_2 u_4 \Big|_{x=T^{-1}(\eta,\xi)} ,
\end{aligned} \tag{4.22}$$

for $j \in \{1, 2\}$. This model (4.22) suggests a natural choice of feedback transformation

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} := D^{-1}(x) \left(\begin{bmatrix} -L_f^4 \alpha_1 \\ -L_f^4 \alpha_2 \\ -L_f^4 \pi_1 \\ -L_f^3 \pi_2 \end{bmatrix} + \begin{bmatrix} v^{\xi_1} \\ v^{\xi_2} \\ v^{\eta_1} \\ v^{\eta_2} \end{bmatrix} \right), \tag{4.23}$$

where $(v^{\xi_1}, v^{\xi_2}, v^{\eta_1}, v^{\eta_2})$ are auxiliary control inputs. By Lemma 4.4.3 this controller (4.23) is well-defined in a neighbourhood of every $x^\star \in \Gamma \backslash \{x \in \mathbb{R}^{14} : x_{10} = x_{11} = \pm 90^0\}$. Thus in a neighbourhood of $x^\star$, the closed loop system is simply reduced to four decoupled chains of integrators

$$\begin{array}{llll} \dot{\xi_{11}} = \xi_{12} & \dot{\xi_{21}} = \xi_{22} & \dot{\eta_{11}} = \eta_{12} & \dot{\eta_{21}} = \eta_{22} \\ \dot{\xi_{12}} = \xi_{13} & \dot{\xi_{22}} = \xi_{23} & \dot{\eta_{12}} = \eta_{13} & \dot{\eta_{22}} = v^{\eta_2}. \\ \dot{\xi_{13}} = \xi_{14} & \dot{\xi_{23}} = \xi_{24} & \dot{\eta_{13}} = \eta_{14} & \\ \dot{\xi_{14}} = v^{\xi_1} & \dot{\xi_{24}} = v^{\xi_2} & \dot{\eta_{24}} = v^{\eta_2} & \end{array} \tag{4.24}$$

and linear control techniques can be used. The output (4.16) is a flat output [67] for the rigid body system (4.13) because these outputs transform the system to a fully linear system.

4.4.1 Auxiliary controller design

After applying the coordinate and feedback transformations (4.20), (4.23) to the extended system (4.13) the auxiliary controller design is straight forward. Stabilizing the origin of the first two chain of integrators, collectively called the ξ-subsystem, corresponds to the stabilization of path following manifold $\Gamma^\star$. When $\xi = 0$ the states of the system are restricted to stay on the path. We propose the following controller to control the ξ-subsystem.

$$v^{\xi_j} = \sum_{i=1}^{4} k_i \xi_{ji}, \tag{4.25}$$

with $k_i < 0$, $j \in \{1, 2\}$. This controller exponentially stabilizes $\xi = 0$. Since $\xi = 0$ is an equilibrium of the ξ-subsystem, the origin is exponentially stable. Moreover, we stabilize the path following manifold $\Gamma^\star$ and hence path invariance is achieved. In other words, **PF1** and **PF2** are satisfied.

To achieve the goal of point stabilization along the curve, controlling the speed along the curve, and forcing the rigid body to follow a given acceleration profile along the curve we propose the following controller

$$v^{\eta_1} = \sum_{i=1}^{4} k_i(\eta_{1i} - \eta_{1i}^{ref}), \tag{4.26}$$

where $k_i < 0$, η_{11} is the path parameter. By setting η_{11}^{ref} to the desired value, point stabilization is achieved. By choosing $k_1 = 0$ and setting η_{12}^{ref} to the desired velocity profile the rigid body follows the given velocity profile. Similarly, by choosing $k_1 = k_2 = 0$ and setting η_{13}^{ref} to the desired acceleration profile the system follows the given acceleration profile. Therefore, by the auxiliary controller given by (4.26), **PF3** has been achieved.

The yaw angle of the rigid body can be controlled by designing a similar controller for the fourth chain of integrator.

$$v^{\eta_2} = k_1(\eta_{21} - \eta_{21}^{ref}) + k_2(\eta_{22} - \dot{\eta}_{21}^{ref}) + \ddot{\eta}_{21}^{ref}, \tag{4.27}$$

where $k_1, k_2 < 0$. By stabilizing the origin of the η_2-subsystem, the yaw angle of the rigid body converges to the desired yaw angle reference function, satisfying the yaw objective in **PF3**.

4.5 Simulation results

For simulation purposes, it is assumed that the rigid body has a mass of $m = 4.493$ kg, and inertias $J_x = J_y = 0.177$ kg.m^2 and $J_z = 0.344$ kg.m^2, and acceleration due to gravity is $g = 9.8$ m/sec^2. It is further assumed that due to modeling uncertainties there is 10% error in J_x, J_y, J_z. The error in the mass of the rigid body is assumed to be 1% because the mass of the quadrotor can be accurately measured by a precise weight measuring instrument. The initial position of the rigid body is indicated by a solid dot. The rigid body is following a curve represented by a fourth order spline. The controller allows the system to follow a spline of any order greater than 1. Moreover, each vehicle belonging to the class of vehicles is capable of following any closed or non-closed curve satisfying Definition 4.2.1. The path chosen for this simulation is a general 4th order spline given by $\sigma : \mathbb{R} \to \mathbb{R}^3$, $\lambda \mapsto \mathrm{col}(\lambda, a_4\lambda^4 + a_3\lambda^3 + a_2\lambda^2 + a_1\lambda + a_0, 3)$. The implicit representation of the same curve is given by $\gamma = \{s_1(y) = s_2(y) = 0\}$, where $s_1(y) = x_3 - a_4x_1^4 + a_3x_1^3 + a_2x_1^2 + a_1x_1 + a_0 = 0$ and $s_2 = x_3 - 3 = 0$. In Figure 4.1, the system

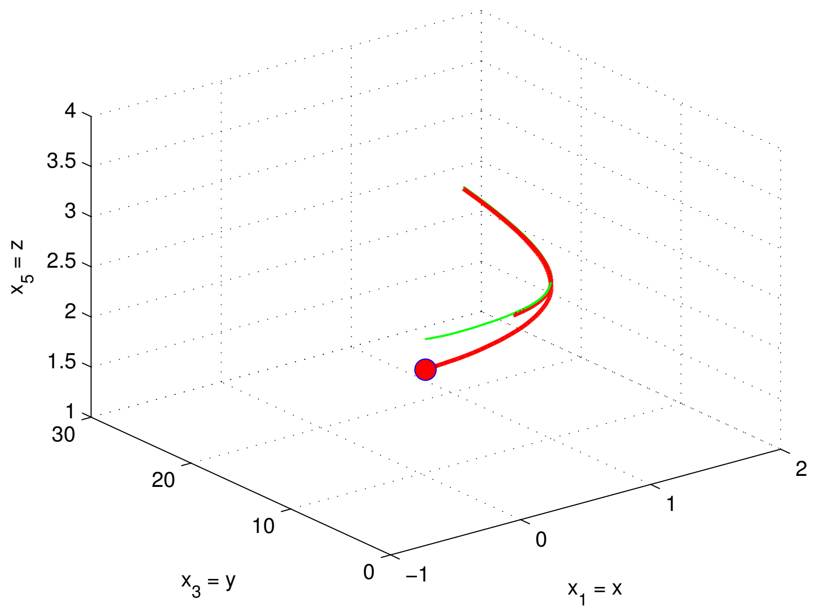


Figure 4.1: Velocity profile simulation. The path followed by the rigid body is represented by a bold red line and the desired path is represented by a solid green line.

is following the desired path and tracking a velocity profile given by $\eta_{12}^{ref} = 1$ for $t \in [0, 20)$, $\eta_{12}^{ref} = 0$, for $t \in [20, 40)$ and $\eta_{12}^{ref} = -1$ for $t \geq 40$.

The above velocity profile forces the rigid body to maintain a velocity of 1 unit/sec for 20 seconds, stopping along the curve for the next 20 seconds, and then reverse the direction of the motion with a velocity of -1 unit/second. This indicates that with the proposed controller, the system is capable of stopping along the curve and changing direction along the curve while staying on the path. The system's velocity is compared to the desired velocity in Figure 4.2. Moreover, the rigid body is following a yaw profile $\eta_{21}^{ref} = \sin(t)$ while following the desired path as shown in Figure 4.2. Stabilizing the yaw angle to a desired value or forcing it to follow a given profile can be of practical importance for certain system belonging to the class of vehicles such as quadrotors, and satellite systems.

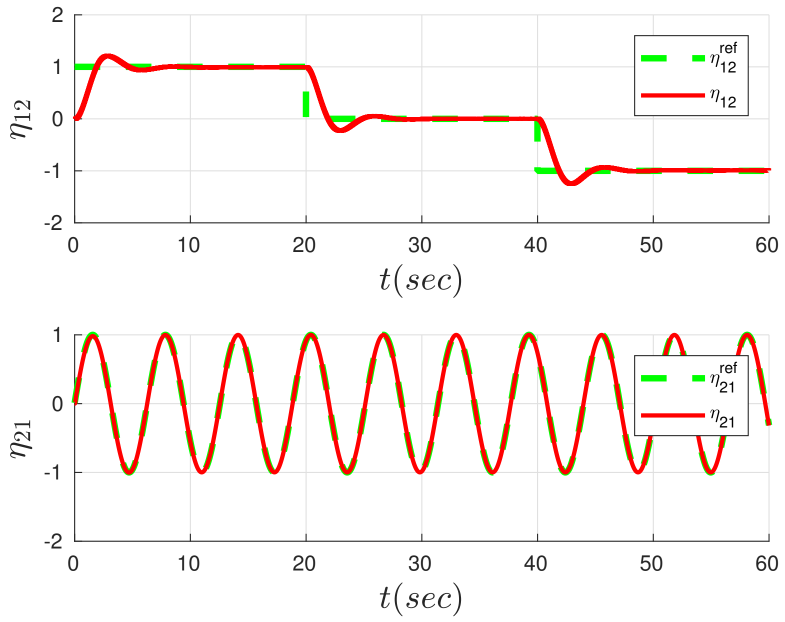
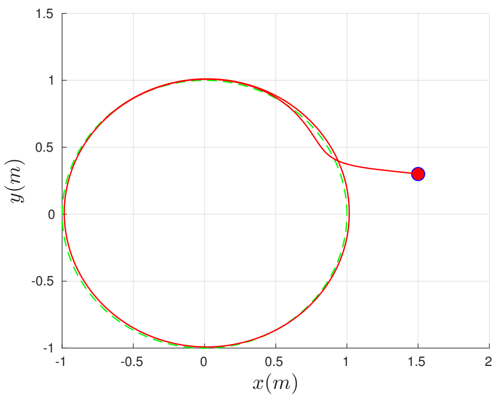


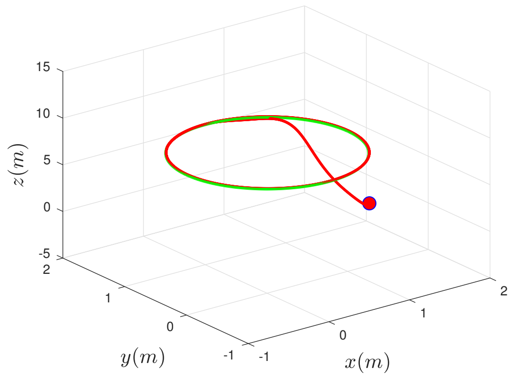
Figure 4.2: Comparison between reference and actual states.

4.6 Application of path following controller on a quadrotor

Next we analyze performance of the path following controller (4.23), designed in the previous section, on one of the vehicles belonging to the class of vehicles presented in Chapter 2 i.e., a quadrotor. The inputs of the quadrotor are motor pwm, which is a natural number between 0 to 200. The control inputs u_1, u_2, u_3, u_4 are converted to motor pwm by the mapping give by (2.9). Moreover, to make the simulation more realistic a first order motor model in also included in simulation. In each simulation the desired path is a unit circle in x, y plane, and the quadrotor is desired to maintain a constant height of $10m$ in z axis, and a constant speed along the path.



(a) Following desired path in x, y plane



(b) Following desired path in 3D

Figure 4.3: path following without noise

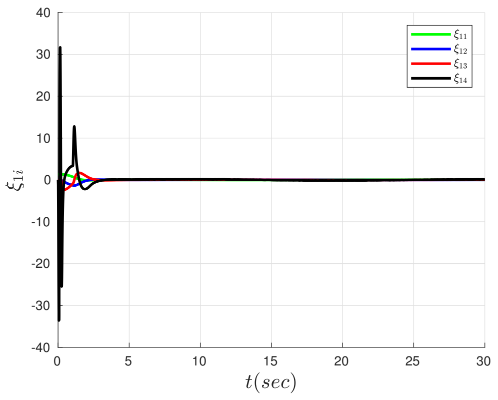
4.6.1 Without sensor noise

We first simulate the controller (4.23) on the quadrotor in the absence of sensor noise. A first order motor model is added in the simulation, and as expected the motor dynamics are fast enough to not have any significant effect on the path following. In the absence of sensor noise, the quadrotor precisely follow the desired path, as shown in Figure 4.3a, and Figure 4.3b. Moreover, all the transformed states, ξ_{ij}, for $i = \{1, 2\}$, and $j = \{1, 2, 3, 4\}$, converge to zero, as shown in Figure 4.4a, and Figure 4.4b. We introduce actuator saturation, in simulation, to cap the motor pwm inputs to 160[3], as shown in Figure 4.5b. Thrust, i.e., ζ_1, and rate of change of thrust, i.e., ζ_2 are shown in Figure 4.5a.

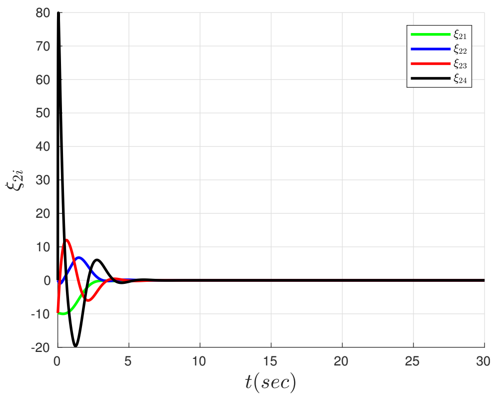
4.6.2 Sensor Noise

Now we simulate quadrotor system, just like the previous case, but in the presence of sensor noise. We assume that the quadrotor states are sensed by either an IMU, or Indoor Positioning System (IPS) or both. The noise is assumed to be a zero mean Gaussian, and is given in

[3]The rotor speed that results from a PWM command of 160 is as large as is safe for indoor flight.

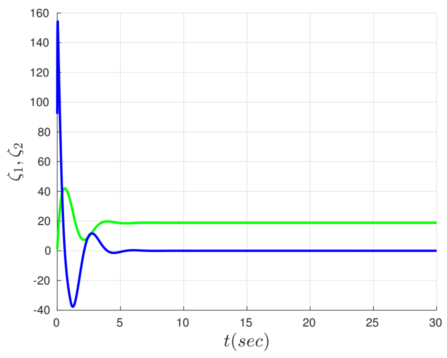


(a) Transformed states ξ_{1i}

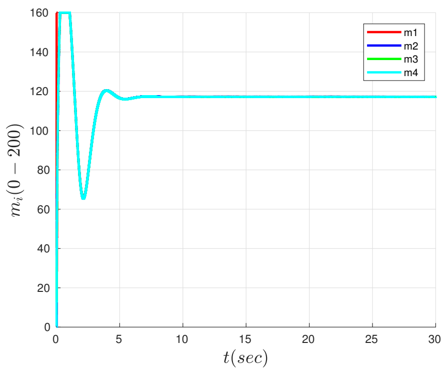


(b) Transformed states ξ_{2i}

Figure 4.4: Transformed states without noise



(a) Augmented states ζ_1, ζ_2



(b) Motor pwm values

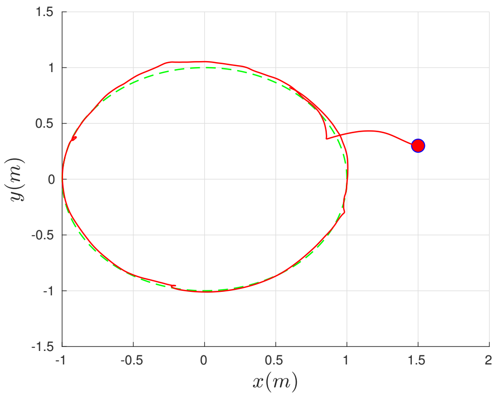
Figure 4.5: Augmented states without noise

Table 4.1: Quadrotor noise levels

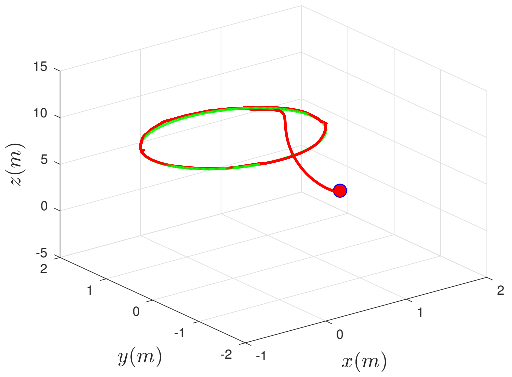
State	Source	Range	Standard Deviation σ
Position (x, y, z)	IPS	$\pm 50e^{-6}m$	$10e^{-6}m$
Velocity $(\dot{x}, \dot{y}, \dot{z})$	IPS	$\pm 2e^{-3}m/sec$	$3e^{-4}m/sec$
Angles (ϕ, θ, ψ)	IPS	$\pm$0.02 deg.	0.005 deg.
Angles (ϕ, θ, ψ)	IMU	$\pm$0.1 deg.	0.01 deg.
Body rates (p, q, r)	IMU	$\pm$0.08 rads/sec	0.02 rads/sec.

Table 4.1 [68, 58, 15] [4]. It should be noted that the quadrotor platform *AscTec* provides basic filtered IMU data at an update rate of up to 1kHz. Although, feedback linearized controllers, sometimes, suffer performance limitations in the presence of sensor noise, the controller (4.23) performs fairly well in terms of path following, as shown in Figure 4.6a, and Figure 4.6b. It can be easily seen in these figures that the performance has degraded compared to the previous case when all the states are known precisely. However, these results are good enough for most of the practical purposes. It can be seen that the transformed states ξ_{ij}, for $i = \{1, 2\}$, and $j = \{1, 2, 3, 4\}$ get noisy but converge to zero, as shown in Figure 4.7a, and Figure 4.7b. It is interesting to see in Figure 4.8a, and Figure 4.8b that the augmented states ζ_1 and ζ_2, and motor pwn values remain noise free. It should be noted that thrust, i.e., ζ_1 is input of the non-extended quadrotor system, but is the state of the extended system. Moreover, for implementing controller (4.23), thrust or ζ_1, and ζ_2 must be known, or in other words must be measured by some sensor. Therefore, in the next simulation we add noise on these states.

[4] http://www.dis.uniroma1.it/~oriolo/fda/matdid/ControlOfAQuadrotorUAV.pdf

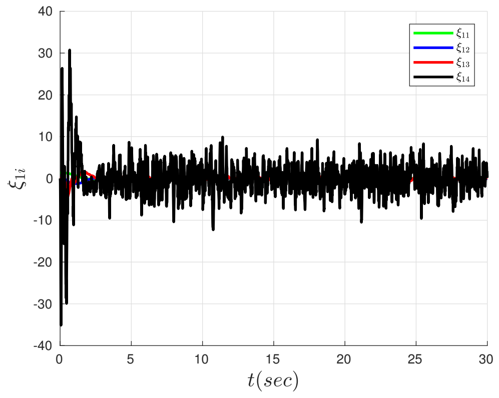


(a) Following desired path in x, y plane

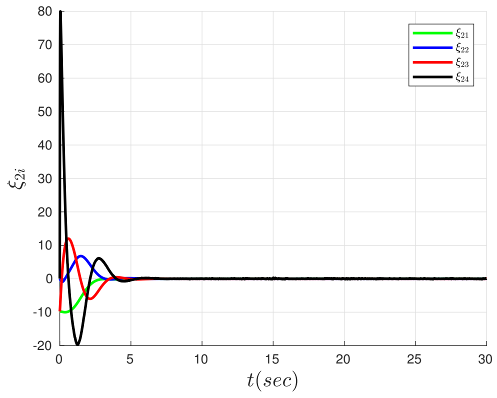


(b) Following desired path in 3D

Figure 4.6: Path following in the presence of sensor noise

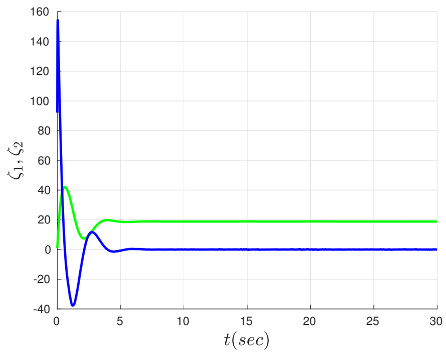


(a) Transformed states ξ_{1i}

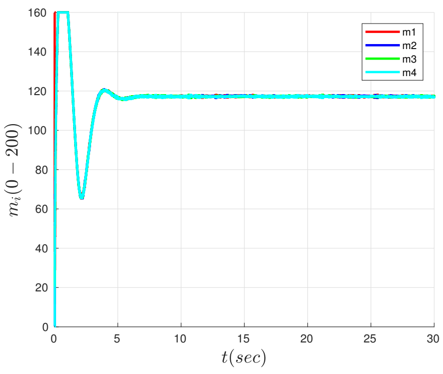


(b) Transformed states ξ_{2i}

Figure 4.7: Transformed states in the presence of sensor noise



(a) Augmented states ζ_1, ζ_2



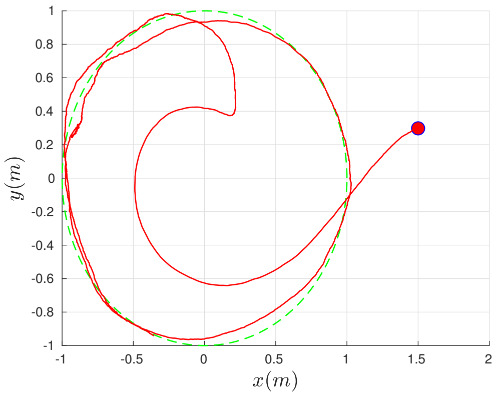
(b) Motor pwm values

Figure 4.8: Augmented states without noise

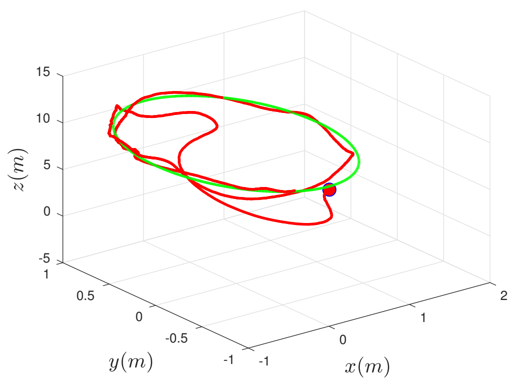
4.6.3 Noise on augmented states

The quadrotor system is capable of measuring motor speed with hall effect sensors mounted on the Pelican system. The motors speeds can be converted to thrust ζ_1, and numerically differentiating thrust gives ζ_2. We add realistic noise levels to ζ_1, and ζ_2. It should be noted that ζ_1 is not filtered at low level processor of Pelican, unlike IMU values which are filtered at low level processor. Numerical differentiation of a noisy ζ_1 results in a larger noise level on ζ_2. The noise levels of these augmented states has a large impact on path following performance as shown in Figure 4.9a, 4.9b, 4.10a , and Figure 4.10b. Finally, the noisy augmented states, and motor pwm values are show in Figure 4.11a, and Figure 4.11b. One can perform some basic online filtering, such as low pass filtering or Kalman filtering, to reduce noise on augmented states, but it leads to delay the input signal, and that can further effect the path following performance. We conclude this chapter with the following comment,

Remark 4.6.1. *In the light of Lemma 4.3.1, and Lemma 4.4.3 the $\mathcal{C}_V$ class of vehicle entails dynamic extension, which forces the system to have two more states ζ_1, ζ_2. As shown by the simulation results, these noisy augmented states greatly affects the path following control performance. This brings up a natural question: can dynamics extension be avoided? Yes, it can be*

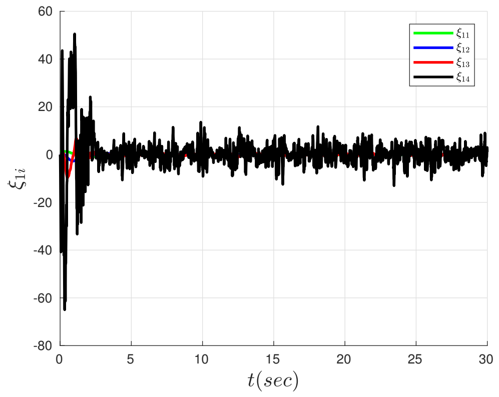


(a) Following desired path in x, y plane

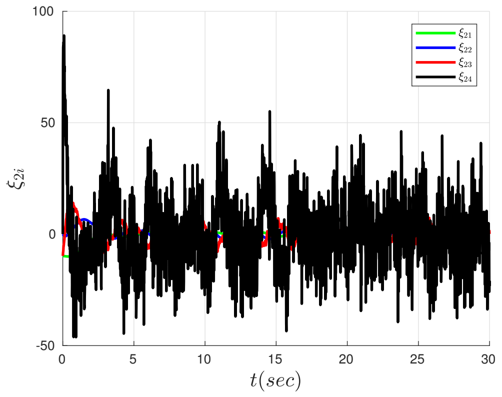


(b) Following desired path in 3D

Figure 4.9: Path following in the presence of sensor and motor noise

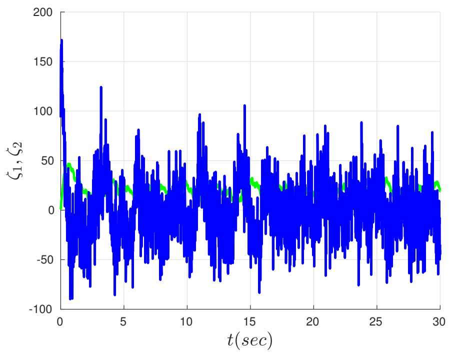


(a) Transformed states ξ_{1i}

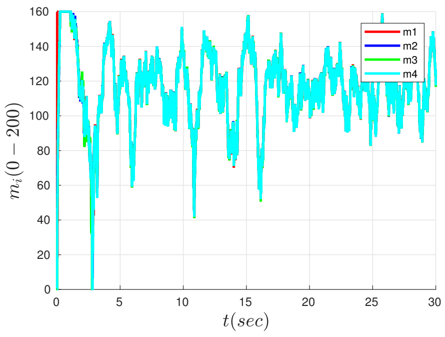


(b) Transformed states ξ_{2i}

Figure 4.10: Transformed states with noise



(a) Augmented states ζ_1, ζ_2



(b) Motor pwm values

Figure 4.11: Augmented states with noise

avoided by adopting an all together different control scheme. In other words, we used a monolithic control design approach in this chapter, that requires dynamic extension. By adopting an "inner-outer loop" approach, dynamic extension can be avoided. The idea of inner-outer loop control in not new, for example see [15]. In Chapter 6, we present a class of novel geometric controllers for rigid bodies on $\mathsf{SO}(3)$ *by exploiting the inner-outer loop approach. Before this, we consider an interesting problem of path following control for quadrotor in case of a rotor failure in the next chapter.*

Chapter 5

Fault Tolerant Path Following of a Quadrotor

In the previous chapter we have shown that the path following controller gives satisfactory performance in the presence of low sensor noise, and under close to ideal conditions the controller allows the system to follow the given curve closely. Using a similar controller design procedure used in the previous chapter we present a path following controller for a specific vehicle in the $\mathcal{C}_V$ class of vehicle, i.e., a quadrotor UAV, and consider the case when the quadrotor experience a single rotor failure. In the single rotor failure case, or when one of the rotor is fully broken we call it a three-rotor case, and when all the four rotors are working without any failure we call it a four-rotor case. Similar to the four-rotor case presented in Chapter 4, we design a smooth, dynamic feedback control law that allows the quadrotor to follow both closed and non-closed embedded curves while maintaining a desired velocity profile along the path when one out of four motors is completely disabled. Unlike the four-rotor case, this quadrotor model is not fully feedback linearizable. Therefore, the nonlinear model of the quadrotor is transformed into a partially linear model by a coordinate and feedback transformation. We prove that even with three rotors path invariance is achieved, and the controller gives satisfactory performance with no sensor noise. We further show that the uncontrolled nonlinear portion of the dynamics (internal dynamics) are bounded.

5.1 Introduction

Informally, a fault tolerant control (FTC) system is a system that can maintain stability in the presence of particular faults. In the literature, fault tolerant controllers can be broadly classified into two groups: active fault tolerant controllers (AFTC), and passive fault tolerant controllers (PFTC). AFTC relies on a fault diagnosis system, which detects a fault and makes an active change to the controller to manage the faulty state control. With a PFTC no switching occurs, rather a continuous controller is designed and optimized for the fault-free situation, while satisfying some degraded performance for the faulty case [69]. The fault tolerant path following control considered in this chapter is based on the assumption that path following must be maintained in the event one of the rotors no longer provides any thrust or moment. Such a scenario generally may occurs when a rotor collides with a static object in the environment, causing the rotor blade to break, a propeller loss, or an electrical failure. During typical operation the quadrotor flies in a fault-free fashion and the path following controller designed in Chapter 4 can be used. However, when a failure occurs, the goal of path following becomes more challenging because the requirements of path invariance and pre-defined velocity profile tracking must be maintained using only three motors. The controller is of practical importance because it allows a quadrotor to recover from a single rotor failure and maintain path invariance at the expense of independent control over yaw angle. The result is a vehicle that spins about its body z-axis while travelling along the path. The rate at which the yaw varies can be bounded, allowing for safe operation in the presence of such a failure.

Compared to the fault-free case, less research has been done for fault tolerant control of quadrotors. In [70] the authors compare existing methods of fault tolerant control systems.

In [71, 72, 73] the authors propose a fault tolerant controller for quadrotors using sliding mode control. In [74] the authors present a learning-based fault tolerant controller for the quadrotors. The authors show in simulation that the proposed method is effective for optimizing the fuzzy tracking controller on-line and counteracting the side effects of actuator faults. In [75] the authors discuss the problem of fault detection and diagnosis of an unmanned quadrotor helicopter in the presence of actuator faults. Moreover, the authors discuss three fault cases: loss of control effectiveness in one signal actuator, loss of control effectiveness in two actuators, and loss of control effectiveness in three actuators, and show experimentally the effectiveness of the proposed method. In [76] the authors proposed a fault tolerant controller based on trajectory linearization when one rotor of the quadrotor fails. In [77] the authors applied feedback linearization and discussed the problem of trajectory tracking by following the inner-outer loop control structure.

In this work, a path following controller is proposed for the case when one rotor of the quadrotor encounters a failure. The control design process is challenging because it requires the controller to manage six degrees of freedom using only three control inputs instead of four. It is shown in this work that using fault tolerant path following, the system can be made to stay precisely on the path, in other words path invariance can be achieved while the quadrotor is running only on three rotors. Unlike the controller designed in the last chapter for the case of fault-free system where the system was shown to be differentially flat, the three rotor system is not differentially flat, but instead presents uncontrolled internal dynamics. Nonetheless, it is shown that the vehicle is still able to follow the given path, maintain the desired speed along the path, and does not rotate at an unbounded rate.

5.2 Mathematical model

Without loss of generality, assume the second motor M_2 fails due to collision. A fault diagnosis system detects a severe effect on the vehicle roll control, and triggers the switch from the fault-free state to the faulty state and turns off M_2 completely. The new control inputs are τ_f, τ_q, τ_r, and (2.8) becomes

$$\begin{bmatrix} u_t \\ \tau_p \\ \tau_q \\ \tau_r \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & l \\ -l & l & 0 \\ d & d & -d \end{bmatrix} \begin{bmatrix} f_1 \\ f_3 \\ f_4 \end{bmatrix}. \tag{5.1}$$

The expression $\tau_p = lf_4 = u_t - \frac{\tau_r}{d}$ can be substituted into the quadrotor model (4.5). We add translational and rotational drag terms in (4.5), and the quadrotor model for the fault tolerant case can be obtained as,

$$\begin{aligned}
\dot{x}_1 &= x_4 + x_5 S_1 T_2 + x_6 C_1 T_2 \\
\dot{x}_2 &= x_5 C_1 - x_6 S_1 \\
\dot{x}_3 &= \sec_2(x_5 S_1 + x_6 C_1) \\
\dot{x}_4 &= -((J_z - J_y)/J_x) x_5 x_5 - (k_r x_4 / J_x) + (1/J_x)\left(\tau_f - \frac{\tau_r}{d}\right) \\
\dot{x}_5 &= -((J_x - J_z)/J_y) x_4 x_6 - (k_r x_5 / J_y) + (1/J_y)\tau_q \\
\dot{x}_6 &= -((J_y - J_x)/J_z) x_4 x_5 - (k_r x_6 / J_z) + (1/J_z)\tau_r \\
\dot{x}_7 &= x_{10} \\
\dot{x}_8 &= x_{11} \\
\dot{x}_9 &= x_{12} \\
\dot{x}_{10} &= (-k_t x_{10}/m) + (1/m)(C_1 S_2 C_3 + S_1 S_3) u_t \\
\dot{x}_{11} &= (-k_t x_{11}/m) + (1/m)(C_1 S_2 S_3 - S_1 C_3) u_t \\
\dot{x}_{12} &= (-k_t x_{12}/m) + g - (1/m)(C_1 C_2) u_t.
\end{aligned} \tag{5.2}$$

The output of the quadrotor is the position of the center of gravity of the quadrotor in the inertial frame.

5.3 Problem statement

Given a path presented in Defintion 4.2.1, we seek a smooth dynamic feedback law

$$\begin{aligned}\dot{\zeta} &= \mathcal{A}(x,\zeta) + \mathcal{B}(x,\zeta)u \\ \overline{u} &= \mathcal{C}(x,\zeta) + \mathcal{D}(x,\zeta)u,\end{aligned} \tag{5.3}$$

with[1] $\zeta \in \mathbb{R}^{\widetilde{q}}$, $u \in \mathbb{R}^3$ and an open subset of initial conditions in a neighborhood of the lift of the path, such that the quadrotor with one damaged rotor meets the following goals,

G1 The system asymptotically approaches the path, $\|h(x(t))\|_\gamma \to 0$ as $t \to \infty$.

G2 The zero level set $s(y)$ is output invariant for all $t \geq 0$.

G3 On the path, the system follows a desired speed profile along the curve.

G4 The body rates p, q, r remain bounded, i.e., $|p| < \infty$, $|q| < \infty$, and $|r| < \infty$ for all $t \geq 0$.

G5 The system does not spin at unbounded rate about its axis, i.e., $|\dot{\psi}| < \infty$ for all $t \geq 0$.

5.4 Dynamic extension

As discussed in the last chapter, the largest controlled invariant submanifold $\Gamma^\star$ contained in Γ can be obtained by applying the zero dynamics algorithm [4] to the function α, as defined in (4.9). Since the faulted quadrotor has only three inputs it is natural to augment the function (4.9) with one additional function to make the number of output functions equal to the number of control inputs and then check the relative degree of the system with respect to the augmented output. To this end let $\pi(x_7, x_8, x_9)$ be any smooth real-valued function. It is easy to show, similar to Lemma 4.3.1, that the "virtual" output $\bar{y} = (\alpha, \pi(x))$ fails to yield a well-defined relative degree for the system (5.2) because the decoupling matrix is always rank deficient.

[1]The dimension $\widetilde{q}$ of the controller state ζ is not fixed *a priori*.

This problem is overcome by delaying the appearance of the control input u_t with the help of two integrators, which are included through two additional states $x_{13} := u_t$ and $x_{14} := \dot{u}_t$. Let $u_d = \ddot{u}_t$ and $u = \mathrm{col}(u_d, u_q, u_r) \in \mathbb{R}^3$, where $u_r := \tau_r$ and $u_q := \tau_q$.

$$\begin{aligned}
\dot{x}_1 &= x_4 + x_5 S_1 T_2 + x_6 C_1 T_2 \\
\dot{x}_2 &= x_5 C_1 - x_6 S_1 \\
\dot{x}_3 &= \sec(x_2)(x_5 S_1 + x_6 C_1) \\
\dot{x}_4 &= -((J_z - J_y)/J_x) x_5 x_5 - (k_r x_4 / J_x) + (l x_{13}/2J_x) - \left(\frac{l}{2J_x d}\right) u_r \\
\dot{x}_5 &= -((J_x - J_z)/J_y) x_4 x_6 - (k_r x_5 / J_y) + (1/J_y) u_q \\
\dot{x}_6 &= -((J_y - J_x)/J_z) x_4 x_5 - (k_r x_6 / J_z) + (1/J_z) u_r \\
\dot{x}_7 &= x_{10} \\
\dot{x}_8 &= x_{11} \\
\dot{x}_9 &= x_{12} \\
\dot{x}_{10} &= (-k_t x_{10}/m) + (1/m)(C_1 S_2 C_3 + S_1 S_3) x_{13} \\
\dot{x}_{11} &= (-k_t x_{11}/m) + (1/m)(C_1 S_2 S_3 - S_1 C_3) x_{13} \\
\dot{x}_{12} &= (-k_t x_{12}/m) + g - (1/m)(C_1 C_2) x_{13} \\
\dot{x}_{13} &= x_{14} \\
\dot{x}_{14} &= u_d.
\end{aligned} \tag{5.4}$$

With a slight abuse of notation we write the extended model compactly as

$$\dot{x} = f(x) + \sum_{i=1}^{3} g_i(x) u_i.$$

As in [32], applying the zero dynamics algorithm to the output (4.9) and the extended system yields the path following manifold

$$\Gamma^\star = \left\{ x \in \mathbb{R}^{14} : L_f^i \alpha(x) = 0, i = 0, 1, 2, 3, 4 \right\}. \tag{5.5}$$

The path following manifold in the faulted case equals the path following manifold for the fault-free case. Once again, by making the path following manifold attractive and invariant, **G1** and **G2** are satisfied.

5.5 Path following controller design

Similar to the fault free case, we define a "virtual output" function $\hat{y}$ to be

$$\hat{y} = \begin{bmatrix} \alpha_1(x_7, x_8, x_9) \\ \alpha_2(x_7, x_8, x_9) \\ \pi(x_7, x_8, x_9) \end{bmatrix} = \begin{bmatrix} s_1 \circ h(\chi) \\ s_2 \circ h(\chi) \\ \varpi \circ h(\chi) \end{bmatrix}. \tag{5.6}$$

Now we check the vector relative degree of the system (5.4),

Lemma 5.5.1. *The extended model of the quadrotor with output* (5.6) *yields a well-defined vector relative degree of* $\{4, 4, 4\}$ *at each point on* $\Gamma^\star \cap \{x \in \mathbb{R}^{14} : x_{13} \neq 0\}$.

Proof. The proof is similar to the proof of Lemma 4.4.3. □

The extended system has a well defined vector relative degree of $\{4, 4, 4\}$, which implies that the dimension of the internal dynamics is $14 - (4 + 4 + 4) = 2$. Two additional functions are needed to define a complete coordinate transformation.

Corollary 5.5.2. *Let* $x^\star \in \Gamma^\star \backslash \{x \in \mathbb{R}^{14} : x_1 \pm 90^0,\ x_{13} \neq 0\}$. *There exists a neighbourhood* $U \subset \mathbb{R}^{14}$ *containing* $x^\star$ *such that the mapping* $T : U \subset \mathbb{R}^{14} \to T(U) \subset \mathbb{R}^{14}$, *defined by*

$$\begin{bmatrix} \xi_{ji} \\ \eta_i \\ \mu_k \end{bmatrix} = T(x) = \begin{bmatrix} L_f^{i-1}\alpha_j(x) \\ L_f^{i-1}\pi(x) \\ \mu(x) \end{bmatrix}, \tag{5.7}$$

for $i \in \{1, 2, 3, 4\}$, $j \in \{1, 2\}$ *and* $k \in \{1, 2\}$, *is a diffeomorphism.*

Proof. The choice of $(\xi, \eta) \in \mathbb{R}^{12}$ is clear from (5.7). However, the relative degree of the extended system is 2 less than the dimension of the state space. Therefore we must select two additional real-valued functions μ_1, μ_2 to complete the definition of T. The distribution $G_0(x) := \mathrm{span}\{g_1, g_2, g_3\}(x)$ is constant and therefore involutive. By [4, Proposition 5.1.2] there exist real-valued functions μ_1 and μ_2 whose differentials belong to the annihilator of $G_0(x)$ and complete the coordinate transformation T. Two possible choices are

$$\begin{aligned} \mu_1 &:= x_3, \\ \mu_2 &:= -\frac{-x_4}{J_z} - \frac{Lx_6}{2J_x d}. \end{aligned} \tag{5.8}$$

With the above choice of μ_1 and μ_2 it is sufficient to check the rank of the 14×14 Jacobian matrix. The determinant of Jacobian matrix $\frac{dT}{dx}$ is given by

$$\det\left(\frac{dT}{dx}\right) = \frac{-l(x_{13})^4 C_1}{2J_x m^6 d}\langle d_\chi \alpha_1, (d_\chi \alpha_2 \times \sigma')\rangle. \tag{5.9}$$

By arguments similar to given in Lemma 4.4.3, Equation (5.9) equals zero at $x^\star \in \Gamma^\star \cap \{x \in \mathbb{R}^{14} : x_{13} \neq 0\}$ if and only if $C_1 = 0$. However, by hypothesis, the gimbal lock condition $\phi = \theta = \pm 90^0$ does not hold at $x^\star$. Therefore, the Jacobian of T is non-singular in a neighbourhood of $x^\star$ and T is a local diffeomorphism. $\square$

Using the coordinate transformation T from Corollary 5.5.2, the system is differentially equivalent in a neighbourhood of $x^\star$ to

$$\begin{aligned}
\dot{\xi}_{11} &= \xi_{12} \\
\dot{\xi}_{12} &= \xi_{13} \\
\dot{\xi}_{13} &= \xi_{14} \\
\dot{\xi}_{14} &= L_f^4\alpha_1 + L_{g_1}L_f^3\alpha_1 u_f\big|_{x=T^{-1}(\eta,\xi,\mu)} \\
\dot{\xi}_{21} &= \xi_{22} \\
\dot{\xi}_{22} &= \xi_{23} \\
\dot{\xi}_{23} &= \xi_{24} \\
\dot{\xi}_{34} &= L_f^4\alpha_2 + L_{g_2}L_f^3\alpha_2 u_r\big|_{x=T^{-1}(\eta,\xi,\mu)} \\
\dot{\eta}_1 &= \eta_2 \\
\dot{\eta}_2 &= \eta_3 \\
\dot{\eta}_3 &= \eta_4 \\
\dot{\eta}_4 &= L_f^4\pi + L_{g_3}L_f^3\pi u_q\big|_{x=T^{-1}(\eta,\xi,\mu)} \\
\dot{\mu}_j &= b_j(\eta,\xi,\mu)\big|_{x=T^{-1}(\eta,\xi,\mu)}
\end{aligned} \tag{5.10}$$

for $i \in \{1,2\}$, $j \in \{1,2,3\}$, $k \in \{1,2\}$ and where b_k are smooth nonlinear functions. The structure of (5.10) suggests the feedback transformation

$$\begin{bmatrix} u_f \\ u_r \\ u_q \end{bmatrix} := D^{-1}(x)\left(\begin{bmatrix} -L_f^4\alpha_1 \\ -L_f^4\alpha_2 \\ -L_f^4\pi \end{bmatrix} + \begin{bmatrix} v^{\xi_1} \\ v^{\xi_2} \\ v^{\eta} \end{bmatrix}\right), \tag{5.11}$$

where $(v^{\xi_1}, v^{\xi_2}, v^{\eta})$ are auxiliary control inputs. By Lemma 5.5.1 the feedback transformation (5.11) is well-defined in a neighborhood of every $x^\star \in \Gamma^\star \cap \{x \in \mathbb{R}^{14} : x_{13} \neq 0\}$. Thus in a neighborhood of $x^\star$, the closed-loop system is reduced to 3 decoupled chains of integrators and the nonlinear internal dynamics of the system.

$$\begin{array}{llll} \dot{\xi}_{11} = \xi_{12} & \dot{\xi}_{21} = \xi_{22} & \dot{\eta}_1 = \eta_2 & \dot{\mu}_1 = b_1(\xi, \eta, \mu) \\ \dot{\xi}_{12} = \xi_{13} & \dot{\xi}_{22} = \xi_{23} & \dot{\eta}_2 = \eta_3 & \dot{\mu}_2 = b_2(\xi, \eta, \mu). \\ \vdots & \vdots & \vdots & \\ \dot{\xi}_{14} = v^{\xi_1} & \dot{\xi}_{24} = v^{\xi_2} & \dot{\eta}_4 = v^{\eta} & \end{array} \tag{5.12}$$

After applying the coordinate and feedback transformations (5.7), (5.11) to the extended system the auxiliary controller design is straight forward for the linear subsystems. A linear controller, similar to the last chapter, can be designed to stabilize the origin of the ξ. Similarly a linear controller can be designed for the $\eta-$subsystem to satisfy **G3**.

5.6 Internal dynamics

The μ-subsystem represents the internal dynamics

$$\begin{aligned} \dot{\mu}_1(\eta, \xi, \mu) &= \sec(x_2)(x_6 C_1 + x_5 S_1)\big|_{x=T^{-1}(\eta,\xi,\mu)}, \\ \dot{\mu}_2(\eta, \xi, \mu) &= -\frac{0.5 l x_{13}}{J_x J_z} + \frac{l k_r x_6}{2 J_x J_z d} + \frac{k_r x_4 + x_5 x_6 (J_x - J_z)}{J_x J_z}\Bigg|_{x=T^{-1}(\eta,\xi,\mu)}. \end{aligned} \tag{5.13}$$

In order to prove boundedness of the internal dynamics (Lemma 5.6.3), we need the following preliminary results. We first analyze the stability of the set of differential equations involving the dynamics of the body rates from when the control inputs are set to zero.

Lemma 5.6.1. *The origin* $(x_4, x_5, x_6) = (0, 0, 0)$ *of*

$$\begin{aligned} \dot{x}_4 &= -((J_z - J_y)/J_x) x_5 x_6 - (k_r x_4)/J_x \\ \dot{x}_5 &= -((J_x - J_z)/J_y) x_4 x_6 - (k_r x_5)/J_y \\ \dot{x}_6 &= -((J_y - J_x)/J_z) x_4 x_5 - (k_r x_6)/J_z. \end{aligned} \tag{5.14}$$

is globally exponentially stable.

Proof. We assume, without the loss of generality, that $J_x \geq J_y \geq J_z$, and let

$$a_1 := \frac{J_y - J_z}{J_x}, \quad a_2 := \frac{J_x - J_z}{J_y}, \quad a_3 := \frac{J_x - J_y}{J_z}$$
$$k_4 := \frac{k_r}{J_x}, \quad k_5 := \frac{k_r}{J_y}, \quad k_6 := \frac{k_r}{J_z}.$$

If $J_x \geq J_y \geq J_z$ does not hold, a_1, a_2, a_3 can be redefined, so that these constants are non-negative. With these definitions, (5.14) becomes

$$\begin{aligned} \dot{x}_4 &= a_1 x_5 x_6 - k_4 x_4 \\ \dot{x}_5 &= -a_2 x_4 x_6 - k_5 x_5 \\ \dot{x}_6 &= a_3 x_4 x_5 - k_6 x_6. \end{aligned} \tag{5.15}$$

Equation (5.15) can be written as $\dot{\overline{x}} = \overline{f}(\overline{x})$, for $\overline{x} := \mathrm{col}(x_4, x_5, x_6)$. Choose as a candidate Lyapunov function $V : \mathbb{R}^3 \to \mathbb{R}$

$$V(\overline{x}) = \overline{x}^\top P \overline{x} := p_1 x_4^2 + p_2 x_5^2 + p_3 x_6^2 \tag{5.16}$$

where $P := \mathrm{diag}\,(p_1, p_2, p_3)$. If p_1, p_2, p_3 are positive then V is a positive definite quadratic form. The Lie derivative of V along the vector field (5.15) is

$$L_{\overline{f}} V = 2x_4 x_5 x_6 \,(a_1 p_1 - a_2 p_2 + a_3 p_3) - 2\overline{x}^\top \,\mathrm{diag}\,(k_4 p_1, k_5 p_2, k_6 p_3)\overline{x}.$$

Now choose $p_1, p_2, p_3 > 0$ so that $a_1 p_1 - a_2 p_2 + a_3 p_3 = 0$. This is always possible, for example $p_1 = J_x$, $p_2 = J_y$, $p_3 = J_z$ works. In summary we have that

(i) For all $\overline{x} \in \mathbb{R}^3$,

$$\min\{p_1, p_2, p_3\} \|\overline{x}\|^2 \leq V(\bar{x}) \leq \max\{p_1, p_2, p_3\} \|\overline{x}\|^2$$

(ii) For all $\overline{x} \in \mathbb{R}^3$,

$$\begin{aligned} L_{\overline{f}}V &= -2\overline{x}^\top \operatorname{diag}(k_4p_1, k_5p_2, k_6p_3)\overline{x} \\ &\leq -2\min\{k_4p_1, k_5p_2, k_6p_3\}\|\overline{x}\|^2 \\ &= -2\min\{k_4p_1, k_5p_2, k_6p_3\}\frac{\max\{p_1,p_2,p_3\}}{\max\{p_1,p_2,p_3\}}\|\overline{x}\|^2 \\ &\leq -2\frac{\min\{k_4p_1, k_5p_2, k_6p_3\}}{\max\{p_1,p_2,p_3\}}V(\overline{x}) \end{aligned}$$

Conditions (i) and (ii) imply, by [78, Theorem 3.1], that $\overline{x} = 0$ is globally exponentially stable.

□

Next we analyze the stability of the body rate equations and show that they are input-to-state stable (ISS-stable) [79]. Let $k_7 := l/2J_x$, $k_8 := -1/dJ_x$, $k_9 := 1/J_y$, $k_{10} := 1/J_z$.

Lemma 5.6.2. *The system* (5.17)

$$\begin{aligned} \dot{x}_4 &= a_1x_5x_6 - k_4x_4 + k_7x_{13} + k_8u_2 \\ \dot{x}_5 &= -a_2x_4x_6 - k_5x_5 + k_9u_3 \\ \dot{x}_6 &= a_3x_4x_5 - k_6x_6 + k_{10}u_2, \end{aligned} \tag{5.17}$$

is input-to-state stable.

Proof. To be consistent with the notation we used for (5.15), write system (5.17) as $\dot{\overline{x}} = \overline{f}(\overline{x}) + B\overline{w}$ where $\overline{w} := \operatorname{col}(x_{13}, u_r, u_q)$. To prove that the system (5.17) is ISS-stable we show that the function (5.16) is an ISS-Lyapunov function. As in the proof of Lemma 5.6.1, choose p_1, p_2, $p_3 > 0$ so that $a_1p_1 - a_2p_2 + a_3p_3 = 0$. Then $Q := \operatorname{diag}(k_4p_1, k_5p_2, k_6p_3)$ and we have

$$\begin{aligned} \dot{V} &= -2\overline{x}^\top Q\overline{x} + 2\overline{x}^\top PB\overline{w} \\ &= -2(1-\theta)\overline{x}^\top Q\overline{x} - 2\theta\overline{x}^\top Q\overline{x} + 2\overline{x}^\top QB\overline{w}, \; (\forall\, \theta \in (0,1)) \\ &\leq -2(1-\theta)\overline{x}^\top Q\overline{x} - 2\theta\min\{k_4p_1, k_5p_2, k_6p_3\}\|\overline{x}\|^2 + 2\overline{x}^\top QB\overline{w} \\ &\leq -2(1-\theta)\overline{x}^\top Q\overline{x} - 2\theta\min\{k_4p_1, k_5p_2, k_6p_3\}\|\overline{x}\|^2 + 2\|\overline{x}\|\|Q\|\|B\|\|\overline{w}\|. \end{aligned}$$

Thus

$$\forall\ \|\overline{x}\| \geq \frac{\|Q\|_2\|B\|_2}{\theta \min\{k_4p_1, k_5p_2, k_6p_3\}}\|\overline{w}\|,$$
$$\dot{V} \leq -(1-\theta)\overline{x}^\top Q\overline{x}$$

where $\theta \in (0,1)$. This shows, by [80, Theorem 4.19], that (5.17) is ISS stable. □

In summary, by Lemmas 5.6.1 and 5.6.2 we have shown that the body rates x_4, x_5, x_6 are bounded.

Lemma 5.6.3. *If the control inputs u_f, u_r, u_q of the quadrotor are bounded and the quadrotor avoids the gimbal lock condition ($x_1 = x_2 = \pm 90^0$), then $\dot{\mu}_1$ and $\dot{\mu}_2$ in (5.13) are bounded. Moreover, μ_2 is bounded.*

Proof. By Lemma 5.6.2 we have shown that for any bounded inputs, the body rates x_4, x_5, x_6 are bounded. By hypothesis the system is bounded away from Euler angle singularities ($x_1 = x_2 = \pm 90^0$). From the expressions (5.13) we have that

$$\begin{aligned}
|\dot{\mu}_1| &\leq \sec(x_2)\left(|x_5| + |x_6|\right), \\
|\dot{\mu}_2| &\leq \frac{lk_r}{2dJ_xJ_z}|x_6| + \frac{l}{2J_xJ_z}|x_{13}| + \frac{k_r}{J_xJ_z}|x_4| + \frac{J_z - J_x}{J_xJ_z}|x_5||x_6|, \\
|\mu_2| &\leq \frac{1}{J_z}|x_4| - \frac{l}{2dJ_x}|x_6|,
\end{aligned} \tag{5.18}$$

which is bounded because the body rates are bounded and $x_2 \neq 0$ during the flight. □

In summary, all the goals **G1-G5** are achieved. It is interesting to note that the internal state μ_1 which represent the yaw angle may become unbounded. This implies that the quadrotor is spinning about its body z-axis while traveling along the path, which is not surprising as there is an imbalance in torques that results when one of the four rotors fails. However, we have shown that the rate at which the quadrotor spins $\dot{\mu}_1 = \dot{\psi}$ is bounded, which is a result of the rotational drag term about the body z-axis that resists overly fast rotation, regardless of the path traveled. The result is a physically meaningful path following controller that sacrifices yaw angle control to maintain path invariance and keep the quadrotor safe when a failure occurs.

5.7 Simulation

The initial position of the quadrotor is indicated by a solid dot. The values of k_t and k_r used in the simulation are taken from the rotational and translational drag models presented in [57, 81]. We consider a curve at varying height given by $\sin(x_7) + 3$ units represented as $\sigma : \mathbb{R} \to \mathbb{R}^3$, $\lambda \mapsto \mathrm{col}(\lambda, \cos(\lambda), \sin(\lambda) + 3)$. The implicit representation of the same curve is given by $\gamma = \{s_1(y) = s_2(y) = 0\}$, where $s_1(y) = y_2 - \cos(y_1) = 0$ and $s_2 = y_3 + \sin(y_1) - 3$. The quadrotor is initialized at $(x_7, x_8, x_9) = (0, 0.9, 0)$. The results are shown in Figure 5.1. While following the

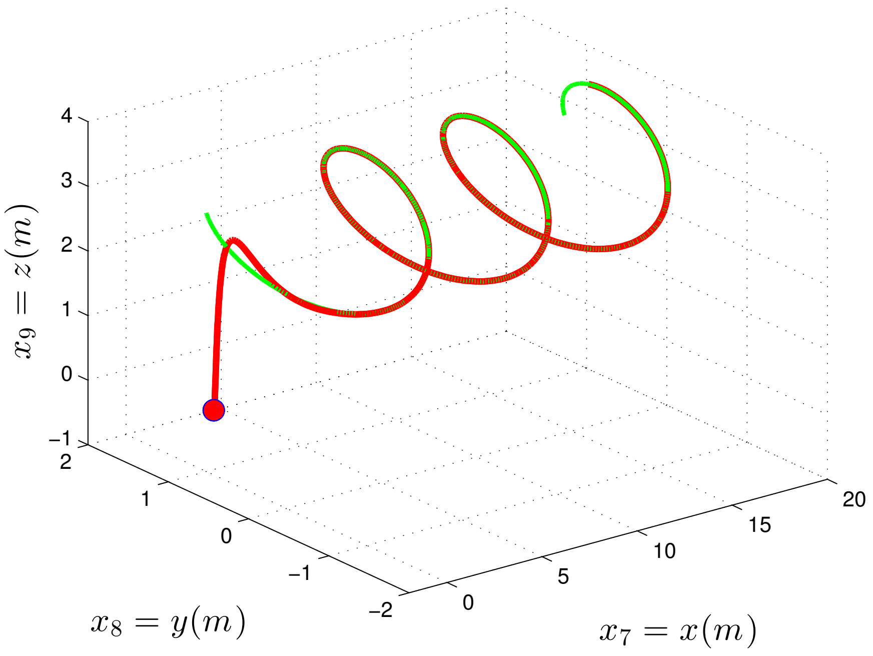


Figure 5.1: The path followed by the quadrotor is represented by a bold red line and the desired path is represented by a dashed green line. The initial position of the quadrotor is represented by a solid dot.

path γ the quadrotor is following the curve at a desired constant speed of 0.3 m/sec, as presented in Figure 5.2.

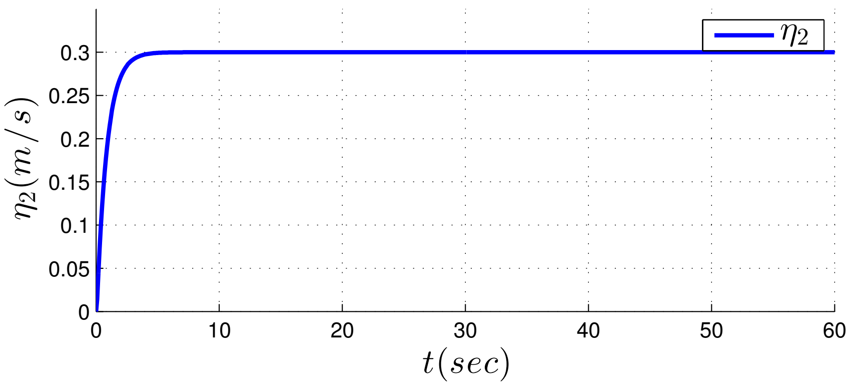


Figure 5.2: The quadrotor is traversing the curve at the desired velocity of 0.3 m/sec.

In Figure 5.3 the internal state $\dot{\mu}_1$ is shown. It represents the rate at which the quadrotor is spinning about its axis which remains bounded.

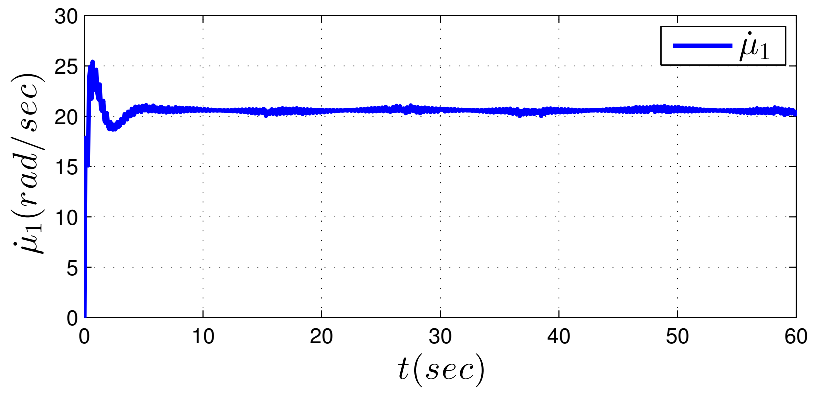


Figure 5.3: The yaw rate $\dot{\mu}_1$ remains bounded as the quadrotor traverses the desired path.

This controller also requires dynamic extension, and an implementation needs the knowledge of augmented states ζ_1, and ζ_2. In ideal or close to ideal situations with none or very little sensor noise levels, such controllers perform very well. However, in case of very low sensor noise, it should be noted that ζ_2 is computed by taking derivative of the thrust, which results in a signal with a relatively large noisy signal. This noisy signal may degrade the overall performance of closed loop system significantly in practical scenarios. In the next chapter, we overcome this issue by adopting a control strategy that does not require dynamic extension, and the need to use augmented states for controller design.

Chapter 6

Controller Class $\mathcal{C}_R$ for Attitude Tracking of $\mathcal{C}_V$ Vehicles

In Chapter 4, and Chapter 5 a monolithic controller design approach is employed which seeks to stabilize and track all states simultaneously. The result for rigid bodies from the $\mathcal{C}_V$ class of vehicles is that the controller relies on dynamic extension, leading to sensitivity to model errors, and sensor noise. An alternative approach is to design a cascaded control system, also known as "inner-outer loop" control, wherein the inner-loop stabilizes a rigid body's attitude, and the outer loop its position and velocity by requesting desired attitudes (and thrust) from the inner loop. In an effort to eliminate the sensitivity of these control designs to noise, this chapter proposes an attitude stabilization and tracking approach for inner loop control that avoids dynamic extension. Further, the design operates directly on $\mathsf{SO}(3)$, eliminating concerns over local chart limitations such as gimbal lock. A Lie algebra valued function family is presented that induces a class of geometric controllers on $\mathsf{SO}(3)$, for which stability is proven.

6.1 Mathematical model

Consider, once again, a rigid body moving in free space. As shown in Chapter 2, the rigid body dynamics are represented by (2.3), and (2.6), and are given here again for convenience,

$$\frac{\mathrm{d}}{\mathrm{d}\,t}R(t) = R(t)\widehat{\Omega}(t) \tag{6.1}$$

$$J\frac{\mathrm{d}}{\mathrm{d}\,t}\Omega(t) = \tau(t) - (\Omega(t) \times J\Omega(t)). \tag{6.2}$$

We apply a preliminary feedback to the system

$$\tau = \Omega(t) \times J\Omega(t) + Ju$$

where $u \in \mathbb{R}^3$ is an auxiliary control input to be designed. This feedback simplifies (6.2) and we obtain

$$\dot{\Omega}(t) = u. \tag{6.3}$$

Given the attitude dynamics of a rigid body (6.1), (6.3) and a desired attitude $R_d : [0, \infty) \to \mathsf{SO}(3)$ generated by the exogenous system

$$\begin{aligned} \dot{R}_d(t) &= R_d(t)\widehat{\Omega}_d(t) \\ \dot{\Omega}_d(t) &= u_d(t), \end{aligned} \tag{6.4}$$

where $u_d : [0, \infty) \to \mathbb{R}^3$ is assumed to be continuously differentiable, we seek a feedback control law for the input u in (6.3) such that $R(t)$ asymptotically approaches $R_d(t)$ when the initial "tracking error" $R(0) - R_d(0)$ is sufficiently small. We further assume that the reference frame $R_d(t)$ does not move "too quickly" in a sense to be made precise. We allow our controllers to depend on the plant states R, Ω as well as the exosystem states R_d, Ω_d and u_d. Thus we call this an attitude tracking problem with full information. With a slight abuse of notion, we omit time dependency of R, and $\widehat{\Omega}$, unless explicitly mentioned for the sake of simplification of time dependent expressions of R, and $\widehat{\Omega}$ and their derivatives. Next we present some basic, yet useful, results essential for the controller design section.

6.2 Problem formulation

First we present a notion of distance on $\mathsf{SO}(3)$. The distance on $\mathsf{SO}(3)$ cannot be defined in a usual Euclidean sense, because $\mathsf{SO}(3)$ is a manifold. To compare the "difference" between two rotation matrices $R_1, R_2 \in \mathsf{SO}(3)$, a distance function or a metric on $\mathsf{SO}(3)$ can be defined, as shown in [56] ,

$$\begin{aligned} \Lambda\colon\ \mathsf{SO}(3)^2 &\to [0, 2\sqrt{2}] \subset \mathbb{R}^+ \\ (R, R_d) &\mapsto \|I - R_1^\top R_2\|_F, \end{aligned} \tag{6.5}$$

where $\|.\|_F$ is the Frobenius norm of the matrix. It can be proven that Λ is a metric on $\mathsf{SO}(3)$, see [56] for details. It should be noted that Λ gives a measure of rotation required to apply to R_1 to align it with R_2.

6.2.1 Problem statement

Consider a rigid body represented by (6.1), (6.3), and an exosystem given by (6.4), there exists $d_\Lambda \in [0, 2\sqrt{2})$, we seek a smooth feedback control law $u(R, R_d, \widehat{\Omega}, \widehat{\Omega}_d)$ such that if, $\|I - R_d^\top R\| < d_\Lambda$ for all $t \geq 0$, the rigid body satisfies the following tracking goals,

T1 The attitude of the rigid body tracks the desired attitude of the exosystem, $\|I - R_d^\top R\|_F \to 0$, as $t \to \infty$.

T2 The body rates of the rigid body tracks the desired body rates of the exosystem, $\|\Omega - \Omega_d\| \to 0$, as $t \to \infty$.

T3 The rigid body is capable of performing multiple flips.

We call this the attitude tracking problem. We characterize d_Λ precisely in the sections to follow.

6.3 Derivatives on $\mathsf{SO}(3)$

The controller design process presented in this chapter requires computing derivatives of certain function, and some key definitions.

Definition 6.3.1. *[5] The matrix exponential function,* $\exp : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n}$*, is defined in terms of the Taylor series expansion of the exponential,*

$$\exp(A) := e^{A} = I + A + \frac{A^2}{2!} + \frac{A^3}{3!} + \cdots .$$

Definition 6.3.2. *[5] The matrix log is defined only for matrices close to the identity matrix* I*,*

$$\mathrm{Log}(X) := (X-I) - \frac{(X-I)^2}{2} + \frac{(X-I)^3}{3} - + \cdots .$$

As shown in Chapter 2, the matrix exponential $(\exp)$ is an analytic diffeomorphism between

$$U_{\mathfrak{so}(3)} := \left\{\widehat{\omega} \in \mathfrak{so}(3) : \omega \in \mathbb{R}^3, \|\omega\|_2 < \pi\right\}$$

and

$$U_{\mathsf{SO}(3)} := \{R \in \mathsf{SO}(3) : \mathrm{trace}\,(R) \neq -1\} .$$

The inverse of exponential map denoted by $\mathrm{Log} : U_{\mathsf{SO}(3)} \to U_{\mathfrak{so}(3)}$ is the principle matrix logarithm defined by 6.3.2.

Proposition 6.3.3. *Given* $R \in \mathsf{SO}(3)$*, and* $\widehat{\Omega} \in \mathfrak{so}(3)$*, satisfying* $\dot{R} = R\widehat{\Omega}$*, then*

$$\frac{\mathrm{d}}{\mathrm{d}\,t}(R^{\top}) = \left(\frac{\mathrm{d}}{\mathrm{d}\,t}R\right)^{\top} . \tag{6.6}$$

Proof. Since $R \in \mathsf{SO}(3)$, by definition,

$$R^{\top}R = I,$$

by taking derivate of both sides,

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}\,t}(R^\top)R + R^\top\dot{R} &= 0 \qquad (6.7)\\
\frac{\mathrm{d}}{\mathrm{d}\,t}(R^\top)R &= -R^\top\dot{R}\\
\frac{\mathrm{d}}{\mathrm{d}\,t}(R^\top)R &= -R^\top(R\widehat{\Omega})\\
\frac{\mathrm{d}}{\mathrm{d}\,t}(R^\top) &= -R^\top(R\widehat{\Omega})R^\top\\
\frac{\mathrm{d}}{\mathrm{d}\,t}(R^\top) &= -\widehat{\Omega}R^\top, \qquad \text{by associativity of matrix product}\\
\frac{\mathrm{d}}{\mathrm{d}\,t}(R^\top) &= \left(\frac{\mathrm{d}}{\mathrm{d}\,t}R\right)^\top.
\end{aligned}$$

□

Proposition 6.3.4. *Given $R, R_d \in \mathsf{SO}(3)$, and $\widehat{\Omega}, \widehat{\Omega}_d \in \mathfrak{so}(3)$, satisfying $\dot{R} = R\widehat{\Omega}$, and $\dot{R}_d = R_d\widehat{\Omega}_d$, then*

(i)

$$\frac{\mathrm{d}}{\mathrm{d}\,t}(R_d^\top R) = (R_d^\top R)\widehat{\Omega} - \widehat{\Omega}_d(R_d^\top R),$$

(ii)

$$\frac{\mathrm{d}}{\mathrm{d}\,t}(R^\top R_d) = (R^\top R_d)\widehat{\Omega}_d - \widehat{\Omega}(R^\top R_d),$$

(iii)

$$\frac{\mathrm{d}}{\mathrm{d}\,t}(R_d^\top R) - \frac{\mathrm{d}}{\mathrm{d}\,t}(R^\top R_d) \in \mathfrak{so}(3),$$

(iv)

$$\frac{\mathrm{d}}{\mathrm{d}\,t}(RR_d^\top) = R\widehat{\Omega}R_d^\top - R\widehat{\Omega}_d R_d^\top,$$

(v)

$$\frac{\mathrm{d}}{\mathrm{d}\,t}(R_dR^\top) = R_d\widehat{\Omega}_d R^\top - R_d\widehat{\Omega}R^\top,$$

(vi)

$$\frac{\mathrm{d}}{\mathrm{d}\,t}(RR_d^\top) - \frac{\mathrm{d}}{\mathrm{d}\,t}(R_dR^\top) \in \mathfrak{so}(3),$$

Proof. We prove (i) by taking time derivative of $R_d^\top R$,

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}\,t}(R_d^\top R) &= \dot{R}_d^\top R + R_d^\top \dot{R}, && \text{by Proposition 6.3.3}\\
&= \left(R_d\widehat{\Omega}_d\right)^\top R + R_d^\top\left(R\widehat{\Omega}\right)\\
&= -\left(\widehat{\Omega}_dR_d^\top\right)R + R_d^\top\left(R\widehat{\Omega}\right)\\
&= -\widehat{\Omega}_d\left(R_d^\top R\right) + \left(R_d^\top R\right)\widehat{\Omega}, && \text{by associativity of matrix product}
\end{aligned}$$

This completes the proof of (i). Proof of (ii) follows by Proposition 6.3.3. To prove (iii), let

$$A = \frac{\mathrm{d}}{\mathrm{d}\,t}(R_d^\top R) - \frac{\mathrm{d}}{\mathrm{d}\,t}(R^\top R_d).$$

It can be easily seen that A is a skew symmetric matrix by definition, i.e., $A + A^\top = 0$. Hence $A \in \mathfrak{so}(3)$, which proves (iii). Proofs of (iv), (v), and (vi) are similar to the proofs of (i), (ii), and (iii), respectively. This completes the proof. $\square$

Definition 6.3.5. *Let $A \in \mathbb{R}^{n\times n}$. Then the sequence $\{S_n\}_{n\geq 0}$ defined by*

$$S_n = I + A + \cdots A^{n-1}$$

is called the geometric series generated by A. The series converges if the sequence $\{S_n\}_{n\geq 0}$ converge.

Next we state a well known result of linear algebra.

Theorem 6.3.6. *[82, Theorem 7.14] The geometric series generated by $A \in \mathbb{R}^{n\times n}$ converges if and only if $|\lambda_i| < 1$, for each eigenvalue λ_i of A. If this condition holds, then $I - A$ is invertible and we have*

$$S_n := \sum_{k=0}^{n-1} A^k = (I-A)^{-1}(I-A^n),$$

and hence the series converges to

$$\sum_{k=0}^{\infty} A^k = (I - A)^{-1}.$$

Remark 6.3.7. *Given $R \in \mathsf{SO}(3)$, it is easy to check that each eigenvalue of $(I - R)$ is strictly less than 1, and by Theorem 6.3.6*

$$\sum_{k=0}^{\infty} (I - R)^k = (I - I + R)^{-1} = R^{-1} = R^\top. \tag{6.8}$$

Proposition 6.3.8. *Let $R(t) \in \mathsf{SO}(3)$ satisfying*

$$\dot{R}(t) = R(t)\widehat{\Omega}(t),$$

then $\frac{\mathrm{d}}{\mathrm{d}\,t} \operatorname{Log}(R(t)) = \left(R^\top\right) \dot{R} = \widehat{\Omega}(t)$

Proof. Let I by the 3 by 3 identity matrix. We prove the result by construction, and applying definition 6.3.2. To simplify notation, time dependency of R and $\widehat{\Omega}$ are omitted,

$$\begin{aligned}
\operatorname{Log}(R) &= \operatorname{Log}(I + (R - I)) \\
&= \sum_{n=1}^{\infty} \frac{(-1)^{n+1}}{n} (R - I)^n \\
&= \frac{(-1)^2}{1}(R - I) + \frac{(-1)^3}{2}(R - I)^2 + \frac{(-1)^4}{3}(R - I)^3 + \cdots ,
\end{aligned}$$

by taking time derivative of both sides of the equation we get,

$$
\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}\,t}\operatorname{Log}(R) &= \dot{R} + \frac{(-1)^3}{2}2(R-I)\dot{R} + \frac{(-1)^4}{3}3(R-I)^2\dot{R} + \frac{(-1)^5}{4}4(R-I)^3\dot{R} + \cdots \\
&= \left(I + (-1)^3(R-I) + (-1)^4(R-I)^2 + (-1)^5(R-I)^3\right)\dot{R} + \cdots \\
&= \left(I + (-1)(R-I) + (-1)^2(R-I)^2 + (-1)^3(R-I)^3\right)\dot{R} + \cdots \\
&= \left(\sum_{n=0}^{\infty}(-1)^n(R-I)^n\right)\dot{R} \\
&= \left(\sum_{n=0}^{\infty}(-R+I)^n\right)\dot{R} \\
&= \left(R^\top\right)\dot{R}, \qquad \text{by (6.8)} \\
&= \left(R^\top\right)(R\widehat{\Omega}) \\
&= \widehat{\Omega}.
\end{aligned}
$$

□

Proposition 6.3.9. *Let $R(t) \in \mathsf{SO}(3)$ satisfying*

$$\dot{R}(t) = R(t)\widehat{\Omega}(t),$$

then $\frac{\mathrm{d}}{\mathrm{d}\,t}\operatorname{Log}(R^\top(t)) = -\widehat{\Omega}(t)$

Proof. The proof is similar to the proof of proposition 6.3.8. □

Proposition 6.3.10. *Given $R, R_d \in \mathsf{SO}(3)$, and $\widehat{\Omega}, \widehat{\Omega}_d \in \mathfrak{so}(3)$, satisfying $\dot{R} = R\widehat{\Omega}$, and $\dot{R}_d = R_d\widehat{\Omega}_d$, then*

(i)

$$\frac{\mathrm{d}}{\mathrm{d}\,t}\left(\operatorname{Log}(R_d^\top R)\right) = \widehat{\Omega} - (R^\top R_d)\widehat{\Omega}_d(R_d^\top R),$$

(ii)

$$\frac{\mathrm{d}}{\mathrm{d}\,t}\left(\operatorname{Log}(R^\top R_d)\right) = \widehat{\Omega}_d - (R_d^\top R)\widehat{\Omega}(R^\top R_d),$$

(iii)

$$\frac{\mathrm{d}}{\mathrm{d}\,t}\left(\mathrm{Log}(RR_d^\top)\right) = R_d\widehat{\Omega}R_d^\top - R_d\widehat{\Omega}_dR_d^\top = \mathrm{Adj}_{R_d}\widehat{\Omega} - \mathrm{Adj}_{R_d}\widehat{\Omega}_d,$$

(iv)

$$\frac{\mathrm{d}}{\mathrm{d}\,t}\left(\mathrm{Log}(R_dR^\top)\right) = R\widehat{\Omega}_dR^\top - R\widehat{\Omega}R^\top = \mathrm{Adj}_{R}\widehat{\Omega}_d - \mathrm{Adj}_{R}\widehat{\Omega},$$

(v) Each of the derivatives listed above, i.e., $(i), (ii), (iii), (iv)$*, is an element of* $\mathfrak{so}(3)$.

Proof. We start by proving (i). Let $A = (R_d^\top R)$. By Proposition 6.3.8, we can write

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}\,t}\mathrm{Log}(A) &= \left(A^\top\right)\dot{A} \\
&= (R_d^\top R)^\top\left(\frac{\mathrm{d}}{\mathrm{d}\,t}(R_d^\top R)\right) \\
&= (R^\top R_d)\left((R_d^\top R)\widehat{\Omega} - \widehat{\Omega}_d(R_d^\top R)\right), \qquad \text{by Proposition 6.3.4} \\
&= \widehat{\Omega} - (R^\top R_d)\widehat{\Omega}_d(R_d^\top R) \\
&= \widehat{\Omega} - \mathrm{Adj}_{(R^\top R_d)}\widehat{\Omega}_d.
\end{aligned}$$

By definition$\mathrm{Adj}_{(R^\top R_d)}\widehat{\Omega}_d \in \mathfrak{so}(3)$, and $\mathfrak{so}(3)$ is closed under addition, therefore the above expression is an element of $\mathfrak{so}(3)$, which proves (i). The proof of (ii) follows a similar sequence. To prove (iii), similar to (i), we write,

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}\,t}\left(\mathrm{Log}((RR_d^\top))\right) &= \left(\left(RR_d^\top\right)^\top\right)\left(\frac{\mathrm{d}}{\mathrm{d}\,t}(RR_d^\top)\right) \\
&= (R_dR^\top)\left(R\widehat{\Omega}R_d^\top - R\widehat{\Omega}_dR_d^\top\right), \qquad \text{by Proposition 6.3.4} \\
&= R_d\widehat{\Omega}R_d^\top - R_d\widehat{\Omega}_dR_d^\top \\
&= \mathrm{Adj}_{R_d}\widehat{\Omega} - \mathrm{Adj}_{R_d}\widehat{\Omega}_d.
\end{aligned}$$

By definition the above expression is in $\mathfrak{so}(3)$. The proof of (iv) follows the similar sequence of (iii) and is not included to avoid repetition of the arguments, which proves (v), This completes the proof. □

6.4 Function family $\mathcal{F}_R$

To solve the attitude tracking problem, we consider a family of function $\mathcal{F}_R$. We call this family $\mathcal{F}_R$ because it consists of functions that depends only on position, i.e., rotation matrices R, and R_d.

Definition 6.4.1. *A function $f : U \subseteq (\mathsf{SO}(3))^2 \to \mathfrak{so}(3)$ is said to belong to $\mathcal{F}_R$ if there exists an open set $U \subseteq (\mathsf{SO}(3))^2$ containing (I, I) such that*

P1 *It is twice continuously differentiable on U.*

P2 $f^{-1}(0) = \{(R, R_d) \in U : R = R_d\}$.

P3 *For all $X = (I, R_d, 0, \widehat{\Omega}_d) \in U \times \mathfrak{so}(3)^2$, the differential of $\frac{\mathrm{d}}{\mathrm{d}\,t} f(R, R_d)$ with respect to $\widehat{\Omega}$ is non-singular.*

Example 6.4.2. *Consider the function*

$$\begin{aligned} f\colon U \subseteq \mathsf{SO}(3)^2 &\to \mathfrak{so}(3) \\ (R, R_d) &\mapsto \log(R_d^\top R). \end{aligned}$$

Take the open set U to be

$$U := \left\{(R, R_d) \in (\mathsf{SO}(3))^2 : \operatorname{trace}(R_d^\top R) \neq -1\right\}.$$

Since the matrix logarithm is analytic in U, $\mathbf{P1}$ *holds. Also, it's clear that $f^{-1}(0)$ is the subset of U in which $R = R_d$. To check* $\mathbf{P3}$*, we take the derivative of f. By Proposition 6.3.10*

$$\frac{\mathrm{d}}{\mathrm{d}\,t} f(R, R_d) = \widehat{\Omega} - (R^\top R_d)\widehat{\Omega}_d(R_d^\top R). \tag{6.9}$$

From the expression above it is easy to check that $\mathrm{d}_{\widehat{\Omega}}(\frac{\mathrm{d}}{\mathrm{d}\,t} f(R, R_d))$ is non-singular everywhere in U, satisfying $\mathbf{P3}$. ▲

Example 6.4.3. *Consider the function*

$$f\colon U \to \mathfrak{so}(3)$$
$$(R, R_d) \mapsto R_d^\top R - R^\top R_d$$

with

$$U := \left\{(R, R_d) \in \mathsf{SO}(3)^2 : \Lambda(R, R_d) < 2\right\}.$$

Similar to Example 6.4.2, one can check that this function satisfies both $\mathbf{P1}$ *and* $\mathbf{P2}$*. Using the results from Proposition 6.3.4 the first time derivative can be written as,*

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}\,t} f(R, R_d) &= \frac{\mathrm{d}}{\mathrm{d}\,t}(R_d^\top R - R^\top R_d) \\ &= (R_d^\top R)\widehat{\Omega} - \widehat{\Omega}_d(R_d^\top R) + \widehat{\Omega}(R^\top R_d) - (R^\top R_d)\widehat{\Omega}_d.\end{aligned} \tag{6.10}$$

which, using property (1.1)*, can be written as*

$$\mathrm{d}_{\widehat{\Omega}}\left(\frac{\mathrm{d}}{\mathrm{d}\,t} f(R, R_d)\right) = \mathrm{trace}(R^\top R_d)I - (R^\top R_d).$$

It can be seen that $\{\mathrm{trace}(R^\top R_d)I - (R^\top R_d)\} \in \mathrm{GL}(3, \mathbb{R})$ *only when* R*, is a neighborhood of* R_d*. In other words* $\mathrm{d}_{\widehat{\Omega}}(\frac{\mathrm{d}}{\mathrm{d}\,t} f(R, R_d))$ *is non-singular in some neighborhood of* R_d*, hence* $\mathbf{P3}$ *is also satisfied. At* $R = R_d$*,* $\{\mathrm{trace}(R^\top R_d)I - (R^\top R_d)\} = \mathrm{diag}(2, 2, 2)$*.* ▲

Some other examples of functions belonging to the family $\mathcal{F}_R$ are:

1. $f = a(R_d^\top R - R^\top R_d), \quad a \in \mathbb{R}$
2. $f = R^\top R_d - R_d^\top R$
3. $f = R R_d^\top - R_d R^\top$
4. $f = R_d R^\top - R R_d^\top$
5. $f = \log\left(R_d^\top R\right)$

6. $f = a \log\left(R^\top R_d\right), \quad a \in \mathbb{R}$

7. $f = \log\left(R R_d^\top\right)$

8. $f = \log\left(R_d R^\top\right)$

9. $f = \log(R) - \log(R_d)$

10. $f = \operatorname{trace}(I_{3\times 3} - R^\top R_d)(R_d^\top R - R^\top R_d)$.

6.4.1 Class $\mathcal{C}_R$ feedback controllers

Let f be any function in $\mathcal{F}_R$. We start by taking the Lie derivative of $f \in \mathcal{F}_R$ along the vector fields of (6.1), (6.3), (6.4). Formally, this yields

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}\,t} f(R, R_d) &= (\mathrm{d}_R\, f)\dot{R} + (\mathrm{d}_{R_d}\, f)\dot{R}_d \\ &= (\mathrm{d}_R\, f)R\widehat{\Omega} + (\mathrm{d}_{R_d}\, f)R_d\widehat{\Omega}_d.\end{aligned} \tag{6.11}$$

Since the control input does not appear we take the second derivative of f,

$$\begin{aligned}\frac{\mathrm{d}^2}{\mathrm{d}\,t^2} f(R, R_d) &= (\mathrm{d}_R(\mathrm{d}_R\, f))\, \dot{R}R\widehat{\Omega} + (\mathrm{d}_R\, f)\, \dot{R}\widehat{\Omega} + (\mathrm{d}_R(\mathrm{d}_{R_d}\, f))\, \dot{R}R_d\widehat{\Omega}_d \\ &+ (\mathrm{d}_{R_d}(\mathrm{d}_R\, f))\, \dot{R}_d R\widehat{\Omega} + (\mathrm{d}_{R_d}(\mathrm{d}_{R_d}\, f))\, \dot{R}_d R_d\widehat{\Omega}_d + (\mathrm{d}_{R_d f})\, \dot{R}_d\widehat{\Omega}_d \\ &+ (\mathrm{d}_R\, f)\, R\dot{\widehat{\Omega}} + (\mathrm{d}_{R_d}\, f)\, R_d\dot{\widehat{\Omega}}_d \\ &= (\mathrm{d}_R(\mathrm{d}_R\, f))\, R\widehat{\Omega}R\widehat{\Omega} + (\mathrm{d}_R\, f)\, R\widehat{\Omega}\widehat{\Omega} + (\mathrm{d}_R(\mathrm{d}_{R_d}\, f))\, R\widehat{\Omega}R_d\widehat{\Omega}_d \\ &+ (\mathrm{d}_{R_d}(\mathrm{d}_R\, f))\, R_d\widehat{\Omega}_d R\widehat{\Omega} + (\mathrm{d}_{R_d}(\mathrm{d}_{R_d}\, f))\, R_d\widehat{\Omega}_d R_d\widehat{\Omega}_d + (\mathrm{d}_{R_d f})\, R_d\widehat{\Omega}_d\widehat{\Omega}_d \\ &+ (\mathrm{d}_R\, f)\, R\widehat{u} + (\mathrm{d}_{R_d}\, f)\, R_d\widehat{u}_d.\end{aligned}$$

By property **P3**, $\mathrm{d}_R f$ is nonsingular when $R = I$. Therefore, by property **P1**, it's invertible in a neighbourhood of $R = I$. Thus the feedback controller

$$\begin{aligned}\widehat{u} := ((\mathrm{d}_R\, f)R)^{-1} \Big\{ &- (\mathrm{d}_{R_d}\, f)\, R_d \widehat{u}_d - (K_1 f^\vee)^\wedge - \left(K_2 \dot{f}^\vee\right)^\wedge \\ &- (\mathrm{d}_R(\mathrm{d}_R\, f))\, R\widehat{\Omega} R\widehat{\Omega} - (\mathrm{d}_R\, f)\, R\widehat{\Omega}\widehat{\Omega} - (\mathrm{d}_R(\mathrm{d}_{R_d}\, f))\, R\widehat{\Omega} R_d \widehat{\Omega}_d \\ &- (\mathrm{d}_{R_d}(\mathrm{d}_R\, f))\, R_d \widehat{\Omega}_d R\widehat{\Omega} - (\mathrm{d}_{R_d}(\mathrm{d}_{R_d}\, f))\, R_d \widehat{\Omega}_d R_d \widehat{\Omega}_d - (\mathrm{d}_{R_d f})\, R_d \widehat{\Omega}_d \widehat{\Omega}_d \Big\}, \end{aligned} \tag{6.12}$$

where $K_1, K_2 \in \mathbb{R}^{3\times 3}$ are Hurwitz, is well-defined in a neighbourhood of $R = I$. We call the control law 6.12 a class $\mathcal{C}_R$ feedback controller. The overall control scheme is represented by the block diagram 6.1.

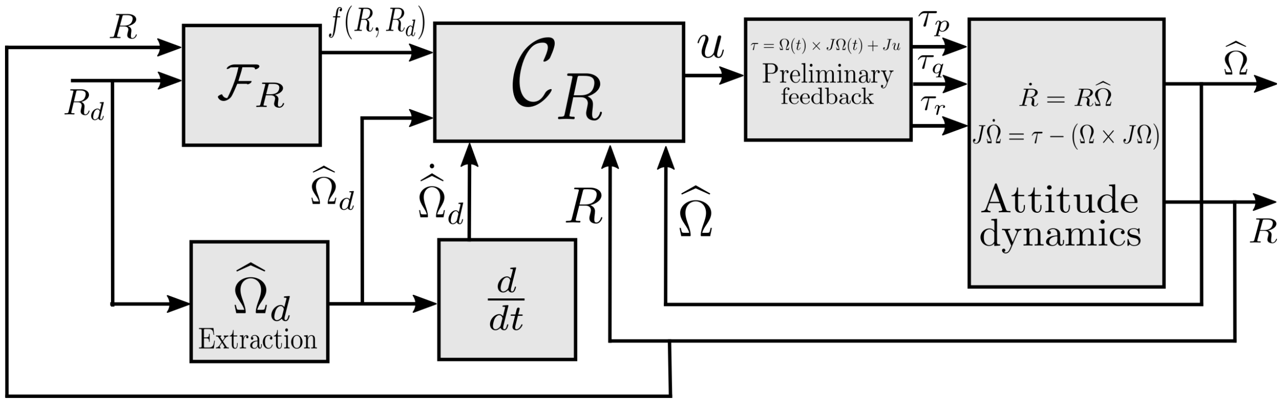


Figure 6.1: Attitude control scheme

Remark 6.4.4. *The controller class $\mathcal{C}_R$ is designed without any local chart. In other words, the controller is designed directly on the manifold* $\mathsf{SO}(3)$*, therefore the controller class $\mathcal{C}_R$ is geometric.*

Now we prove the main results.

Theorem 6.4.5. *Given rigid body dynamics by* (6.1)*, and* (6.3)*, and an exogenous system satisfying* (6.4)*, each controller in the class $\mathcal{C}_R$ asymptotically stabilizes the rigid body.*

Proof. Let $f(R, R_d)$ belong to families of function $\mathcal{F}_R$. The time derivative of $f(R, R_d)$ is given by,

$$\frac{\mathrm{d}}{\mathrm{d}\,t} f(R, R_d) := g = (\mathrm{d}_R\, f) R\widehat{\Omega} + (\mathrm{d}_{R_d}\, f) R_d \widehat{\Omega}_d. \tag{6.13}$$

We pick a real valued positive define Lyapunov function

$$\begin{aligned} V\colon\ \left((\mathsf{SO}(3))^2 \times (\mathfrak{so}(3))^2\right) &\to \mathbb{R} \\ (R, R_d, \widehat{\Omega}, \widehat{\Omega}_d) &\mapsto \frac{1}{2}||g^\vee||_2^2 + \frac{1}{2}||K_1 f^\vee||_2^2. \end{aligned} \tag{6.14}$$

We take the derivative of the Lyapunov function,

$$\begin{aligned} \dot{V} &= (g^\vee)^\top \frac{\mathrm{d}}{\mathrm{d}\,t} g^\vee + (K_1 f^\vee)^\top \frac{\mathrm{d}}{\mathrm{d}\,t} f^\vee \\ &= (g^\vee)^\top \Big\{ (\mathrm{d}_R(\mathrm{d}_R\, f))\, R\widehat{\Omega} R\widehat{\Omega} + (\mathrm{d}_R\, f)\, R\widehat{\Omega}\widehat{\Omega} + (\mathrm{d}_R(\mathrm{d}_{R_d}\, f))\, R\widehat{\Omega} R_d \widehat{\Omega}_d \\ &+ (\mathrm{d}_{R_d}(\mathrm{d}_R\, f))\, R_d\widehat{\Omega}_d R\widehat{\Omega} + (\mathrm{d}_{R_d}(\mathrm{d}_{R_d}\, f))\, R_d\widehat{\Omega}_d R_d\widehat{\Omega}_d + (\mathrm{d}_{R_d f})\, R_d\widehat{\Omega}_d\widehat{\Omega}_d \\ &+ (\mathrm{d}_R\, f)\, R\widehat{u} + (\mathrm{d}_{R_d}\, f)\, R_d\widehat{u}_d \}^\vee + (K_1 f^\vee)^\top (g^\vee)\,. \end{aligned} \tag{6.15}$$

We apply the controller $\mathcal{C}_R$ from (6.12) in the above expression,

$$\begin{aligned} \dot{V} &= (g^\vee)^\top \left\{ -(K_1 f^\vee)^\wedge - \left(K_2 \dot{f}^\vee\right)^\wedge \right\}^\vee + (K_1 f^\vee)^\top (g^\vee) \\ &= (g^\vee)^\top \left\{ -(K_1 f^\vee) - \left(K_2 \dot{f}^\vee\right) \right\} + (K_1 f^\vee)^\top (g^\vee) \\ &= (g^\vee)^\top \{ -(K_1 f^\vee) - (K_2 g^\vee) \} + (K_1 f^\vee)^\top (g^\vee) \\ &= -(g^\vee)^\top (K_1 f^\vee) - (g^\vee)^\top (K_2 g^\vee) + (K_1 f^\vee)^\top (g^\vee) \\ &= -(K_1 f^\vee)^\top (g^\vee) - (g^\vee)^\top (K_2 g^\vee) + (K_1 f^\vee)^\top (g^\vee)\,, \quad \textit{by Proposition A.3.1} \\ &= -||\sqrt{K_2} g^\vee||_2^2, \quad \textit{by Proposition A.3.2}. \end{aligned} \tag{6.16}$$

Hence, by [80, Theorem 4.1] the system is (at least locally) asymptotically stable. ☐

Theorem 6.4.5 solves the attitude tracking problem, and satisfies **T1,T2**, and **T3**.

Example 6.4.6. *The example is the continuation of Example 6.4.2. Following the controller design procedure, we take the second derivative of* (6.9)

$$\begin{aligned}\frac{\mathrm{d}_2}{\mathrm{d}\,t^2} f(R, R_d) = \hat{\dot{\Omega}} - \left\{\frac{\mathrm{d}}{\mathrm{d}\,t}(R^\top R_d)\right\}\widehat{\Omega}_d(R_d^\top R) - (R^\top R_d)\hat{\dot{\Omega}}_d(R_d^\top R)\\ - (R^\top R_d)\widehat{\Omega}_d\left\{\frac{\mathrm{d}}{\mathrm{d}\,t}(R_d^\top R)\right\}\end{aligned}\tag{6.17}$$

Using results from Proposition 6.3.4, the above equation can be written as,

$$\begin{aligned}\frac{\mathrm{d}_2}{\mathrm{d}\,t^2} f(R, R_d) &= \widehat{u} - (R^\top R_d)\hat{\dot{\Omega}}_d(R_d^\top R)\\ &\quad - \left\{(R^\top R_d)\widehat{\Omega}_d - \widehat{\Omega}(R^\top R_d)\right\}\widehat{\Omega}_d(R_d^\top R)\\ &\quad - (R^\top R_d)\widehat{\Omega}_d\left\{(R_d^\top R)\widehat{\Omega} - \widehat{\Omega}_d(R_d^\top R)\right\}.\end{aligned}\tag{6.18}$$

It can be seen that the term multiplied by $\widehat{u}$ is I, therefore by property $\mathbf{P1}$ *it is invertible everywhere. The closed form expression of the controller can be written as,*

$$\begin{aligned}\widehat{u} = (R^\top R_d)\hat{\dot{\Omega}}_d(R_d^\top R) + \left\{(R^\top R_d)\widehat{\Omega}_d - \widehat{\Omega}(R^\top R_d)\right\}\widehat{\Omega}_d(R_d^\top R)\\ + (R^\top R_d)\widehat{\Omega}_d\left\{(R_d^\top R)\widehat{\Omega} - \widehat{\Omega}_d(R_d^\top R)\right\} - (K_1 f^\vee)^\wedge - \left(K_2 \dot{f}^\vee\right)^\wedge.\end{aligned}\tag{6.19}$$

It is easy to see that the controller $\widehat{u} \in \mathcal{C}_R$ is not global, because $f = \log(R_d^\top R)$ is not defined globally. Precisely, $f = \log(R_d^\top R)$ is not defined when $\mathrm{trace}(R_d^\top R) = -1$. On $\mathsf{SO}(3)$ $\mathrm{trace}(R_d^\top R) = -1$ at the following three points,

1. $(R_d^\top R) = \mathrm{diag}(-1, -1, 1)$,
2. $(R_d^\top R) = \mathrm{diag}(-1, 1, -1)$,
3. $(R_d^\top R) = \mathrm{diag}(1, -1, -1)$.

We can consider a set consisting of these three points, and since this set consists of finite elements, its Lebesgue measure is zero. By Theorem 6.4.5, the controller is asymptotically stable everywhere except on this set of Lebesgue measure zero, hence the controller is almost globally asymptotically stable. ▲

Remark 6.4.7. *The controller $\widehat{u}$ given in* (6.19) *is not globally stable. In fact, no continuous time-invariant feedback controller can globally asymptotically stabilize an equilibrium attitude of a rigid body, or globally track a reference attitude because of topological obstructions [34, 83]. The strongest stability or tracking property that can be achieved is almost global asymptotic stability. Informally, almost global asymptotic stability is global asymptotic stability everywhere excluding a "small" set of zero measure. In terms of global asymptotic stability (or region of convergence), the controller* (6.19) *is the best possible controller. Other controllers also belong to the class $\mathcal{C}_R$ that enjoys almost global asymptotic property.*

Remark 6.4.8. *Physically, the condition when* $\mathrm{trace}(R_d^\top R) = -1$ *happens when the desired orientation is furtherest apart from the current orientation. In other words, when the distance metric Λ achieves the maximum value $2\sqrt{2}$. More intuitively,* $\mathrm{trace}(R_d^\top R) = -1$*, when, for example, a rigid body is upside down, and the desired orientation is upright. In local coordinate (e.g., Euler Angles) this condition happens when the desired orientation is π radians apart in either roll, pitch or yaw axis.*

Example 6.4.9. *This example is the continuation of Example 6.4.3. Following the controller design procedure, we take the second derivative of* (6.10)

$$\begin{aligned}\frac{\mathrm{d}_2}{\mathrm{d}\,t^2} f(R, R_d) = &\left\{\frac{\mathrm{d}}{\mathrm{d}\,t}(R_d^\top R)\right\}\widehat{\Omega} + (R_d^\top R)\widehat{\dot{\Omega}} - \widehat{\dot{\Omega}}_d(R_d^\top R) - \widehat{\Omega}_d\left\{\frac{\mathrm{d}}{\mathrm{d}\,t}(R_d^\top R)\right\}\\ &+ \widehat{\dot{\Omega}}(R^\top R_d) + \widehat{\Omega}\left\{\frac{\mathrm{d}}{\mathrm{d}\,t}(R^\top R_d)\right\} - \left\{\frac{\mathrm{d}}{\mathrm{d}\,t}(R^\top R_d)\right\}\widehat{\Omega}_d - (R^\top R_d)\widehat{\dot{\Omega}}_d.\end{aligned} \tag{6.20}$$

Using results from Proposition 6.3.4, the above equation can be written as,

$$\begin{aligned}\frac{\mathrm{d}_2}{\mathrm{d}\,t^2} f(R, R_d) = &\left\{(R_d^\top R)\widehat{\Omega} - \widehat{\Omega}_d(R_d^\top R)\right\}\widehat{\Omega} + (R_d^\top R)\widehat{u}\\ &- \widehat{\dot{\Omega}}_d(R_d^\top R) - \widehat{\Omega}_d\left\{(R_d^\top R)\widehat{\Omega} - \widehat{\Omega}_d(R_d^\top R)\right\}\\ &+ \widehat{u}(R^\top R_d) + \widehat{\Omega}\left\{(R^\top R_d)\widehat{\Omega}_d - \widehat{\Omega}(R^\top R_d)\right\}\\ &- \left\{(R^\top R_d)\widehat{\Omega}_d - \widehat{\Omega}(R^\top R_d)\right\}\widehat{\Omega}_d - (R^\top R_d)\widehat{\dot{\Omega}}_d.\end{aligned} \tag{6.21}$$

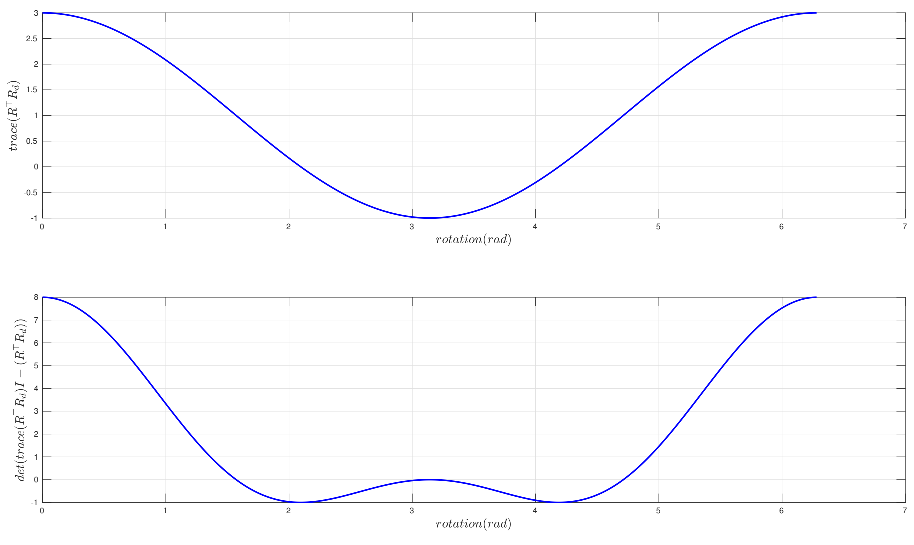


Figure 6.2: $\det(D)$

Combining the terms involving $\widehat{u}$, and using the property 1.1, we can write

$$\begin{aligned}\frac{\mathrm{d}_2}{\mathrm{d}\,t^2} f(R, R_d) &= \left(\left\{\mathrm{trace}(R^\top R_d)I - (R^\top R_d)\right\} u\right)^\wedge - \widehat{\dot{\Omega}}_d(R_d^\top R) - (R^\top R_d)\widehat{\dot{\Omega}}_d \\ &\quad + \left\{(R_d^\top R)\widehat{\Omega} - \widehat{\Omega}_d(R_d^\top R)\right\}\widehat{\Omega} + \widehat{\Omega}\left\{(R^\top R_d)\widehat{\Omega}_d - \widehat{\Omega}(R^\top R_d)\right\} \\ &\quad - \widehat{\Omega}_d\left\{(R_d^\top R)\widehat{\Omega} - \widehat{\Omega}_d(R_d^\top R)\right\} - \left\{(R^\top R_d)\widehat{\Omega}_d - \widehat{\Omega}(R^\top R_d)\right\}\widehat{\Omega}_d.\end{aligned} \tag{6.22}$$

By $\mathbf{P3}$ *the matrix* $\left(\left\{\mathrm{trace}(R^\top R_d)I - (R^\top R_d)\right\}\right)$ *is invertible in some neighborhood of* $R = I$*. Let* $D := \left(\left\{\mathrm{trace}(R^\top R_d)I - (R^\top R_d)\right\}\right)$*. It can be shown that the matrix* D *is not invertible for all* $R, R_d \in \mathsf{SO}(3)$*, moreover, it loses rank whenever the following conditions hold,*

1. $\mathrm{trace}\left(R^\top R_d\right) = 1$*, and*

2. $\mathrm{trace}\left(R^\top R_d\right) = -1$.

The condition when D *loses rank is shown in Figure 6.2. At the start of the simulation* R *is aligned with* R_d*, i.e.,* $R = R_d$*. In other words* $\mathrm{trace}(R^\top R_d) = 3$*, then the rigid body is rotated* *about any arbitrary axis to 2π. As seen in Figure 6.2. The matrix determinant* $\det(D)$ *goes to zero, when* $\operatorname{trace}(R^\top R_d) = 1$*, or* $\operatorname{trace}(R^\top R_d) = -1$*, and this happens when the rotation value is $\pm\pi/2$ or π. Another way to interpret this condition is that, on the distance metric Λ, i.e., D loses rank whenever $\Lambda = 2$ and $\Lambda = 2\sqrt{2}$. This motivates us to pick a neighborhood around R_d such that the distance Λ between each point in the neighborhood and R_d is less than 2, i.e,*

$$U := \left\{(R, R_d) \in \mathsf{SO}(3)^2 : \Lambda < 2\right\}.$$

Using $\mathbf{P1}$ *the closed form expression of the controller can be written form* (6.22)

$$\begin{aligned}\widehat{u} = &\left(\left\{\operatorname{trace}(R^\top R_d)I - (R^\top R_d)\right\}\right)^{-1}\Big[\dot{\widehat{\Omega}}_d(R_d^\top R) + (R^\top R_d)\dot{\widehat{\Omega}}_d \\ &-\left\{(R_d^\top R)\widehat{\Omega} - \widehat{\Omega}_d(R_d^\top R)\right\}\widehat{\Omega} - \widehat{\Omega}\left\{(R^\top R_d)\widehat{\Omega}_d - \widehat{\Omega}(R^\top R_d)\right\} \\ &+\widehat{\Omega}_d\left\{(R_d^\top R)\widehat{\Omega} - \widehat{\Omega}_d(R_d^\top R)\right\} + \left\{(R^\top R_d)\widehat{\Omega}_d - \widehat{\Omega}(R^\top R_d)\right\}\widehat{\Omega}_d\,(K_1 f^\vee)^\wedge - \left(K_2\dot{f}^\vee\right)^\wedge\Big].\end{aligned} \tag{6.23}$$

By Theorem 6.4.5, the controller (6.23) *is asymptotically stable everywhere on U, and is local.*

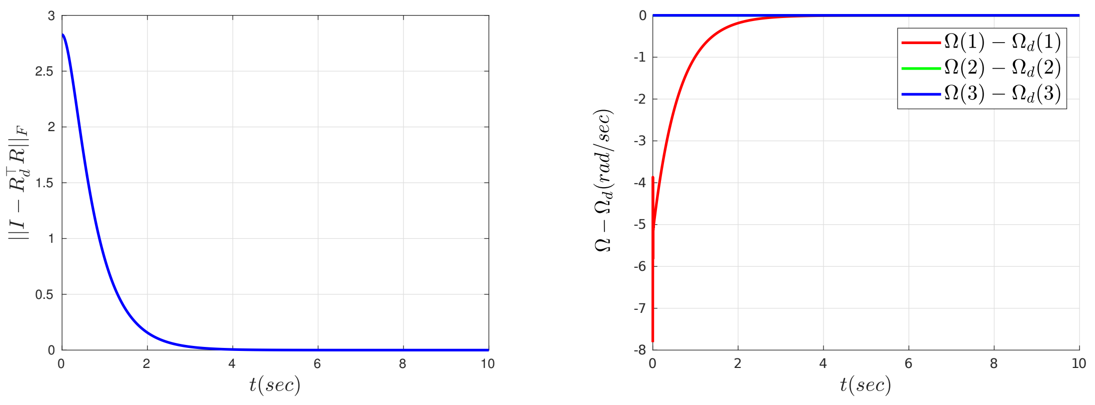
▲

6.5 Almost global controller simulation

In this section we present simulation results of the controller (6.19) presented in Example 6.4.2, and Example 6.4.6.

6.5.1 Stabilization

First we discuss when R_d is not moving with time. Without loss of generality, let $R_d = I$. Starting from an "almost" upside down position, i.e., the initial orientation at $t = 0$ is $R(0) = \exp((\pi - \epsilon)\widehat{e}_1)$, where $|\epsilon| \approx 0 \in \mathbb{R}$ is close to zero but not identically zero. The target is to achieve upright position, i.e., $R = R_d = I$. It can be seen in Figure 6.3a, that at $t = 0$ the error



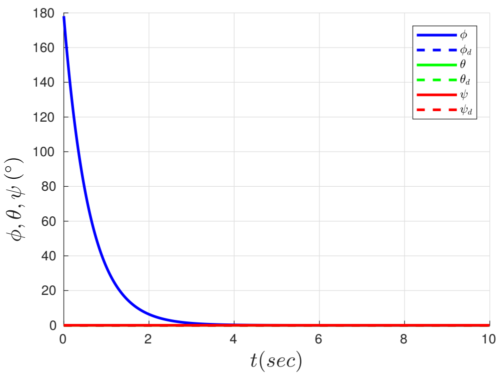
(a) Rotation matrix error converging to zero (b) Body rates error $\Omega - \Omega_d$ converging to zero

Figure 6.3: Attitude errors

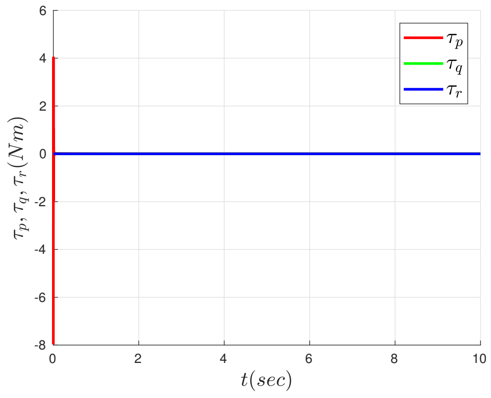
was almost $2\sqrt{2}$, which in other words is the almost upside down position. Starting from this maximum error in attitude the error converges to zero. The body rate errors $\Omega - \Omega_d$ about each body axis also converge to zero, as shown in Figure 6.3b. Finally, we represent the stabilization of rigid body in terms of Euler angles in Figure 6.4a. It should be noted that Euler angles are used only for representation purposes, and are not used for control design. As seen in the figure, at $t = 0$ the roll angle ϕ is almost π, and then converges to zero. Figure 6.4b shows the control effort required to achieve the stabilization task.

Stabilization with noise

Again, we consider the stabilization case, i.e., when the desired rigid body pose R_d is I. We investigate the tracking errors in the presence of noise. We consider that the rigid body is attached with an IMU, and a low level processing unit that gives full attitude information, i.e., both Ω and R, with noise levels used in the simulation based on *AscTec* pelican noise level from Table 4.1. This assumption is quite practical, as the quadrotor platform, *AscTec*, is equipped with such an IMU, and a low level processing unit that is capable of giving full attitude information at an update rate of up to 1 KHz. Figure 6.5a, and 6.5b represent attitude and body rates errors



(a) Euler angles converging to zero



(b) Inputs of the system τ_p, τ_q, τ_r

Figure 6.4: Euler angles and system inputs

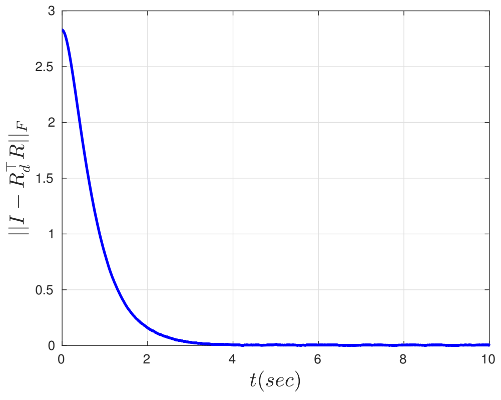
converging to zero.

6.5.2 Sinusoidal signal tracking

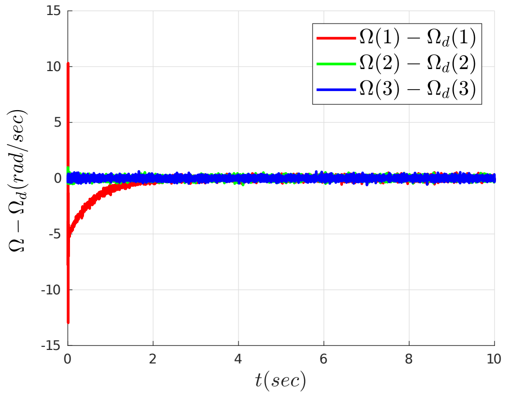
In this section we show simulation results for the tracking case, i.e., when $R_d(t)$ is changing with time. Starting from an initial pose of $R(0) = \exp((\pi/2)\widehat{e}_2)$, the target is to track the desired moving reference attitude

$$R_d(t) = \exp\left((170^\circ(\pi/180)\sin(0.005)t)\widehat{e}_1\right).$$

In local coordinates, the initial condition can be interpreted as a pitch angle of $\pi/2$, and the desired reference attitude R_d can be seen as a 170° sinusoidal movement about roll axis. It can be seen in Figure 6.6a that the tracking errors converge to zero. Figure 6.7a shows the rigid body tracking the sinusoidal signal. Again, the figure shows Euler angles just for the purpose of demonstration, as it is relatively intuitive to visualize the pose of a rigid body in terms of Euler angles. It can be seen that the rigid body crosses the gimbal lock point. This is one of the advantage of this geometric controller: as the controller is designed without selecting a local chart

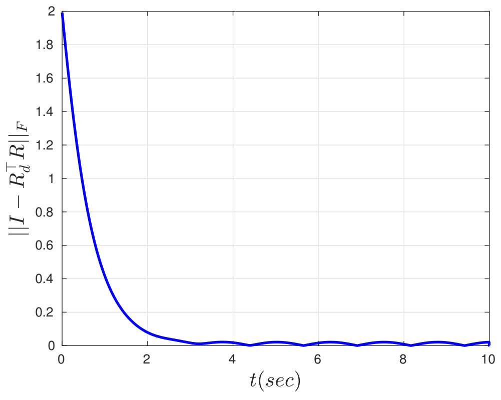


(a) Rotation matrix error converging to zero

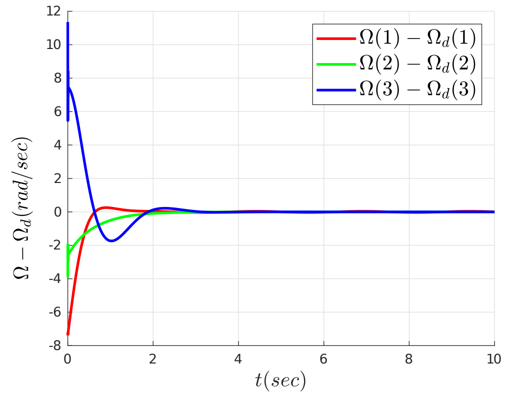


(b) Body rates error $\Omega - \Omega_d$ converging to zero

Figure 6.5: Attitude errors in the presence of noise

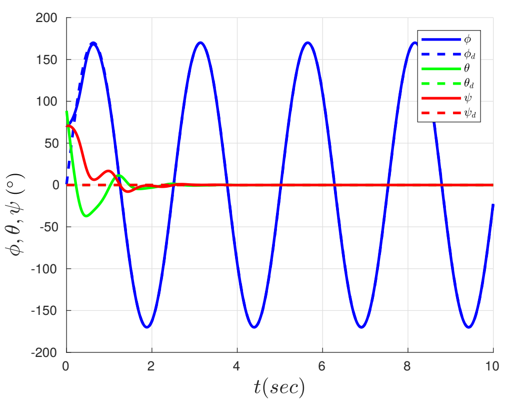


(a) Rotation matrix error converging to zero

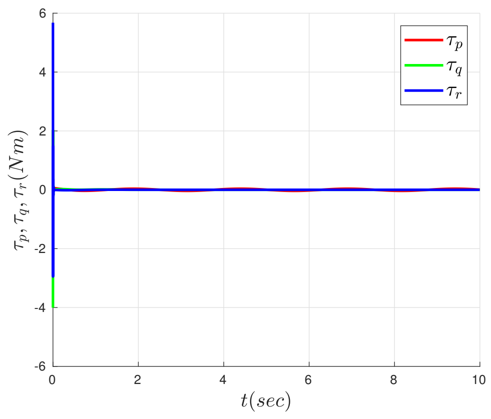


(b) Body rates error $\Omega - \Omega_d$ converging to zero

Figure 6.6: Attitude errors



(a) Euler angles converging to zero



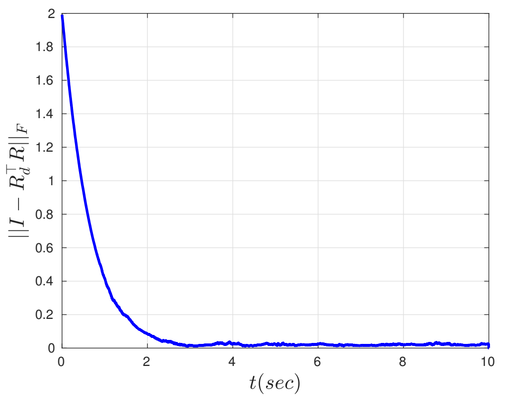
(b) Inputs of the system τ_p, τ_q, τ_r

Figure 6.7: Euler angles and system inputs

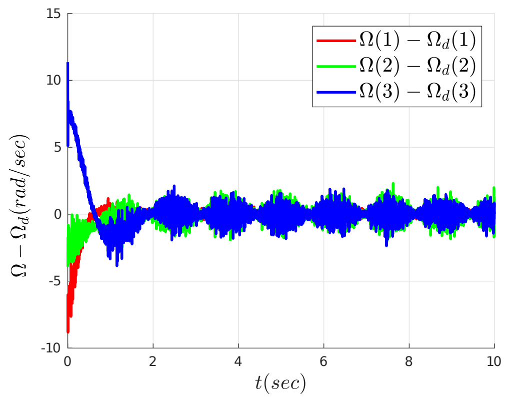
such as Euler angles, singularities such as gimbal lock can be avoided. Figure 6.6b represents all of the body rates converging to zero. Figure 6.7b shows the control signals required to track the given sinusoidal signal. As shown in figure, at $t = 0$ the controller applies body torques up to 6 N.m to follow the desired reference signal.

Sinusoidal tracking with noise

Now we show the simulation results of the controller tracking the same sinusoidal reference attitude $R_d(t) = \exp\left((170^\circ(\pi/180)\sin(0.005)t)\widehat{e}_1\right)$, starting from the same initial attitude $R(0) = \exp((\pi/2)\widehat{e}_2)$, but in the presence of noise. Figure 6.8a, and Figure 6.8b, represent the errors converging to zero. It can be seen that even in the presence of noise the controller tracks the desired attitude signal closely.



(a) Rotation matrix error converging to zero



(b) Body rates error $\Omega - \Omega_d$ converging to zero

Figure 6.8: Attitude errors in the presence of noise

6.5.3 Multiple flips

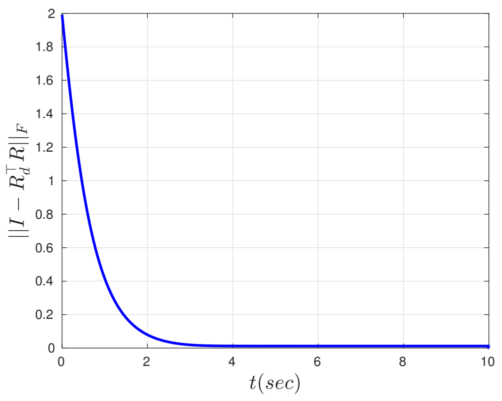
In this section we show simulation results of the rigid body performing multiple flips. Starting from an initial pose of $R(0) = \exp((\pi/2)\widehat{e}_2)$, the target is to track the desired moving reference attitude

$$R_d(t) = \exp\left(0.05t\widehat{e}_1\right).$$

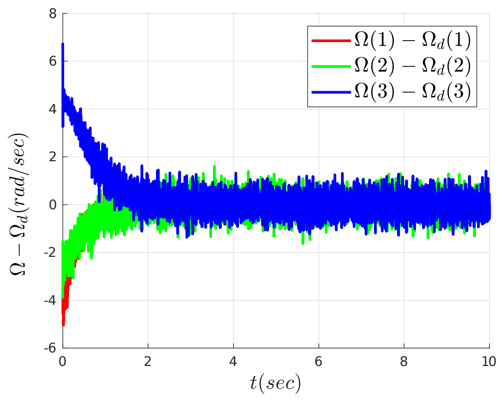
Informally, the controller is capable of performing multiple flips as long as the rigid body is not moving "too fast". Formally this is equivalent to saying that the distance between R and R_d on the metric Λ is less than $2\sqrt{2}$ for all time, i.e., $\Lambda(R(t), R_d(t)) < 2\sqrt{2}$. It can be seen in Figure 6.10 that the rigid body is tracking a time varying signal along the roll axis, and performs more than six flips, and the attitude error converges to zero as shown in Figure 6.9a. The body rate errors are shown in Figure 6.9b.

Multiple flips with noise

Next we show the performance of this controller in the presence of noise. Again starting from the same initial condition $R(0) = \exp((\pi/2)\widehat{e}_2)$, and tracking the same attitude reference $R_d(t) =$



(a) Rotation matrix error converging to zero



(b) Body rates error $\Omega - \Omega_d$ converging to zero

Figure 6.9: Attitude errors

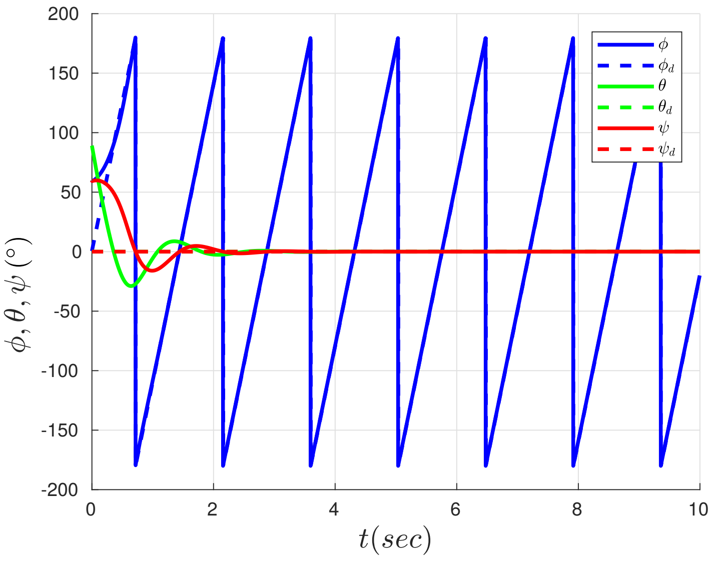
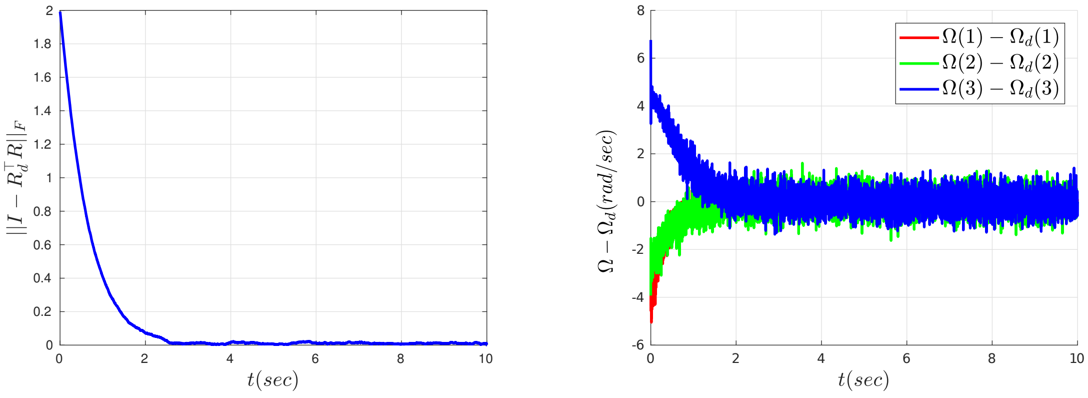


Figure 6.10: Euler angles showing multiple flips

.

$\exp\left(0.05t\widehat{e}_1\right)$ the target is to track the desired moving reference attitude. Figure 6.11a, and Figure 6.11b show the attitude and body rates errors converging to zero in the presence of noise.

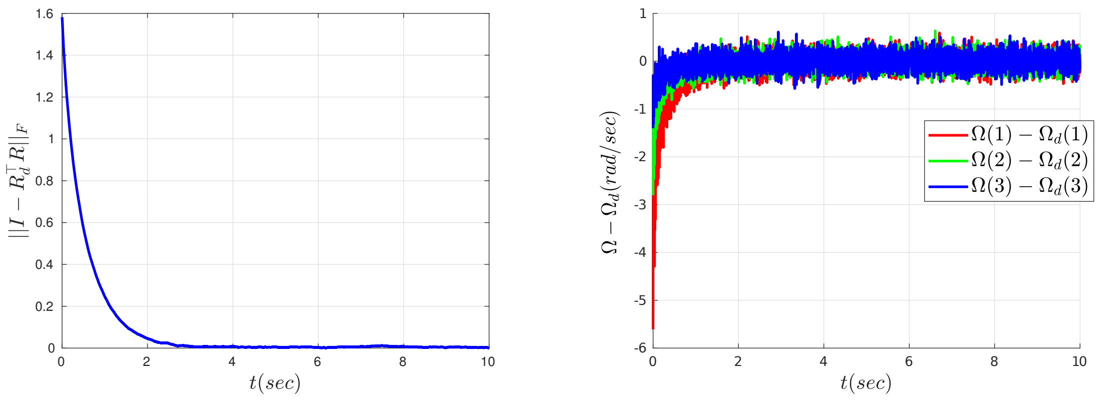


(a) Rotation matrix error converging to zero (b) Body rates error $\Omega - \Omega_d$ converging to zero

Figure 6.11: Attitude errors in the presence of noise

6.6 Local controller simulation

In this section we present simulation results of the local controller 6.23 presented in Example (6.4.3), and Example (6.4.9). We call this controller local because the region of convergence of this controller is in a small neighborhood of the desired point, and not all or almost-all of the state space. Moreover, we compare this local controller with the local controller we designed using Euler angles in Chapter 4. Since the tracking performance of this local geometric controller is similar to the geometric tracking controller except the region of convergence of this controller is smaller, we provide simulation results only with sensor noise for the case of stabilization, and multiple flips to avoid repetition of similar looking figures. Throughout this section we select the same level of sensor noise that we selected in the previous section.



(a) Rotation matrix error converging to zero (b) Body rates error $\Omega - \Omega_d$ converging to zero

Figure 6.12: Attitude errors in the presence of noise

6.6.1 Stabilization with noise

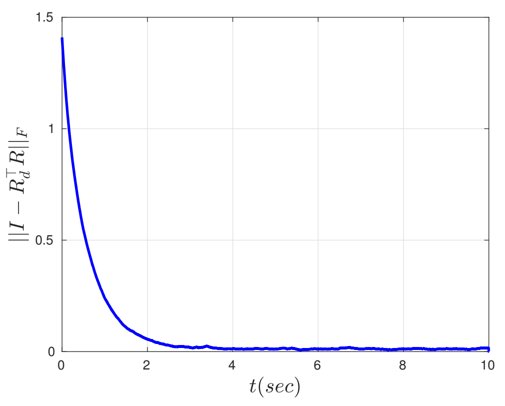
First we discuss when R_d is constant. Again, without loss of generality, let $R_d = I$. The rigid body is initialized at,

$$R(0) = \exp\left(60\frac{\pi}{180}\widehat{e}_1 + 30\frac{\pi}{180}\widehat{e}_2 + 15\frac{\pi}{180}\widehat{e}_3\right).$$

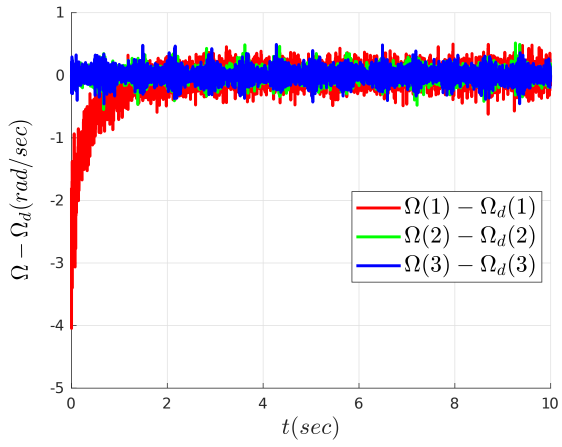
The target is to achieve an upright position, i.e., $R = R_d = I$. It can be seen in Figure 6.12a, that at $t = 0$ the error was almost 1.6 on the metric Λ. It should be noted that for all time the tracking error has to be less than 2, unlike the almost-global case when the tracking error can be less than $2\sqrt{2}$. Figure 6.12b represents the body rate errors converging to zero.

6.6.2 Multiple flips with noise

In this section we show simulation results of the rigid body performing multiple flips in the presence of sensor noise. Although this controller is local, it can perform multiple flips, unlike a local controller designed using a local chart such as Euler angle. Starting from an initial pose of



(a) Rotation matrix error converging to zero



(b) Body rates error $\Omega - \Omega_d$ converging to zero

Figure 6.13: Attitude errors in the presence of noise

$R(0) = \exp(60(\pi/180)\widehat{e}_1)$, the target is to track the desired moving reference attitude

$$R_d(t) = \exp\left(0.05t\widehat{e}_1\right).$$

Similar to the almost global case, the controller is capable of performing multiple flips as far as the rigid body stays "close" to the desired target. The only difference is that in this local case the distance between R and R_d on metric Λ needs to be less than 2 for all time, i.e., $\Lambda(R(t), R_d(t)) < 2$, unlike the almost global case when the distance between R and R_d on metric Λ needs to be less than $2\sqrt{2}$. Similar to the simulation of almost global controller, it can be seen in Figure 6.14 the rigid body is tracking a time varying signal along roll axis, and perform more than six flips, and the convergence of attitude error to zero is shown in Figure 6.13a. The body rate errors are shown in Figure 6.13b.

This simulation demonstrates an important point which is, although this controller is local yet it allows the system to perform flips or multiple flips. However, it is not possible if a smooth controller is designed using a local chart such as Euler angles. This local simulation highlights another important point that the controller class $\mathcal{C}_R$ contains a rich collection of both almost global and local controllers, and depending on an application a designer can pick a suitable con-

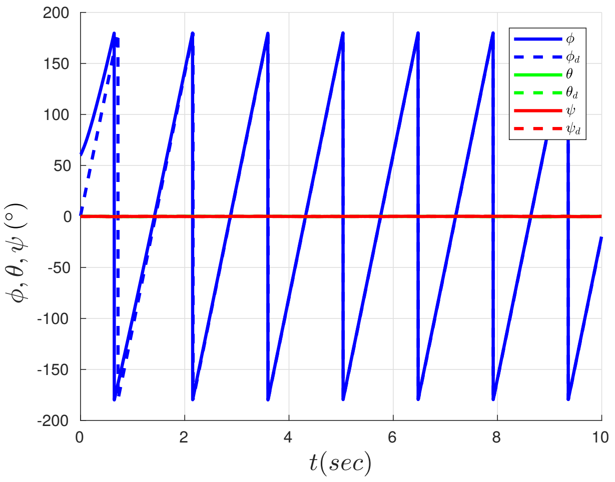


Figure 6.14: Euler angles showing multiple flips

.

troller from this class of controllers. This make our geometric controller class $\mathcal{C}_R$ a broader class compared to the geometric controllers proposed in [45]. The characteristics of each controller form this controller class such as noise sensitivity, and robustness is a future work. In summary, we have solved the attitude tracking problem and show by simulation results in the presence of noise as well that the rigid body satisfies **T1,T2**, and **T3**.

Chapter 7

Application of $\mathcal{C}_R$ Controller Class on $\mathcal{C}_V$ Class of Vehicles

In this chapter we consider the class of vehicles $\mathcal{C}_V$, and design a controller that enables each vehicle belonging to the class $\mathcal{C}_V$ to track a given desired curve. We follow the inner-outer loop control strategy, see [84, 85, 42, 15]. Generally, the inner loop runs at a faster rate, compared to the outer loop in a typical inner-outer loop control framework. In our case, inner loop represents the attitude of the rigid body, and we use the controller class $\mathcal{C}_R$ for attitude control designed in Chapter 6. This chapter focus on outer loop. Specifically, we design a controller for the outer loop, and demonstrate through simulations that it works with the controller class $\mathcal{C}_R$. The overall scheme of the inner-outer loop controller is summarized by Figure 7.1. The outer loop, also called the position loop, assigns a desired attitude R_d to the controller class $\mathcal{C}_R$, and a thrust command τ_f to the translational dynamics. The controller class $\mathcal{C}_R$ assigns body torques to the attitude dynamics.

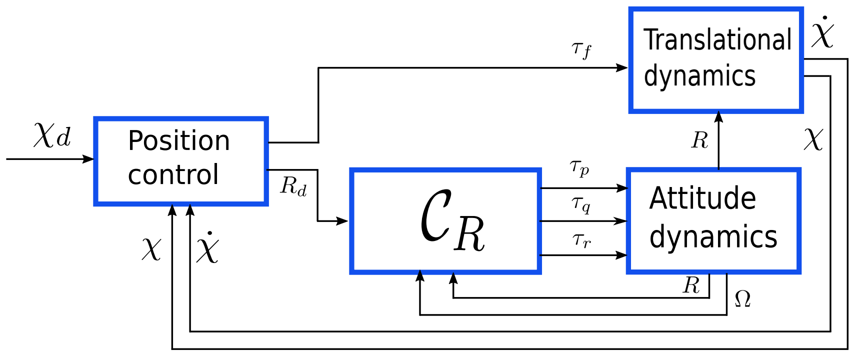


Figure 7.1: Block diagram showing inner-outer loop control scheme

7.1 Tracking position control

Consider the translational dynamics of the rigid body (2.7), repeated here for convenience,

$$\dot{\chi} = v$$
$$\dot{v} = gb_3 - \frac{1}{m}u_t R b_3.$$

We assume that a desired trajectory, parameterized by time, that needs to be followed is given in three dimensional space, such that the desired position, desired velocity, and desired acceleration is given by $\chi_d(t), \dot{\chi}_d(t), \ddot{\chi}_d(t)$, respectively. The time dependency of the desired position, desired velocity, and desired acceleration is removed for the simplification of expressions, whenever obvious from the context. We define $e_\chi := \chi - \chi_d$, and $e_v := v - \dot{\chi}_d$. By taking the time derivative of e_χ,

$$\begin{aligned} \dot{e}_\chi &= \dot{\chi} - \dot{\chi}_d \\ e_v &= v - \dot{\chi}_d. \end{aligned} \tag{7.1}$$

By taking the second derivative of the above expression, we can write,

$$\begin{aligned} \dot{e}_v &= \dot{v} - \ddot{\chi}_d \\ &= gb_3 - \frac{1}{m}u_t R b_3 - \ddot{\chi}_d. \end{aligned} \tag{7.2}$$

Let,

$$T := -\frac{1}{m}u_t R b_3. \tag{7.3}$$

Using (7.2), and (7.3), the translational dynamics (2.7) can be expressed in terms of error coordinates,

$$\begin{aligned} \dot{e}_\chi &= e_v \\ \dot{e}_v &= g b_3 + T - \ddot{\chi}_d. \end{aligned} \tag{7.4}$$

It is easy stabilize the origin of the above system. For example by selecting

$$T = -g b_3 + \ddot{\chi}_d - k_1 e_\chi - k_2 e_v, \tag{7.5}$$

for some positive k_1, and k_2, the system (7.4) takes the form,

$$\begin{aligned} \dot{e}_\chi &= e_v \\ \dot{e}_v &= -k_1 e_\chi - k_2 e_v. \end{aligned} \tag{7.6}$$

It is easy to see that the above system is asymptotically stable.

7.1.1 Thrust and attitude extraction

It should be noted that the control input of the error dynamics (7.4) is $T \in \mathbb{R}^3$. We need to extract the actual control input of the rigid body $u_t \in \mathbb{R}$. We assume that mass of the rigid is m, and thrust produced by the propeller mechanism u_t of each vehicles in $\mathcal{C}_V$ is strictly positive. Consider (7.3),

$$\begin{aligned} T &= -\frac{1}{m} u_t R b_3 \\ \|T\| &= \| -\frac{1}{m} u_t R b_3 \| \\ \| - g b_3 + \ddot{\chi}_d - k_1 e_\chi - k_2 e_v \| &= \frac{1}{m} u_t \|R b_3\|, \qquad m, u_t > 0 \\ \| - g b_3 + \ddot{\chi}_d - k_1 e_\chi - k_2 e_v \| &= \frac{1}{m} u_t, \qquad \|R b_3\| = 1 \\ m \| - g b_3 + \ddot{\chi}_d - k_1 e_\chi - k_2 e_v \| &= u_t. \end{aligned} \tag{7.7}$$

It should be noted that Rb_3 represent the third column of the rotation matrix, which has unit norm by definition. This completes the thrust extraction part.

Next we extract the desired attitude $R_d \in \mathsf{SO}(3)$ from the control definition T. Let $R_d := \operatorname{col}(b_{d_3}, b_{d_1}, b_{d_3}) \in \mathsf{SO}(3)$ be the desired attitude. Intuitively, the z-axis of the desired reference frame (or in other words, the third column vector of R_d) must align with the z-body axis. Another way of saying this is, b_{3_d} must align with the thrust produced by the rigid body. Therefore, we pick

$$b_{3_d} := -\frac{T}{\|T\|} = Rb_3. \tag{7.8}$$

This leaves us to pick two more columns of the desired rotation matrix R_d. There are infinitely many choices to pick the other two columns of the desired rotation matrix. Let $v_d \in \mathbb{R}^3$ be an arbitrary vector of unit length such that $\langle v_d, b_{3_d} \rangle \neq 0$, then we pick

$$b_{1_d} := \frac{T}{\|T\|} \times v_d,$$

and

$$b_{2_d} := -\frac{T}{\|T\|} \times b_{1_d}.$$

By definition $R_d = \operatorname{col} b_{d_1}, b_{d_2}, b_{d_3} \in \mathsf{SO}(3)$. This completes the attitude extraction part.

Remark 7.1.1. *It can be seen that the controller (7.5) asymptotically stabilizes the translational subsystem (2.7), and the controller class $\mathcal{C}_R$, designed in Chapter 6, stabilizes the rotational subsystem (2.3), (2.6). In general, asymptotic stability of each subsystem does not guarantee asymptotic stability of over all (cascade) system, see [86, 20, 85, 18]. However, for the system under study, i.e., rigid body, the rotational dynamics are "faster" compared to translational dynamics, and it has been practically demonstrated by researches, see [15, 16, 87, 52, 72], that given each subsystem is asymptotic stability the overall (cascade) system remains stable under reasonable maneuvers. In other words, as far as the attitude control loop runs at a sufficiently higher rate (about 5 times the speed of the translational loop or more), the over all system practically demonstrate stable behavior.*

Although, asymptotic stability of the overall system, i.e., both translational and rotational subsystem is an interesting theoretical problem but is not very critical for the case of rigid body control. The main reason is the attitude dynamics, and the sensor update rate of attitude loop, is at least five to ten times higher than the translational loop. Most practical systems are attached with an IMU that provides an update rate of 1 KHz. On the other hand, the translation states are either updated by a GPS which provides an update rate of 10 Hz to 50 Hz, or an indoor positioning system with an update rate of 100 Hz to 150 Hz. The faster attitude dynamics provide a control design separation. i.e., one can design control of each subsystem without worrying about the overall stability of the cascade system. We will not discuss the overall stability of the connected system, i.e., stability of the cascade system as cascade control is beyond the scope of this thesis, and is left as a future research topic. Since the attitude dynamics are faster compared to the translational dynamics, one can design any translational or outer loop controller, even a standard PID controller [15], and by selecting any controller from the controller class $\mathcal{C}_R$, one can achieve desired tracking performance. In the next section we design a translational controller using the control design procedure discuss in Chapter 3, 4.

7.2 Path following position control

In the last section the controller (7.5) allows the rigid body to track a trajectory parameterized by time. In this section, instead of tracking a function parameterized by time the goal is to follow a path specified as a function of state variables. Let r_{ij} be the i^{th} row and j^{th} column entry of $R \in \mathsf{SO}(3)$ for $i = \{1, 2, 3\}$, and $j = \{1, 2, 3\}$. The translational subsystem (2.7) can be written as,

$$\ddot{x} = \frac{u_t}{m}(r_{13}), \tag{7.9}$$

$$\ddot{y} = \frac{u_t}{m}(r_{23}), \tag{7.10}$$

$$\ddot{z} = -g + \frac{r_{33}}{m}(u_t). \tag{7.11}$$

We design the position controller in two steps. In the first step we design a path following height controller, and in the second step we design a controller that performs path following in the $x-y$

plane.

7.2.1 Height controller

To control the height dynamics given by (7.11), we stabilize a path as a function of z

$$\begin{aligned} f_z \colon \mathbb{R} &\to \mathbb{R} \\ z &\mapsto f_z(z), \end{aligned} \tag{7.12}$$

such that the function f_z is at least C^1. It should be noted that $f_z(z)$ is a function of state variable z, and not time. We design the controller by following a procedure similar to one used in Chapter 3, Chapter 4, and Chapter 5. By taking the second derivative of f_z the control input u_t appears,

$$\ddot{f}_z = \frac{\partial^2 f_z}{\partial z^2}\dot{z} + \frac{\partial f_z}{\partial z}\left[-g + \frac{r_{33}}{m}u_t\right] \tag{7.13}$$

for $r_{33} \neq 0$, and by selecting

$$u_t = \left[\frac{m}{r_{33}} + g + \frac{1}{\partial f_z/\partial z}\left(\frac{\partial^2 f_z}{\partial z^2}\dot{z} + u_z\right)\right]. \tag{7.14}$$

Let $\xi_1^z := f_z$ and $\xi_2^z := \dot{f}_z$. By following the procedure in [13, 32, 46], by some diffeomorphism the height dynamics can be represented as,

$$\dot{\xi}_1^z = \xi_z^z \tag{7.15}$$

$$\dot{\xi}_2^z = u_z \tag{7.16}$$

The above system is a linear double integrator system, and following proofs similar to Chapter 4, it can proven that the system is exponentially stable by some appropriate choice of u_z.

7.2.2 $x - y$ Position controller

The $x - y$ position dynamics is given by (7.9), and (7.10). We assume that the trust control input u_t is already selected by the height controller stage, and for $x - y$ position stage the control inputs are r_{13}, and r_{23}. To this end designing a tracking controller for this simple subsystem, given by (7.9), and (7.10), is straightforward, however we seek a path following controller. We pick two functions

$$\begin{aligned} \alpha_i \colon \mathbb{R}^2 &\to \mathbb{R} \\ (x,y) &\mapsto \alpha_i(x,y), \end{aligned} \tag{7.17}$$

for $i = \{1, 2\}$. Let the gradient vectors be represented by

$$d_{xy}\alpha_i := \mathrm{col}\left(\frac{\partial \alpha_i}{\partial x}, \frac{\partial \alpha_i}{\partial y}\right),$$

$i = \{1, 2\}$. We assume that $\mathrm{span}\{d_{xy}\alpha_1, d_{x,y}\alpha_2\} = \mathbb{R}^2$. With a slight abuse of notation, (7.9), and (7.10) can be written in the control affine form

$$\dot{\overline{x}} = f(\overline{x}) + g_1(\overline{x})r_{13} + g_2(\overline{x})r_{23}, \tag{7.18}$$

where, $\overline{x} := \mathrm{col}(x, y, v_x, v_y)$, $f(\overline{x}) := \mathrm{col}(v_x, v_y, 0, 0)$, $g_1(\overline{x}) := \mathrm{col}(0, 0, u_t/m, 0)$, and $g_2(\overline{x}) := \mathrm{col}(0, 0, 0, u_t/m)$. By following a reasoning similar to Section 4.4, it is easy to see that control inputs r_{13}, r_{23} appear by taking second derivatives of functions α_i, i.e., $L_{g_j}\alpha_i = 0$, for $i = \{1, 2\}$, and $j = \{1, 2\}$, and the decoupling matrix is given by,

$$D(\overline{x}) = \begin{bmatrix} L_{g_1}L_f\alpha_1 & L_{g_2}L_f\alpha_1 \\ L_{g_1}L_f\alpha_2 & L_{g_2}L_f\alpha_2 \end{bmatrix} = \frac{u_t}{m}\begin{bmatrix} \frac{\partial \alpha_1}{\partial x} & \frac{\partial \alpha_1}{\partial y} \\ \frac{\partial \alpha_2}{\partial x} & \frac{\partial \alpha_2}{\partial y} \end{bmatrix}. \tag{7.19}$$

Direct computations give

$$\det(D(\overline{x}) = \left(\frac{u_t}{m}\right)^2 \left[\left\langle d_{xy}\alpha_1, (d_{xy}\alpha_2)_{-\pi/2}\right\rangle\right], \tag{7.20}$$

where $(d_{xy}\alpha_2)_{-\pi/2}$ represents the vector $d_{xy}\alpha_2$ rotated by $-\pi/2$. Since $\mathrm{span}\{d_{xy}\alpha_1, d_{x,y}\alpha_2\} = \mathbb{R}^2$, this implies that the decoupling matrix (7.19) is non singular whenever $u_t \neq 0$. To this end, a path following controller for the $x - y$ position can be easily designed similar to Section 4.4.

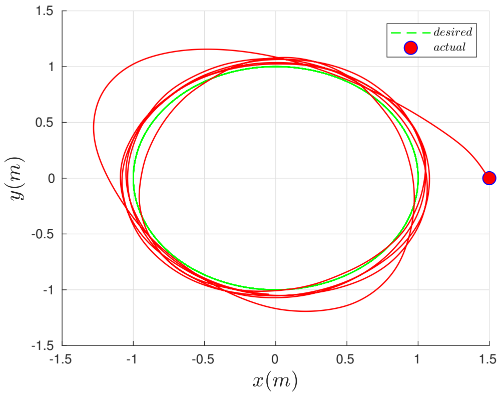
7.3 Simulation results

In this section we present simulation results for each of the outer loop controllers presented in this chapter with the inner loop controller class $\mathcal{C}_R$. In the presence of sensor noise, we present results under two cases: in the first case the outer loop controllers demands the vehicle to move at a normal speed, while in the other case the vehicle is required to perform aggressive maneuvers. For the simulation purposes, we use the global tracking controller from the controller class $\mathcal{C}_R$.

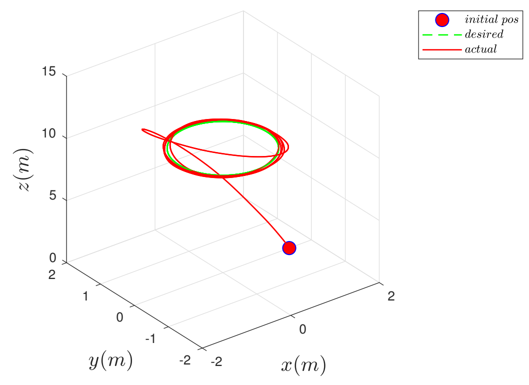
7.3.1 Tracking position control

In the first simulation, the system is required to track a unit circle at a relatively low speed in the presence of practical sensor noise. Roughly, the system is required to traverse the unit circle at a speed of $1.2 m/sec$. Figure 7.2a shows a 2D view of the system tracking the desired unit circle. The desired circle is shown with the green line, and the system's actual trajectory is shown by the red line. The initial position is represented by a solid red dot. The figure shows that when the system is initialized in the neighborhood of the desired trajectory it converges, and tracks it. Figure 7.2b shows the system follows the given trajectory in 3D. The position controllers assigns a desired attitude, and body rates to the inner loop. As seen in Figure 7.3a, and Figure 7.3b, the rotation matrix error, and body rates error converge to zero. Figure 7.4 shows the desired and actual attitude commands (in terms of local coordinates). It is interesting to see that tracking controller assigns an attitude command of $\pm 10^\circ$ to track the circle at the desired slow speed. As seen from the figure the controller belonging to the class $\mathcal{C}_R$ tracks the desired angles with an error of less than a degree.

In the second case, we test the same outer loop tracking controller, on the same trajectory, i.e., unit circle in the presence of noise, but this time it is required to follow the trajectory at a relatively aggressive speed of 3.3 m/sec. As shown in Figure 7.5a, and Figure 7.5b, the system tracks the desired trajectory. Moreover, Figure 7.6a, and Figure 7.6b shows the attitude and body

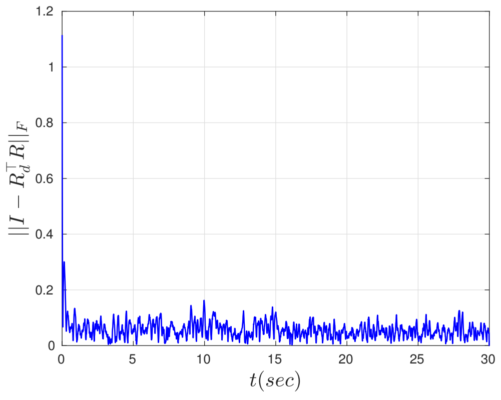


(a) Tracking the desired path, planar view

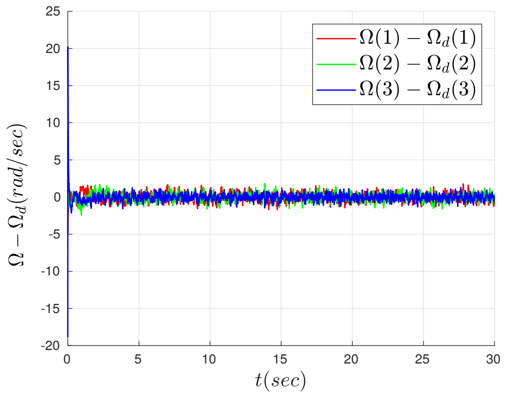


(b) Tracking the desired path, 3d view

Figure 7.2: Translational errors in the presence of noise



(a) Rotation matrix error converging to zero



(b) Body rates error $\Omega - \Omega_d$ converging to zero

Figure 7.3: Attitude errors in the presence of noise

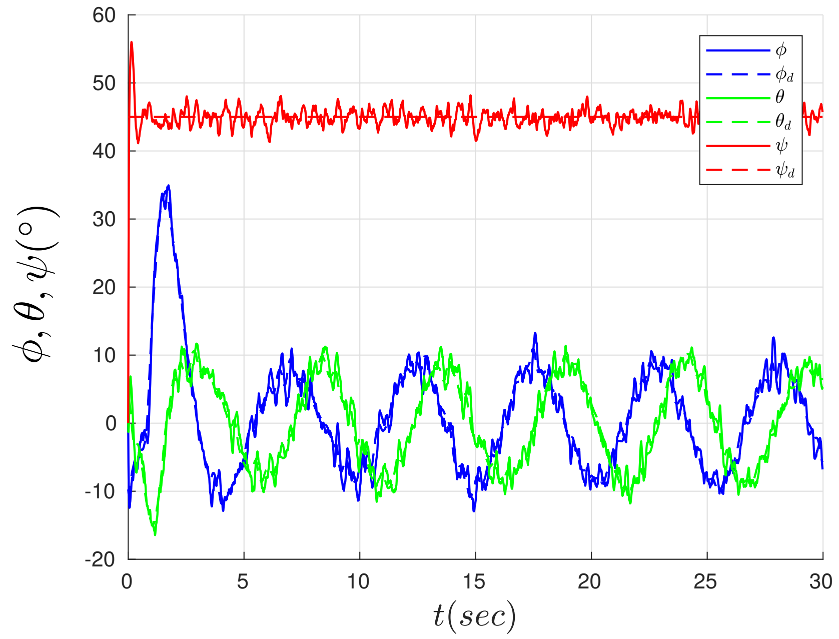
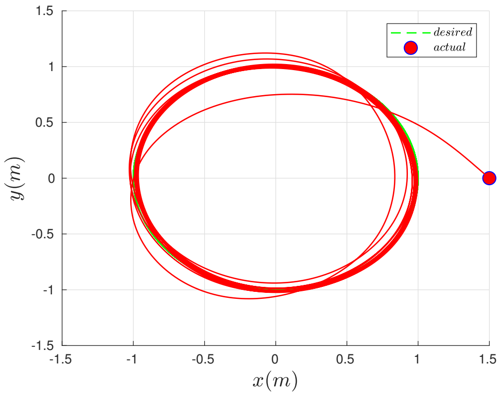


Figure 7.4: Desired and actual attitude, represented in local coordinates

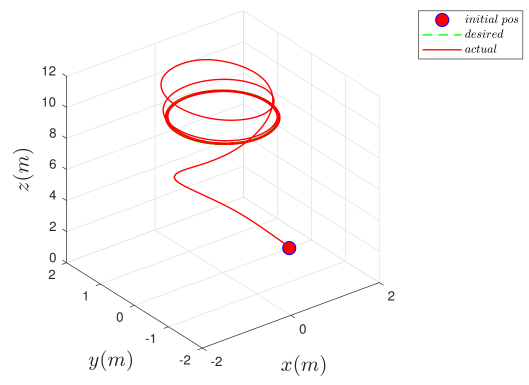
rates error converge to zero. It can be seen in Figure 7.7 that in order to track the unit circle at the desired speed the system outer loop or the tracking controllers assigns larger attitude commands (i.e., in terms of Euler angles about $\pm 40^\circ$). The inner loop controller belonging to the controller class $\mathcal{C}_R$ tracks the desired signal in a satisfactory manner (with an attitude error of few degrees). It should be noted that since the desired attitude angles are large, most of linear controller may fail to perform attitude tracking.

7.3.2 Path following position control

Now we present simulation results for the outer loop path following controller. Similar to the previous case, the outer loop path following controller assigns a desired attitude command to the inner loop attitude tracking controller belonging to the class $\mathcal{C}_R$. In the first case the unit

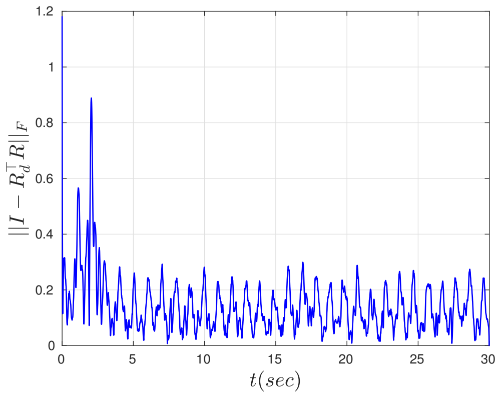


(a) Tracking the desired path, planar view

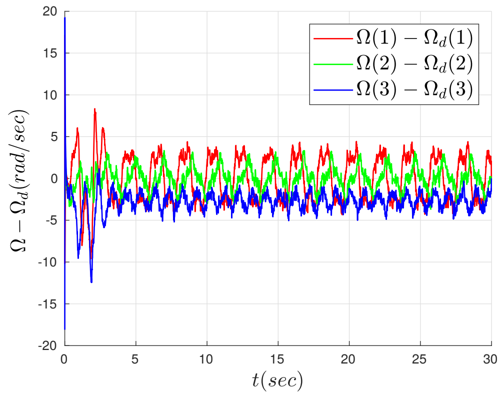


(b) Tracking the desired path, 3d view

Figure 7.5: Translational errors in the presence of noise



(a) Rotation matrix error converging to zero



(b) Body rates error $\Omega - \Omega_d$ converging to zero

Figure 7.6: Attitude errors in the presence of noise

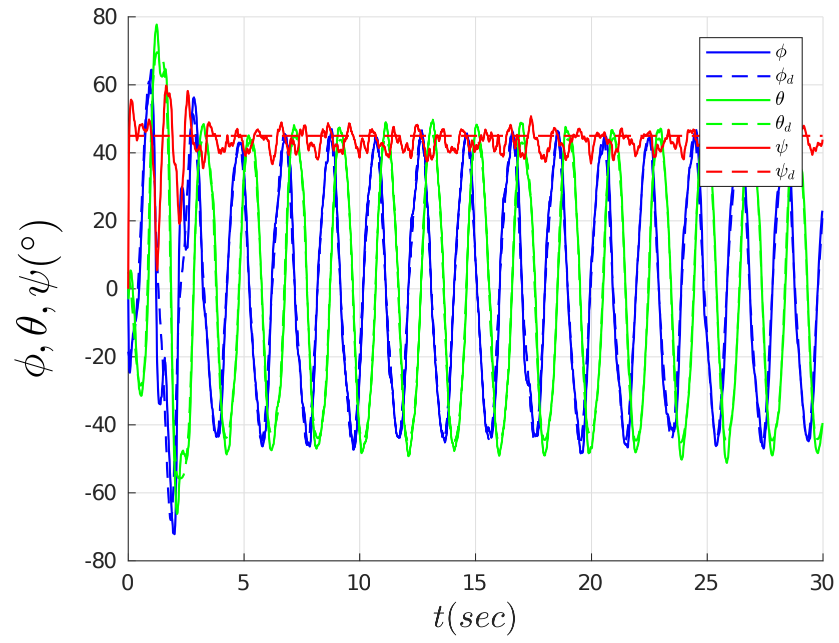
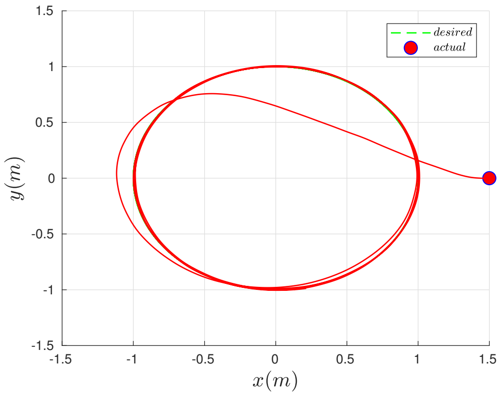


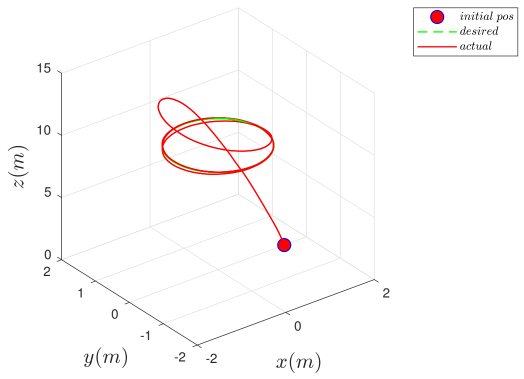
Figure 7.7: Desired and actual attitude, represented in local coordinates

circle is required to follow at a relatively slower speed, about 1.2 m/sec in the presence of sensor noise. Figure 7.8a, and Figure 7.8b shows the system following the desired path. Compared to the tracking case, it can be seen that the controller performance is quite similar at low speeds. Figure 7.9a, and Figure 7.9b show rotation matrix, and body rates errors. Similar to the outer loop tracking case, it can be seen in Figure 7.10 that the outer loop path following controller assigns small desired attitude commands (in terms of Euler angles $\pm 10^\circ$) to the inner loop, and the controller tracks the desired attitude commands within an error of less than a degree.

Now we test the outer loop path following controller at relatively higher speed around (3.3 m/sec). It can be seen from Figure 7.11a, and Figure 7.11b that at higher speed the outer loop path following controller follows the curve "closely" when compared to the tracking controller. The attitude errors and body rate errors are shown in Figure 7.12a, and Figure 7.12b, respectively. It is interesting to note that (see Figure 7.12a) in this case, the attitude error is around

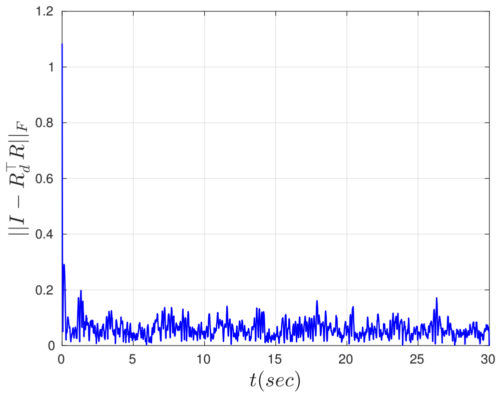


(a) Tracking the desired path, planar view

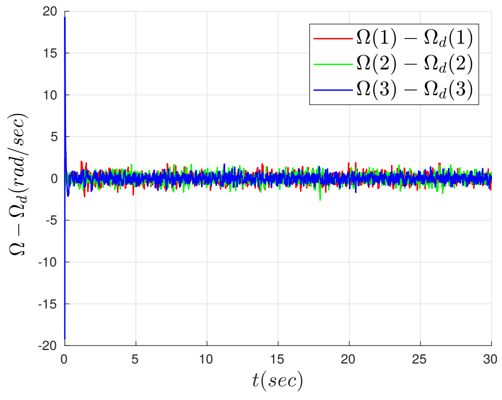


(b) Tracking the desired path, 3d view

Figure 7.8: Translational errors in the presence of noise



(a) Rotation matrix error converging to zero



(b) Body rates error $\Omega - \Omega_d$ converging to zero

Figure 7.9: Attitude errors in the presence of noise

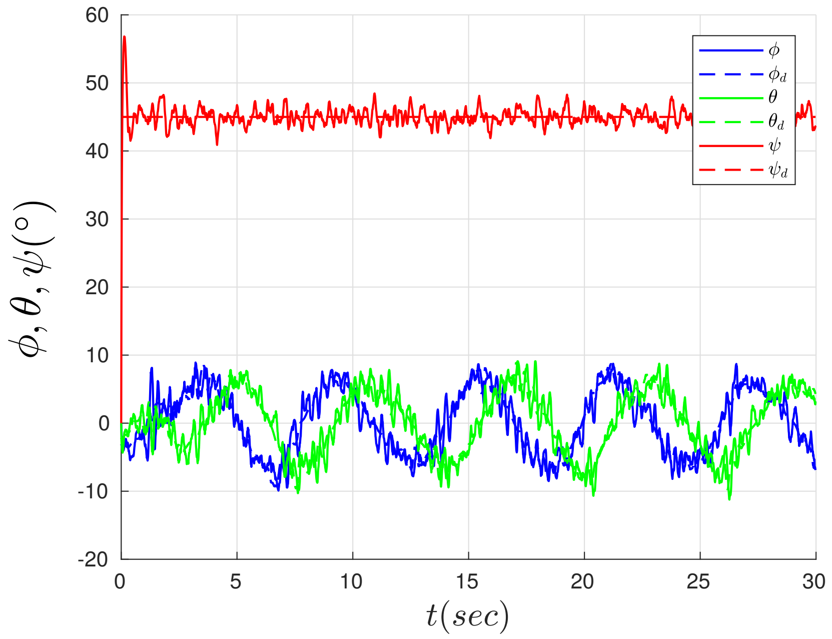
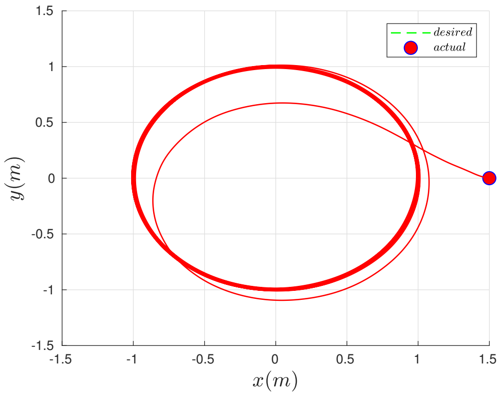


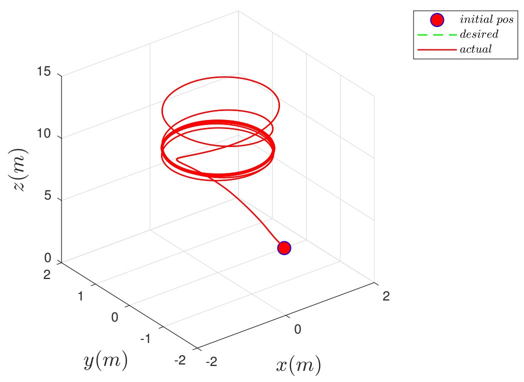
Figure 7.10: Desired and actual attitude, represented in local coordinates

0.1 units at steady state, while in the tracking case (as seen in Figure 7.6a) the attitude error is around 0.2 units. Therefore, in terms of attitude error the outer loop path following controller is better compared to outer loop tracking controller. Similar to the outer loop tracking case at higher speed, it can be seen in Figure 7.13 that the path following outer loop assigns an attitude command of around $\pm 40^\circ$. As shown in the figure the inner loop controllers tracks the desired attitude commands with an attitude error of up-to few degree.

In summary, as shown by the simulation results, the path following outer loop controller performs better compared to the trajectory tracking outer loop controller in terms of attitude tracking error. We show through simulation that, since the attitude loop operates at a higher rate compared to the outer loop, the overall system exhibits stable behavior. Theoretically, the overall stability of the system is a cascade control problem which is beyond the scope of this thesis, and is left as a future direction.

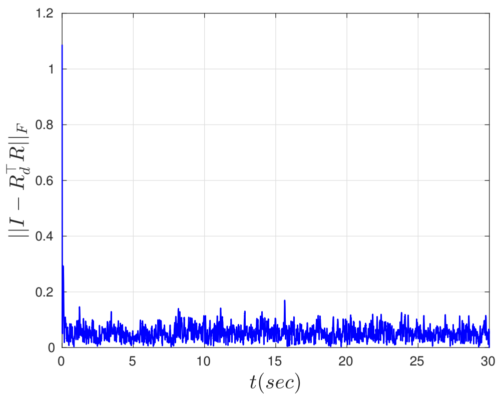


(a) Tracking the desired path, planar view

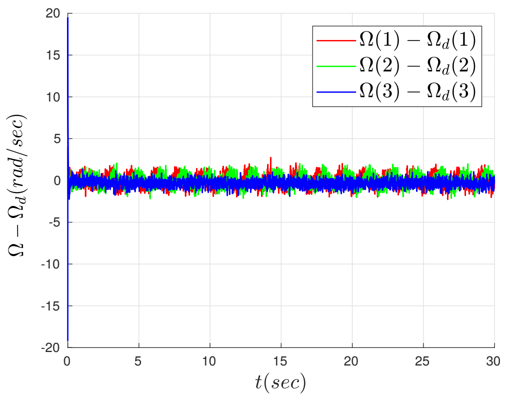


(b) Tracking the desired path, 3d view

Figure 7.11: Translational errors in the presence of noise



(a) Rotation matrix error converging to zero



(b) Body rates error $\Omega - \Omega_d$ converging to zero

Figure 7.12: Attitude errors in the presence of noise

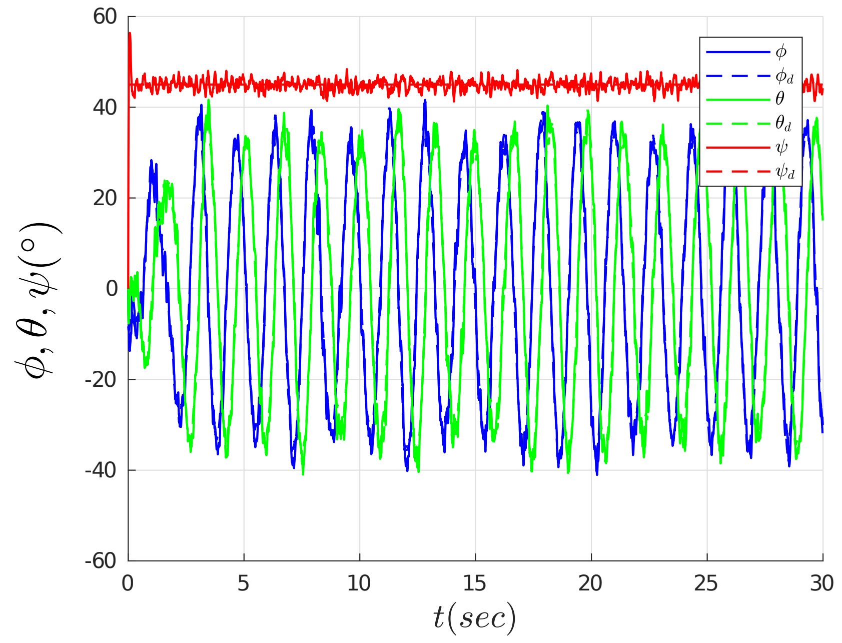


Figure 7.13: Desired and actual attitude, represented in local coordinates

Chapter 8

Conclusion and Future Work

In this chapter we briefly summarize the main highlights of the thesis, and conclude some of the limitations and advantages of the controller design procedure adopted in this thesis. Both geometric and local controllers lead to interesting future directions, and we conclude this chapter on the future work note.

8.1 Conclusion

In this thesis we consider the motion control problem of the $\mathcal{C}_V$ class of vehicles which includes satellites, quadrotors, under water vehicles, and tail sitting vehicles. Informally, the motion control problem is the following: given each vehicle from the class $\mathcal{C}_V$, and given a "curve" in the three dimensional space, the task is to follow the curve using the control inputs. We treat the curve either as a path (parameterized by a path parameter), or as a trajectory (parameterized by time). In this thesis we consider the motion control problem under two settings: a local controller design problem, and a global (or geometric) control problem. We call the motion control problem local when the system's model is represented by some local chart such as Euler angles, and call the motion control problem a geometric control problem when the system dynamics are represented without any local chart, or directly on $\mathsf{SO}(3)$.

Broadly, in this thesis, we solve the motion control problem in two ways: a path following approach, and a trajectory tracking approach. In the path following problem we not only consider the path parameterized by the path parameter, but also treat the problem in a unified setting. In other words, under path following setting, we do not divide the problem into the so-called "inner loop" and "outer loop" approach. A set stabilization approach is used to solve the path following problem for the class of vehicles $\mathcal{C}_V$. Before presenting the path following problem for the class of vehicles, we present, in Chapter 3, how a path following controller is design for a planar mobile robot using the path following and set stabilization approach. This chapter has two purposes: first it explains the controller design process by considering a system with much "simpler" dynamics compared to $\mathcal{C}_V$, and second, it highlights the important fact that although the controller is based on the concept of feedback linearization the controller enjoys precise path following when tested on a real platform. Again, most of the theoretical contributions of this chapter is part of the author's masters work, but the practical implementation is part of the author's doctoral work.

Chapter 4 presents a novel path following controller for $\mathcal{C}_V$. Moreover the controller is designed when the system dynamics are represented by a local chart, i.e., Euler angles. The path following controller is based on the idea on set stabilization which allows path invariance. Informally, path invariance means once the system is on the path it will never leave the path. This controller allows each vehicle belonging to the class of vehicle to follow a broad class of both closed and non-closed curves. Although, this path following controller provides invariance property, it suffers two limitations. The first limitation is gimbal lock, which arises because of the choice of Euler angles as a local chart. The second, practical, limitation is that the control design procedure requires dynamic extension. Dynamic extension requires finding two derivatives of the thrust input, and treating thrust input and rate of change of thrust input as system states. The derivatives of the thrust input (already a noisy signal) are even more noisy which, although proven to be practically working well on a ground robot despite this, makes the controller for $\mathcal{C}_V$ practically less feasible. We believe that practically, with low-noise sensors and careful state observer design, the controller could be used on real $\mathcal{C}_V$ class of vehicles.

Chapter 6 addresses both limitations, i.e., gimbal lock, and sensitivity to noise caused by dynamics extension. Gimbal lock is avoided by treating the geometric version of the problem. Secondly, we adopt an inner-outer loop control design approach which eliminates the need to perform dynamic extension. The heart of the inner-outer loop motion control design problem is the inner loop control, which is the focus of this chapter. To solve the inner loop control problem, we propose a novel family of functions $\mathcal{F}_R$, which induces a novel geometric class of controllers $\mathcal{C}_R$. The controller class $\mathcal{C}_R$ consists of both global and local controllers that stabilize $\mathcal{C}_V$. We show in simulation that this controller class is capable of performing better in the presence of noisy sensor data compared to the controllers that require dynamic extension. Moreover, the geometric nature of the controller class $\mathcal{C}_R$ allows the class of vehicle $\mathcal{C}_V$ to perform multiple flips.

For the class of vehicles $\mathcal{C}_V$ under study, the inner loop or attitude dynamics are “faster” compared to the outer loop or translational dynamics. This faster inner loop dynamics make the so-called separation principle hold, which means one can design an asymptotically stable outer loop controller and use it with an asymptotically stable inner loop controller without direct consideration of overall stability of the full system. This is not strictly true, as if both the inner loop and outer-loop are asymptotically stable, the overall system can be unstable [20]. However, as practically previously demonstrated, such as in [15], under reasonable assumption, i.e., if the inner-loop is running at least two to three times faster than the outer loop, the overall system exhibits stability. In Chapter 7 we design two asymptotically stable outer loop controllers and demonstrate in simulation that these controllers work well with the geometric controller class $\mathcal{C}_R$ even in the presence of sensor noise.

8.2 Future work

In this section we informally discuss some of the future research directions that stem from the work considered in this thesis.

8.2.1 Function family $\mathcal{F}_{R,\Omega}$

In Chapter 6 we propose a family of functions $\mathcal{F}_R$, i.e., a family that depends only on positions R, here we propose another novel family of functions that depends both on positions (R, R_d), and velocities ($\widehat{\Omega}, \widehat{\Omega}_d$), and represent this family by $\mathcal{F}_{R,\Omega}$. In other words $\mathcal{F}_{R,\Omega}$ consists of functions that depend on all the state information, i.e, all positions and velocities. Let $X = (R, R_d, \widehat{\Omega}, \widehat{\Omega}_d)$ denote a point in $(\mathsf{SO}(3))^2. \times (\mathfrak{so}(3))^2$.

Definition 8.2.1. *Let f_Ω be a velocity error, and $f_R \in \mathcal{F}_R$. A function*

$$\begin{aligned} f : U \subseteq (\mathsf{SO}(3))^2 \times (\mathfrak{so}(3))^2 &\to \mathfrak{so}(3) \\ (f_\Omega, f_R) &\mapsto f_\Omega + f_R \end{aligned}$$

is said to belong to $\mathcal{F}_{R,\Omega}$ if there exists an open set $U \subseteq (\mathsf{SO}(3))^2 \times (\mathfrak{so}(3))^2$ containing $(I, I, 0, 0)$ such that

A1 *It is continuously differentiable on U.*

A2 *For all $(R, R_d, 0, \widehat{\Omega}_d) \in U$, the differential f with respect to $\widehat{\Omega}$ is non-singular.*

A3 *f_Ω is a compatible velocity error with f_R.*

Precise definitions of velocity error and compatible velocity errors are part of future work, along with controller design, and stability proof.

Some other examples of functions belonging to the class $\mathcal{F}_{R,\Omega}$ are

1. $f = \widehat{\Omega}_d - \widehat{\Omega} + \log\left(R^\top R_d\right)$

2. $f = \widehat{\Omega} - \mathrm{Adj}_{R_d^\top R}\,\widehat{\Omega}_d + \log\left(R_d^\top R\right)$

3. $f = \widehat{\Omega} - \mathrm{Adj}_{R_d^\top R}\,\widehat{\Omega}_d + R_d^\top R - R^\top R_d$

4. $f = \widehat{\Omega} - \widehat{\Omega}_d + a(R_d^\top R - R^\top R_d), \quad a \in \mathbb{R}$

5. $f = \mathrm{trace}(I_{3\times 3} - R^\top R_d)\left(\Omega - \mathrm{Adj}_{R_d^\top R}\,\widehat{\Omega}_d + \log\left(R_d^\top R\right)\right)$

This problem is theoretically novel and interesting because it is like partial feedback linearization but on manifolds, and this requires careful handling of notions such as vector relative degree, internal dynamics, and zero dynamics.

8.2.2 Stability of the cascade system

The controller class $\mathcal{C}_R$ described in Chapter 6 asymptotically stabilizes the attitude dynamics (inner loop) of the rigid body, while the position controllers presented in Chapter 7 asymptotically stabilize the translational dynamics (outer loop) of $\mathcal{C}_V$. As highlighted in Chapter 7, although both subsystem are asymptotically stable this does not guarantee stability of overall system. This leads to an interesting theoretical cascade control problem, which is proving stability of the overall system when each subsystem is asymptotically stable. The problem can be approached in two ways. The first approach is to use the so called standard techniques of cascade control that involve proving one of the subsystem to be input to state stable, for more details see [20, 88, 21]. The second approach is to solve the attitude control problem as a geometric set stabilization problem, which would be a nontrivial task. In other words, this requires proving the controller class $\mathcal{C}_R$ using set stabilization, and then using reduction theorems, and nested set scheme, as described by [84], to prove stability of the overall system. The main challenge in both cases would be the geometric analysis for the whole controller class. Moreover, in the second case the problem is defined on a non-compact set, while the standard reduction theorems deal with compact sets. This leads to an interesting novel research problem.

8.2.3 Geometric path following

Chapter 4 presents a path invariant controller for the rigid bodies that does not require additional stability tools as the problem was solved in a unified approach. However, as discussed before, one of the limitation of the controller was the gimbal lock which arises because the system's dynamics were represented in local coordinates. One way to avoid gimbal lock is to consider a geometric version of the problem. It is our conjecture that the system still needs dynamics extension, or it would be required to add virtual states to the system. This would require one to check the so called "vector relative degree" of a system defined on a manifold, which is a challenging and nontrivial task at this point. Moreover, rest of the analysis would require geometric tools and may leads to a novel and almost-global path following controller.

8.2.4 Practical implementation

A natural extension of the work considered in this thesis is to implement controllers from each controller class on an actual platform such as a quadrotor, or an underwater vehicle. This task is quite challenging in its own ways, as these controller are novel and have never been tested and implemented on a real platform. This can lead to more interesting practical questions. Moreover, investigating robustness of these controllers, or stability of these controller with a geometric filter would be an interesting and practical research direction.

Appendix A

Basic Concepts and Notations

This Appendix reviews some of the basic concepts used in this proposal document. Some definitions from algebra, analysis and differential geometry are very briefly reviewed that is used periodically in this book. The purpose of this appendix is to give an informal and intuitive review of some of the basic tools used in this proposal. These concepts are taken from [4, 80, 5, 66, 61, 2].

A.1 Review of Algebra, Analysis and Differential Geometry

Informally a map is an operator taking elements from its domain, and generating elements in its co-domain. Let U and V be open subsets of $\mathbb{R}^n$ and $\mathbb{R}^m$ respectively. A function f is sometimes called a mapping, and we say that f maps a domain element $a \in U$ to its codomain element $b \in V$, sometimes called the image of a. In symbols, we might write $f : U \to V$ and $f : a \mapsto b$. Surjective, injective and bijective maps are the basis properties of maps.

Definition A.1.1. *A map $f : U \subseteq \mathbb{R}^n \to V \subseteq \mathbb{R}^m$ is surjective or onto if for each $y \in V$ there exist at least one $x \in U$ such that $f(x) = y$.*

Definition A.1.2. *A map $f : U \subseteq \mathbb{R}^n \to V \subseteq \mathbb{R}^m$ is injective or one-to-one if, $x_1, x_2 \in U, f(x_1) = f(x_2)$ implies $x_1 = x_2$.*

Definition A.1.3. *A map $f : U \subseteq \mathbb{R}^n \to V \subseteq \mathbb{R}^m$ is bijective if it is both injective and surjective.*

Definition A.1.4. *A group G is a set with a binary operation $(.) : G \times G \mapsto G$, such that the following properties are satisfied:*

1. *associativity: $(a.b).c = a.(b.c)$ for all $a, b, c \in G$*
2. *$\exists$ an identity element e such that $e.a = a.e = a$ for all $a \in G$*
3. *$\forall a \in G$ there exists an inverse a^{-1} such that $a.a^{-1} = a^{-1}.a = e$*

Definition A.1.5. *A homomorphism between groups, $\phi : G \mapsto H$, is a map which preserves the group operation*

$$\phi(a.b) = \phi(a).\phi(b).$$

Definition A.1.6. *An isomomorphism is a homomorphism that is bijective.*

Smooth Manifold and Smooth Maps Roughly speaking, manifolds are, locally, vector spaces but are globally curved spaces. For example the surface of a sphere is "locally flat" but globally curved and globally the surface of a sphere is not a vector field. Although manifolds resemble Euclidean spaces near each point ("locally"), the global structure of a manifold may be more complicated. For example, any point on the usual two-dimensional surface of a sphere is surrounded by a circular region that can be flattened to a circular region of the plane, as in a geographical map. However, the sphere differs from the plane.

Let U and V be open subsets of $\mathbb{R}^n$ and $\mathbb{R}^m$ respectively. A mapping $f : U \mapsto V$ is called smooth if f is differentiable and the derivative of the map $\partial f/\partial x$ is continuous. In this case the function f is of class C^1. If f is r^{th} order differentiable and $\partial^r f/\partial x$ is continuous then we say f is of class C^r. If f is smooth for all finite r then we say f is $smooth$ or of class C^∞.

Definition A.1.7. *A map $f : U \subset \mathbb{R}^n \to V \subset \mathbb{R}^m$ is diffeomorphism if f is a homeomorphism (i.e., a one-to-one or injective continuous map with a continuous inverse) and if both f and f^{-1} are smooth.*

Definition A.1.8. *A subset $M \subset \mathbb{R}^k$ is called a smooth manifold of dimension m if for each $x \in M$ there is a neighborhood $W \cap M$, where $W \subset \mathbb{R}^k$, that is a diffeomorphic to an open subset $U \subset \mathbb{R}^m$*

A unit circle $\mathbb{S}^1 \subset \mathbb{R}^2$ defined by $\{(\cos\theta, \sin\theta)\}, \theta \in [0, 2\pi]$ is an example of a manifold. A submanifold is simply a smaller manifold inside a larger manifold.

Definition A.1.9. *A manifold M is said to be an invariant manifold if whenever $y \in M$ and $t_0 \geq 0$, we have*

$$\phi(t, y, t_0) \in M,$$

Theorem A.1.10. *(Inverse Function Theorem [66]) Let U be an open subset of $\mathbb{R}^n$ and $f : U \to \mathbb{R}^n$, a C^∞ mapping. If the Jacobian, $df_{x^\star}$, is nonsingular at some $x^\star$ in U, then there exists an open neighborhood V of $x^\star$ in U such that $W = f(U)$ is open in $\mathbb{R}^n$ and $f|_V$ is a diffeomorphism onto W.*

Regular Values Let $f : M \to N$ be a smooth map between manifolds of same dimensions. A point $x \in M$ is said to be a regular point of f if the derivative Df_x is nonsingular. If x is a regular point it follows from the inverse function theorem that f maps a neighborhood of x diffeomorphically onto a neighborhood of $y = f(x)$.

Definition A.1.11. *Given a map $f : M \to N$, $y \in N$ is said to be a regular value if every point in the set $f^{-1}(y)$ is a regular point.*

A.1.1 Vector fields and their Derivatives

A vector field is an assignment of a vector to each point in a subset of Euclidean space. A vector field in the plane for instance can be visualized as an arrow, with a given magnitude and direction, attached to each point in the plane. Vector fields are often used to model speed and direction of a moving objects throughout space, for example speed and direction of a mobile robot. The following demonstrates the notion of vector field,

$$f(x) = \begin{bmatrix} x_3^2 \\ x_2 \\ 1 + x_1^2 \end{bmatrix} = \frac{\partial x_3^2}{\partial x_1} + \frac{\partial x_2}{\partial x_2} + \frac{\partial(1 + x_1^2)}{\partial x_3}. \tag{A.1}$$

The Lie derivative also called the direction derivative evaluates the change of a vector field along the flow of another vector field. This change is coordinate invariant and therefore the Lie derivative is defined on any differentiable manifold.

Definition A.1.12. *Consider a vector field f and a real valued function,*

$$\lambda : U \subseteq \mathbb{R}^n \to \mathbb{R}, \tag{A.2}$$

the derivative of λ along f is a function $L_f\lambda : U \to \mathbb{R}$ defined as

$$L_f\lambda(x) := \langle d\lambda(x), f(x)\rangle = \frac{\partial \lambda}{\partial x} f(x) = \sum_{i=1}^{n} \frac{\partial \lambda}{\partial x_i} f_i(x), \tag{A.3}$$

which is also called the Lie derivative or directional derivative of λ along f, where $\langle ., .\rangle$ is the Euclidean inner product.

Repeated use of this operator is possible and the following notation can be used,

$$L_g L_f \lambda(x) := \frac{\partial L_f \lambda(x)}{\partial x} g(x) = \sum_{i=1}^{n} \frac{\partial L_f \lambda}{\partial x_i} g_i(x). \tag{A.4}$$

The operation can be recursively defined, such that taking the k derivatives of λ along f would be denoted by L_f^k where,

$$L_f^k \lambda(x) := \frac{\partial L_f^{k-1} \lambda(x)}{\partial x} f(x) = \sum_{i=1}^{n} \frac{\partial L_f^{k-1} \lambda}{\partial x_i} f_i(x). \tag{A.5}$$

Consider a parameterized curve $\sigma : \mathbb{R} \to \mathbb{R}^n$. It is clear that $\sigma(\lambda)$ represents location of a moving point along the curve. The velocity vector of σ at point λ can be represented by $\sigma'(\lambda)$. The speed at λ is the length $\|\sigma'(\lambda)\|$.

Definition A.1.13. *The parameterization $\sigma(\lambda)$ is unit-speed if $\|\sigma'(\lambda)\| = 1$.*

Definition A.1.14. *The curve $\sigma(\lambda)$ is regular if $\sigma'(\lambda) \neq 0$ for all $\lambda \in \mathbb{R}$.*

Consider a curve $\sigma(\lambda) = (a\lambda \cos \lambda, a\lambda \sin \lambda)$. It has velocity

$$\sigma'(\lambda) = a(\cos \lambda - \lambda \sin \lambda, \sin \lambda + \lambda \cos \lambda),$$

and speed

$$\|\sigma'(\lambda)\| = |a| \sqrt{(\cos \lambda - \lambda \sin \lambda)^2 + (\sin \lambda + \lambda \cos \lambda)^2} = |a| \sqrt{1 + \lambda^2} \neq 0.$$

Therefore the parameterization is regular.

A.2 Nonlinear Control Systems

Consider a time-invariant, finite-dimensional, deterministic control-affine system with m inputs, $u := [u_1 \cdots u_m]^\top \in \mathbb{R}^m$ and p outputs and $f : \mathbb{R}^n \to \mathbb{R}^n$, $g_i : \mathbb{R}^n \to \mathbb{R}^n$ and $h : \mathbb{R}^n \to \mathbb{R}^p$ are smooth C^r maps.

$$\dot{x} = f(x) + \sum_{i=1}^{m} g_i(x) u_i := f(x) + g(x)u, \tag{A.6}$$

$$\tag{A.7}$$

and consider a function,

$$y = h(x) = \begin{bmatrix} h_1(x) \\ \vdots \\ h_p(x) \end{bmatrix}, \forall y \in \mathbb{R}^p, \tag{A.8}$$

which is the output of the system. The relative degree is the key concept in solving feedback linearization problems.

Definition A.2.1. *Consider system* (A.6) *with* $u \in \mathbb{R}$ *and with output function* (A.8) *with* $m = p = 1$ *i.e.,* $y = h(x)$, $y \in \mathbb{R}$. *The system has a relative degree of* r *at a point* x_0 *if*

1. $L_g L_f^k h(x) = 0, \forall x \in$ *a neighborhood of* x_0 *and* $\forall k < r - 1$,
2. $L_g L_f^{r-1} h(x_0) \neq 0$.

The relative degree of a single input single output (SISO) system is the number of times we need to differentiate the output before the control input appears. A notion, called the vector relative degree can be defined for the multiple-input multiple-output (MIMO) systems.

Definition A.2.2. *Consider system* A.6 *with* $m = p$. *We define an* $m \times m$ *matrix,*

$$A(x) := \begin{bmatrix} L_{g_1} L_f^{r_1 - 1} h_1(x) & \cdots & L_{g_m} L_f^{r_1 - 1} h_1(x) \\ L_{g_1} L_f^{r_2 - 1} h_2(x) & \cdots & L_{g_m} L_f^{r_2 - 1} h_2(x) \\ \vdots & \ddots & \vdots \\ L_{g_1} L_f^{r_m - 1} h_m(x) & \cdots & L_{g_m} L_f^{r_m - 1} h_m(x) \end{bmatrix}. \tag{A.9}$$

The system has a vector relative degree of $\{r_1, \ldots r_m\}$ *at a point* x_0 *if*

1. $L_{g_j} L_f^k h_i(x) = 0, \forall 1 \leq j \leq m$ *for all* $k < r_i - 1$ *for all* $1 \leq i \leq m$ *and for all* x *in a neighborhood of* x_0.
2. *The matrix* $A(x)$ *is nonsingular at* $x = x_0$,

A.3 Elementary results

In this section we present some elementary results. Although, these results are basic and easy to prove, yet we find them valuable for writing proofs of some of the main results of this thesis.

Proposition A.3.1.

Let $k_1, k_2, k_3 \in \mathbb{R}^+$, and $v, w \in \mathbb{R}^3$, and $K = \mathrm{diag}(k_1, k_2, k_3)$, then

$$v^\top(Kw) = (Kw)^\top v$$

Proof.

$$\begin{aligned} v^\top(Kw) &= \langle v, Kw\rangle \\ &= \langle Kw, v\rangle \\ &= (Kw)^\top v. \end{aligned}$$

□

Proposition A.3.2. *If $k_1, k_2, k_3 \in \mathbb{R}^+$ and $v \in \mathbb{R}^3$, then*

$$v^\top(\mathrm{diag}(k_1, k_2, k_3)v) = \left\|\mathrm{diag}\left(\sqrt{k_1}, \sqrt{k_2}, \sqrt{k_3}\right) v\right\|_2^2 \tag{A.10}$$

Proof. let $K := \operatorname{diag}(k_1, k_2, k_3)$, by expanding the left hand side of (A.10)

$$\begin{aligned}
v^\top(Kv) &= \begin{bmatrix} v_1 & v_2 & v_3 \end{bmatrix} \begin{bmatrix} k_1 & 0 & 0 \\ 0 & k_2 & 0 \\ 0 & 0 & k_3 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \\
&= \begin{bmatrix} v_1 & v_2 & v_3 \end{bmatrix} \begin{bmatrix} k_1 v_1 \\ k_2 v_2 \\ k_3 v_3 \end{bmatrix} \\
&= k_1 v_1^2 + k_2 v_2^2 + k_3 v_3^2 \\
&= \left\| \begin{bmatrix} \sqrt{k_1} & 0 & 0 \\ 0 & \sqrt{k_2} & 0 \\ 0 & 0 & \sqrt{k_3} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \right\|_2^2 \\
&= \left\| \sqrt{K} v \right\|_2^2 \\
&= \left\| \operatorname{diag}\left(\sqrt{k_1}, \sqrt{k_2}, \sqrt{k_3}\right) v \right\|_2^2
\end{aligned}$$

which proves the result. □

Bibliography

[1] A. Roza, "Motion control of rigid bodies in SE(3)," Master's thesis, University of Toronto, 2012.

[2] C. Nielsen, "Maneuver regulation, transverse feedback linearization and zero dynamics," Master's thesis, Department of Electrical and Computer Engineering University of Toronto, 2004.

[3] M. I. El-Hawwary and M. Maggiore, "Global path following for the unicycle and other results," in *American Control Conference (ACC)*, June 2008, pp. 3500 –3505.

[4] A. Isidori, *Nonlinear Control Systems*. Secaucus, NJ, U.S.A: Springer-Verlag New York, Inc., 1995.

[5] S. Sastry, *Nonlinear systems: analysis, stability, and control*. New York: Springer, 1999.

[6] R. Skjetne, T. I. Fossen, and P. V. Kokotovié, "Robust output maneuvering for a class of nonlinear systems," *Automatica*, vol. 40, no. 3, pp. 373 – 383, 2004.

[7] R. M. Murray, M. Rathinam, and W. Sluis, "Differential flatness of mechanical control systems: A catalog of prototype systems," in *Proceedings of the ASME International Congress and Exposition*, 1995.

[8] A. Banaszuk and J. Hauser, "Feedback linearization of transverse dynamics for periodic orbits," *Systems and Control Letters*, vol. 26, no. 2, pp. 95 – 105, 1995.

[9] C. Nielsen and M. Maggiore, “Maneuver regulation via transverse feedback linearization: Theory and examples,” *Symposium on Nonlinear Control Systems (NOLCOS)*, September 2004.

[10] C. Nielsen, C. Fulford, and M. Maggiore, “Path following using transverse feedback linearization: Application to a maglev positioning system,” *Automatica*, vol. 46, no. 3, pp. 585–590, March 2010.

[11] L. Consolini, M. Maggiore, C. Nielsen, and M. Tosques, “Path following for the pvtol aircraft,” *Automatica*, vol. 46, pp. 1284–1296, August 2010.

[12] A. Hladio, C. Nielsen, and D. Wang, “Path following for mechanical systems: Experiments and examples,” in *American Control Conference (ACC)*, June 2011.

[13] A. Akhtar and C. Nielsen, “Path following for a car-like robot using transverse feedback linearization and tangential dynamic extension,” in *50th IEEE Conference on Decision and Control and European Control Conference, (CDC)-ECC*, Dec. 2011, pp. 7949 –7979.

[14] J. How, B. Bethke, A. Frank, D. Dale, and J. Vian, “Real-time indoor autonomous vehicle test environment,” *IEEE Control Systems*, vol. 28, no. 2, pp. 51–64, April 2008.

[15] G. M. Hoffmann, H. Huang, S. L. Waslander, and C. J. Tomlin, “Precision flight control for a multi-vehicle quadrotor helicopter testbed,” *Control Engineering Practice*, vol. 19, no. 9, pp. 1023–1036, September 2011.

[16] R. Mahony, V. Kumar, and P. Corke, “Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor,” *IEEE Robotics Automation Magazine*, vol. 19, no. 3, pp. 20–32, Sept 2012.

[17] S. Bouabdallah and R. Siegwart, “Full control of a quadrotor,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2007, pp. 153–158.

[18] E. Panteley and A. Loria, “Global uniform asymptotic stability of cascaded non-autonomous nonlinear systems,” in *European Control Conference (ECC)*, July 1997, pp. 973–978.

[19] R. Naldi, M. Furci, R. G. Sanfelice, and L. Marconi, “Global trajectory tracking for under-actuated vtol aerial vehicles using a cascade control paradigm,” in *52nd IEEE Conference on Decision and Control (CDC)*, Dec 2013, pp. 4212–4217.

[20] V. Sundarapandian, “Global asymptotic stability of nonlinear cascade systems,” *Applied Mathematics Letters*, vol. 15, no. 3, pp. 275 – 277, 2002.

[21] A. Chaillet, “On stability and robustness of nonlinear cascaded systems - Application to mechanical systems,” Theses, Université Paris Sud - Paris XI, Jul. 2006.

[22] H. Voos, “Nonlinear control of a quadrotor micro-UAV using feedback-linearization,” in *IEEE International Conference on Mechatronics*, April 2009, pp. 1–6.

[23] T. Madani and A. Benallegue, “Control of a quadrotor mini-helicopter via full state back-stepping technique,” in *45th IEEE Conference on Decision and Control (CDC)*, Dec. 2006, pp. 1515–1520.

[24] Y. Fan, Y. Cao, and Y. Zhao, “Sliding mode control for nonlinear trajectory tracking of a quadrotor,” in *36th Chinese Control Conference (CCC)*, July 2017, pp. 6676–6680.

[25] R. Xu and U. Ozguner, “Sliding mode control of a quadrotor helicopter,” in *45th IEEE Conference on Decision and Control (CDC)*, Dec. 2006, pp. 4957–4962.

[26] A. Mokhtari, A. Benallegue, and B. Daachi, “Robust feedback linearization and **GH**-∞ controller for a quadrotor unmanned aerial vehicle,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, August 2005, pp. 1198–1203.

[27] D. Lee, H. Jin Kim, and S. Sastry, "Feedback linearization vs. adaptive sliding mode control for a quadrotor helicopter," *International Journal of Control, Automation and Systems*, vol. 7, no. 3, pp. 419–428, 2009.

[28] V. Utkin, *Sliding Mode Control in Electro-Mechanical System.* CRC Press., 1999.

[29] J.-L. Chang, "Dynamic sliding mode controller design for reducing chattering," *Journal of the Chinese Institute of Engineers*, vol. 37, no. 1, pp. 71–78, 2014.

[30] S. Kumar and R. Gill, "Path following for quadrotors," in *IEEE Conference on Control Technology and Applications (CCTA)*, Aug 2017, pp. 2075–2081.

[31] A. Roza and M. Maggiore, "Path following controller for a quadrotor helicopter," in *American Control Conference (ACC)*, June 2012.

[32] A. Akhtar, S. L. Waslander, and C. Nielsen, "Path following for a quadrotor using dynamic extension and transverse feedback linearization," in *51st IEEE Conference on Decision and Control (CDC)*, Dec. 2012, pp. 3551 –3556.

[33] Z. Liu, L. Ciarletta, C. Yuan, Y. Zhang, and D. Theilliol, "Path following control of unmanned quadrotor helicopter with obstacle avoidance capability," in *International Conference on Unmanned Aircraft Systems (ICUAS)*, June 2017, pp. 304–309.

[34] N. A. Chaturvedi, A. K. Sanyal, and N. H. McClamroch, "Rigid-body attitude control," *IEEE Control Systems*, vol. 31, no. 3, pp. 30–51, June 2011.

[35] F. Goodarzi, D. Lee, and T. Lee, "Geometric nonlinear PID control of a quadrotor UAV on $\mathsf{SE}(3)$," in *European Control Conference (ECC).* IEEE, 2013, pp. 3845–3850.

[36] R. Bayadi and R. N. Banavar, "Almost global attitude stabilization of a rigid body for both internal and external actuation schemes," *European Journal of Control*, vol. 20, no. 1, pp. 45 – 54, 2014.

[37] Y. Zhu, X. Chen, and C. Li, “Some discussions about the error functions on $\mathsf{SO}(3)$ and $\mathsf{SE}(3)$ for the guidance of a uav using the screw algebra theory,” *Advances in Mathematical Physics*, 2017.

[38] N. A. Chaturvedi and N. H. McClamroch, “Almost global attitude stabilization of an orbiting satellite including gravity gradient and control saturation effects,” in *American Control Conference (ACC)*, June 2006.

[39] A. Safa, M. Baradarannia, H. Kharrati, and S. Khanmohammadi, “Global attitude stabilization of rigid spacecraft with unknown input delay,” *Nonlinear Dynamics*, vol. 82, no. 4, pp. 1623–1640, 2015.

[40] J. Markdahl, J. Hoppe, L. Wang, and X. Hu, “A geodesic feedback law to decouple the full and reduced attitude,” *Systems and Control Letters*, vol. 102, pp. 32 – 41, 2017.

[41] T. Lee, M. Leoky, and N. McClamroch, “Geometric tracking control of a quadrotor uav on $\mathsf{SE}(3)$,” in *49th IEEE Conference on Decision and Control (CDC)*, 2010, pp. 5420–5425.

[42] T. Lee, “Global exponential attitude tracking controls on $\mathsf{SO}(3)$,” *IEEE Transactions on Automatic Control*, vol. 60, no. 10, pp. 2837–2842, Oct 2015.

[43] K. Sreenath, T. Lee, and V. Kumar, “Geometric control and differential flatness of a quadrotor uav with a cable-suspended load,” in *52nd IEEE Conference on Decision and Control (CDC)*, Dec 2013, pp. 2269–2274.

[44] L. Taeyoung, L. Melvin, and M. N. Harris, “Nonlinear robust tracking control of a quadrotor UAV on SE(3),” *Asian Journal of Control*, vol. 15, no. 2, pp. 391–408, 2012.

[45] F. Bullo and R. M. Murray, “Tracking for fully actuated mechanical systems: A geometric framework,” *Automatica*, vol. 35, no. 1, pp. 17–34, Jan 1999.

[46] A. Akhtar, C. Nielsen, and S. L. Waslander, “Path following using dynamic transverse feedback linearization for car-like robots,” *IEEE Transactions on Robotics*, vol. 31, no. 2, pp. 269–279, April 2015.

[47] A. Akhtar, S. Waslander, and C. Nielsen, “Fault tolerant path following for a quadrotor,” in *52nd IEEE Conference on Decision and Control (CDC)*, Dec 2013, pp. 847–852.

[48] V. I. Arnold, *Mathematical Methods of Classical Mechanics*, ser. Graduate Texts in Mathematics. Springer, 1989.

[49] J. E. Marsden and T. S. Ratiu, *Introduction to Mechanics and Symmetry*, ser. A Basic Exposition of Classical Mechanical Systems. Springer-Verlag, 1999.

[50] F. Bullo and A. D. Lewis, *Geometric Control of Mechanical Systems*, ser. Texts in Applied Mathematics. New York-Heidelberg-Berlin: Springer Verlag, 2004, vol. 49.

[51] A. Agrachev and Y. Sachkov, *Control Theory from the Geometric Viewpoint*. Springer, 01 2002.

[52] S. Bouabdallah, P. Murrieri, and R. Siegwart, “Design and control of an indoor micro quadrotor,” in *IEEE International Conference on Robotics and Automation (ICRA)*, vol. 5, April 2004, pp. 4393–4398 Vol.5.

[53] M. D. Shuster, “Survey of attitude representations,” *Journal of the Astronautical Sciences*, vol. 41, pp. 439–517, Oct. 1993.

[54] E. W. Grafarend and W. Kühnel, “A minimal atlas for the rotation group SO(3),” *International Journal on Geomathematics - GEM*, vol. 2, no. 1, pp. 113–122, Jun 2011.

[55] J. Stuelpnagel, “On the parametrization of the three-dimensional rotation group,” *SIAM Review*, vol. 6, no. 4, pp. 422–430, 1964.

[56] D. Q. Huynh, "Metrics for 3d rotations: Comparison and analysis," *Journal of Mathematical Imaging and Vision*, vol. 35, no. 2, pp. 155–164, 2009.

[57] P.-J. Bristeau, P. Martin, E. Salaün, and N. Petit, "The role of propeller aerodynamics in the model of a quadrotor UAV," in *European Control Conference (ECC)*, 2009, pp. 683–688.

[58] Mohajerin, Nima, "Modeling dynamic systems for multi-step prediction with recurrent neural networks," Ph.D. dissertation, University of Waterloo, 2017.

[59] N. Mohajerin and S. L. Waslander, "State initialization for recurrent neural network modeling of time-series data," in *International Joint Conference on Neural Networks (IJCNN)*, May 2017, pp. 2330–2337.

[60] R. Ritz and R. D'Andrea, "A global controller for flying wing tailsitter vehicles," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 2731–2738.

[61] A. Akhtar, "Dynamic path following controllers for planar mobile robots," Master's thesis, Department of Electrical and Computer Engineering University of Waterloo, 2011.

[62] V. Guillemin and A. Pollack, *Differential Topology*. Englewood Cliffs NJ: Prentice-Hall, 1974.

[63] L. Consolini, M. Maggiore, C. Nielsen, and M. Tosques, "Path following for the PVTOL aircraft," *Automatica*, vol. 46, no. 8, pp. 1284–1296, August 2010.

[64] A. Hladio, C. Nielsen, and D. Wang, "Path following controller design for a class of mechanical systems," in *18th World Congress of the International Federation of Automatic Control*, Milano, Italy, Aug. 2011.

[65] G. Strang, *Linear Algebra and its applications*. 111 Fifth Avenue, New York, New York 10003: Academic Press, INC, 1976.

[66] C. Pugh, *Real Mathematical Analysis*. New York, U.S.A.: Springer, 2002.

[67] M. van Nieuwstadt, M. Rathinam, and R. M. Murray, "Differential flatness and absolute equivalence of nonlinear control systems," *SIAM J. Control and Optimization.*, vol. 36, no. 4, pp. 1225–1239, July 1998.

[68] P. Merriaux, Y. Dupuis, R. Boutteau, P. Vasseur, and X. Savatier, "A study of vicon system positioning performance," *Sensors*, vol. 17, no. 7, 2017.

[69] C. Sloth, T. Esbensen, and J. Stoustrup, "Active and passive fault-tolerant LPV control of wind turbines," in *American Control Conference (ACC)*, July 2010, pp. 4640–4646.

[70] Y. Zhang and J. Jiang, "Bibliographical review on reconfigurable fault-tolerant control systems," *Annual Reviews in Control*, vol. 32, no. 2, pp. 229 – 252, 2008.

[71] T. Li, "Nonlinear and fault-tolerant control techniques for a quadrotor unmanned aerial vehicle," Concordia University, Tech. Rep., 2011.

[72] Y. Zhang and A. Chamseddine, *Fault Tolerant Flight Control Techniques with Application to a Quadrotor UAV Testbed, Automatic Flight Control Systems*. InTech, 2012.

[73] T. Li, Y. Zhang, and B. W. Gordon, "Nonlinear fault-tolerant control of a quadrotor UAV based on sliding mode control technique," *IFAC Proceedings Volumes*, vol. 45, no. 20, pp. 1317 – 1322, 2012, 8th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes.

[74] Z. Liu, C. Yuan, Y. Zhang, and J. Luo, "A learning-based fault tolerant tracking control of an unmanned quadrotor helicopter," *Journal of Intelligent & Robotic Systems*, vol. 84, no. 1, pp. 145–162, Dec 2016.

[75] Y. Yi and Y. Zhang, "Fault diagnosis of an unmanned quadrotor helicopter based on particle filter," in *International Conference on Unmanned Aircraft Systems (ICUAS)*, June 2017, pp. 1432–1437.

[76] H. Jiang, Y. Yu, X. Ding, and J. Zhu, "A fault tolerant control strategy for quadrotor UAVs based on trajectory linearization approach," in *International Conference on Mechatronics and Automation (ICMA)*, Aug. 2012, pp. 1174–1179.

[77] A. Freddi, A. Lanzon, and S. Longhi, "A feedback linearization approach to fault tolerance in quadrotor vehicles," in *IFAC World Congress*, Milan, Italy, 2011, pp. 5413–5418.

[78] W. Haddad and V. Chellaboina, *Nonlinear Dynamical Systems and Control.* Prentice University Press, 2008.

[79] E. Sontag, "Smooth stabilization implies coprime factorization," *IEEE Transactions on Automatic Control*, vol. 34, no. 4, pp. 435–443, 1989.

[80] H. K. Khalil, *Nonlinear systems*, 3rd ed. Prentice Hall, 2002.

[81] J. Meyer, A. Sendobry, S. Kohlbrecher, U. Klingauf, and O. von Stryk, "Comprehensive simulation of quadrotor UAVs using ROS and Gazebo," in *3rd International Conference on Simulation, Modeling and Programming for Autonomous Robots (SIMPAR)*, 2012.

[82] N. R. E. Kani, N. Pullman, "Lecture notes on algebraic methods," in *Lecture notes on Algebraic Methods*, ser. Course notes Math 211, 2017.

[83] S. P. Bhat and D. S. Bernstein, "A topological obstruction to continuous global stabilization of rotational motion and the unwinding phenomenon," *Systems and Control Letters*, vol. 39, no. 1, pp. 63 – 70, 2000.

[84] M. Maggiore, "Reduction principles for hierarchical control design," in *Reduction Principles for Hierarchical Control Design*, ser. Mini Course, Norwegian University of Science and Technology, Trondheim, 2015.

[85] A. Roza and M. Maggiore, "A class of position controllers for underactuated vtol vehicles," *IEEE Transactions on Automatic Control*, vol. 59, no. 9, pp. 2580–2585, Sept 2014.

[86] E. D. Sontag and Y. Wang, “On characterizations of the input-to-state stability property,” *Systems & Control Letters*, vol. 24, no. 5, pp. 351 – 359, 1995.

[87] M. A. Hsieh, L. Chaimowicz, A. Cowley, B. Grocholsky, J. F. Keller, V. Kumar, C. J. Taylor, Y. Endo, R. C. Arkin, B. Jung, D. F. Wolf, G. Sukhatme, and D. C. MacKenzie, “Adaptive teams of autonomous aerial and ground robots for situational awareness,” *Journal of Field Robotics*, vol. 24, no. 11-12, p. 991 – 1014, Nov 2007.

[88] E. D. Sontag, “Input to state stability: Basic concepts and results,” in *Nonlinear and Optimal Control Theory*. Springer, 2006, pp. 163–220.