PIECEWISE BÉZIER CURVE TRAJECTORY GENERATION AND CONTROL FOR QUADROTORS

BY

ARUN LAKSHMANAN

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Aerospace Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2016

Urbana, Illinois

Adviser:

Professor Naira Hovakimyan

Abstract

Quadrotors have the capability of being immensely useful vehicles to aid humans in labor intensive tasks. The critical challenge of using quadrotors inside homes is the efficient navigation of these vehicles in tight spaces while avoiding obstacles. Although methods exists to generate collision free trajectories, they often do not account for the dynamics of quadrotor.

This thesis presents an approach for trajectory generation and control that can harness the complete dynamics of the quadrotor to achieve efficient navigation in cluttered spaces. First, the equations of motion for a quadrotor model is derived. It is also shown that the quadrotor system is differentially flat, which allows the analytical conversion of a time parameterized trajectory to states and outputs of the vehicle. Next, the thesis describes a control design approach in the non-Euclidean state space of quadrotors to achieve improved tracking performance for complex trajectories.

A novel trajectory generation method is presented to achieve smooth and graceful paths for quadrotors. The trajectory generation is formulated as an optimization problem that generates piecewise Bézier curves which minimize snap over the complete trajectory. The optimization method generates these trajectories from collision-free waypoints indicative of the constrained environment. Further, the Bézier curves are time parameterized to satisfy the dynamic constraints and ensure feasibility.

Acknowledgements

During the past two years at the University of Illinois, I have had the chance to work and interact with some great people who have contributed to this work in many positive ways.

I would especially like to thank my adviser, Professor Naira Hovakimyan, for all her support and guidance during my time at ACRL. I appreciate her advice and counsel on all the hurdles that I faced as I was beginning to shape my research.

I would like thank all my friends and colleagues at ACRL for their support and engaging discussions. Specifically, I would like to thank everyone at the IRL for all the fun times both inside and outside of the lab.

I am deeply grateful to my parents for their love, care and support and for cherishing the curious and inquisitive nature in me since childhood. At last, I would like thank my girlfriend, Mridula for her love, encouragement and understanding through all my busy schedules and late nights.

Table of Contents

Chapter 1

Introduction

In recent years we have seen a rising interest in the research and development of quadrotors for autonomous applications. Their relative compact size, high maneuverability, fewer moving components and low manufacturing cost are some of the very appealing factors that attract both consumers and researchers alike. With the rise of low-power computational chipsets and sensors, such vehicles can now perform complex tasks and achieve autonomy. The primary reason for such a keen interest in quadrotors is a result of the numerous real world applications that these vehicles can play a crucial role in. Quadrotors have been used in agriculture surveys, aerial photography, videography, urban surveillance, civil inspection and e-commerce package delivery to name a few.

Quadrotors can be immensely useful in cluttered environments such as households and warehouses, as their dynamics allow them to easily navigate such spaces without colliding with obstacles. Marinho et al. [1] discuss how quadrotors can aid elderly citizens with simple household tasks such as moving furniture or delivering medicine, while flying in a way that accounts for the perceived safety of humans in the vicinity.

These vehicles have four rotors mounted diagonally from each other, as pictured in Fig. 1.1. They exhibit four control actuations through the differential mapping of the rotor forces, namely, thrust, roll moment, pitch moment and yaw moment.

Figure 1.1: Crazyflie 2.0 quadrotor platform

Quadrotors are outfitted with an Inertial Measurement Unit (IMU) which allows them to measure accelerations, angular velocities, and heading angle. Although, this sensor is fairly reliable for attitude stabilization, it largely falls short when the vehicle needs to track a given position in 3D space because of the noise and biases in sensor measurements. Newer sensors such as ultra-wideband [2] and vision [3] provide fairly good localization information to the quadrotors, but they too are often prone to noise and failure. Some quadrotors also carry onboard computers for high-level tasks such as vision based mapping [4], and trajectory planning.

Many research groups in the controls and robotics community have contributed significantly to the fundamental research of these vehicles. Mellinger and Kumar [5] showed how the differential flatness properties of quadrotors can be exploited to generate aggressive trajectories. Hehn and D'Andrea [6], and Mueller et al. [2] describe approaches using model predictive control for generating real-time trajectories. Co-operative finite horizon control in Turpin

et al. [7] allows multiple quadrotors to fly in tight formation. Large swarm formations using micro aerial vehicles has been demonstrated in Preiss et al. [8], and Kushleyev et al. [9]. Designing controllers for quadrotors in non-Euclidean manifolds as prescribed by the vehicle dynamics has been explored in Lee et al. [10]. Mallikarjunan et al. [11] investigated different controller architectures using $\mathcal{L}_1$ adaptive control methods for attitude stabilization of quadrotors in the presence of unmodeled dynamics and external disturbances.

1.1 Motivation and Thesis Overview

Primary challenges for quadrotors to effectively fly in dense cluttered spaces are in sensing, motion planning and control. Most controllers available on quadrotors today often do not use the full scale of dynamic capabilities that the vehicle has to offer. Motion planning has remained a hot topic in the robotics community for nearly two decades but the available planners are simply not very good at generating dynamically feasible paths while being collision free which is crucial to quadrotors. In this thesis we present a computationally efficient approach of trajectory generation and control that takes advantage of the quadrotor dynamics to generate and track trajectories in cluttered environments (see Fig. 1.2).

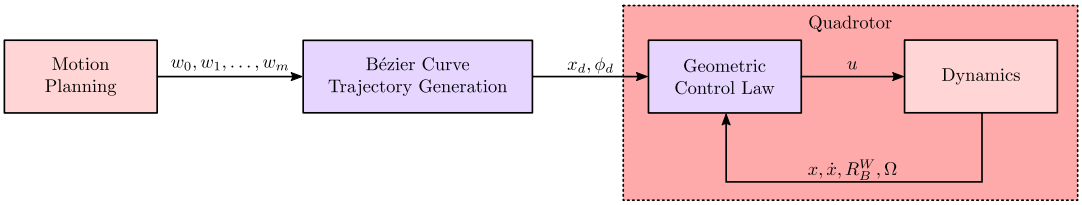

Figure 1.2: Trajectory generation and control problem structure

In chapter 2, we describe the equations of motions for a quadrotor vehicle. We will derive the differential flatness properties of such systems in order to analytically evaluate the vehicle states and outputs required to follow any desired trajectory.

Chapter 3 explores the design of a controller for the complete dynamics of the quadrotor. We

use the non-Euclidean manifold of the quadrotor's configuration space to design high performance controllers capable of tracking aggressive trajectories. We will also present some simulations and results of the tracking performance of the geometric controller.

Chapter 4 formulates an optimization problem to generate piecewise Bézier curves which minimize the snap of the entire trajectory while passing through a prescribed set of waypoints. It describes a two-step approach which optimizes flight time allowable by the quadrotor physics and sensing capabilities while maintaining minimum snap across the trajectory. The chapter will also include some simulation results using the optimization approach.

Chapter 2

Dynamic Model

Quadrotors are underactuated systems with six degrees of freedom and four control actuators. In order to track the time-parameterized position and orientation an accurate system model is required. Quadrotors are highly nonlinear as a result of the strong coupling between actuation and the dynamics, aerodynamic effects and mechanical design. They are susceptible to unmodeled behavior such as rotor blade flapping and vehicle induced drag, both of which have been extensively studied in Vervoorst [12] and Mahony et al. [13]. Dynamic modeling of non-planar multirotors analyzed in Brescianini and D'Andrea [14] and Crowther et al. [15] give insight into the control design methodology for such systems. We will make the following common assumptions as described in [16] during our analysis of the quadrotor model:

(i) *The quadrotor body has a rigid structure.*

(ii) *The body is axially symmetric.*

(iii) *The center of mass coincides with the center of symmetry.*

(iv) *Linear motor dynamics.*

(v) *No induced drag, blade flapping, and ground effects.*

In this chapter we define the dynamic model of quadrotors. We describe the coordinate frames which help represent the vehicle dynamics from an inertial and non-inertial reference

system. Further in this chapter, we also look at the differential flatness properties of such vehicles and how they can be advantageous during trajectory generation.

2.1 Coordinate Frames

Two frames of reference are defined to model the dynamics of a quadrotor as shown in Fig 2.1 using conventional right hand coordinate systems. The axes x_B, y_B and z_B represent the body fixed frame $\mathcal{B}$ of the vehicle and axes x_W, y_W and z_W represent the inertial frame $\mathcal{W}$. As the thesis focuses on indoor applications for quadrotors the z-up reference frame is adopted instead of the conventional North-East-Down (NED) frame of reference used in most aerospace applications.

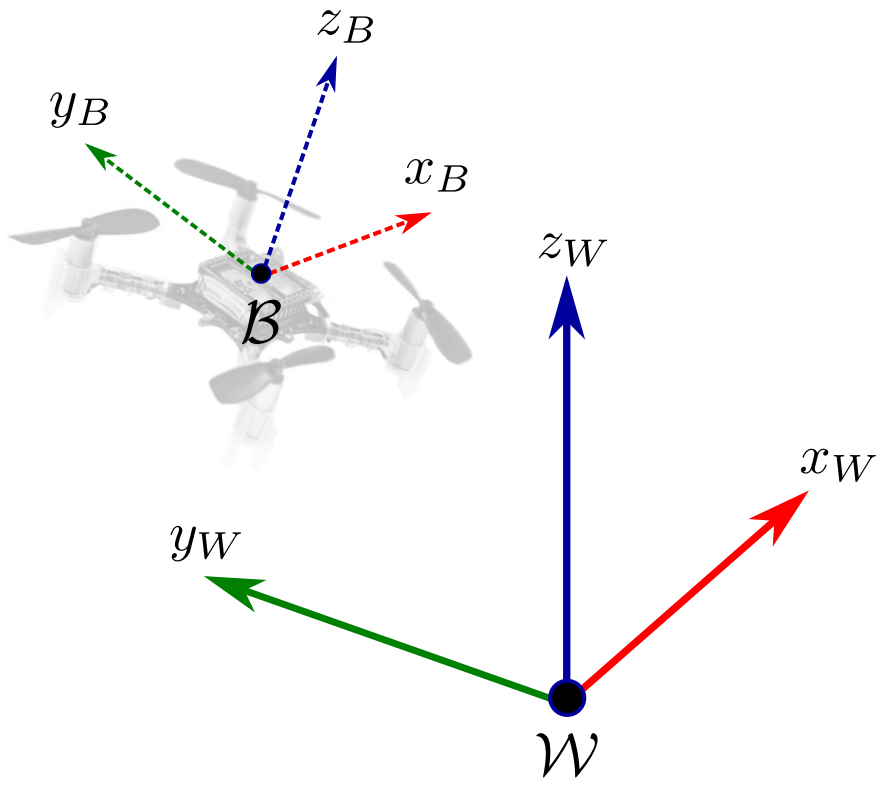

Figure 2.1: Inertial and body reference frames

The body-fixed frame is specified by the 'X' configuration of the quadrotor with the origin at the vehicle's center of mass. The 'X' configuration of quadrotor is defined such that the axes between the motors M1 and M4 describe the forward direction of the vehicle, as shown in Fig 2.2. Such a configuration allows for higher body torques and better reachability for a serial manipulator if it is attached underneath the body of the quadrotor.

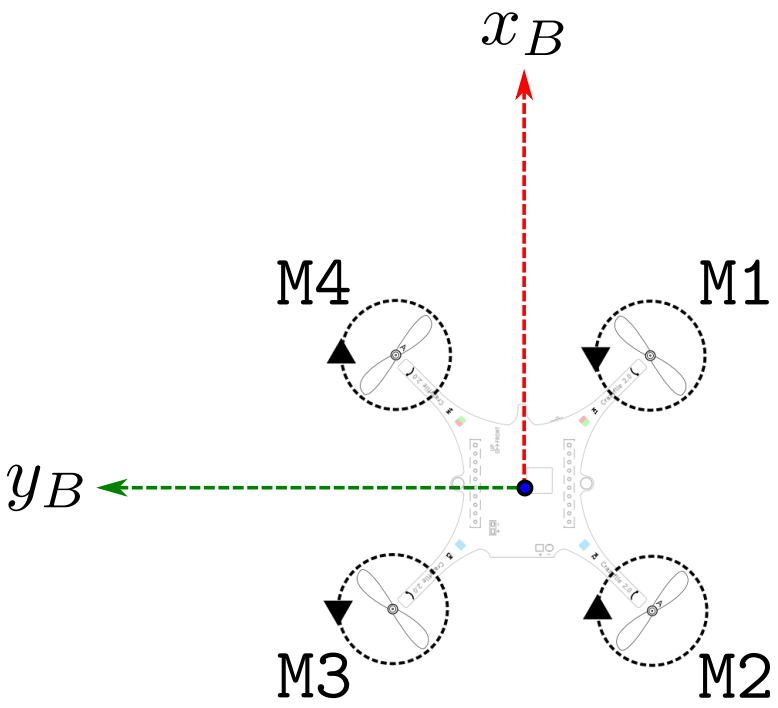

Figure 2.2: 'X' quadrotor configuration

In order to transform between reference frames, successive rotations are performed about the inertial $Z-Y-X$ axes by angles ψ, θ and ϕ. First, the frame is rotated about z_W by the yaw angle ψ, then it is rotated about y_W by the pitch angle θ, and finally rotated about x_W by the roll angle ϕ. The following equations describe the rotation from body to inertial frame R_B^W, using the $Z-Y-X$ Euler angle convention

$$R_B^W = R_{z,\psi} R_{y,\theta} R_{x,\psi} \tag{2.1}$$

$$= \begin{bmatrix} c_\psi & s_\psi & 0 \\ -s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_\theta & 0 & -s_\theta \\ 0 & 1 & 0 \\ s_\theta & 0 & c_\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\phi & s_\phi \\ 0 & -s_\phi & c_\phi \end{bmatrix} \tag{2.2}$$

$$= \begin{bmatrix} c_\psi c_\theta & c_\psi s_\theta s_\phi - c_\phi s_\psi & s_\psi s_\phi + c_\psi c_\phi s_\theta \\ c_\theta s_\psi & c_\psi c_\phi + s_\psi s_\theta s_\phi & c_\phi s_\psi s_\theta - c_\psi s_\phi \\ -s_\theta & c_\theta s_\phi & c_\theta c_\phi \end{bmatrix}, \tag{2.3}$$

where $\cos(\alpha)$ and $\sin(\alpha)$ are represented by c_α and s_α for brevity.

Therefore, the body axes can be written in $\mathcal{W}$ using the aforementioned rotation matrix as

$$x_B = R_B^W \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T, \tag{2.4}$$

$$y_B = R_B^W \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T, \tag{2.5}$$

$$z_B = R_B^W \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T. \tag{2.6}$$

2.2 Equations of Motion

2.2.1 Linear Acceleration

Using Newton's second law of motion, we can define the body acceleration in $\mathcal{W}$ as the addition of gravitational force and thrust component in the z_W as

$$m\ddot{x} = -mgz_W + u_f z_B \tag{2.7}$$

$$\ddot{x} = -gz_W + \frac{u_f}{m} z_B, \tag{2.8}$$

where $x \in \mathbb{R}^3$ is the position vector in $\mathcal{W}$, m is the mass of the vehicle, g is the gravitation constant and $u_f \in \mathbb{R}$ is the thrust force applied across all four rotors.

2.2.2 Angular Acceleration

Under the assumption that the body is axially symmetric, the body inertia tensor $\mathcal{I}$, does not contain any non-zero off diagonal elements. The inertia tensor matrix can then be simplified as

$$\mathcal{I} = \begin{bmatrix} \mathcal{I}_{xx} & 0 & 0 \\ 0 & \mathcal{I}_{yy} & 0 \\ 0 & 0 & \mathcal{I}_{zz} \end{bmatrix}, \tag{2.9}$$

where $\mathcal{I}_{xx}$, $\mathcal{I}_{yy}$, $\mathcal{I}_{zz}$ are scalar values computed using the mass and geometry properties of the quadrotor vehicle.

The moments $u_m = \begin{bmatrix} u_\phi & u_\theta & u_\psi \end{bmatrix}^T$, of the vehicle in $\mathcal{B}$ are given by the rate of change of angular momentum and an additional term for the gyroscopic forces, can be derived as

$$u_m = \frac{d}{dt} H \tag{2.10}$$

$$u_m = \frac{\partial}{\partial t} H + \Omega_B \times H \tag{2.11}$$

$$u_m = \mathcal{I} \cdot \dot{\Omega}_B + \Omega_B \times (\mathcal{I} \cdot \Omega_B), \tag{2.12}$$

where H is the angular momentum of the vehicle in body frame, and $\Omega_B = \begin{bmatrix} p & q & r \end{bmatrix}^T$ is the angular velocity in the body fixed frame $\mathcal{B}$. The angular accelerations in the body frame can be evaluated from this expression as

$$\dot{\Omega}_B = \mathcal{I}^{-1}(-\Omega_B \times \mathcal{I} \cdot \Omega_B + u_m) \tag{2.13}$$

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \frac{1}{\mathcal{I}_{xx}} & 0 & 0 \\ 0 & \frac{1}{\mathcal{I}_{yy}} & 0 \\ 0 & 0 & \frac{1}{\mathcal{I}_{zz}} \end{bmatrix} \left(- \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} \mathcal{I}_{xx}p \\ \mathcal{I}_{yy}q \\ \mathcal{I}_{zz}r \end{bmatrix} + \begin{bmatrix} u_\phi \\ u_\theta \\ u_\psi \end{bmatrix} \right) \tag{2.14}$$

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \frac{1}{\mathcal{I}_{xx}} & 0 & 0 \\ 0 & \frac{1}{\mathcal{I}_{yy}} & 0 \\ 0 & 0 & \frac{1}{\mathcal{I}_{zz}} \end{bmatrix} \left(- \begin{bmatrix} (-\mathcal{I}_{yy} + \mathcal{I}_{zz})qr \\ (\mathcal{I}_{xx} - \mathcal{I}_{zz})pr \\ (-\mathcal{I}_{xx} + \mathcal{I}_{yy})pq \end{bmatrix} + \begin{bmatrix} u_\phi \\ u_\theta \\ u_\psi \end{bmatrix} \right) \tag{2.15}$$

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \frac{1}{\mathcal{I}_{xx}} & 0 & 0 \\ 0 & \frac{1}{\mathcal{I}_{yy}} & 0 \\ 0 & 0 & \frac{1}{\mathcal{I}_{zz}} \end{bmatrix} \left(\begin{bmatrix} (\mathcal{I}_{yy} - \mathcal{I}_{zz})qr + u_\phi \\ (\mathcal{I}_{zz} - \mathcal{I}_{xx})pr + u_\theta \\ (\mathcal{I}_{xx} - \mathcal{I}_{yy})pq + u_\psi \end{bmatrix} \right). \tag{2.16}$$

The angular acceleration about each of the body axes can be defined as

$$\dot{p} = \frac{1}{\mathcal{I}_{xx}}((\mathcal{I}_{yy} - \mathcal{I}_{zz})qr + u_\phi), \tag{2.17}$$

$$\dot{q} = \frac{1}{\mathcal{I}_{yy}}((\mathcal{I}_{zz} - \mathcal{I}_{xx})pr + u_\theta), \tag{2.18}$$

$$\dot{r} = \frac{1}{\mathcal{I}_{zz}}((\mathcal{I}_{xx} - \mathcal{I}_{yy})pq + u_\psi). \tag{2.19}$$

From the linear acceleration (2.8) and angular acceleration (2.17-2.18) equations, we can describe the vehicle states and control inputs to the actuators as

$$\mathbf{x} = \begin{bmatrix} x^T & \Theta^T & \dot{x}^T & \Omega_B^T \end{bmatrix}^T, \tag{2.20}$$

$$\mathbf{u} = \begin{bmatrix} u_f & u_\phi & u_\theta & u_\psi \end{bmatrix}^T, \tag{2.21}$$

where $\Theta = \begin{bmatrix} \phi & \theta & \psi \end{bmatrix}^T$ is the orientation of the vehicle defined as Euler angles.

2.2.3 Motor Mapping

The control inputs to the rotors $\begin{bmatrix} u_1 & u_2 & u_3 & u_4 \end{bmatrix}^T$ are defined by the following linear mapping to the thrust u_f, and moments $u_m = \begin{bmatrix} u_\phi & u_\theta & u_\psi \end{bmatrix}^T$ as

$$\begin{bmatrix} u_f \\ u_\phi \\ u_\theta \\ u_\psi \end{bmatrix} = \begin{bmatrix} k_F & k_F & k_F & k_F \\ -\frac{k_F}{\sqrt{2}}L & -\frac{k_F}{\sqrt{2}}L & \frac{k_F}{\sqrt{2}}L & \frac{k_F}{\sqrt{2}}L \\ -\frac{k_F}{\sqrt{2}}L & \frac{k_F}{\sqrt{2}}L & \frac{k_F}{\sqrt{2}}L & -\frac{k_F}{\sqrt{2}}L \\ -k_Q & k_Q & -k_Q & k_Q \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix}, \tag{2.22}$$

where k_F is the thrust coefficient, k_Q is the torque coefficient and L is the arm length defined by the quadrotor geometry. As the matrix has linearly independent columns if $k_F, k_Q, L \neq 0$, the rotor inputs can be calculated through matrix inversion as

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} k_F & k_F & k_F & k_F \\ -\frac{k_F}{\sqrt{2}}L & -\frac{k_F}{\sqrt{2}}L & \frac{k_F}{\sqrt{2}}L & \frac{k_F}{\sqrt{2}}L \\ -\frac{k_F}{\sqrt{2}}L & \frac{k_F}{\sqrt{2}}L & \frac{k_F}{\sqrt{2}}L & -\frac{k_F}{\sqrt{2}}L \\ -k_Q & k_Q & -k_Q & k_Q \end{bmatrix}^{-1} \begin{bmatrix} u_f \\ u_\phi \\ u_\theta \\ u_\psi \end{bmatrix} \tag{2.23}$$

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \frac{1}{4} \begin{bmatrix} \frac{1}{k_F} & -\frac{\sqrt{2}}{k_F L} & -\frac{\sqrt{2}}{k_F L} & -\frac{1}{k_Q} \\ \frac{1}{k_F} & -\frac{\sqrt{2}}{k_F L} & \frac{\sqrt{2}}{k_F L} & \frac{1}{k_Q} \\ \frac{1}{k_F} & \frac{\sqrt{2}}{k_F L} & \frac{\sqrt{2}}{k_F L} & -\frac{1}{k_Q} \\ \frac{1}{k_F} & \frac{\sqrt{2}}{k_F L} & -\frac{\sqrt{2}}{k_F L} & \frac{1}{k_Q} \end{bmatrix} \begin{bmatrix} u_f \\ u_\phi \\ u_\theta \\ u_\psi \end{bmatrix}. \tag{2.24}$$

2.3 Differential Flatness

Differential flatness for quadrotors is a well explored area of research [17, 18]. Quadrotors possess the flatness property if all of the states $\mathbf{x}$, and inputs $\mathbf{u}$, can be represented as functions of flat outputs $\boldsymbol{\sigma}$, and a finite number of their derivatives without the need for a time integral. If there exists a flat output represented as

$$\boldsymbol{\sigma} = h(\mathbf{x}, \mathbf{u}, \dot{\mathbf{u}}, \ldots, \mathbf{u}^{(k)}), \tag{2.25}$$

such that the functions $g(\cdot)$ and $k(\cdot)$ satisfy

$$\mathbf{x} = g(\boldsymbol{\sigma}, \dot{\boldsymbol{\sigma}}, \ldots, \boldsymbol{\sigma}^{(j)}), \tag{2.26}$$

$$\mathbf{u} = k(\boldsymbol{\sigma}, \dot{\boldsymbol{\sigma}}, \ldots, \boldsymbol{\sigma}^{(j)}), \tag{2.27}$$

then the vehicle is said to be differentially flat.

Similar to the structure in [5], the flat outputs for our system can be defined by the desired position and heading angle of a trajectory as

$$\boldsymbol{\sigma} = \begin{bmatrix} x_d^T & \psi_d \end{bmatrix}^T, \tag{2.28}$$

where $x_d \in \mathbb{R}^3$ is the desired position vector and $\psi_d \in SO(2)$ is the desired heading angle.

2.3.1 Translation

The following equations describe how the translational states and their derivatives can be represented as derivatives of the flat outputs. The mapping from the flat outputs to the position is

$$x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \boldsymbol{\sigma}, \tag{2.29}$$

the velocity is

$$\dot{x} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \dot{\boldsymbol{\sigma}}, \tag{2.30}$$

and the acceleration is

$$\ddot{x} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \ddot{\boldsymbol{\sigma}}. \tag{2.31}$$

Similar to equations above, the jerk and snap trajectory states can also be derived by taking higher order derivatives of the flat outputs.

2.3.2 Orientation

In order to construct the body orientation R_B^W (as shown in Fig. 2.3) in the inertial frame, the desired z_B vector is first obtained from (2.8) and (2.31) as the direction of the thrust vector given by

$$z_B = \frac{\begin{bmatrix} I_{3\times3} & 0 \end{bmatrix} \cdot \ddot{\boldsymbol{\sigma}} + g z_W}{\left\| \begin{bmatrix} I_{3\times3} & 0 \end{bmatrix} \cdot \ddot{\boldsymbol{\sigma}} + g z_W \right\|}. \tag{2.32}$$

An intermediate vector x_I is chosen based on the heading angle of the trajectory as

$$x_I = \frac{\begin{bmatrix} \cos(\psi_d) & \sin(\psi_d) & 0 \end{bmatrix}^T}{\left\| \begin{bmatrix} \cos(\psi_d) & \sin(\psi_d) & 0 \end{bmatrix}^T \right\|}. \tag{2.33}$$

y_B is chosen orthonormal to z_B and x_I, and x_B is chosen orthonormal to y_B and z_B as

$$y_B = z_B \times x_I, \tag{2.34}$$

$$x_B = y_B \times z_B. \tag{2.35}$$

From the definition of R_B^W in equations (2.4), (2.5), (2.6), the body orientation can be constructed as

$$R_B^W = \begin{bmatrix} x_B & y_B & z_B \end{bmatrix}. \tag{2.36}$$

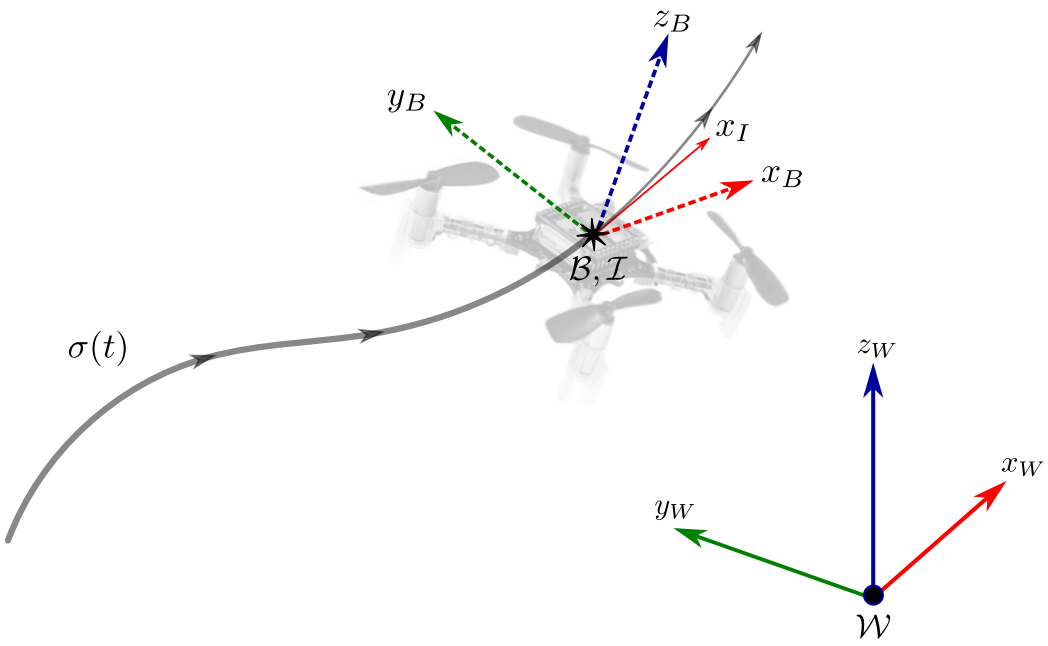

Figure 2.3: Body and intermediate reference frames

2.3.3 Angular Velocity

Defining the component of jerk on the z_B axes, we obtain the rate of change of thrust $\dot{u}_f$ as,

$$\dot{u}_f = m(\dddot{x} \cdot z_B) \tag{2.37}$$

$$\frac{\dot{u}_f}{m} = \dddot{x} \cdot z_B. \tag{2.38}$$

On taking the derivative of the Newton equation (2.8), the jerk of the trajectory is given by

$$\dddot{x} = \frac{\dot{u}_f}{m} z_B + \Omega_B \times \frac{u_f}{m} z_B. \tag{2.39}$$

The angular accelerations can be expressed as

$$\Omega_B \times z_B = \frac{m}{u_f}(\dddot{x} - \frac{\dot{u}_f}{m} z_B) \tag{2.40}$$

$$\begin{bmatrix} q \\ -p \\ 0 \end{bmatrix} = \frac{m}{u_f}(\dddot{x} - \frac{\dot{u}_f}{m} z_B). \tag{2.41}$$

Substituting from (2.38) it follows that

$$\begin{bmatrix} q \\ -p \\ 0 \end{bmatrix} = \frac{m}{u_f}(\dddot{x} - (\dddot{x} \cdot z_B) z_B). \tag{2.42}$$

The pitch and roll angular velocities can be defined as

$$p = -\frac{m}{u_f}(\dddot{x} - (\dddot{x} \cdot z_B) z_B) \cdot y_B, \tag{2.43}$$

$$q = \frac{m}{u_f}(\dddot{x} - (\dddot{x} \cdot z_B) z_B) \cdot x_B. \tag{2.44}$$

The body yaw acceleration is obtained by taking the component of the desired heading angle velocity along z_B as

$$r = \Omega_I \cdot z_B \tag{2.45}$$

$$r = \dot{\psi}_d (z_W \cdot z_B). \tag{2.46}$$

where Ω_I is the angular rate of the body in the intermediate frame.

2.3.4 Angular Acceleration

The double derivative of the thrust vector u_f is obtained from the differentiation of (2.38) as

$$\ddot{u_f} = m(\dddot{x} \cdot z_B) - (\Omega_B \times \Omega_B \times u_f z_B) \cdot z_B \tag{2.47}$$

$$\frac{\ddot{u}_f}{m} = \dddot{x} \cdot z_B - (\Omega_B \times \Omega_B \times z_B) \cdot \frac{u_f}{m} z_B. \tag{2.48}$$

On taking the derivative of the Newton equation (2.39), the snap of the trajectory is given by

$$\ddddot{x} = \frac{\ddot{u}_f}{m} z_B + 2\Omega_B \times \frac{\dot{u}_f}{m} z_B + \Omega_B \times \Omega_B \times \frac{u_f}{m} z_B + \dot{\Omega}_B \times \frac{u_f}{m} z_B. \tag{2.49}$$

The angular accelerations can be expressed as

$$\dot{\Omega}_B \times z_B = \frac{m}{u_f} \left(\ddddot{x} - \frac{\ddot{u}_f}{m} z_B - 2\Omega_B \times \frac{\dot{u}_f}{m} z_B - \Omega_B \times \Omega_B \times \frac{u_f}{m} z_B \right) \tag{2.50}$$

$$\begin{bmatrix} \dot{q} \\ -\dot{p} \\ 0 \end{bmatrix} = \frac{m}{u_f} \left(\ddddot{x} - \frac{\ddot{u}_f}{m} z_B - 2\Omega_B \times \frac{\dot{u}_f}{m} z_B - \Omega_B \times \Omega_B \times \frac{u_f}{m} z_B \right). \tag{2.51}$$

The pitch and roll angular accelerations can be defined as

$$\dot{p} = -\frac{m}{u_f} \left(\ddddot{x} - \frac{\ddot{u}_f}{m} z_B - 2\Omega_B \times \frac{\dot{u}_f}{m} z_B - \Omega_B \times \Omega_B \times \frac{u_f}{m} z_B \right) \cdot y_B, \tag{2.52}$$

$$\dot{q} = \frac{m}{u_f} \left(\ddddot{x} - \frac{\ddot{u}_f}{m} z_B - 2\Omega_B \times \frac{\dot{u}_f}{m} z_B - \Omega_B \times \Omega_B \times \frac{u_f}{m} z_B \right) \cdot x_B. \tag{2.53}$$

The body yaw acceleration is obtained by taking the component of the desired heading angle acceleration along z_B as

$$\dot{r} = \dot{\Omega}_I \cdot z_B \tag{2.54}$$

$$\dot{r} = \ddot{\psi}_d (z_W \cdot z_B) \tag{2.55}$$

where $\dot{\Omega}_I$ is the angular acceleration of the body in the intermediate frame.

2.3.5 Control Inputs

Thrust of the quadrotor vehicle is follows from the desired acceleration (2.32) as

$$u_f = m \left\| \begin{bmatrix} I_{3\times 3} & 0 \end{bmatrix} \cdot \ddot{\boldsymbol{\sigma}} + g z_W \right\|. \tag{2.56}$$

The moment inputs can be evaluated from the angular velocity (2.43),(2.44),(2.46),the angular accelerations (2.52),(2.53),(2.55) and Euler equations (2.13) as

$$u_m = \mathcal{I}\dot{\Omega}_B + \Omega_B \times \mathcal{I}\Omega_B. \tag{2.57}$$

The pitch, roll and yaw moments can be defined as

$$u_\phi = \mathcal{I}_{xx}\dot{p} - \mathcal{I}_{yy}qr + \mathcal{I}_{zz}qr, \tag{2.58}$$

$$u_\theta = \mathcal{I}_{xx}pr + \mathcal{I}_{yy}\dot{q} - \mathcal{I}_{zz}pr, \tag{2.59}$$

$$u_\psi = -\mathcal{I}_{xx}pq + \mathcal{I}_{yy}pq + \mathcal{I}_{zz}\dot{r}. \tag{2.60}$$

Importantly, we can see that the flat outputs enter the body moments as fourth derivatives. This property will be used in later sections for generating efficient trajectories for quadrotor vehicles.

Chapter 3

Geometric Control

Effective controller design is crucial for a quadrotor vehicle to autonomously track a given reference. In recent years, reference tracking controllers for multirotors have generated a lot of interest in academia owing to the improvements in low-power and efficient computer chips which can be used on-board the vehicles.

Controllers designed for linearized dynamics of the quadrotors have been widely studied in [19, 16, 20]. Hehn and D'Andrea [21] and Mueller et al. [22] have proposed computationally efficient methods to solve time-optimal interception problems using model predictive control techniques. Nonlinear control design using sliding mode and backstepping approaches are explored in Bouabdallah and Siegwart [23].

Path following controllers ensure that a vehicle converges to, and tracks a desired reference path without any temporal constraints. This allows for a free control input to satisfy other dynamic objectives of the problem. In Cichella et al. [24], the authors use the free control input to vary the time parametrization of a given reference path for collision avoidance and time-coordination between multiple vehicles.

In our problem, trajectories are generated for minimum snap characteristics for a minimum feasible flight duration. Naturally, this implies that the references are time parametrized,

and that the control design needs to account for temporal specification set during the trajectory generation.

In this chapter, we will formulate the controller design as a geometric control problem, as seen in [10, 5]. We will define equations that describe the trajectory tracking errors and the control law. Later in the chapter, we will discuss the *Simulink* implementation for a quadrotor vehicle tracking 3D nonlinear trajectories and compare the performance with that of a cascaded PID position controller.

3.1 Problem Formulation

The reference trajectory is defined by the following time parametrized variables

$$\sigma_d(t) = \begin{bmatrix} x_d(t)^T & \psi_d(t)^T \end{bmatrix}^T, \tag{3.1}$$

where $x_d(t)$ is the desired position and $\psi_d(t)$ is the desired heading of the vehicle at time t. It is easy to see that $\sigma_d \in \mathbb{R}^3 \times \mathbb{S}$. We will assume that the reference satisfies the physical bounds of the quadrotor on linear acceleration and angular velocity, defined as

$$\ddot{x}_d < a_{\max}, \tag{3.2}$$

$$\Omega_D < \omega_{\max}. \tag{3.3}$$

A quadrotor has six degrees of freedom and its configuration space is defined as $SE(3) \equiv \mathbb{R}^3 \times SO(3)$. Geometric control techniques can be used to track references in such nonlinear manifolds. Specifically with respect to quadrotors, this means that controllers can be designed directly on $SO(3)$ and prevent any singularities that occur in purely Euclidean space based controllers.

We will define the following mathematical operators, which will be used extensively in this

section. The *hat* operator: $^\wedge$, is a smooth map from $\mathbb{R}^3 \to SO(3)$ given by

$$\widehat{\begin{bmatrix} a \\ b \\ c \end{bmatrix}} = \begin{bmatrix} 0 & -c & b \\ c & 0 & -a \\ -b & a & 0 \end{bmatrix}. \tag{3.4}$$

And the *vee* operator: $^\vee$, is a smooth map from $SO(3) \to \mathbb{R}^3$ given by

$$\begin{bmatrix} 0 & -c & b \\ c & 0 & -a \\ -b & a & 0 \end{bmatrix}^\vee = \begin{bmatrix} a \\ b \\ c \end{bmatrix}. \tag{3.5}$$

The control design follows the structure illustrated in Fig. 3.1. As described in the previous chapter, the controller inputs are comprised of the force and moment outputs as

$$\mathbf{u} = \begin{bmatrix} u_f & u_\phi & u_\theta & u_\psi \end{bmatrix}^T, \tag{3.6}$$

where $u_f \in \mathbb{R}$ is the thrust output, and $u_\phi, u_\theta, u_\psi \in \mathbb{R}$ are the moment outputs to the quadrotor about each axis.

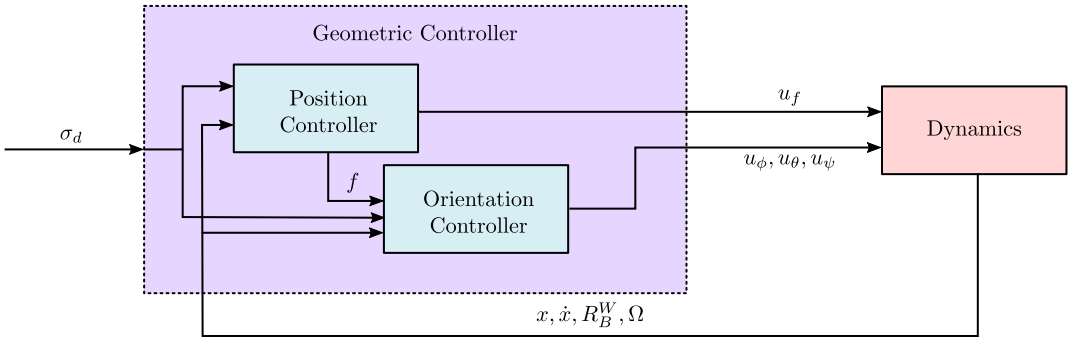

Figure 3.1: Geometric controller design

3.2 Controller Design

3.2.1 Thrust output

Translational error quantities are computed as

$$e_x = x_d - x, \tag{3.7}$$

$$e_v = \dot{x}_d - \dot{x}, \tag{3.8}$$

where e_x and e_v are the position error and velocity error respectively. The magnitude of the magnitude of desired force vector f, and the thrust input u_f, can be computed from (2.8) as

$$f = \left\| K_x e_x + K_v e_v + mgz_W + m\ddot{x}_d \right\|, \tag{3.9}$$

$$u_f = (K_x e_x + K_v e_v + mgz_W + m\ddot{x}_d) \cdot z_B, \tag{3.10}$$

$$\tag{3.11}$$

such that

$$K_x \succ 0, \tag{3.12}$$

$$K_v \succ 0, \tag{3.13}$$

where K_x and K_v are gain matrices for position and velocity tracking errors. We can see that the above equation is a PD controller with the gravity and desired trajectory accelerations appearing as feedforward terms.

3.2.2 Moment output

In order to prevent singularity when the desired thrust output is zero, the desired body z-axis $z_{B,d}$, is constructed as

$$z_{B,d} = \frac{(K_x + s_x I)e_x + (K_v + s_v I)e_v + (mg + s_a)z_W + m\ddot{x}_d}{\|(K_x + s_x I)e_x + (K_v + s_v I)e_v + (mg + s_a)z_W + m\ddot{x}_d\|}, \tag{3.14}$$

where

$$s_x = \begin{cases} 0, & \text{if } f \neq 0 \\ \text{sgn}(e_x \cdot z_W), & \text{if } f = 0 \end{cases}, \tag{3.15}$$

$$s_v = \begin{cases} 0, & \text{if } f \neq 0 \\ \text{sgn}(e_v \cdot z_W), & \text{if } f = 0 \end{cases}, \tag{3.16}$$

$$s_a = \begin{cases} 0, & \text{if } f \neq 0 \\ 1, & \text{if } \begin{cases} f = 0 \\ e_x \equiv 0 \\ e_v \equiv 0 \end{cases} \end{cases}. \tag{3.17}$$

An intermediate vector $x_{I,d}$ is chosen based on the heading angle of the trajectory as

$$x_{I,d} = \frac{\begin{bmatrix} \cos(\psi_d) & \sin(\psi_d) & 0 \end{bmatrix}^T}{\left\| \begin{bmatrix} \cos(\psi_d) & \sin(\psi_d) & 0 \end{bmatrix}^T \right\|}. \tag{3.18}$$

$y_{B,d}$ is chosen orthonormal to $z_{B,d}$ and $x_{I,d}$, and $x_{B,d}$ is chosen orthonormal to $y_{B,d}$ and $z_{B,d}$ as

$$y_{B,d} = z_{B,d} \times x_{I,d} \tag{3.19}$$

$$x_{B,d} = y_{B,d} \times z_{B,d}. \tag{3.20}$$

The rotation R_D^W from desired body orientation frame $\mathcal{D}$ to $\mathcal{W}$ is defined by the desired body axes as

$$R_D^W \begin{bmatrix} x_{B,d} & y_{B,d} & z_{B,d}. \end{bmatrix} \tag{3.21}$$

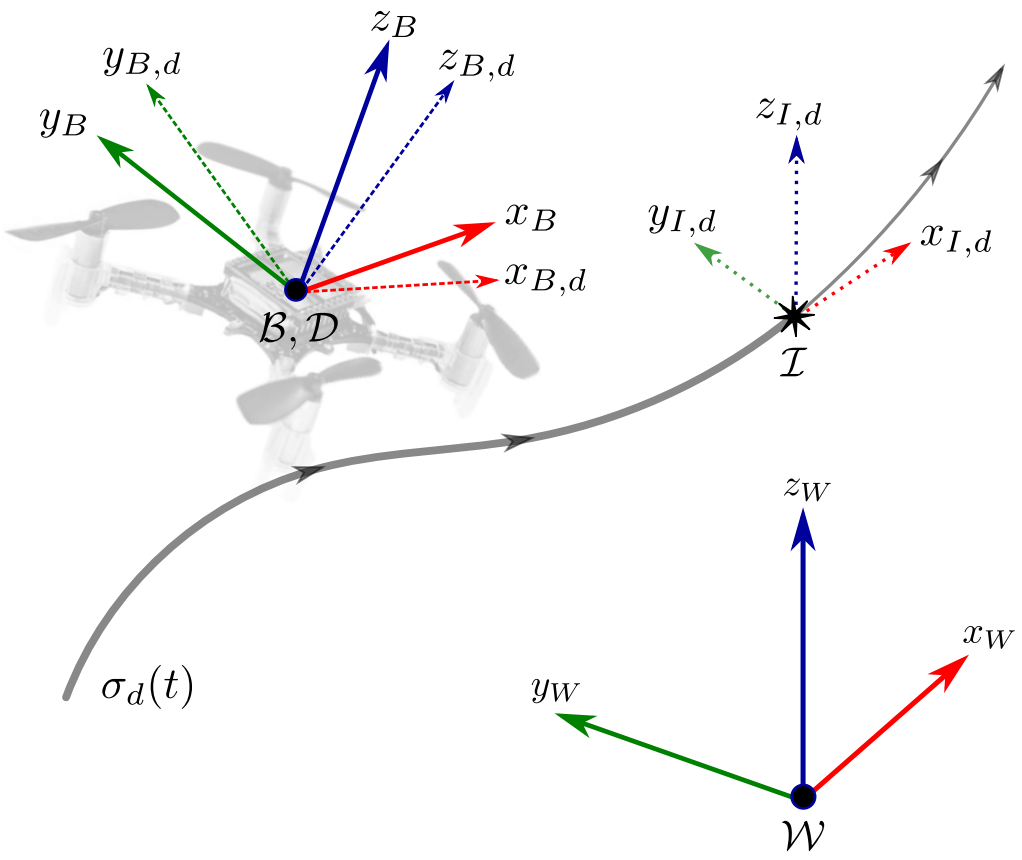

Figure 3.2: Desired and body frames of reference

Let us define the rotation from $\mathcal{B}$ to the desired orientation $\mathcal{D}$ as

$$R_B^D = R_W^D R_B^W = (R_D^W)^T R_B^W, \tag{3.22}$$

and the derivative of this orientation is given by

$$\dot{R}_B^D = R_B^D(\Omega_B - (R_B^D)^T \Omega_D). \tag{3.23}$$

The attitude tracking error e_R, can be chosen as

$$e_R = \frac{1}{2}(R_B^D - (R_B^D)^T)^\vee, \tag{3.24}$$

and the angular velocity tracking error e_Ω, is given by the time derivative of R_B^D in the $\mathcal{B}$ frame as

$$e_\Omega = R_D^B \dot{R}_B^D = \Omega_B - (R_B^D)^T \Omega_D. \tag{3.25}$$

The desired moment output is given as

$$u_m = -K_R e_R - K_\Omega e_\Omega + \Omega_B \times \mathcal{I}\Omega_B - \mathcal{I}\left(\hat{\Omega}_B (R_B^D)^T \Omega_D - (R_B^D)^T \dot{\Omega}_D\right), \tag{3.26}$$

$$\tag{3.27}$$

such that,

$$K_R \succ 0, \tag{3.28}$$

$$K_\Omega \succ 0, \tag{3.29}$$

where K_R and K_Ω are diagonal gain matrices for orientation and angular velocity errors. We can see that the above equation is a PD controller on SO(3) with the Coriolis term and desired angular accelerations appearing as feedforward terms.

3.2.3 Properties

The control inputs are given by the equations (3.10) and (3.26). Importantly, this controller improves on the standard PID design by mapping the thrust outputs to the motor axis (z_B) and defining the orientation error on $SO(3)$ to avoid singularities. This control design satisfies exponential stability of complete dynamics under the following initial conditions constraints

$$\text{tr}\left[I - R_B^D(0)\right] < 2, \tag{3.30}$$

$$\|e_\Omega(0)\| < \frac{k_R}{\max(\mathcal{I})}\left(2 - \text{tr}\left[I - R_B^D(0)\right]\right), \tag{3.31}$$

and for almost global exponential attractiveness of complete dynamics, the region of attraction is characterized by

$$2 \leq \text{tr}\left[I - R_B^D(0)\right] < 4, \tag{3.32}$$

$$\|e_\Omega(0)\| < \frac{k_R}{\max(\mathcal{I})}\left(4 - \text{tr}\left[I - R_B^D(0)\right]\right). \tag{3.33}$$

Proofs for these propositions are available in Lee et al. [25].

3.3 Simulations

Two different simulations are presented in the following section. In the first simulation, the quadrotor model is given a helical trajectory to follow and the tracking performance of the geometric controller is investigated. The second simulation is a comparison between the geometric controller and a cascaded PID controller. The simulations are performed on *Simulink* using a model attributed with the physical properties of a Crazyflie 2.0 quadrotor.

3.3.1 Tracking Performance

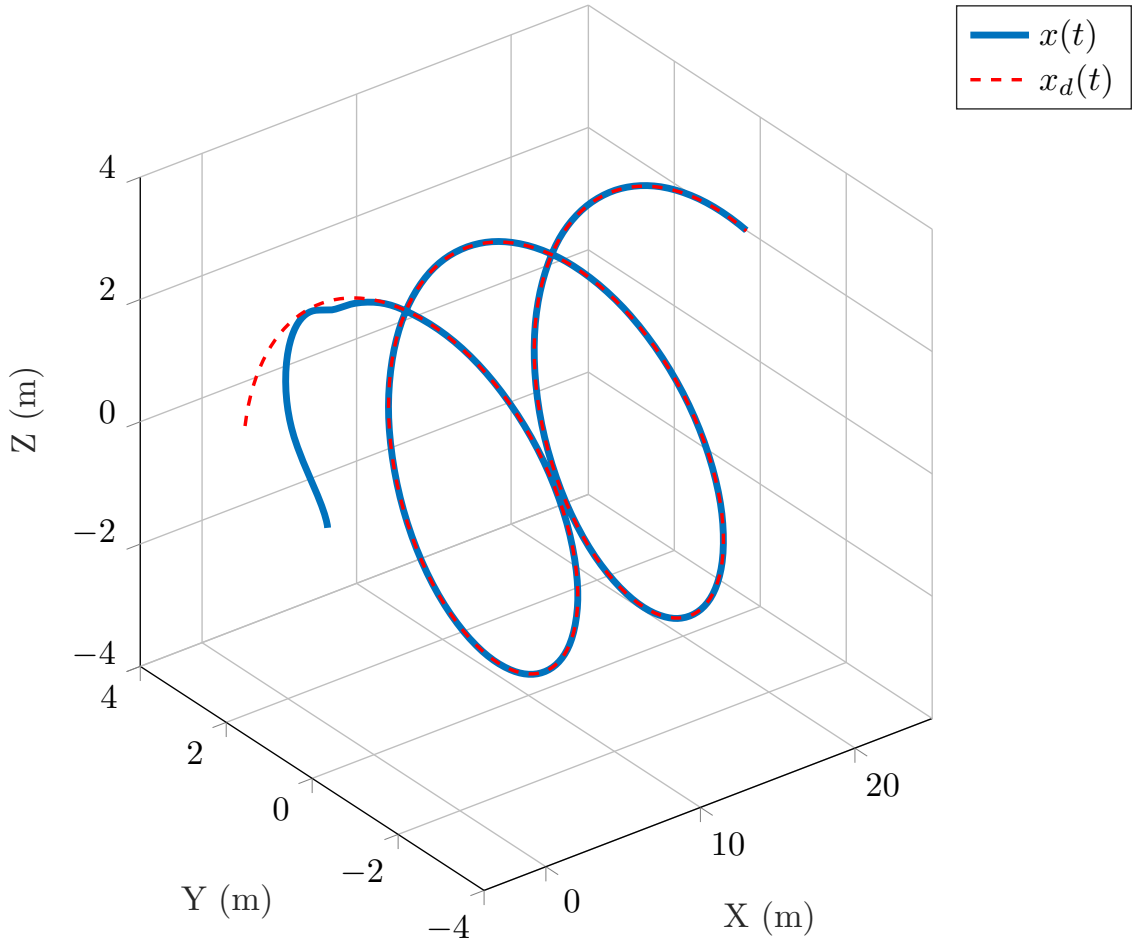

Figure 3.3: Reference and quadrotor trajectory

In this simulation, the quadrotor is tasked to follow a helical trajectory defined along the X-axis as shown in Fig. 3.3. The reference trajectory moves along the horizontal axis at 1 m/s with an angular velocity of $\frac{2}{3}$ rad/s and a radius of 3 m. The quadrotor is initialized from a considerable offset at $\begin{bmatrix} -3 & 0 & 0 \end{bmatrix}^T$, whereas the reference has its initial position at $\begin{bmatrix} 0 & 3 & 0 \end{bmatrix}^T$. Figure 3.3 shows the convergence of the actual quadrotor trajectory to the desired trajectory. The convergence of the translational and rotational tracking errors to zero is shown in Fig. 3.4-3.7 show . The time history of the force and moment control inputs is illustrated in Fig. 3.8 prior to the motor mixing and rotor thrust output.

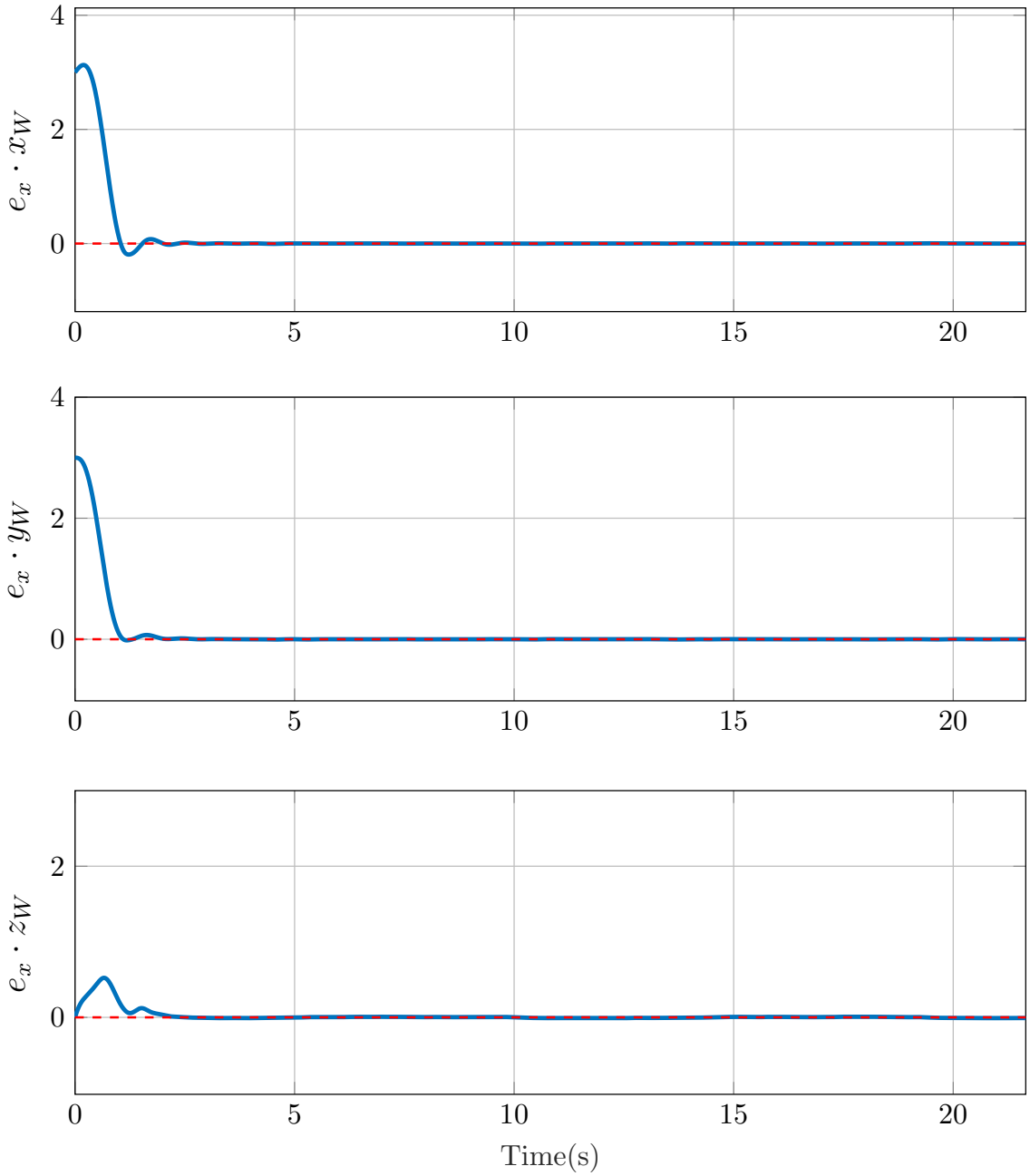

Figure 3.4: Time history of position errors along each axis

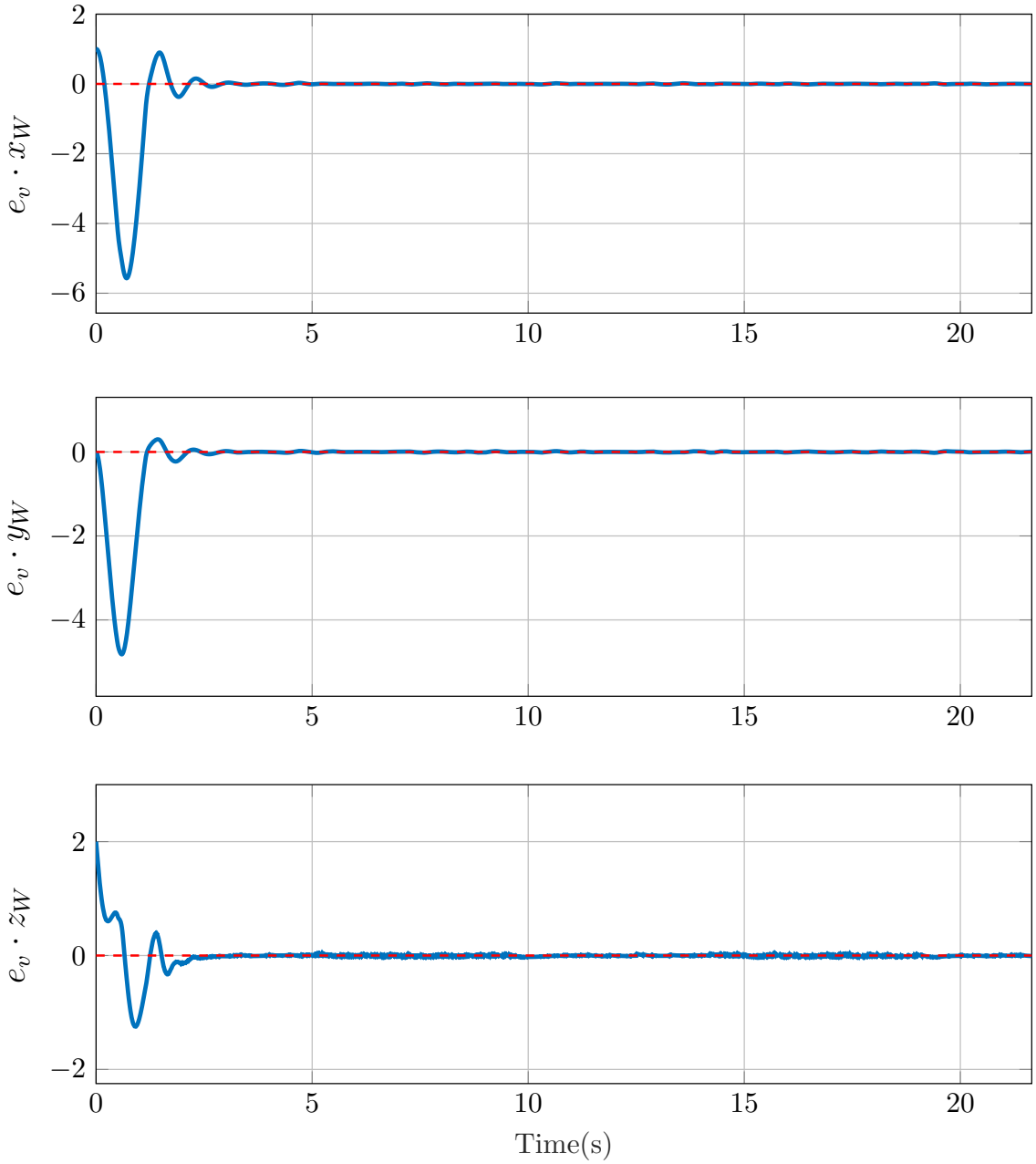

Figure 3.5: Time history of velocity errors along each axis

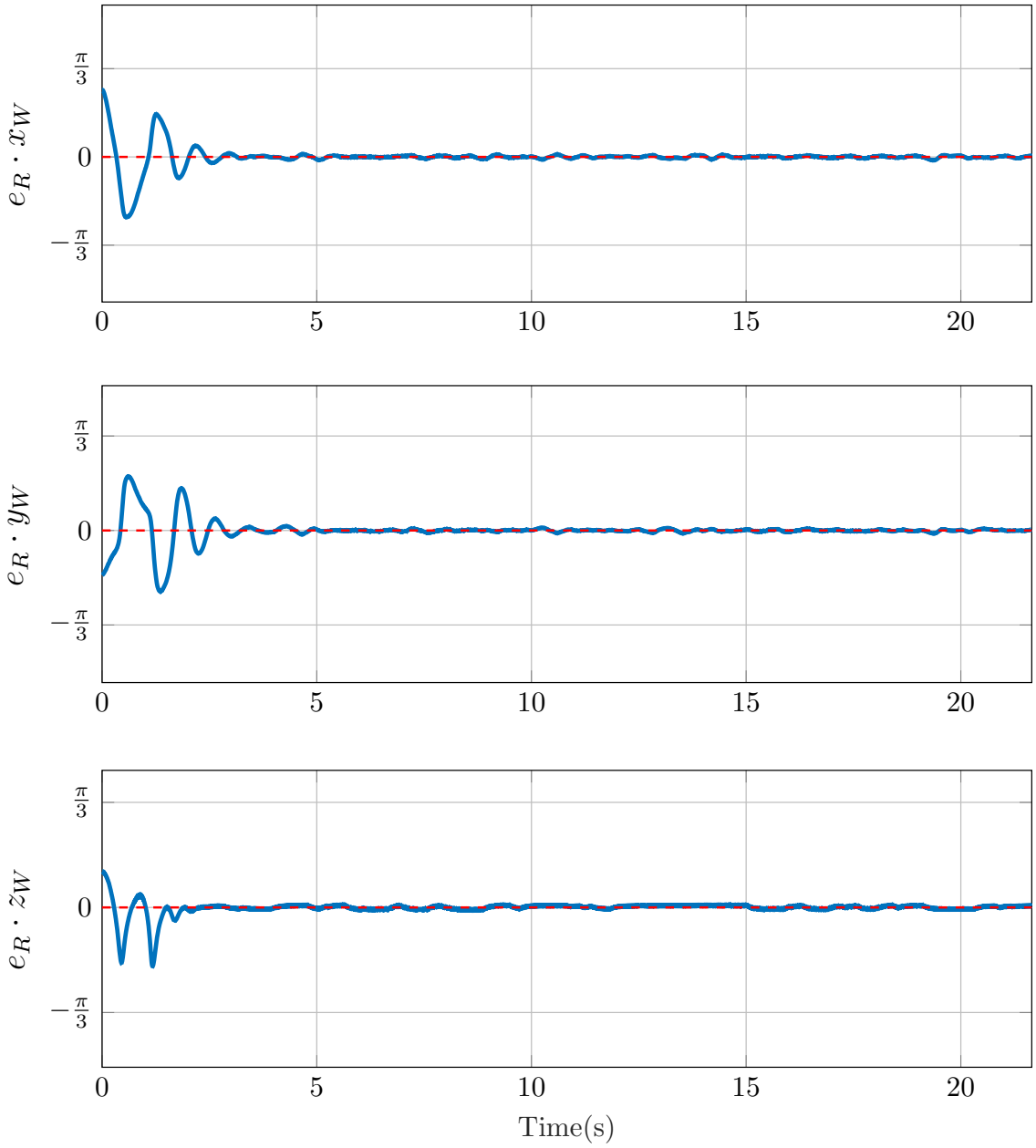

Figure 3.6: Time history of rotational errors along each axis

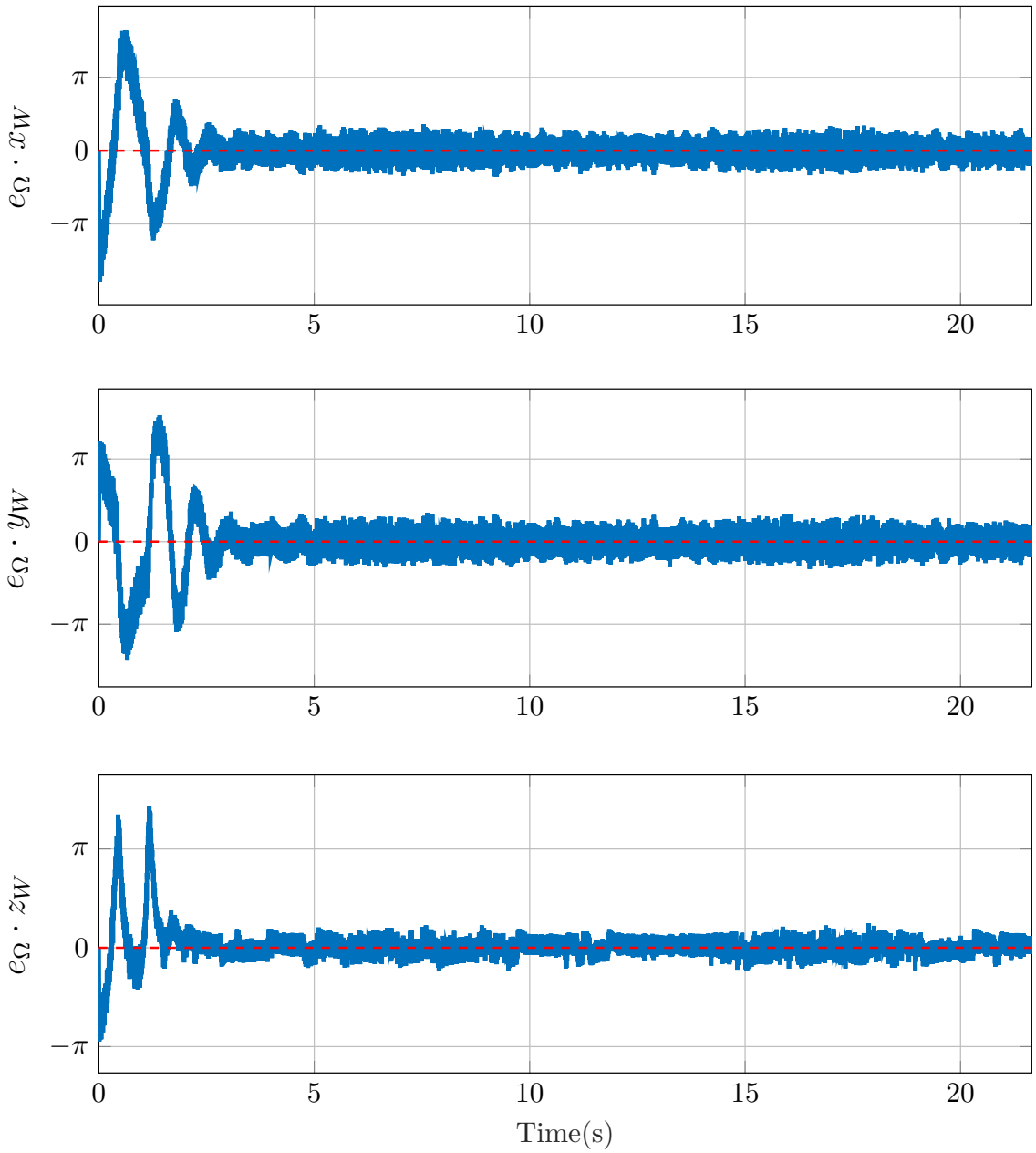

Figure 3.7: Time history of angular velocity errors along each axis

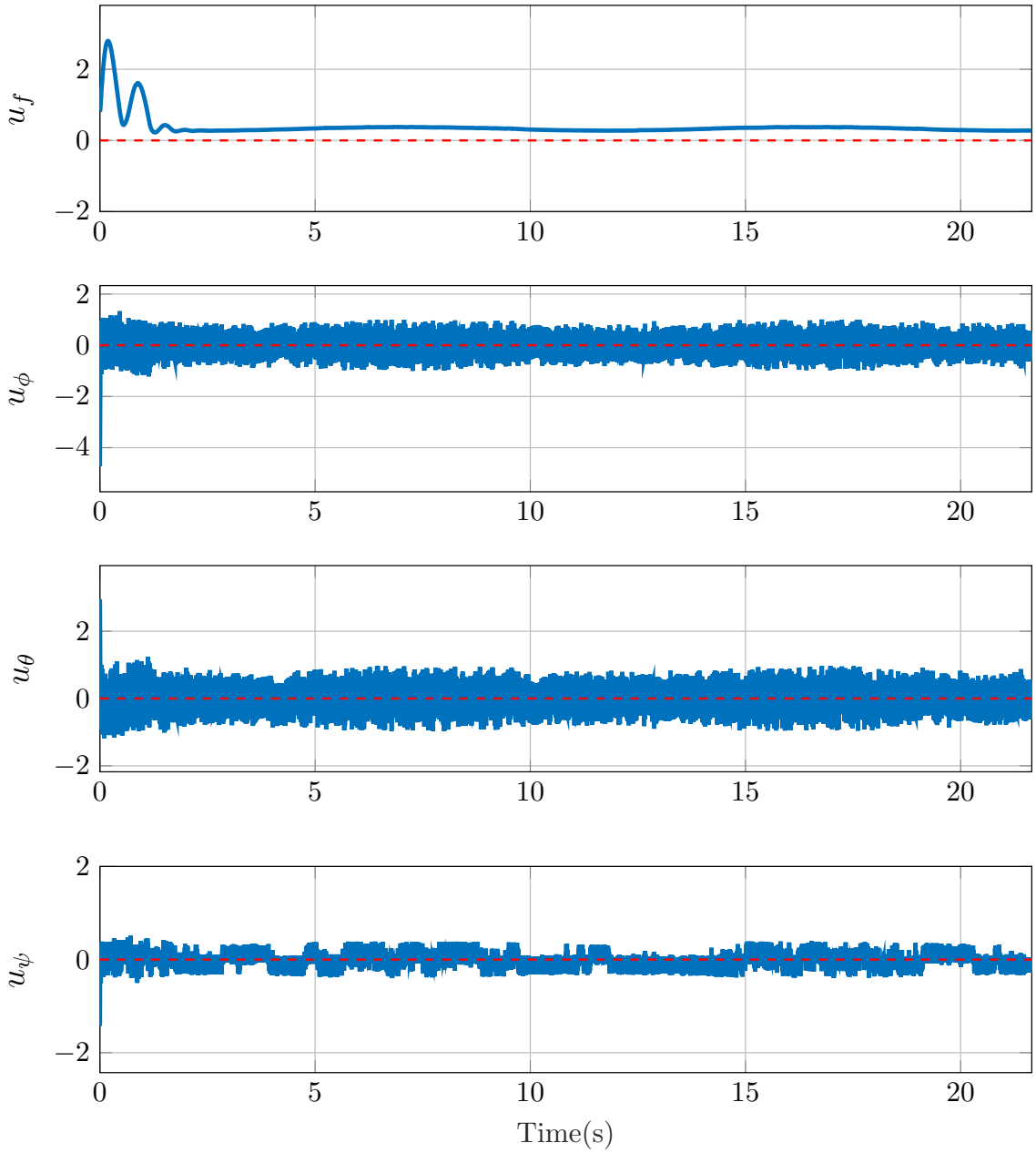

Figure 3.8: Time history of force and moment control inputs

3.3.2 Comparison with a cascaded PID controller

In the following simulations, we will compare the tracking performance of the geometric controller and a cascaded PID controller. The cascaded structure for the PID is comprised of three separate PID controllers, namely a position controller, attitude controller and angular rate controller. This structure does not account for any feedforward terms in the control design except for a thrust bias to account for its own weight during hover.

The following conclusions are drawn from the simulations of two different scenarios. The reference trajectories are defined as circular trajectories with a radius of 3 m and an angular velocity of $\frac{2}{3}$ rad/s on the Y-Z plane (Fig. 3.9) in the first case and on the X-Y plane (Fig. 3.11) in the second. In both scenarios, the PID controller structure performs poorly and fails to converge to the desired trajectory, where as the geometric controller design achieves convergence on the trajectory tracking errors. This is mainly attributed to the presence of acceleration and angular moment feedforward terms in the structure of the geometric controller. Faster convergence is also achieved as the orientation errors are defined using $SO(3)$, rather than in $\mathbb{R}^3$, allowing the quadrotor to orient itself arbitrarily without being affected by the gimbal-lock phenomenon.

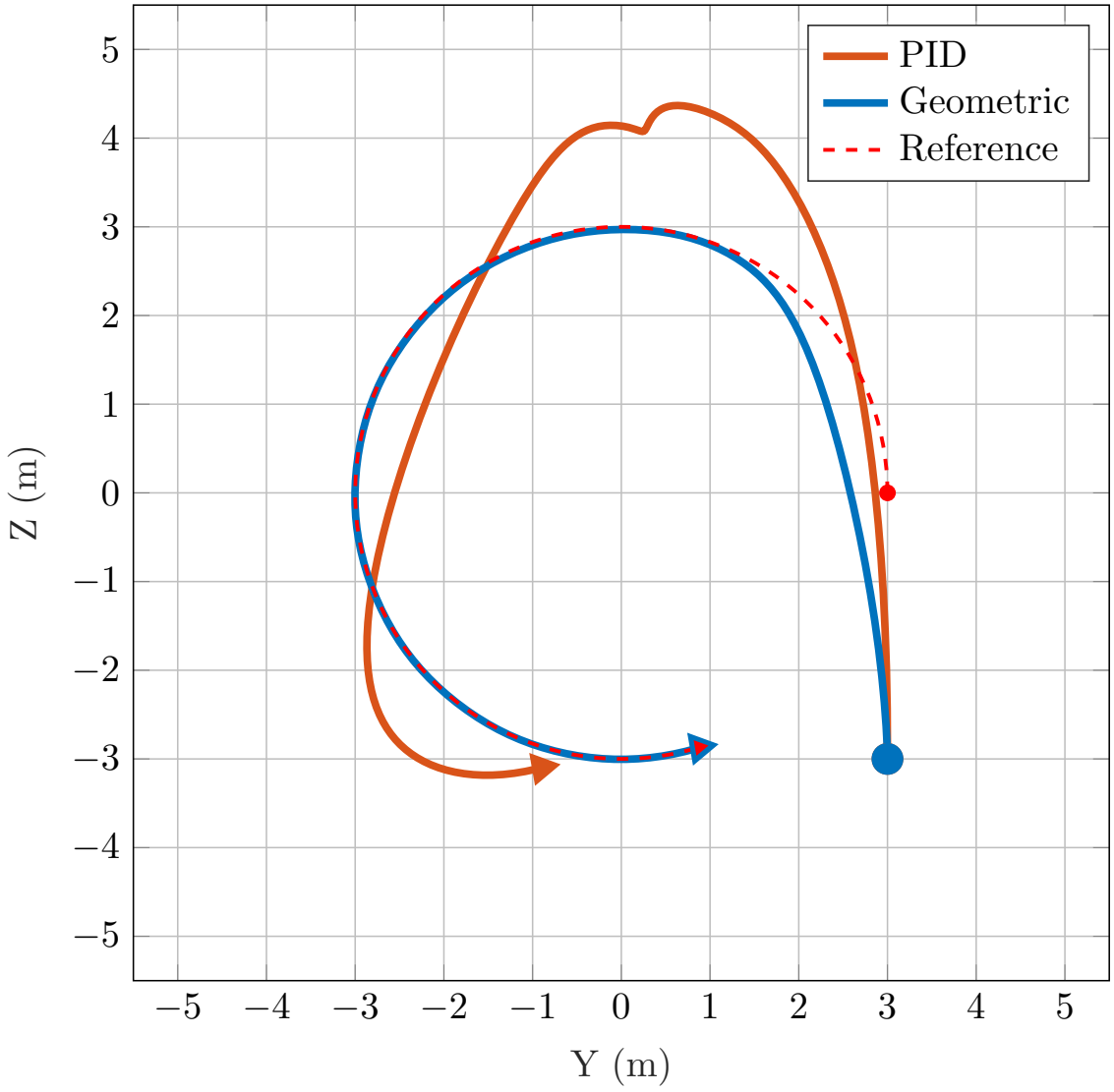

Figure 3.9: Controller performance with reference trajectory in Y-Z plane

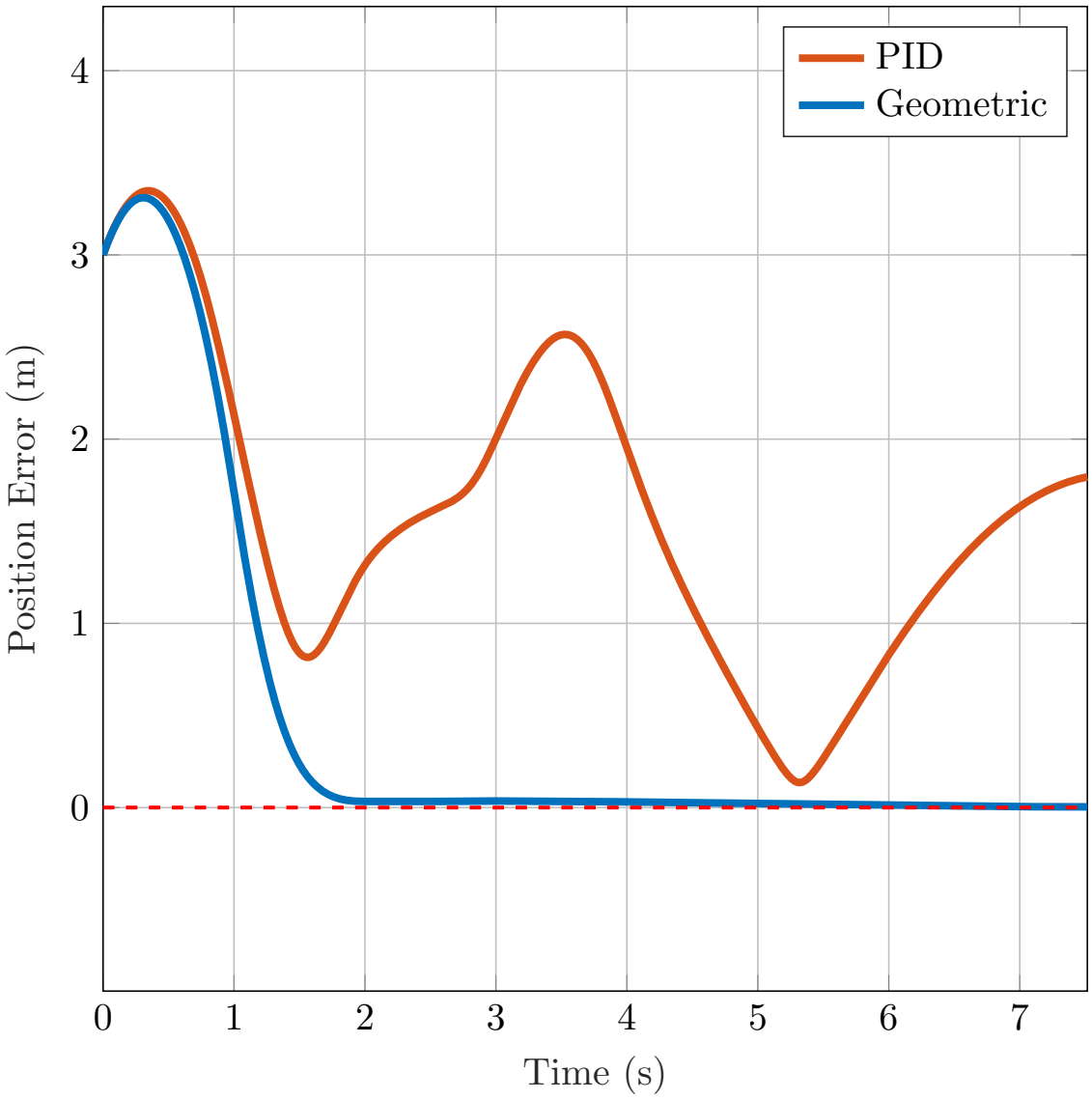

Figure 3.10: Time history of magnitude of the position errors given by Fig. 3.9

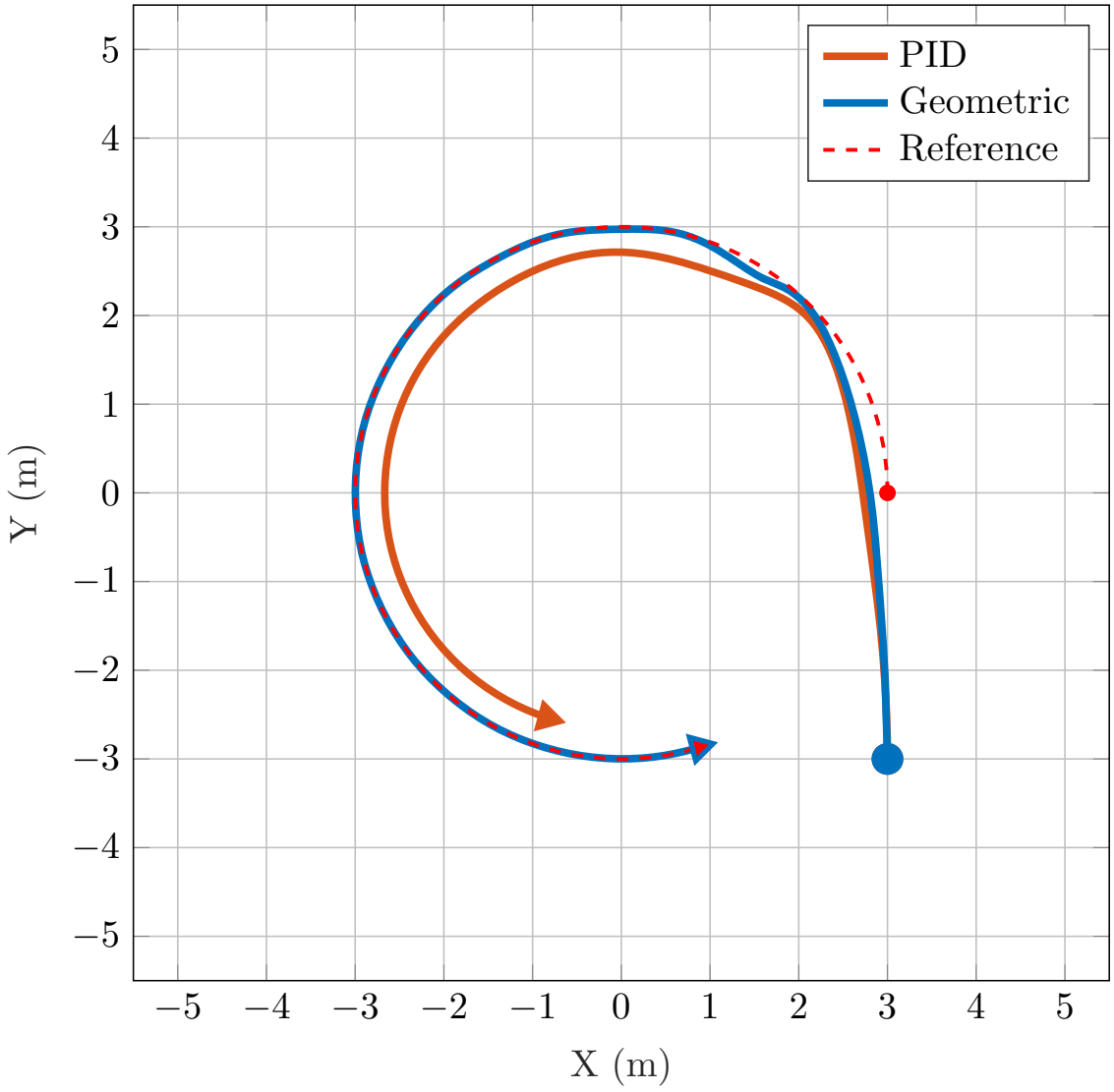

Figure 3.11: Controller performance with reference trajectory in X-Y plane

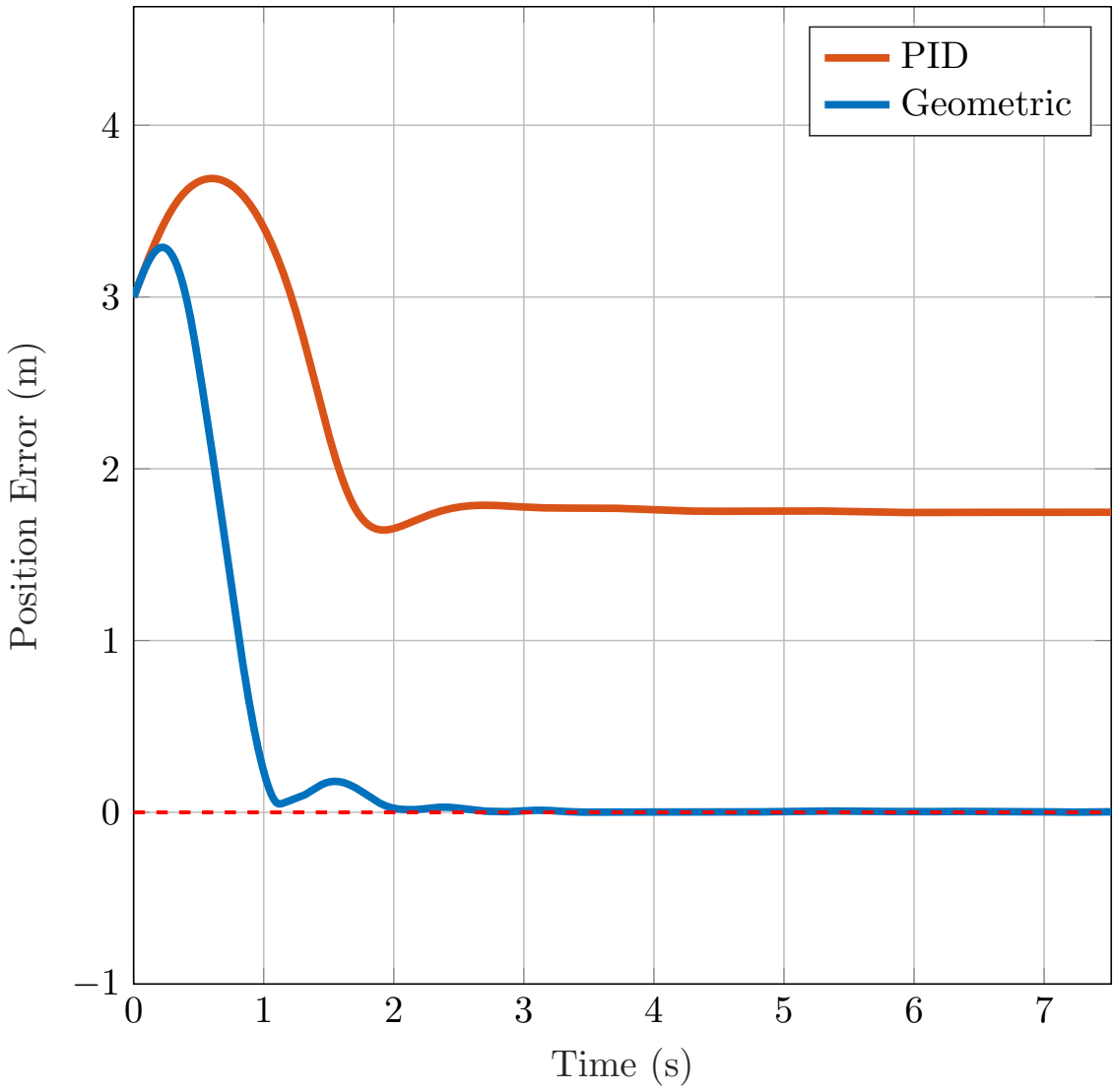

Figure 3.12: Time history of magnitude of the position errors given by Fig. 3.11

Chapter 4

Piecewise Bézier Curve Trajectory Generation

Optimization during trajectory generation involves the minimization of some functional, such as final time, distance or control effort. This results in the construction of a reference path, which achieves the objective in minimum time, distance or control input respectively.

While trajectory generation is a well explored area of research, only few methods exist for tackling this problem in obstacle-rich environments for vehicles with complex nonlinear dynamics such as quadrotors. Although motion planning methods such as RRT* [26] can arguably find the distance optimal solution while satisfying dynamic constraints, this problem is extremely computationally inefficient as the algorithm finds the optimal solution to every point in the workspace while propagating the tree in a six dimensional manifold.

The optimal solutions for the trajectory are obtained as control points for a time parameterized Bernstein polynomial basis. This allows for easy collision checking procedures of the solution with obstacles in the environment as seen from Mehdi et al. [27]. Convex hull properties of Bézier curves can be used to satisfy spatial and temporal separation constraints for multi-agent trajectory generation as shown in Choe et al. [28].

In this chapter we will describe optimization methods to generate minimum snap piecewise Bézier curves. Firstly, we will discuss some important properties of Bézier curves and why such a representation of a polynomial is useful in robotics applications. In later sections, we will define a two-step optimization problem as, *(i)* time-allocation along each flight segment and, *(ii)* feasibility-based total flight time optimization. Later in the chapter, we will discuss the simulation results from a *MATLAB* implementation of this optimization problem.

4.1 Problem Formulation

Heuristic approaches applied to sampling-based motion planning algorithms such as Informed RRT* [29] are excellent at generating distance optimal solutions very quickly. But the solution obtained using such an approach are dynamically infeasible piecewise linear paths.

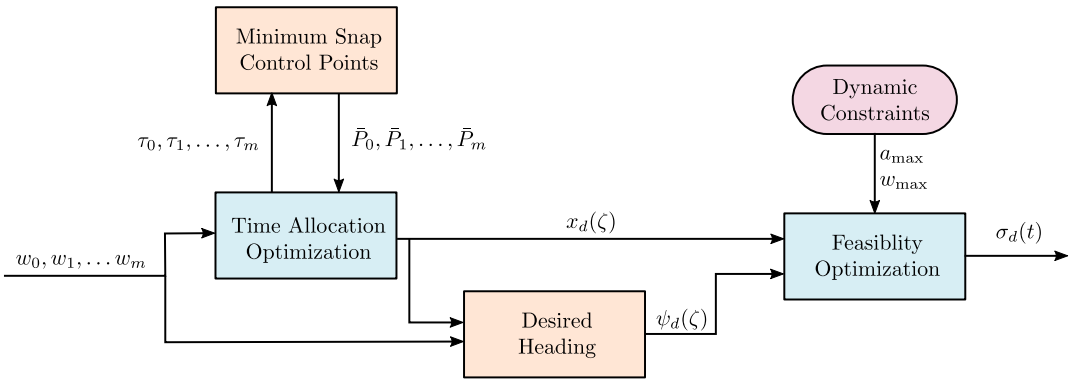

Figure 4.1: Bézier curve trajectory generation structure

In order to improve the quality of the solution, we propose an optimization procedure (see Fig. 4.1) to minimize snap and ensure dynamic feasibility from a set of waypoints obtained from a motion planning procedure. Minimizing snap of a trajectory gives the vehicle an essence of *gracefulness* in its motion behavior as the required control inputs are continuous for such a reference. It is assumed that the motion planner can obtain collision-free waypoints prior to the trajectory generation procedure.

The feasible waypoints are defined as

$$\mathbf{w} = \begin{bmatrix} w_0 & w_1 & w_2 & \dots & w_m \end{bmatrix}^T. \tag{4.1}$$

where $w_i \in \mathbb{R}^3$ is the i^{th} desired position in x, y and z coordinates.

The optimization process will generate time parameterized piecewise Bézier curves with desired heading angle of the quadrotor. The solution trajectory can be defined as

$$\sigma_d(t) = \begin{cases} \begin{bmatrix} x_{0,d}(t)^T & \psi_{0,d}(t) \end{bmatrix}^T & t_0 \leq t < t_1, \\ \begin{bmatrix} x_{1,d}(t)^T & \psi_{1,d}(t) \end{bmatrix}^T & t_1 \leq t < t_2, \\ \vdots & \\ \begin{bmatrix} x_{m-1,d}(t)^T & \psi_{m-1,d}(t) \end{bmatrix}^T & t_{m-1} \leq t < t_m, \end{cases} \tag{4.2}$$

where $x_{i,d}$ is the time parametrized 3D Bézier curve, and $\psi_{i,d}$ is the heading angle time parametrization for the i^{th} segment of the trajectory.

4.2 Properties of Bézier Curves

A Bézier curve, as shown in Fig 4.2, is a parametric polynomial curve defined by a set of control points over the interval $[0, 1]$ as

$$p(\zeta) = \sum_{n=0}^{N} \bar{p}_n b_n^N(\zeta) \qquad \zeta \in [0, 1], \tag{4.3}$$

where the $b_n^N(\zeta)$ is the Bernstein polynomial basis function given by

$$b_n^N(\zeta) = \binom{N}{n} (1-\zeta)^{N-n} \zeta^n \qquad \zeta \in [0, 1], \tag{4.4}$$

where ζ is the parameter variable, N is the order of the polynomial, and $\bar{p}$ are the control points.

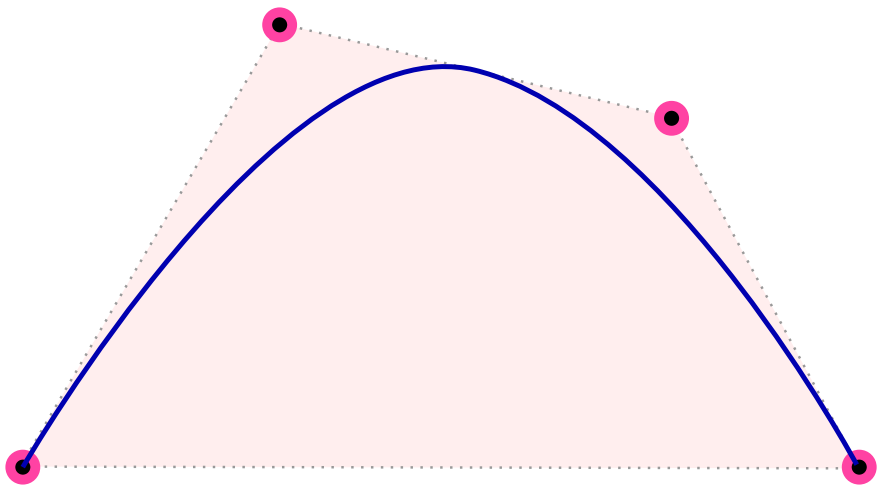

Figure 4.2: Cubic Bézier curve and the control polygon

Bézier curves have the following useful properties,

(i) An n^{th} order Bézier curve defined using $N+1$ control points. The first and last control points, also known as anchor points lie on the curve such that

$$p(0) = \bar{p}_0, \tag{4.5}$$

$$p(1) = \bar{p}_N, \tag{4.6}$$

whereas, the curves never pass through the intermediate control points.

(ii) The Bézier curve lies completely within the convex hull defined by the control points.

(iii) The differentiation of a Bézier curve can be represented as a Bézier curve. For example, the velocity, acceleration, jerk, and snap Bézier curves of are defined by the following

control points

$$\bar{v}_n = N(\bar{p}_{n+1} - \bar{p}_n) \qquad n = 0, 1, \ldots, N-1, \tag{4.7}$$

$$\bar{a}_n = N(N-1)(\bar{p}_{n+2} - \bar{p}_n) \qquad n = 0, 1, \ldots, N-2, \tag{4.8}$$

$$\bar{j}_n = N(N-1)(N-2)(\bar{p}_{n+3} - \bar{p}_n) \qquad n = 0, 1, \ldots, N-3, \tag{4.9}$$

$$\bar{s}_n = N(N-1)(N-2)(N-3)(\bar{p}_{n+4} - \bar{p}_n) \qquad n = 0, 1, \ldots, N-4. \tag{4.10}$$

In particular, it is interesting to note that the Bézier curves are tangential to the control polygons at the anchor points.

(iv) Bézier curves can be obtained through a change of basis from monomials as

$$p(\zeta) = \begin{bmatrix} 1 & \zeta & \zeta^2 & \ldots & \zeta^N \end{bmatrix} M\bar{p}, \tag{4.11}$$

where M is the blending matrix. M is characterized by the binomial coefficients of expansion. For example, a 3^{rd} order Bézier curve can be mapped to a 3^{rd} order polynomial using the following blending matrix

$$p(\zeta) = \begin{bmatrix} 1 & \zeta & \zeta^2 & \zeta^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} \bar{p}_0 \\ \bar{p}_1 \\ \bar{p}_2 \\ \bar{p}_3 \end{bmatrix}. \tag{4.12}$$

4.2.1 Piecewise Bézier Curves

Bézier curves can be concatenated such that they share an anchor point and satisfy derivative matching to ensure some form of continuity across the curve, as shown in Fig 4.3. Such a

composition of Bézier curves can be defined as

$$\mathbf{p}(t) = \begin{cases} \sum_{n=0}^{N} \bar{p}_{0,n} b_n^N \left(\frac{t}{t_1 - 0} \right) & 0 \leq t < t_1, \\ \sum_{n=0}^{N} \bar{p}_{1,n} b_n^N \left(\frac{t}{t_2 - t_1} \right) & t_1 \leq t < t_2, \\ \vdots & \\ \sum_{n=0}^{N} \bar{p}_{m-1,n} b_n^N \left(\frac{t}{t_m - t_{m-1}} \right) & t_{m-1} \leq t < t_m. \end{cases} \tag{4.13}$$

The problems explored in this research will have $\mathcal{C}^4$ continuity, implying that all the derivatives up to and including snap will be matched at the shared anchor points.

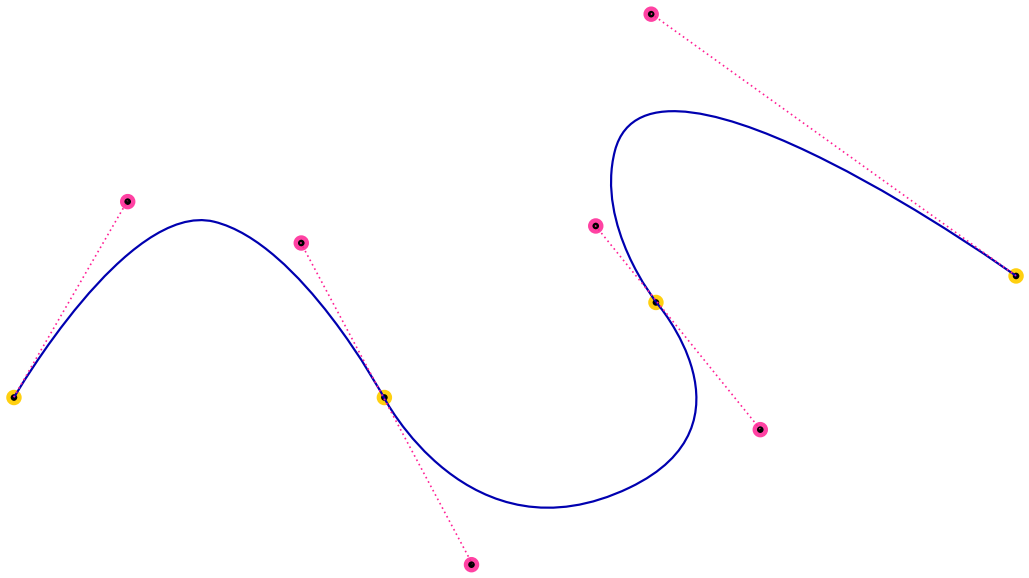

Figure 4.3: Piecewise Bézier curve with velocity vectors at anchor points

4.3 Cost Function

As we saw in Section 2.3, the trajectory for the quadrotors was given by independent flat outputs namely, the three translational coordinates and the heading angle. In the following formulation the Bézier curves are generated independently for each of the output dimensions. Each Bézier curve segment in the composite curve is given by

$$p\left(\zeta\right) = \sum_{n=0}^{N} \bar{p}_n b_n^N\left(\zeta\right), \tag{4.14}$$

such that

$$\zeta = \frac{t}{\tau}, \tag{4.15}$$

where τ is the time allocated for the particular segment.

In order to generate *graceful* trajectories, we minimize the integral of square norm of snap. This follows from the idea expounded in [5], that the flat outputs appear as the fourth derivatives in the control input equations. We will define the cost function for minimizing the r^{th} derivative as

$$J(\tau) = \min_{\tau} \quad \int_0^{\tau} \left(\frac{d^r}{dt^r}\, p\left(\frac{t}{\tau}\right)\right)^2 dt, \tag{4.16}$$

where J is the cost and $p(\cdot)$ is the Bézier curve segment. Changing the Bernstein parameter t/τ to ζ, it follows that

$$t = \tau\zeta, \tag{4.17}$$

which implies

$$dt = \tau d\zeta, \tag{4.18}$$

$$\frac{d}{dt^r} = \frac{1}{\tau^r}\frac{d}{d\zeta^r}. \tag{4.19}$$

Now the expression in (4.16) becomes

$$J(\tau) = \min_{\tau} \quad \frac{1}{\tau^{2r-1}} \int_0^1 \left(\frac{d^r}{d\zeta^r}\, p\left(\zeta\right) \right)^2 d\zeta. \tag{4.20}$$

It is important to note that the time allocation variable appears outside of the integral, implying that the control points that define the minimum snap trajectory for this segment are independent of the time allocated to that segment.

The r^{th} derivative of a Bézier curve can be represented as

$$\frac{d^r}{d\zeta^r} p(\zeta) = \frac{N!}{(N-r)!} \sum_{n=0}^{N} \Delta^r \bar{p}_n b_n^{N-r}(\zeta), \tag{4.21}$$

where Δ^r is the r^{th} forward difference operator given by

$$\Delta^r \bar{p}_n = \bar{p}_{n+r} - \bar{p}_n. \tag{4.22}$$

Expressing the Bézier curve derivative from (4.21) in matrix form, we obtain

$$\frac{d^r}{d\zeta^r} p(\zeta) = B_r(\zeta)^T D_r \bar{P}, \tag{4.23}$$

where $B_r(\cdot)$ and $\bar{P}$ are defined as

$$B_r(\zeta) = \begin{bmatrix} b_0^{N-r}(\zeta) & b_1^{N-r}(\zeta) & \dots & b_{N-r}^{N-r}(\zeta) \end{bmatrix}^T, \tag{4.24}$$

$$\bar{P} = \begin{bmatrix} \bar{p}_0 & \bar{p}_1 & \dots & \bar{p}_N \end{bmatrix}^T. \tag{4.25}$$

D_r is the forward difference operator as a $((N-r)\times N)$ matrix with '-1's on the main diagonal (i.e $i=j$) and '1's on the r^{th} superdiagonal (i.e $i+r=j$) for $r>0$ defined as

$$D_r = \begin{bmatrix} -1 & & & 1 & & \\ & \ddots & & & \ddots & \\ & & -1 & & & 1 \end{bmatrix}. \tag{4.26}$$

The matrix formulation of the cost function from (4.20) and (4.24) can be evaluated as

$$J(\tau) = \min_{\tau} \quad \frac{1}{\tau^{2r-1}} \int_0^1 \left(\bar{P}^T D_r^T B_r(\zeta) B_r(\zeta)^T D_r \bar{P}\right) d\zeta \tag{4.27}$$

$$J(\tau) = \min_{\tau} \quad \frac{1}{\tau^{2r-1}} \bar{P}^T D_r^T \left(\int_0^1 \left(B_r(\zeta) B_r(\zeta)^T\right) d\zeta \right) D_r \bar{P}. \tag{4.28}$$

The integral over the tensor product of Bernstein polynomials can be evaluated as

$$H_r = \int_0^1 \left(B_r(\zeta) B_r(\zeta)^T\right) d\zeta. \tag{4.29}$$

The $(i,j)^{th}$ value of the tensor product can be independently integrated as

$$H_r^{i,j} = \int_0^1 \left(b_i^{N-r}(\zeta) b_j^{N-r}(\zeta)\right) d\zeta \tag{4.30}$$

$$H_r^{i,j} = \binom{N-r}{i} \binom{N-r}{j} \int_0^1 \left((1-\zeta)^{2N-r-(i+j)} \zeta^{i+j}\right) d\zeta \tag{4.31}$$

$$H_r^{i,j} = \frac{\binom{N-r}{i}\binom{N-r}{j}}{\binom{2N-r}{i+j}} \frac{1}{2N-r+1}. \tag{4.32}$$

As a result the cost function (4.28) can be expressed purely as following matrix product

$$J(\tau) = \min_{\tau} \quad \frac{1}{\tau^{2r-1}} \bar{P}^T D_r^T H_r D_r \bar{P}. \tag{4.33}$$

The cost function for m Bézier curve segments can be evaluated by concatenating (4.33)

along the diagonals as

$$\overbrace{J(\tau_0,\ldots,\tau_{m-1})}^{\mathbf{J}} = \min_{(\tau_0,\ldots,\tau_{m-1})} \overbrace{\begin{bmatrix} \bar{P}_0 \\ \vdots \\ \bar{P}_{m-1} \end{bmatrix}^T}^{\bar{\mathbf{P}}^{\mathbf{T}}} \overbrace{\begin{bmatrix} \frac{1}{\tau_0^{2r-1}} D_r^T H_r D_r & & \\ & \ddots & \\ & & \frac{1}{\tau_{m-1}^{2r-1}} D_r^T H_r D_r \end{bmatrix}}^{\mathbf{Q_r}} \overbrace{\begin{bmatrix} \bar{P}_0 \\ \vdots \\ \bar{P}_{m-1} \end{bmatrix}}^{\bar{\mathbf{P}}}. \tag{4.34}$$

The cost function for the entire trajectory can then be expressed as

$$\mathbf{J} = \min_{\boldsymbol{\tau}} \quad \bar{\mathbf{P}}^T \mathbf{Q_r} \bar{\mathbf{P}}. \tag{4.35}$$

where $\boldsymbol{\tau} = (\tau_0, \tau_1, \ldots, \tau_{m-1})$ is the associated segment-wise time allocation variable.

4.3.1 Constraints

The constraints which are required to be satisfied during the optimization of the piecewise Bézier curve are:

(i) *Fixed anchor points.*

(ii) *Fixed derivatives at the initial and final location of the entire trajectory based on required initial and final states.*

(iii) *Matched derivatives at all intermediate anchor points until the r^{th} derivative to ensure $\mathcal{C}^r$ continuity of the curve.*

In matrix form these constraints can be described as

$$A\bar{P} = d, \tag{4.36}$$

such that

$$A = \left[\begin{array}{c} A_0 \\ \hline A_1 \end{array}\right] = \left[\begin{array}{c} B_0^T(0)D_0 \\ \vdots \\ B_r^T(0)D_r \\ \hline B_0^T(1)D_0 \\ \vdots \\ B_r^T(1)D_r \end{array}\right], \tag{4.37}$$

where A is the Bernstein component of each of the derivatives at the anchor locations, and

$$d = \left[\begin{array}{c} d_0 \\ \hline d_1 \end{array}\right] = \left[\begin{array}{c} \frac{d^0}{dt^0}p(0) \\ \vdots \\ \frac{d^r}{dt^r}p(0) \\ \hline \frac{d^0}{dt^0}p(1) \\ \vdots \\ \frac{d^r}{dt^r}p(1) \end{array}\right], \tag{4.38}$$

where d consists of the values for the fixed or matched derivative values at the anchor locations.

The constraints presented above hold only for one Bézier segment. The constraint entries are filled out in block-diagonal fashion for the entire trajectory as

$$\overbrace{\begin{bmatrix} A_0 & & & & \\ A_1 & & & & \\ A_1 & -A_0 & & & \\ & A_1 & & & \\ & \vdots & \ddots & & \\ & & & & A_1 \end{bmatrix}}^{\mathbf{A}} \overbrace{\begin{bmatrix} \bar{P}_0 \\ \bar{P}_1 \\ \vdots \\ \vdots \\ \vdots \\ \bar{P}_{m-1} \end{bmatrix}}^{\bar{\mathbf{P}}} = \overbrace{\begin{bmatrix} d_{0,0} \\ d_{0,1} \\ 0 \\ d_{1,1} \\ \vdots \\ d_{m-1,1} \end{bmatrix}}^{\mathbf{d}}, \tag{4.39}$$

where $d_{k,l}$ is the l^{th} derivative values for the k^{th} segment. The full constraint equation can be expressed as

$$\mathbf{A}\bar{\mathbf{P}} = \mathbf{d}. \tag{4.40}$$

4.3.2 Unconstrained representation

As constraint equations are defined by large sparse matrices, the nonlinear optimization process is hard and often leads to the generation of singular or badly conditioned matrices. In order to avoid such failure scenarios and make the optimization process faster we can represent the constraints directly in the cost function through matrix inversion as described in [30]. The optimal control points for a given time allocation can be directly extracted from the unconstrained representation of the cost function.

In order to invert the constraint equation (4.40) we can calculate the left psuedoinverse of the A matrix as,

$$\bar{\mathbf{P}} = \mathbf{A}^{+}\mathbf{d}. \tag{4.41}$$

The inverse of A can also be evaluated if the matrix is square. However, this is only possible when the order of the matrix is expressed as a function of the minimized derivative (i.e $n = 2r + 1$).

The cost function of the unconstrained representation can be expressed as

$$\mathbf{J} = \min_{\boldsymbol{\tau}} \quad \mathbf{d}^T \mathbf{A}^{+T} \mathbf{Q_r} \mathbf{A}^{+} \mathbf{d}. \tag{4.42}$$

4.4 Optimal Control Points

The $\mathbf{d}$ column vector consists of anchor locations, matching conditions at the joints of the Bézier segments and the unknown derivative values. Rearranging $\mathbf{d}$ by separating the known values from the unknown derivatives using a permutation matrix $\mathbf{M}$ as

$$\mathbf{M}\mathbf{d} = \begin{bmatrix} \mathbf{d_k} \\ \hline \mathbf{d_u} \end{bmatrix}, \tag{4.43}$$

where $\mathbf{d_k}$ are the known derivative values and $\mathbf{d_u}$ are the unknown values. From the inversion of permutation matrix, we obtain

$$\mathbf{d} = \mathbf{M}^T \begin{bmatrix} \mathbf{d_k} \\ \hline \mathbf{d_u} \end{bmatrix}. \tag{4.44}$$

Following this permutation, the cost function (4.42) can be represented as

$$\mathbf{J} = \min_{\boldsymbol{\tau}} \quad \begin{bmatrix} \mathbf{d_k} \\ \hline \mathbf{d_u} \end{bmatrix}^T \overbrace{\mathbf{M}\mathbf{A}^{+T} \mathbf{Q_r} \mathbf{A}^{+} \mathbf{M}^T}^{\mathbf{I}} \begin{bmatrix} \mathbf{d_k} \\ \hline \mathbf{d_u} \end{bmatrix}. \tag{4.45}$$

where I is the intermediate matrix shown as the matrix multiplication of inner terms in the cost function. Partitioning the matrix to match the dimensions of $\mathbf{d_k}$ and $\mathbf{d_u}$ we obtain

$$\mathbf{J} = \min_{\boldsymbol{\tau}} \quad \begin{bmatrix} \mathbf{d_k} \\ \hline \mathbf{d_u} \end{bmatrix}^T \left[\begin{array}{c|c} \mathbf{I_{kk}} & \mathbf{I_{ku}} \\ \hline \mathbf{I_{uk}} & \mathbf{I_{uu}} \end{array}\right] \begin{bmatrix} \mathbf{d_k} \\ \hline \mathbf{d_u} \end{bmatrix}. \tag{4.46}$$

The variables can be separated through the matrix multiplication of the expression described

above as

$$\mathbf{J} = \min_{\boldsymbol{\tau}} \quad \mathbf{d_k}^T \mathbf{I_{kk}} \mathbf{d_k} + \mathbf{d_k}^T \mathbf{I_{ku}} \mathbf{d_u} + \mathbf{d_u}^T \mathbf{I_{uk}} \mathbf{d_k} + \mathbf{d_u}^T \mathbf{I_{uu}} \mathbf{d_u}. \tag{4.47}$$

We can obtain the optimal solution to the unknown derivatives $\mathbf{d_u^*}$, by evaluating the partial differential of the cost function with respect to $\mathbf{d_u}$ as

$$\frac{\partial \mathbf{J}}{\partial \mathbf{d_p}} = \mathbf{d_k}^T \mathbf{I_{ku}} + \mathbf{d_k}^T \mathbf{I_{uk}}^T + \mathbf{d_u^*}^T (\mathbf{I_{uu}}^T + \mathbf{I_{uu}}) = 0 \tag{4.48}$$

$$\mathbf{d_u^*} = -\left(\mathbf{I_{uu}} + \mathbf{I_{uu}}^T\right)^{-1} \left(\mathbf{I_{ku}}^T + \mathbf{I_{uk}}\right) \mathbf{d_k}. \tag{4.49}$$

Following this, the optimal control points for a given time allocation can be evaluated from (4.41) and (4.44) as

$$\bar{\mathbf{P}}^* = \mathbf{A}^+ \mathbf{M}^T \left[\begin{array}{c} \mathbf{d_k} \\ \hline \mathbf{d_u^*} \end{array}\right]. \tag{4.50}$$

4.5 Time Allocation Optimization

The configuration of the control points do not change regardless of the scaling in the time allocation as the time variable does not enter the integral cost for an independent Bézier segment. In order to discover the optimum configuration of control points, the sum of allocated times is constrained to unity as

$$\sum_{k=0}^{m-1} \tau_k = 1. \tag{4.51}$$

The new cost function defined under this constraint can be represented as

$$\mathbf{J} = \min_{\boldsymbol{\tau_n}} \quad \mathbf{d}^T \mathbf{A}^{+T} \mathbf{Q_r} \mathbf{A}^+ \mathbf{d}, \tag{4.52}$$

where $\boldsymbol{\tau}_\mathbf{n}$ is given by

$$\boldsymbol{\tau}_\mathbf{n} = \left(\tau_0, \tau_1, \ldots, \tau_{m-2}, 1 - \sum_{k=0}^{m-2} \tau_k \right), \tag{4.53}$$

such that every time allocation value is always positive.

4.6 Desired Heading

The desired heading is not involved in the optimization problem as yaw is a free variable and can be used to do other independent tasks such as vision tracking of objects or aerial manipulation tasks. In this research, we will design the yaw to simply face in the direction of desired travel. The desired heading at specified waypoints can either be obtained from the motion planning task or from the direction of velocity of the optimized trajectory.

After the heading angles have been specified at the anchor locations the minimum snap approach can be used to find the control points that describe the heading trajectory from equations (4.49) and (4.50) for the following cost function

$$J = \min_{\tau} \int_0^1 \left(\frac{d}{dt} \psi_p(\zeta) \right)^2 d\zeta, \tag{4.54}$$

where $\psi_p(\cdot)$ is the 1D Bézier curve for heading angles.

4.7 Feasibility Optimization

As the optimized trajectory is parametrized between $t \in [0, 1]$ irrespective of the scale of the solution the paths may not be flyable. A second optimization is performed to solve for the total optimal flight time that satisfies the dynamic constraints defined by the vehicle. A quadrotor has bounds on maximum flyable angular velocities and accelerations defined by its physical and sensing capabilities. The cost function $\mathbf{J_T}$, for total flight time can then be expressed by the following constrained optimization

$$\mathbf{J_T}(k) = \min_{k} \quad \mathbf{J}(k\boldsymbol{\tau}), \tag{4.55}$$

such that

$$\ddot{x} < a_{\max}, \tag{4.56}$$

$$\Omega < \omega_{\max}, \tag{4.57}$$

where $\mathbf{J}$ is defined by as the unconstrained cost function for the trajectory (4.42), k is the positive scalar value, $\ddot{x}$ and Ω are the acceleration and angular velocities of the trajectory obtained as a result of the differential flatness properties (Section 2.3), and a_{max} and ω_{max} are bounds defined by the vehicle.

4.8 Simulations

The following section provides the simulation results of the optimization problem defined in the previous sections. We will generate a minimum snap piecewise Bézier curve which passes through the waypoints defined as w_0, w_1, w_2, w_3 and w_4 as illustrated in Fig. 4.4.

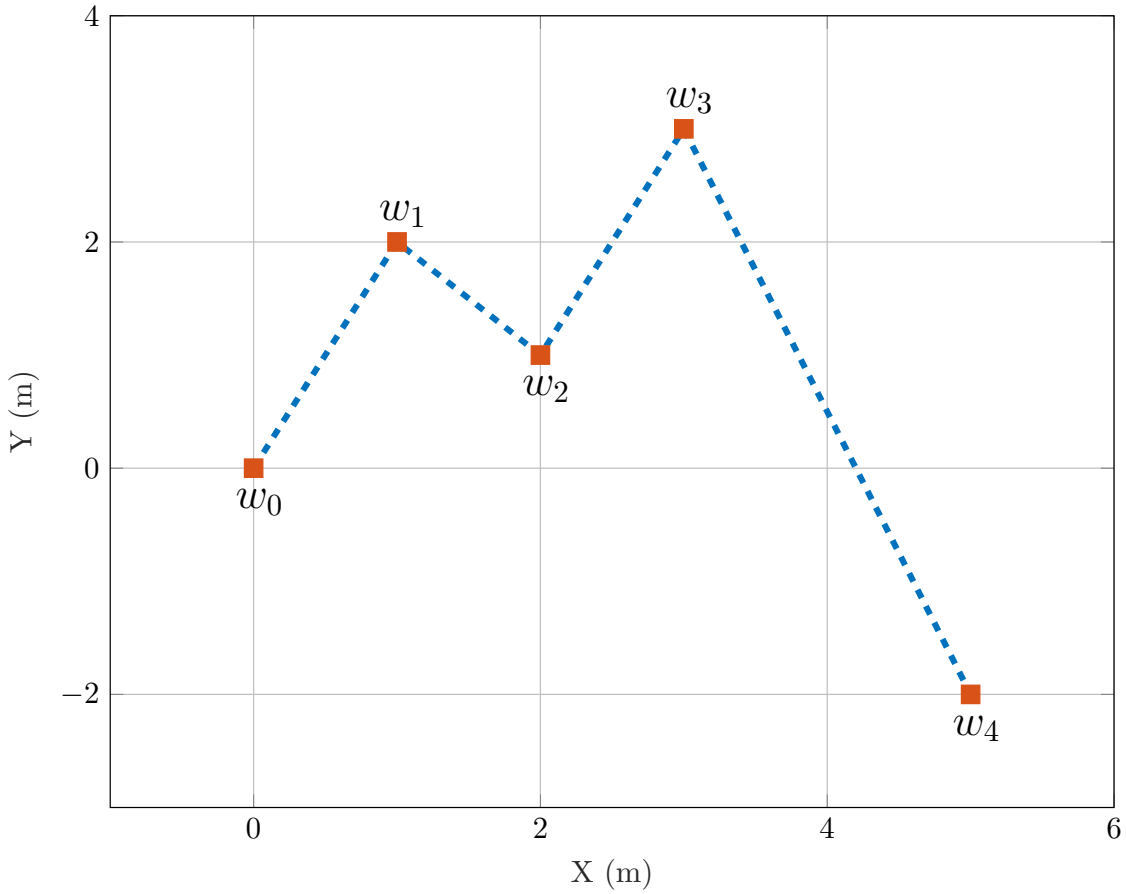

Figure 4.4: Waypoints for trajectory generation (red squares)
Waypoints (red squares) and connecting segments (dotted blue lines) are collision free.

Following the approach described in the section 4.5, the time allocated for each segment is optimized. The desired trajectories at different intervals along the optimization process is shown in Fig. 4.5.

The costs and time allocations associated with the curves generated in Fig. 4.5 is described in the following table,

iter	Cost	τ_0 (s)	τ_1 (s)	τ_2 (s)	τ_3 (s)
0	8.57×10^9	0.2500	0.2500	0.2500	0.2500
4	6.37×10^9	0.2417	0.2250	0.2542	0.2792
8	5.60×10^9	0.2250	0.1931	0.2546	0.3273
12	4.66×10^9	0.2500	0.1889	0.2074	0.3537
16	4.44×10^9	0.2806	0.1707	0.1746	0.3741
32	$\mathbf{4.38 \times 10^9}$	**0.2837**	**0.1574**	**0.1949**	**0.3640**

Table 4.1: Time allocation and total cost

Clearly, the trajectory speeds up in the smaller segments and slows down at the end segments where it needs to leave from or arrive to a complete stop. The optimal control points and the resulting minimum snap Bézier curve obtained from the trajectory generation approach is illustrated in Fig. 4.6.

Next, we define the desired heading angle at the waypoints as the direction of the velocity at these points (see Fig. 4.7). The desired position and heading angle trajectories obtained from the optimization is illustrated in Fig. 4.8. As the trajectory is parameterized for $t \in [0, 1]$, the feasibility optimization is performed to satisfy dynamic constraints as described in section 4.7. The scaling factor obtained from the feasibility optimization for the Crazyflie 2.0 quadrotor model is

$$k^* = 5.9262 \implies t \in \Big[0, \quad 5.9262\Big] .$$

The force and moment control inputs required to follow the optimized trajectory remain well bounded and continuous as seen from Fig. 4.9.

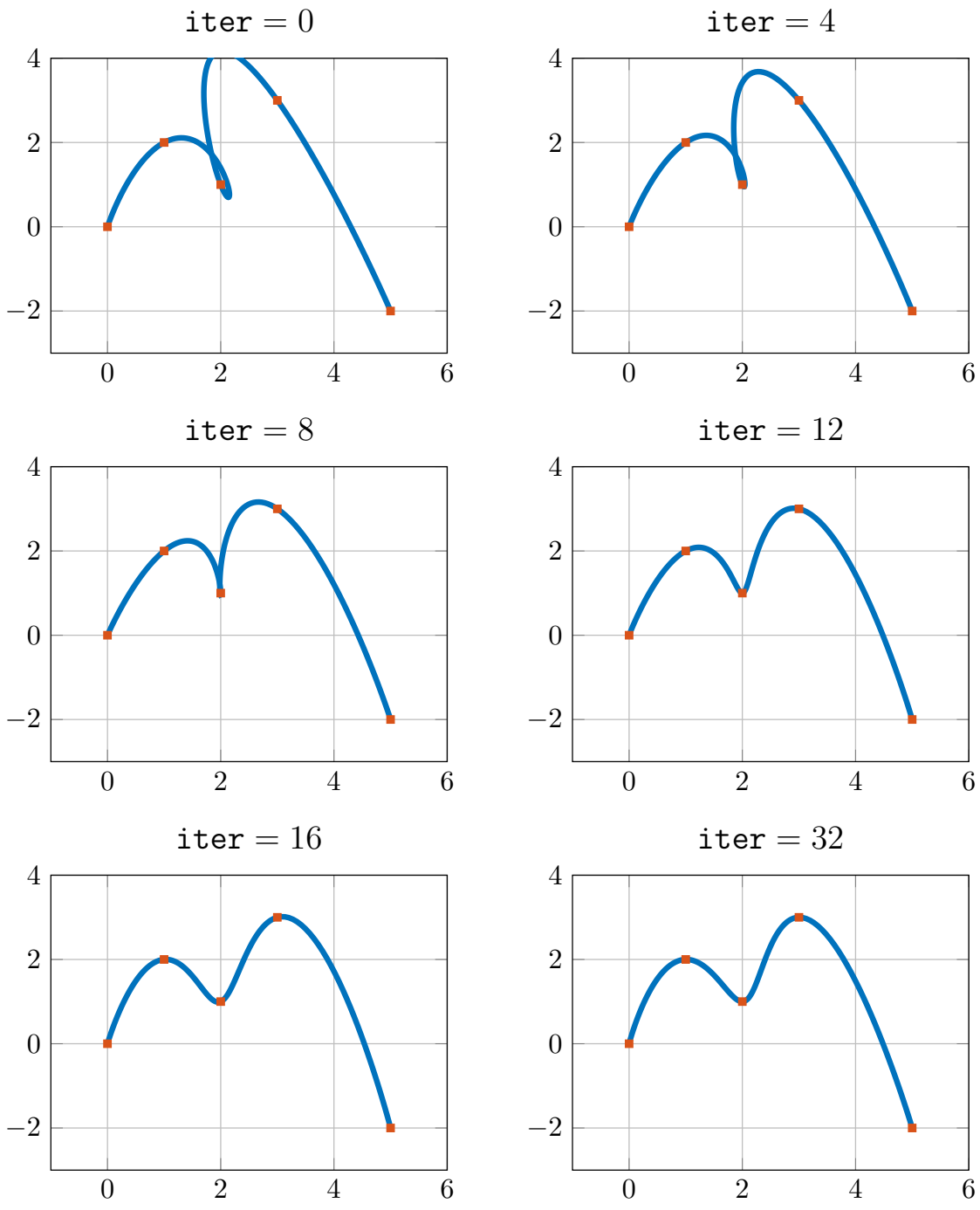

Figure 4.5: Time allocation optimization at different iteration intervals

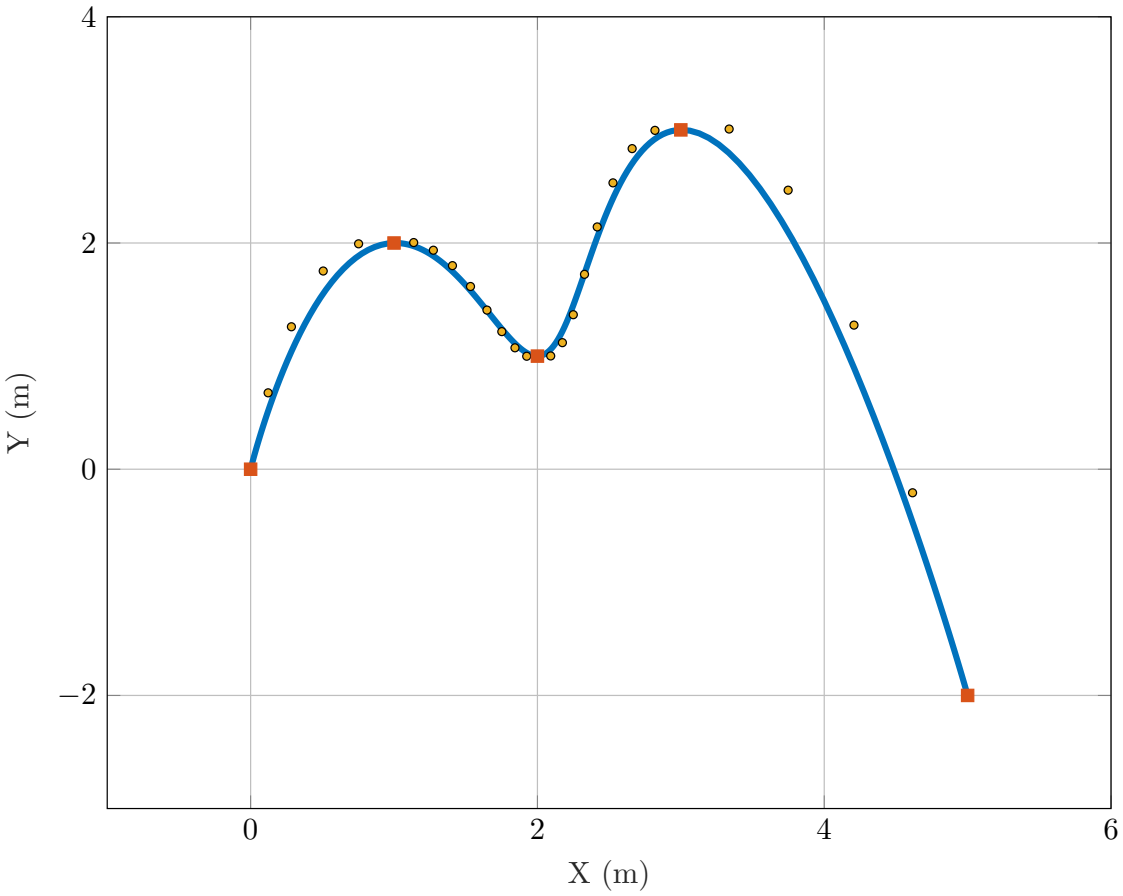

Figure 4.6: Minimum snap piecewise Bézier curve.
Bézier curve (blue), anchor points (red) and the optimal control points (yellow).

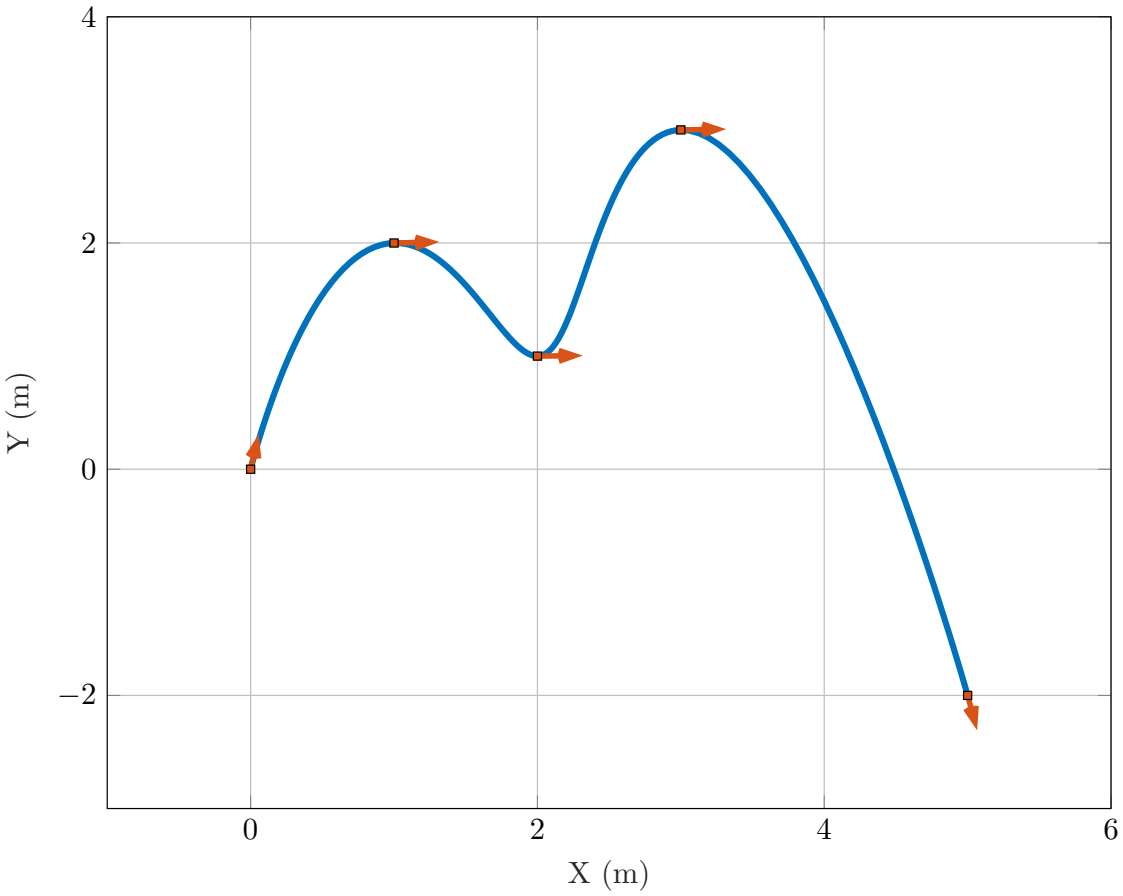

Figure 4.7: Desired heading defined at the waypoints
Heading orientation (red quivers) defined by the tangential velocity vector.

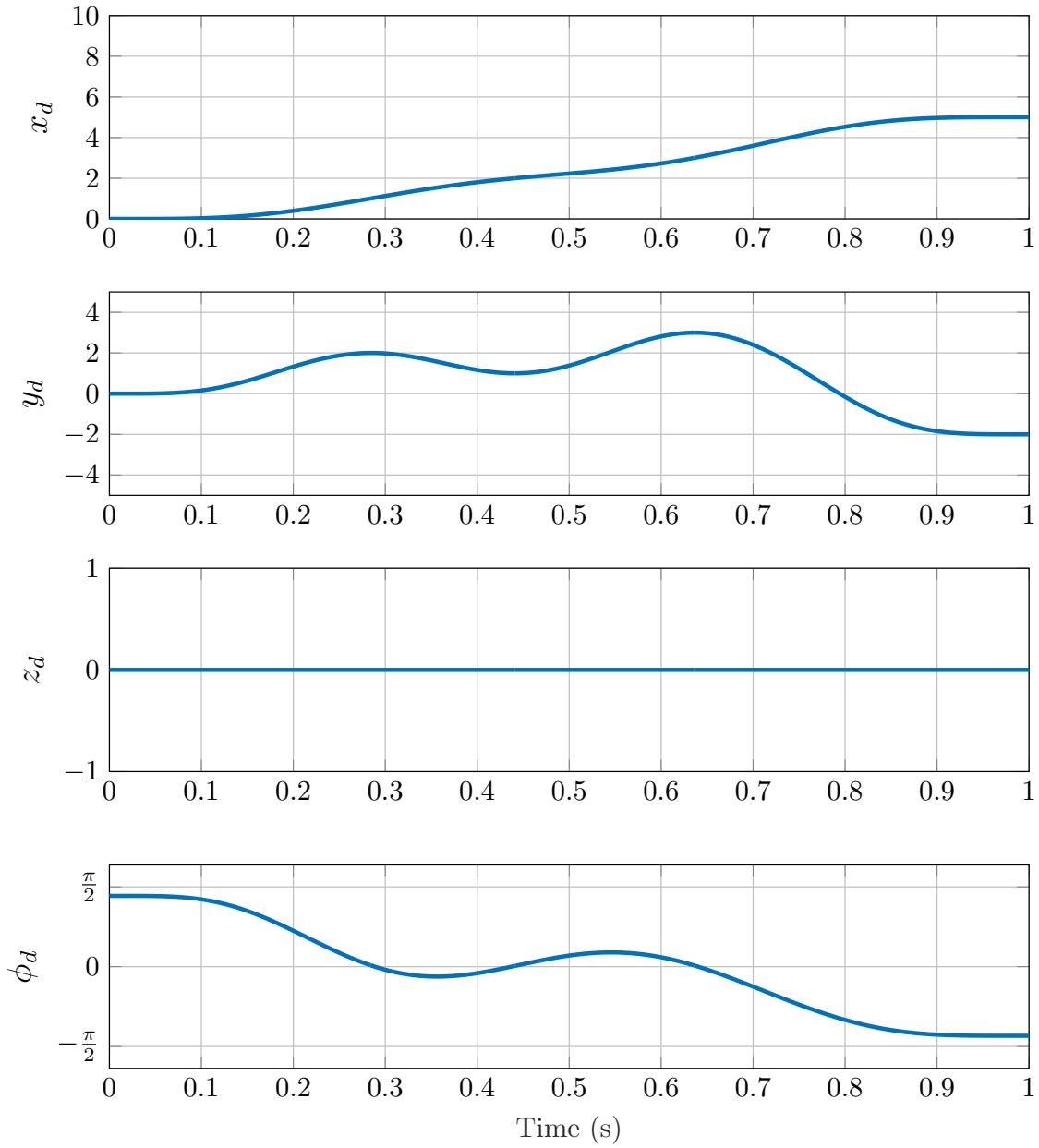

Figure 4.8: Flat outputs

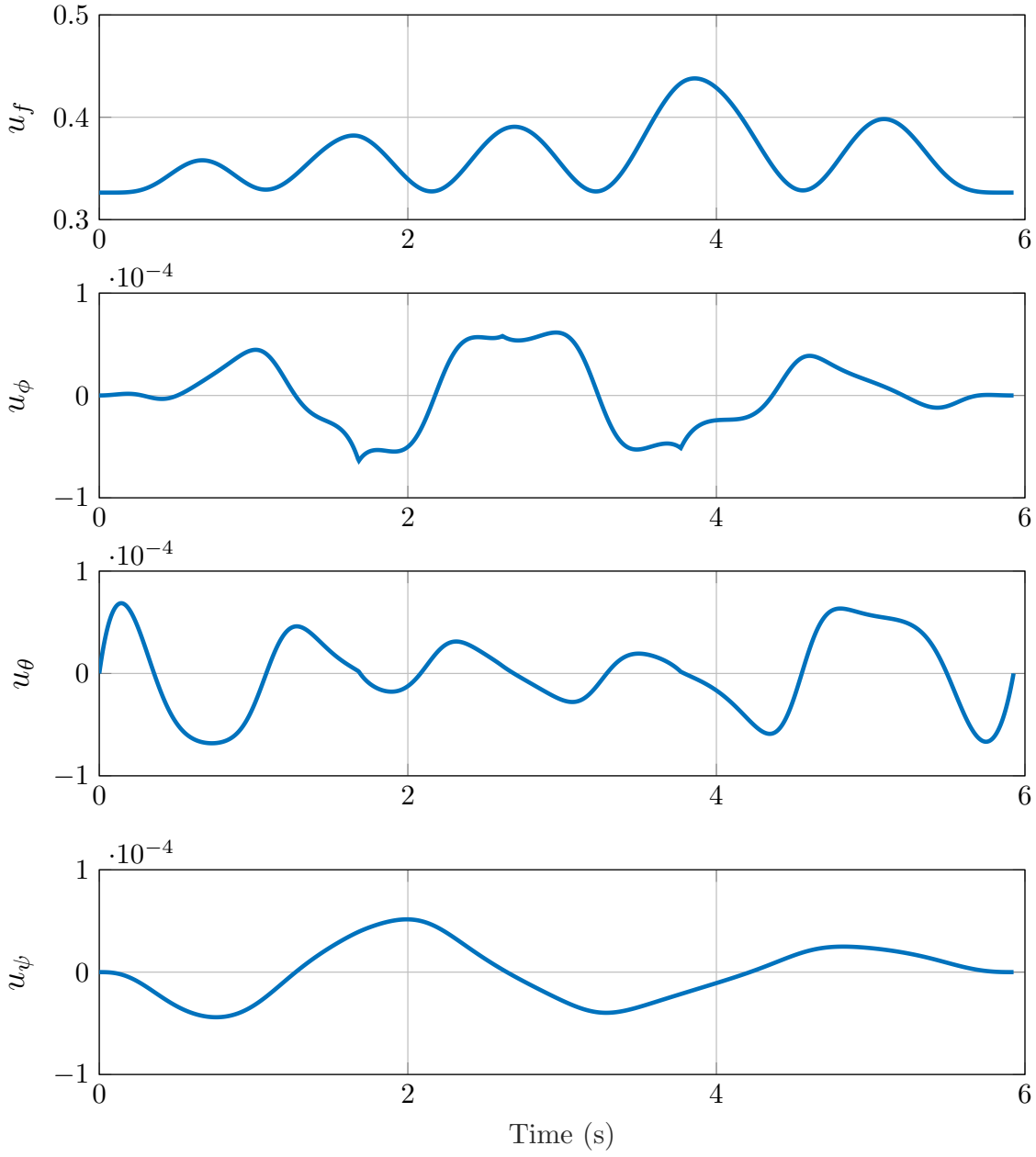

Figure 4.9: Control outputs after feasibility optimization
Optimal scaling factor $k^* = 5.9262$.

Chapter 5

Conclusions

5.1 Summary

This thesis presented contributions for the optimized trajectory generation and control for quadrotor vehicles. In chapter 2 we derived the equations of motion for a quadrotor model and explored the differential flatness properties of such systems. In chapter 3 we presented a controller design in $SO(3)$ to improve the tracking performance of quadrotors following tight trajectories. In chapter 4 we formulated an optimization approach to generate piecewise Bézier curve trajectories for quadrotors. The formulation involved a two-step optimization process of extracting the optimal control points for minimum snap trajectory and the scaling of flight duration to ensure dynamic feasibility for any type of quadrotor vehicle.

The codebase used in the geometric control and optimization simulations described in this thesis is available at [31] under an open source license.

5.2 Future Work

The control design methods used in this research was formulated in non-Euclidean manifolds and strongly improves the controller performance. However, the controller uses a known model and full state feedback in the control design which is unrealistic. Designing the controller to account for model uncertainties and noise will aid in the robustness of the

quadrotor. Adaptive control methodologies such as the ones defined in Hovakimyan and Cao [32] can be used to deal with uncertainties and time-delays in the output channel (namely, position and velocity).

The cost function defined in the optimization problem provides solutions for the optimal trajectories of a single quadrotor. Extending the optimization to solve for multiple vehicles which satisfy some temporal and spatial constraints can allow co-operative trajectory generation for a swarm of quadrotors.

The optimization procedure described in this thesis depends on the availability of collision-free waypoints obtained from motion planning algorithms. Developing an active optimization procedure to perform both tasks in the same step and generating efficient collision-free trajectories will simplify the implementation on actual quadrotors. As the Bézier curve control points can describe the convex hull within which the trajectory lies, this optimization procedure can contribute to the designing of a unified optimization problem.

The trajectory generation procedure relies on perfect knowledge of the environment to generate trajectories. Optimization for finite horizons with model predictive control techniques can offset these problems and provide a viable solution which can be used in real-world scenarios.

The extensions suggested in this section will expand the knowledge in optimization, geometric control and motion planning. Quadrotors are only now beginning to operate simple autonomous tasks (indoor position hold using visual odometry) and researchers have a long and arduous journey ahead before they can design a truly autonomous vehicle.

Bibliography

[1] Thiago Marinho, Christopher Widdowson, Amy Oetting, Arun Lakshmanan, Hang Cui, Naira Hovakimyan, Ranxiao Frances Wang, Alex Kirlik, Amy Laviers, and Dusan Stipanovic. Carebots: Prolonged elderly indeprendence using small mobile robots. *ASME Mechanical Engineering*, 138(9), 2016.

[2] Mark W Mueller, Michael Hamer, and Raffaello D'Andrea. Fusing ultra-wideband range measurements with accelerometers and rate gyroscopes for quadrocopter state estimation. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1730–1736. IEEE, 2015.

[3] Anastasios I Mourikis and Stergios I Roumeliotis. A multi-state constraint Kalman filter for vision-aided inertial navigation. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 3565–3572. IEEE, 2007.

[4] Jakob Engel, Jürgen Sturm, and Daniel Cremers. Camera-based navigation of a low-cost quadrocopter. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2815–2821. IEEE, 2012.

[5] Daniel Mellinger and Vijay Kumar. Minimum snap trajectory generation and control for quadrotors. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2520–2525. IEEE, 2011.

[6] Markus Hehn and Raffaello D'Andrea. Quadrocopter trajectory generation and control. *IFAC Proceedings Volumes*, 44(1):1485–1491, 2011.

[7] Matthew Turpin, Nathan Michael, and Vijay Kumar. Trajectory design and control

for aggressive formation flight with quadrotors. *Autonomous Robots*, 33(1-2):143–156, 2012.

[8] James A Preiss, Wolfgang Hönig, Gaurav S Sukhatme, and Nora Ayanian. Crazyswarm: A large nano-quadcopter swarm.

[9] Alex Kushleyev, Daniel Mellinger, Caitlin Powers, and Vijay Kumar. Towards a swarm of agile micro quadrotors. *Autonomous Robots*, 35(4):287–300, 2013.

[10] Taeyoung Lee, Melvin Leoky, and N Harris McClamroch. Geometric tracking control of a quadrotor UAV on SE (3). In *49th IEEE conference on decision and control (CDC)*, pages 5420–5425. IEEE, 2010.

[11] Srinath Mallikarjunan, Bill Nesbitt, Evgeny Kharisov, Enric Xargay, Naira Hovakimyan, Chengyu Cao, et al. L1 adaptive controller for attitude control of multirotors. In *AIAA Guidance, Navigation and Control Conference, Minneapolis, AIAA-2012-48312012*, 2012.

[12] Jan Willem Vervoorst. *A modular simulation environment for the improved dynamic simulation of multirotor unmanned aerial vehicles*. PhD thesis, 2016.

[13] Robert Mahony, Vijay Kumar, and Peter Corke. Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor. *IEEE robotics & automation magazine*, 19(3): 20–32, 2012.

[14] Dario Brescianini and Raffaello D'Andrea. Design, modeling and control of an omni-directional aerial vehicle. In *Proceedings of the 2016 IEEE International Conference on Robotics and Automation, Stockholm, Sweden, May 16th 21st*, pages 3261–3266, Piscataway, NJ, 2016. IEEE.

[15] Bill Crowther, Alexander Lanzon, Martin Maya-Gonzalez, and David Langkamp. Kinematic analysis and control design for a nonplanar multirotor vehicle. *Journal of Guidance, Control, and Dynamics*, 34(4):1157–1171, 2011.

[16] Pedro Castillo, Rogelio Lozano, and Alejandro Dzul. Stabilization of a mini rotorcraft with four rotors. *IEEE control systems magazine*, 25(6):45–55, 2005.

[17] Michael J Van Nieuwstadt and Richard M Murray. Real time trajectory generation for differentially flat systems. 1997.

[18] Steven M LaValle. *Planning algorithms*. Cambridge university press, 2006.

[19] Samir Bouabdallah, Pierpaolo Murrieri, and Roland Siegwart. Towards autonomous indoor micro VTOL. *Autonomous robots*, 18(2):171–183, 2005.

[20] Gabriel M Hoffmann, Haomiao Huang, Steven L Waslander, and Claire J Tomlin. Quadrotor helicopter flight dynamics and control: Theory and experiment. In *Proc. of the AIAA Guidance, Navigation, and Control Conference*, volume 2, page 4, 2007.

[21] Markus Hehn and Raffaello D'Andrea. Real-time trajectory generation for interception maneuvers with quadrocopters. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4979–4984. IEEE, 2012.

[22] Mark W Mueller, Markus Hehn, and Raffaello D'Andrea. A computationally efficient algorithm for state-to-state quadrocopter trajectory generation and feasibility verification. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3480–3486. IEEE, 2013.

[23] Samir Bouabdallah and Roland Siegwart. Backstepping and sliding-mode techniques applied to an indoor micro quadrotor. In *Proceedings of the 2005 IEEE international conference on robotics and automation*, pages 2247–2252. IEEE, 2005.

[24] Venanzio Cichella, Isaac Kaminer, Enric Xargay, Vladimir Dobrokhodov, Naira Hovakimyan, A Pedro Aguiar, and António M Pascoal. A lyapunov-based approach for time-coordinated 3D path-following of multiple quadrotors. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 1776–1781. IEEE, 2012.

[25] Taeyoung Lee, Melvin Leok, and N Harris McClamroch. Control of complex maneuvers for a quadrotor uav using geometric methods on SE (3). *arXiv preprint arXiv:1003.2005*, 2010.

[26] Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, 30(7):846–894, 2011.

[27] Syed Bilal Mehdi, Ronald Choe, and Naira Hovakimyan. Avoiding multiple collisions through trajectory replanning using piecewise Bézier curves. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pages 2755–2760. IEEE, 2015.

[28] Ronald Choe, Javier Puig, Venanzio Cichella, Enric Xargay, and Naira Hovakimyan. Trajectory generation using spatial pythagorean hodograph bezier curves. In *AIAA Guidance, Navigation, and Control Conference*, page 0597, 2015.

[29] Jonathan D Gammell, Siddhartha S Srinivasa, and Timothy D Barfoot. Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic. *arXiv preprint arXiv:1404.2334*, 2014.

[30] Charles Richter, Adam Bry, and Nicholas Roy. Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments. In *Robotics Research*, pages 649–666. Springer, 2016.

[31] Arun Lakshmanan. Trajectory generation and control for Crazyflie 2.0. `https://github.com/arunlakshmanan/cf-model.git`, 2016.

[32] Naira Hovakimyan and Chengyu Cao. *L1 adaptive control theory: guaranteed robustness with fast adaptation*, volume 21. Siam, 2010.