The Thesis committee for Can Pehlivanturk
certifies that this is the approved version of the following thesis:

Lossless Convexification of Quadrotor Motion Planning with Experiments

APPROVED BY

SUPERVISING COMMITTEE:

Raul Longoria, Supervisor

Behçet Açıkmeşe, Co-Supervisor

Lossless Convexification of Quadrotor Motion Planning with Experiments

by

Can Pehlivantürk, B.S.

THESIS

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN ENGINEERING

THE UNIVERSITY OF TEXAS AT AUSTIN

August 2014

Dedicated to my family, friends, Pınar, and everybody who makes life meaningful.

Acknowledgments

First and foremost, I would like to thank my advisor, Dr. Behçet Açıkmeşe for his guidance throughout the project and the development of this thesis. I would also like to thank Dr. Raul Longoria for his very valuable feedback, especially on my writing.

I would like to express my gratitude to my colleagues and friends. Nazlı Demir with whom the foundation of the algorithm was developed; Tim Lowery who devoted much of his time managing the lab and the experimental setup; Utku Eren, whose countless visits downstairs helped refine many problems. Stephanie Taylor, Natalie and Katie Hansen, Gurpreet Singh, and Engin Hassamancı for their help in editing my thesis and their support that kept me going through the home stretch.

A special thanks goes to my friends from METU and I would like to thank Dr. Derek Baker for being a constant source of wisdom and counsel throughout my undergraduate and graduate studies.

Last but not least, I would like to thank my parents, Berna and Necip, my sister Melis, my girlfriend Sara Pinar, and the rest of my family members for their huge support throughout my studies and being understanding of the recent reduction in phone call frequency.

Lossless Convexification of Quadrotor Motion Planning with Experiments

Can Pehlivantürk, M.S.E.
The University of Texas at Austin, 2014

Supervisors: Raul Longoria
Behçet Açıkmeşe

This thesis describes a motion planning method that is designed to guide an autonomous quadrotor. The proposed method is based on a novel lossless convexification, which was first introduced in [12], that allows convex representations of many non-convex control constraints, such as that of the quadrotors. The second contribution of this thesis is to include two separate methods to generate path constraints that capture non-convex position constraints. Using the convexified optimal trajectory generation problem with physical and path constraints, an algorithm is developed that generates fuel optimal trajectories given the initial state and desired final state. As a proof of concept, a quadrotor testbed is developed that utilize a state-of-the-art motion tracking system. The quadrotor is commanded via a ground station where the convexified optimal trajectory generation algorithm is successfully implemented together with a trajectory tracking feedback controller.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Unmanned aerial vehicles (UAV), first making an appearance in military applications, have been aggressively expanding into the civilian arena. Control engineers has successfully pioneered the automation of unmanned vehicles and devices by replacing human operators with algorithms. UAVs, or drones, have many valuable qualities. With no pilot to fatigue, they can carry out long or tedious observation missions. With no pilot to support, they can be smaller and lighter. Overall, with the removal of the pilot and the addition of autonomy, UAVs can be more versatile than traditional aircraft.

Usage of drones for food delivery has been on the agenda of restaurant chains [6].While people entertain the idea of flying pizzas, the world's largest online retailer, Amazon, has been working on a new delivery system called Amazon Prime Air, which promises 30 minute deliveries using drones [1]. Although it is viewed as a publicity stunt by some[5] [8], Amazon claims this system will be ready as early as 2015 [1]. The proposed usage of drones for shipping is a very exciting development for the field of autonomous vehicles.

UAVs are also becoming increasingly important in environmental conservation and field ecology. The low ecological footprint and increased safety of UAVs are especially valuable for environmental missions [27] [38]. Projects that utilize UAVs for aerial land and wildlife surveys are deemed efficient and successful and are becoming common practice[43] [26]. Another environmental application is forest fire monitoring with infrared imaging [46]. Emerging companies propose usage of drones for disaster relief and healthcare supply [9]. There are a vast number of other fields that take advantage of the observation and land survey capabilities such as, real estate photography, agriculture [23], and traffic monitoring and management [28]. It is increasingly possible to find UAVs on movie sets and in the entertainment industry. There are countless functions UAVs currently serve right now, with the possibility of many more on the horizon.

Tasks related to autonomous flight often involve traveling from the initial position to the target while passing through or avoiding certain areas. Constraints on the path might be necessary to avoid any hazards or navigate through obstacles. It is crucial to have a fast, reliable, and robust guidance algorithm that can meet these requirements. The aim is to develop a solution to the trajectory generation problem and implement it on an autonomous quadrotor.

1.2 Objectives and Contributions

The main objective of this work is to develop a fuel and time optimal quadrotor guidance algorithm to autonomously generate optimal trajectories given initial and final positions with multiple physical and geometric constraints. The algorithm builds upon the previous work “Lossless Convexification of a Class of Non-Convex Optimal Control Problems,” [12] which enables the convexification of non-convex thrust constraints inherent to many thrusters. A similar algorithm utilizing this method is previously suggested for the planetary soft landing optimal control problem with the intent of reaching inaccessible but scientifically valuable targets on Mars for sample return or human class planetary missions [13]. The main contribution is the development and implementation of a convexified optimal trajectory generation problem formulated for indoor quadrotor flight.

The secondary objective is to develop a testbed to demonstrate the algorithm. The experimental setup is developed around a nano-quadrotor with motion tracking and an external trajectory tracking controller. This system is used together with the guidance algorithm and test flights are conducted.

1.3 Literature Survey

There are a considerable number of researchers from many institutions and research groups working on UAVs, and more specifically quadrotors. Almost all of these groups have developed trajectory generation strategies. Iterative methods that define trajectories as segments are developed and used for aggressive maneuvers [35]. Although not guaranteed to be optimal, the closest path can be found using RRT* algorithms [18]. Once the closest path is found, the trajectory can be designed with polynomial segments [39]. The general problem of finding the optimal trajectory with non-linear dynamics is solved using genetic algorithms[31]. The simplified problem with non-convex acceleration constraints is also studied and solved using iterative methods without a convex programming approach [25].

Guidance, trajectory generation, and task assignment for large robot networks or swarm coordination are interesting and ongoing topics of research [14] [40] [29]. Convex optimization is also applied in swarm guidance. [19] [20]. For convexifying the trajectory generation problem, the second-order cone progamming approach is studied with approximated non-convex constraints [32]. The geometric path constraints are transformed to convex representations [42]. Convex transformation and second-order cone programming approach to trajectory planning is not limited to UAVs but has been extended to robot systems [44].

Chapter 2

Trajectory Generation Problem Definition

In this chapter, the trajectory generation problem is formulated and convexified. This convexified optimal control problem is the foundation of the guidance algorithm proposed in this thesis. This is the main path finding strategy used by the quadrotors in the inventory of the Autonomous Guidance Navigation and Control Group (AutoGNC). The optimal trajectory generation problem is defined with a convex minimum fuel cost and convex state constraints. The non-convex input constraint, namely the control magnitude constraint, is replaced by equivalent constraint with a convex cone. This relaxation is established by lossless convexification in which the optimal solution of the relaxed problem is also an optimal solution for the original non-convex problem. [11] [12]. The main advantage of convexifing the optimization problem and subsequently having a convex problem that represents the trajectory generation is the ability to use interior point methods which are very well established in convex programming [16] [36] [30]. Custom algorithms can find the global minimum of similar size convex optimization problems in micro to millisecond time scales [10] [34] [21].

2.1 Formulation

This section introduces the formulation of the optimal trajectory generation problem. The purpose of a quadrotor path plan can be thought of as going from one translational state to another, or from point A to point B. When there are no obstacles or path constraints and the initial and desired final states are stationary the problem is quite simple. However, often times this is not the case and the quadrotor has to follow a constrained path. Also, the ability to change the trajectory in mid flight where the new initial condition has a velocity component can be very beneficial. When these are taken into consideration, the problem becomes more interesting and complex.

When solved, the problem should provide a translational state trajectory and a thrust or acceleration profile over this said trajectory. The trajectory should comply with the desired state and input constraints and be fuel optimal. Flight duration is determined by battery capacity, and due to low payloads it is quite limited for quadrotors. The quadrotor is modeled as a lumped mass with an acceleration vector for control, and has the following dynamics:

$$\dot{x}(t) = Ax(t) + B(g + u(t)), \tag{2.1}$$

where x is the state vector composed of the position and the velocity $x(t) = (r(t), \dot{r}(t)) : R_+ \rightarrow R^6$.

The input vector u is composed of the accelerations in x, y, and z directions which can be expressed as the thrust force in the respective directions, over the mass of the quadrotor:

$$u(t) = \frac{(T_x, T_y, T_z)}{m_{quad}}, \tag{2.2}$$

and the corresponding state matrices are defined as the following:

$$A = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}, \quad B = \begin{bmatrix} \mathbf{0} \\ \mathbf{I} \end{bmatrix}. \tag{2.3}$$

For clarity, the state and input constraints are grouped into physical and path constraints.

2.1.1 Physical Constraints

Physical constraints on the system are the constraints that are the product of the environment and the dynamics of the vehicle. To accommodate some of the critical limitations of the system the following constraints are imposed:

- Thrust lower and upper bound: Due to the physical capacity of the motors and the propellers, a lower and upper bound constraint on the thrust is needed. This can be represented as an constraint on the acceleration input:

  $$0 < \rho_1 \leq \|u(t)\| \leq \rho_2, \tag{2.4}$$

  where ρ_1 and ρ_2 are the lower and the upper bounds on the net acceleration respectively.

- Thrust pointing constraint: It is desired to keep the thrust vector in a prescribed cone around the inertial $z-axis$ in order to always have control over the altitude. This is done to prevent complications related to sudden altitude loss such as falling through the wake which is a highly turbulent disturbed flow. The pointing constraint can be represented as follows:

$$u_3(t) \geq \|u(t)\|\, cos(\theta). \tag{2.5}$$

- Position constraints: For an indoor real time implementation, we need to put bounds on the position vector in order to keep the trajectory in the volume defined by the room dimensions. The position constraint can be represented as the following:

$$\phi_i \leq x_i(t) \leq \delta_i, \quad i = 1, 2, 3 \tag{2.6}$$

  where ϕ and δ are 3 dimensional vectors defining the limits on X, Y, and Z directions.

2.1.2 Path Constraints

To determine the trajectory the following path constraints are imposed:

- Prescribed initial and final positions and velocities: The guidance algorithm will take the quadrotor from an initial state to a final state. The first and last state of the trajectory should follow

$$x_0 = \xi_0, \quad x_f = \xi_f. \tag{2.7}$$

Along the way, there might be obstacles or other situations that would require the quadrotor to avoid an area or to stay in an area. With this algorithm it is possible to constrain the position and the velocity states at each time step, this capability transitioned into two path determination strategies:

- Prescribed multiple waypoints along the trajectory where the quadrotor has to be at the prescribed states at the prescribed times.
- Prescribed corridors or zones along the trajectory in which the quadrotor has to stay in during prescribed time frames.

Depending on the strategy implemented, one of the following constraints are imposed:

- Prescribed multiple waypoints along the trajectory:

$$x(\hat{t}_i) = \xi_i, \quad 0 \leq \hat{t}_i, \dots, \hat{t}_n \leq t_f, \quad i = 1, \dots, n \tag{2.8}$$

  where $\hat{t}$ is a prescribed time step at which the trajectory should pass through the waypoint. Note that there must be adequate time steps between each waypoint to have a feasible solution.

- Prescribed corridors or zones along the trajectory:

$$\beta_i \leq x(t) \leq \gamma_i, \quad \hat{t}_i \leq t < \hat{t}_{i+1}, \quad 0 = \hat{t}_i, \dots, \hat{t}_{n+1} = t_f, \quad i = 1, \dots, n. \tag{2.9}$$

2.1.3 Optimization Problem

For this problem, we are looking for the minimum fuel solution for which the cost function can be set as [16]:

$$\min_{x,u} \int_0^{t_f} \|u(t)\| \, dt \; . \tag{2.10}$$

With the given constraints and the cost function, the following optimization problems can be solved to find the optimal trajectory:

Problem 1. Non-Convex Trajectory Generation Problem with Waypoints

$$\min_{x,u} \int_0^{t_f} \|u(t)\| \, dt \qquad \text{subject to:} \tag{2.10}$$

$$\begin{aligned}
&\dot{x}(t) = Ax(t) + B(g + u(t)) && (2.1)\\
&0 < \rho_1 \leq \|u(t)\| \leq \rho_2 && (2.4)\\
&u_3(t) \geq \|u(t)\| \, cos(\theta) && (2.5)\\
&\phi_i \leq x_i(t) \leq \delta_i \quad i = 1, 2, 3 && (2.6)\\
&x_0 = \xi_0, \quad x_f = \xi_f && (2.7)\\
&x(\hat{t}_i) = \xi_i, \quad 0 \leq \hat{t}_i, \ldots, \hat{t}_n \leq t_f, \quad i = 1, \ldots, n && (2.8)
\end{aligned}$$

Problem 2. Non-Convex Trajectory Generation Problem with Corridors

$$\min_{x,u} \int_0^{t_f} \|u(t)\| \, dt \qquad \text{subject to:} \tag{2.10}$$

$$\begin{aligned}
&\dot{x}(t) = Ax(t) + B(g + u(t)) && (2.1)\\
&0 < \rho_1 \leq \|u(t)\| \leq \rho_2 && (2.4)\\
&u_3(t) \geq \|u(t)\| \, cos(\theta) && (2.5)\\
&\phi_i \leq x_i(t) \leq \delta_i \quad i = 1, 2, 3 && (2.6)\\
&x_0 = \xi_0, \quad x_f = \xi_f && (2.7)\\
&\beta_i \leq x(t) \leq \gamma_i, \quad \hat{t}_i \leq t < \hat{t}_{i+1}, \quad 0 = \hat{t}_i, \ldots, \hat{t}_{n+1} = t_f, \quad i = 1, \ldots, n && (2.9)
\end{aligned}$$

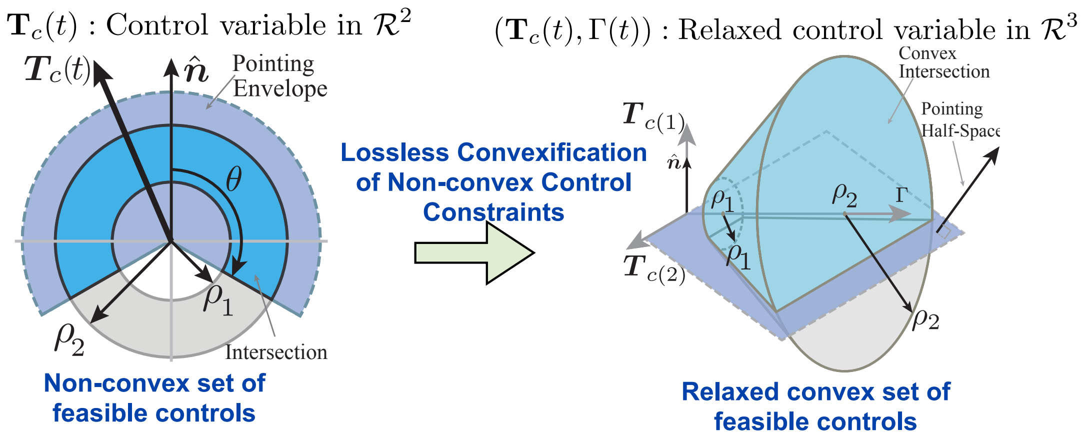
2.2 Convexification

In this section the optimal control problem is converted into a convex problem with second order cone constraints using the lossless convexification method described in [12]. In order to convexify the control constraints, the non-convex thrust constraint in Equation 2.4 is relaxed by replacing the non-convex constraint with the following constraints. Defining a slack variable $\sigma(t)$ such that:

$$\|u(t)\| \leq \sigma(t), \tag{2.11}$$

and

$$0 < \rho_1 \leq \sigma(t) \leq \rho_2. \tag{2.12}$$



$\mathbf{T}_c(t)$: Control variable in $\mathcal{R}^2$
$\boldsymbol{T}_c(t)$
$\hat{\boldsymbol{n}}$
Pointing Envelope
θ
ρ_1
ρ_2
Intersection
Non-convex set of feasible controls
Lossless Convexification of Non-convex Control Constraints
$(\mathbf{T}_c(t), \Gamma(t))$: Relaxed control variable in $\mathcal{R}^3$
Convex Intersection
Pointing Half-Space
$\boldsymbol{T}_{c(1)}$
$\hat{n}$
$\boldsymbol{T}_{c(2)}$
ρ_1
ρ_2
Γ
ρ_1
ρ_2
Relaxed convex set of feasible controls

Figure 2.1: Relaxation of the non-convex control set to a convex set [13]

To demonstrate this relaxation, the non-convex control constraint for a 2D case and the geometric interpretation of the relaxation is illustrated in Figure 2.1 [13]. With this relaxation, it is possible that a feasible solution to the relaxed problem such as $(\|u(t)\|, \sigma(t)) = (0, \rho_1)$ has a solution for control, $\|u(t)\| = 0$, that violates the original control constraint and is not a feasible solution to the original problem [24]. However, Theorem 2 of *"Lossless Convexification for a Class of Optimal Control Problems with Nonconvex Control Constraints"*, proves that the optimal solutions to the relaxed problem are also optimal solutions to the original problem and the relaxation is lossless, hence the name "Lossless Convexification" [12, Theorem 2]. The solutions are on the boundary of the relaxed cone when $\|u(t)\| = \sigma(t)$ which always holds true for optimal solutions. The new cost can be set as:

$$\min_{x,u} \int_0^{t_f} \sigma(t)dt \ . \tag{2.13}$$

After replacing the control constraints and the cost function, the convexified problem reads:

Problem 3. Convexified Trajectory Generation Problem with Waypoints

$$\min_{x,u} \int_0^{t_f} \sigma(t)dt \qquad \text{subject to:} \tag{2.13}$$

$$\dot{x}(t) = Ax(t) + B(g + u(t)) \tag{2.1}$$
$$\|u(t)\| \leq \sigma(t) \tag{2.11}$$
$$0 < \rho_1 \leq \sigma(t) \leq \rho_2 \tag{2.12}$$
$$u_3(t) \geq \|u(t)\| \, cos(\theta) \tag{2.5}$$
$$\phi_i \leq x_i(t) \leq \delta_i \quad i = 1, 2, 3 \tag{2.6}$$
$$x_0 = \xi_0, \quad x_f = \xi_f \tag{2.7}$$
$$x(\hat{t}_i) = \xi_i, \quad 0 \leq \hat{t}_i, \ldots, \hat{t}_n \leq t_f, \quad i = 1, \ldots, n \tag{2.8}$$

Problem 4. Convexified Trajectory Generation Problem with Corridors

$$\min_{x,u} \int_0^{t_f} \sigma(t)dt \qquad \text{subject to:} \tag{2.13}$$

$$\dot{x}(t) = Ax(t) + B(g + u(t)) \tag{2.1}$$

$$\|u(t)\| \leq \sigma(t) \tag{2.11}$$

$$0 < \rho_1 \leq \sigma(t) \leq \rho_2 \tag{2.12}$$

$$u_3(t) \geq \|u(t)\| \, cos(\theta) \tag{2.5}$$

$$\phi_i \leq x_i(t) \leq \delta_i \quad i = 1, 2, 3 \tag{2.6}$$

$$x_0 = \xi_0, \quad x_f = \xi_f \tag{2.7}$$

$$\beta_i \leq x(t) \leq \gamma_i, \quad \hat{t}_i \leq t < \hat{t}_{i+1}, \quad 0 = \hat{t}_i, \ldots, \hat{t}_{n+1} = t_f, \quad i = 1, \ldots, n \tag{2.9}$$

Chapter 3

Implementation and Experimental Setup

3.1 Overview

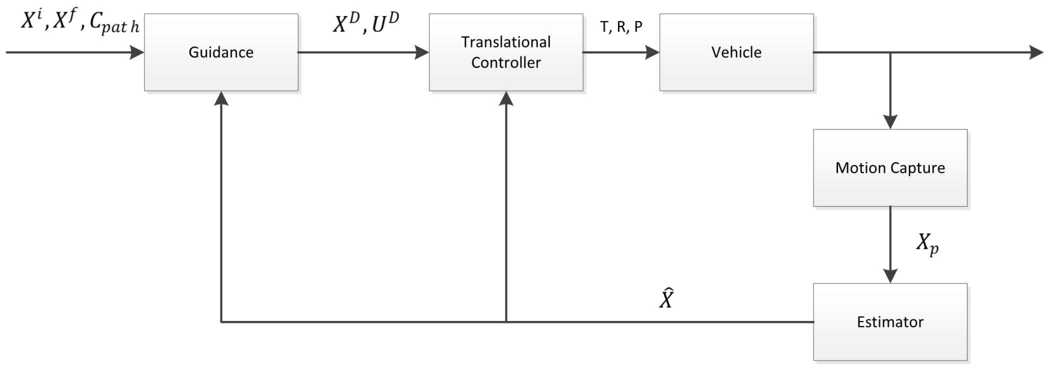


Figure 3.1: Block diagram of the guidance algorithm test bed

In this chapter, the implementation of the guidance algorithm and the experimental setup is explained. The convexified trajectory generation problem is discretized and the resultant guidance algorithm is scripted in MATLAB. In order to fly the generated trajectories, a testbed is developed around a nano-quadrotor Bitcraze Crazyflie. The block diagram of the guidance algorithm experimental setup is presented in Figure 3.1. The guidance algorithm produces desired position, velocity states, and desired acceleration input for each timestep. After the trajectory is generated, the external and onboard controllers keep the quadrotor on the desired trajectory. The translational and attitude controllers are handled separately. The agility of the quadrotor enables it to perform attitude maneuvers much quicker than the translational motion, therefore the translational dynamics are decoupled from the attitude dynamics. The ground station is responsible for the translational control. The position of the quadrotor is determined by a Vicon motion tracking system and the velocity is estimated. The trajectory tracking controller determines the acceleration input and this input is converted to attitude and thrust commands. These commands are sent to the quadrotor and the onboard controller handles the attitude control and the determination of actuator inputs.

3.2 Implementation of the Guidance Algorithm

In order to find the optimal solution to the formulated and convexified trajectory generation problem it should first be discretized. The problem then can be solved using convex programming methods for second order cone constraints. The problem is then modeled in MATLAB using the YALMIP modeling language[33] and solved using SDPT3 [41]. A line search strategy is used to find the minimum total time. The Autonomous Guidance Navigation and Control Laboratory (AutoGNC Lab) has developed an automated custom code generation algorithm for embedded real-time second order cone problems [21]. However the customization and the in-house solver was not implemented in this experimental setup. Upcoming quadrotors of AutoGNC lab will fly the guidance algorithm defined in this thesis with the customization.

3.2.1 Discretization

As mentioned earlier, the quadrotor is modeled as a lumped mass with a thrust vector for control, and has the following discrete time dynamics:

$$x_{k+1} = Ax_k + B(g + u_k), \quad k = 0, \dots, N \tag{3.1}$$

where x is the 6×1 state vector composed of the positions and the velocities in x, y and z directions:

$$x = [x\ y\ z\ \dot{x}\ \dot{y}\ \dot{z}]',$$

and the input vector u is composed of the accelerations along the x, y, and z directions:

$$u(t) = \frac{[T_x, T_y, T_z]}{m_{quad}}.$$

The corresponding state matrices are given as follows:

$$A = \begin{bmatrix} \mathbf{I} & \mathbf{I} \cdot \mathbf{\Delta t} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}, \quad B = \begin{bmatrix} \mathbf{I} \cdot \frac{\mathbf{\Delta t^2}}{\mathbf{2}} \\ \mathbf{I} \end{bmatrix}, \tag{3.2}$$

where Δt is the timestep. The actual time in seconds corresponding to each step can be determined by: $t_k = t_0 + (k-1)\Delta t$ where t_0 is the initial time.

The physical and path constraints with time dependencies are also modified to represent discrete time and the final form of the convexified and discretized optimal control problems for trajectory generation is given below.

Problem 5. Convexified and Discretized Trajectory Generation Problem with Waypoints

$$\min_{x,u} \sum \sigma_k \qquad \text{subject to:} \tag{3.3}$$

$$x_{k+1} = Ax_k + B(g + u_k) \quad k = 0, \ldots, N-1 \tag{3.1}$$
$$\|u_k\| \leq \sigma_k, \tag{3.4}$$
$$0 < \rho_1 \leq \sigma_k \leq \rho_2 \tag{3.5}$$
$$u_{3,k} \geq \|u_k\| \, cos(\theta) \tag{3.6}$$
$$\phi_i \leq x_{i,k} \leq \psi_i, \quad i = 1, 2, 3 \tag{3.7}$$
$$x_0 = \xi_0, \quad x_N = \xi_N \tag{3.8}$$
$$x_{\hat{k}_i} = \xi_i, \quad 0 \leq \hat{k}_i, \ldots, \hat{k}_n \leq t_f, \quad i = 1, \ldots, n \tag{3.9}$$

Problem 6. Convexified and Discretized Trajectory Generation Problem with Corridors

$$\min_{x,u} \sum \sigma_k \qquad \text{subject to:} \tag{3.3}$$

$$x_{k+1} = Ax_k + B(g + u_k), \quad k = 0, \ldots, N-1 \tag{3.1}$$
$$\|u_k\| \leq \sigma_k, \tag{3.4}$$
$$0 < \rho_1 \leq \sigma_k \leq \rho_2 \tag{3.5}$$
$$u_{3,k} \geq \|u_k\| \, cos(\theta) \tag{3.6}$$
$$\phi_i \leq x_{i,k} \leq \psi_i, \quad i = 1, 2, 3 \tag{3.7}$$
$$x_0 = \xi_0, \quad x_N = \xi_N \tag{3.8}$$
$$\beta_i \leq x_k \leq \gamma_i, \quad \hat{k}_i \leq k < \hat{k}_{i+1}, \quad 0 = \hat{k}_i, \ldots, \hat{k}_{n+1} = N-1, \quad i = 1, \ldots, n \tag{3.10}$$

3.3 Motion Tracking and Velocity Estimation

The algorithm is tested in an indoor laboratory environment, therefore the Global Positioning System could not be utilized as it is difficult to locate satellite signals indoors due to the lack of line of sight. Moreover, the constrained nature of the flight area in the indoor laboratory space greatly increases the positioning accuracy requirement and GPS simply can not provide the necessary accuracy. The lack of an onboard solution to the positioning problem made it necessary to seek a motion tracking system. For the experiments, the position of the quadrotor is tracked using a Vicon motion capture system. The motion capture cameras are shown in Figure 3.2. The Vicon system provides millimeter level accuracy and can have a position capture frequency of up to 500 Hz. Another big constraint for nano quadrotors is the payload capacity. One of the advantages of the system used was the ability to use passive markers with only reflective coating and no electronic parts which weigh significantly less compared to active markers.

The motion tracking system provides excellent position of the mass center data at a more than sufficient frequency. The velocity states are observable thus they can be estimated using a simple observer[17]. Consistent with the previous lumped mass assumption, for the linear time invariant system, a discrete time observer of the following form is designed to estimate the velocity:

$$\hat{x}(k+1) = A\hat{x}(k) + Bu(k) + L(y(k) - \hat{y}(k)),$$

$$\hat{y}(k) = C\hat{x}(k).$$

The Observer gain is chosen such that the observer is robust in a frequency range of 100Hz to 500Hz. It should be noted that in order to get rid of the small error is position data, a low-pass filter is used.

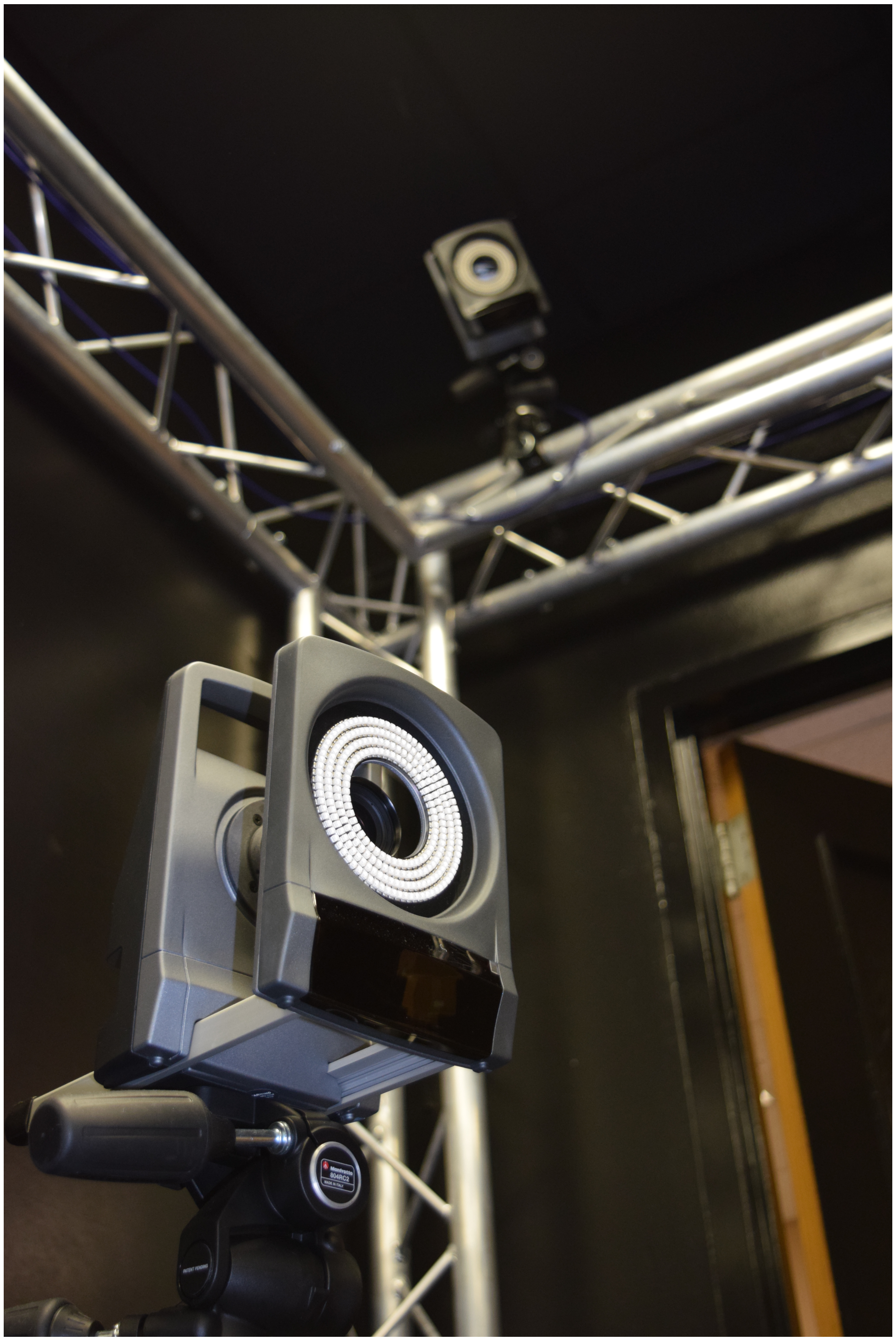


Figure 3.2: Vicon motion capture cameras

3.4 Trajectory Tracking Controller

The trajectory generation algorithm produces desired states and acceleration vs. time. Then, the feedforward force is generated using the lumped mass approximation. A proportional-integral-derivative (PID) controller is used as the feedback mechanism on top of the feedforward control input that is generated by the trajectory algorithm. The error in the trajectory is defined as the following:

- The position error is the difference of the first 3 states of the estimated state vector and the desired state vector:

$$e_p = \hat{x}_j - x_j^d, \quad j = 1, 2, 3.$$

- The velocity error is the difference of the last 3 states of the estimated state vector and the desired state vector:

$$e_v = \hat{x}_j - x_j^d, \quad j = 4, 5, 6.$$

- To get rid of the steady state error in the global Z direction, integral term with accumulated error is also utilized and determined as follows:

$$e_i(t) = \int_0^t e_p(\tau) \mathrm{d}t.$$

Combining the feedback and feedforward thrust forces, the controller produces the following desired acceleration input. Note that the integral term only contributes in the Z direction:

$$U^d = K_p e_p + K_d e_v + K_i e_i + U^{ff}.$$

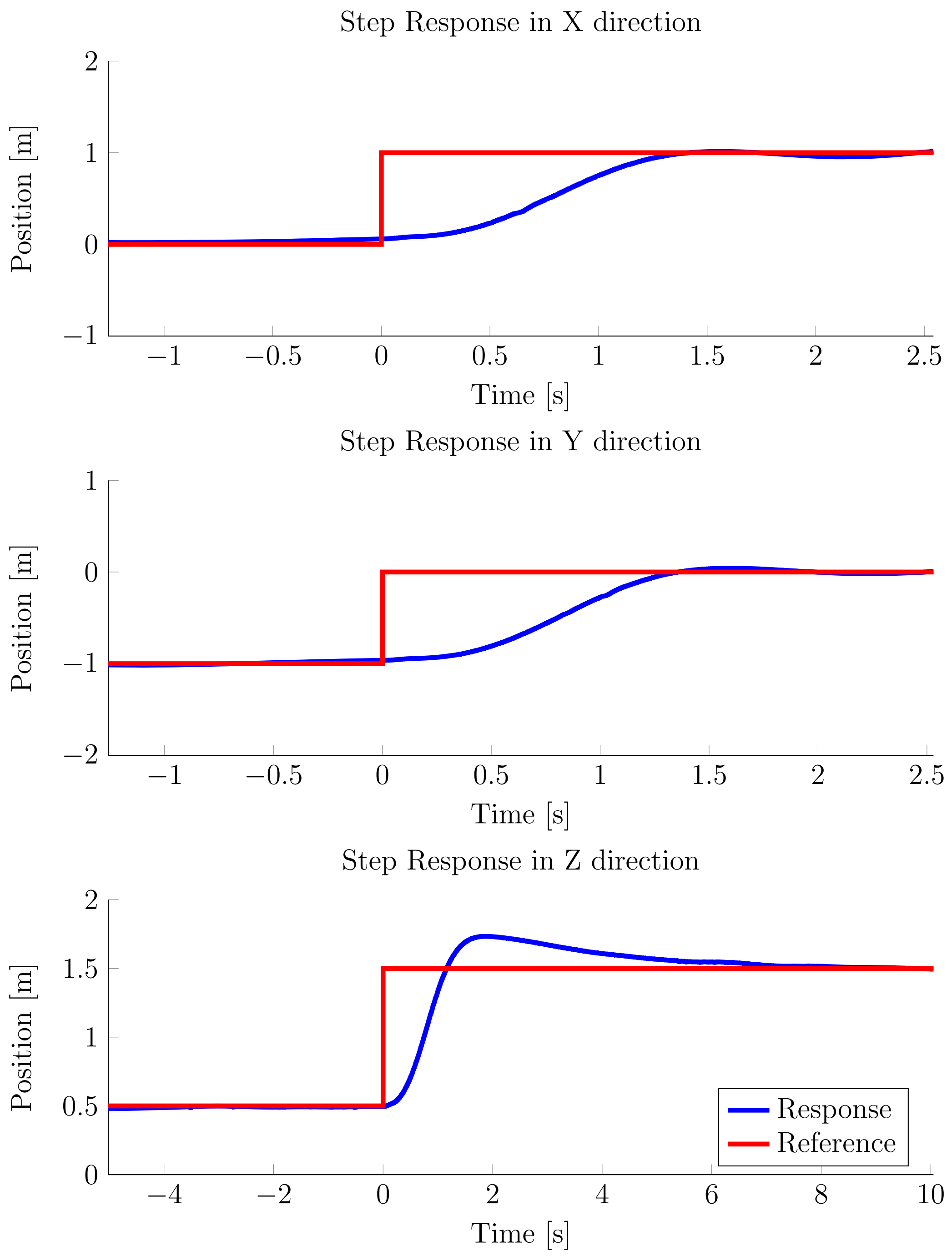


Figure 3.3: Individual step responses in 3 principal directions of the quadrotor translation

The controller can be adjusted in-flight and fine tuning can be achieved. There are many different types of tuning rules for the controller gains for a PID controller [37]. These rules such as Ziegler-Nichols, provided good starting points for the controller. However, the gains were adjusted manually to achieve the desired response from the system. The step responses of the translation controller in global X, Y ,and Z directions are presented in Figure 3.3. The controller is both stiff and designed to minimize overshoot. However because of the integral term the altitude step response has a longer settling time and overshoots. This is due to high error accumulation during the response, and should not occur during normal flight where the generated trajectories are continuous and differentiable. The integral term would only get rid of the steady state error which should stay the same throughout normal flight.

3.5 The Quadrotor

The Quadrotor used for the experiments is the BitCraze Crazyflie. The Crazyflie Quadcopter is a nano quadcopter which weighs about 19 grams and measures about 90 mm from motor to motor. The 170 mAh Li-Po battery powers the Crazyflie for a flight time of up to 7 minutes [3]. The Crazyflie has a 32 bit 72 MHz micro-controller onboard which runs the stabilization system and the attitude controller [7]. Both the hardware and software are open source, giving us crucial flexibility in implementing our own guidance software which makes it possible to combine all of the pieces together. All in all, the Crazyflie proved instrumental in both single quadrotor and swarm

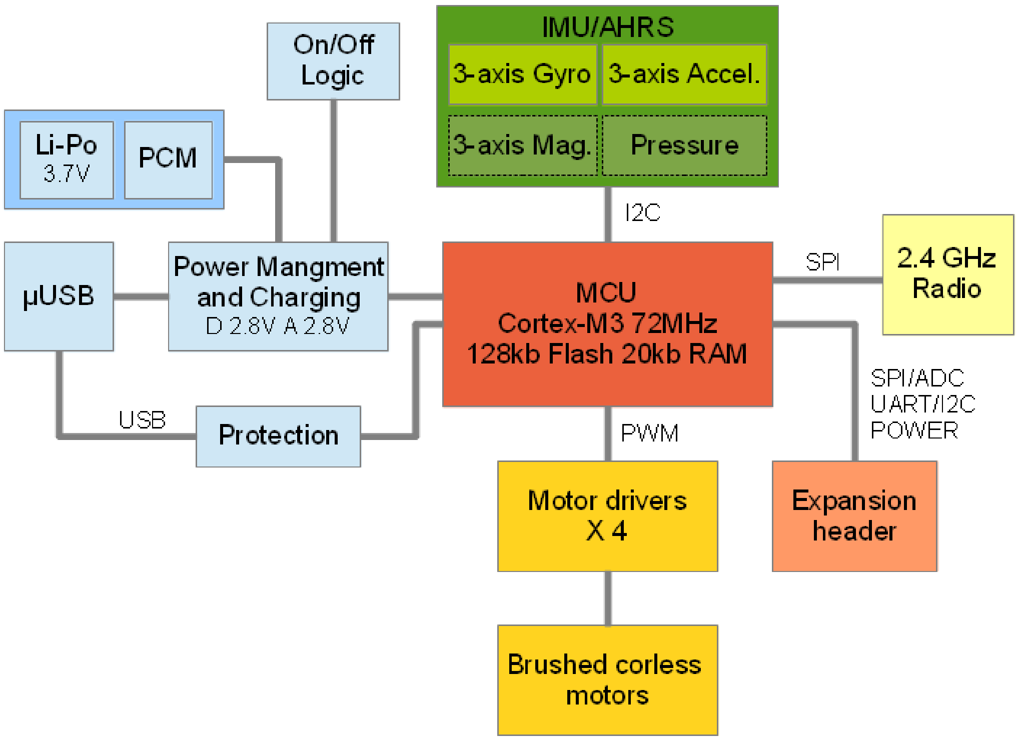


Figure 3.4: The Bitcraze Crazyflie electronics overview [2]

applications. An overview of the electronic setup of the Crazyflie taken from the Bitcraze website is presented in Figure 3.4 [2].

The Crazyflie is commanded from the purpose-built Crazyradio which is a 2.4 GHz radio USB dongle [4]. Depending on the environment, the radio connection has a range up to 80 m. The dongle enables the connection with the PC that runs the trajectory algorithms.

3.5.1 Onboard controller

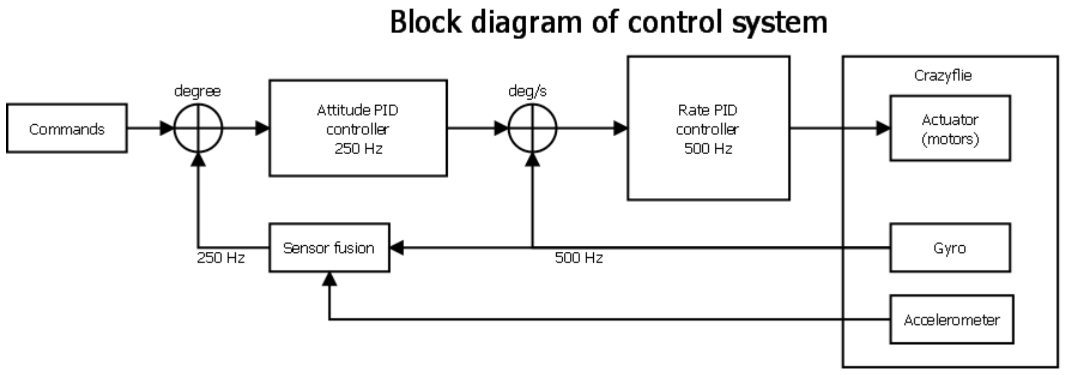

Block diagram of control system
Commands
degree
Attitude PID controller 250 Hz
deg/s
Rate PID controller 500 Hz
Crazyflie
Actuator (motors)
Sensor fusion
250 Hz
500 Hz
Gyro
Accelerometer

Figure 3.5: The block diagram of the Bitcraze Crazyflie onboard attitude controller [7]

The onboard controller brings the orientation of the quadrotor to the given desired orientation. The four actuators on the quadrotor are commanded by the onboard micro-controller to produce the necessary torques for attitude control and commanded net force. The Crazyflie client is written in Python and the same functions are used in the control algorithm to connect to and command the Crazyflie. The main command function of the Crazyflie looks like the following:

```
commander.send_setpoint(roll, pitch, yawrate, thrust)
```

Here the desired roll, pitch, yawrate and net thrust is sent to the quadrotor using the usb radio dongle. The onboard controller then commands the motors to track these desired values. A block diagram taken from the Bitcraze website of the Crazyflie onboard controller is presented in Figure 3.5 [7].

3.5.2 Thruster Characterization

Table 3.1: The results of the thruster characterization test

Test #	Thrust Input	rpm	Mean Thrust Force $[N]$
1	10001	6270	0.028
2	15000	8025	0.048
3	20000	9600	0.069
4	25000	10900	0.091
5	30000	12100	0.113
6	35000	13250	0.137
7	40000	14250	0.162
8	45000	15200	0.187
9	50000	16400	0.218
10	55000	17450	0.247
11	60000	18450	0.279

The quadrotor handles the attitude control internally. It is only necessary to map the net desired force the quadrotor should apply to the actual control input to the motors. The Crazyflie has a thrust PWM input range from 10000 to 60000. Here 10000 is the lowest input, and anything lower halts the motors and 60000 is the continuous input for maximum thrust. The thrust inputs change in small increments. The thrusters are assumed to reach

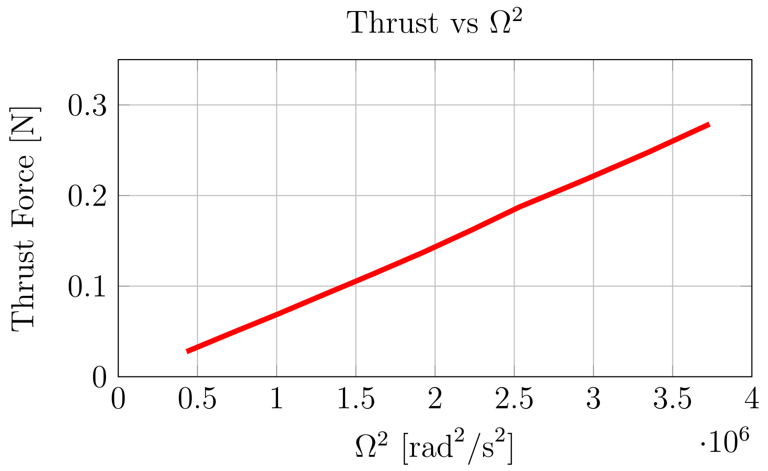


Figure 3.6: The thrust force vs. angular velocity squared

steady state before the control input is updated considering similar motors have settling times less than 100ms under step input [45].

In order for the desired acceleration to be mapped to the thrust input, an experimental thrust test is conducted. The thrust characterization test setup with force and rpm sensors is presented in Figure 3.7. The forces and torques are measured using a load cell and rpm of one of the propellers is measured using a laser photo-diode setup.

The result of the test is presented in Table 3.1. The thrust input is applied for 5 seconds and the resultant force and rpm data is averaged to determine the mean thrust force and rpm corresponding to each input. The thrust force vs. angular velocity squared data from the test is presented in Figure 3.6. The relationship between the two is linear, as expected, and this supports the validity of the test.

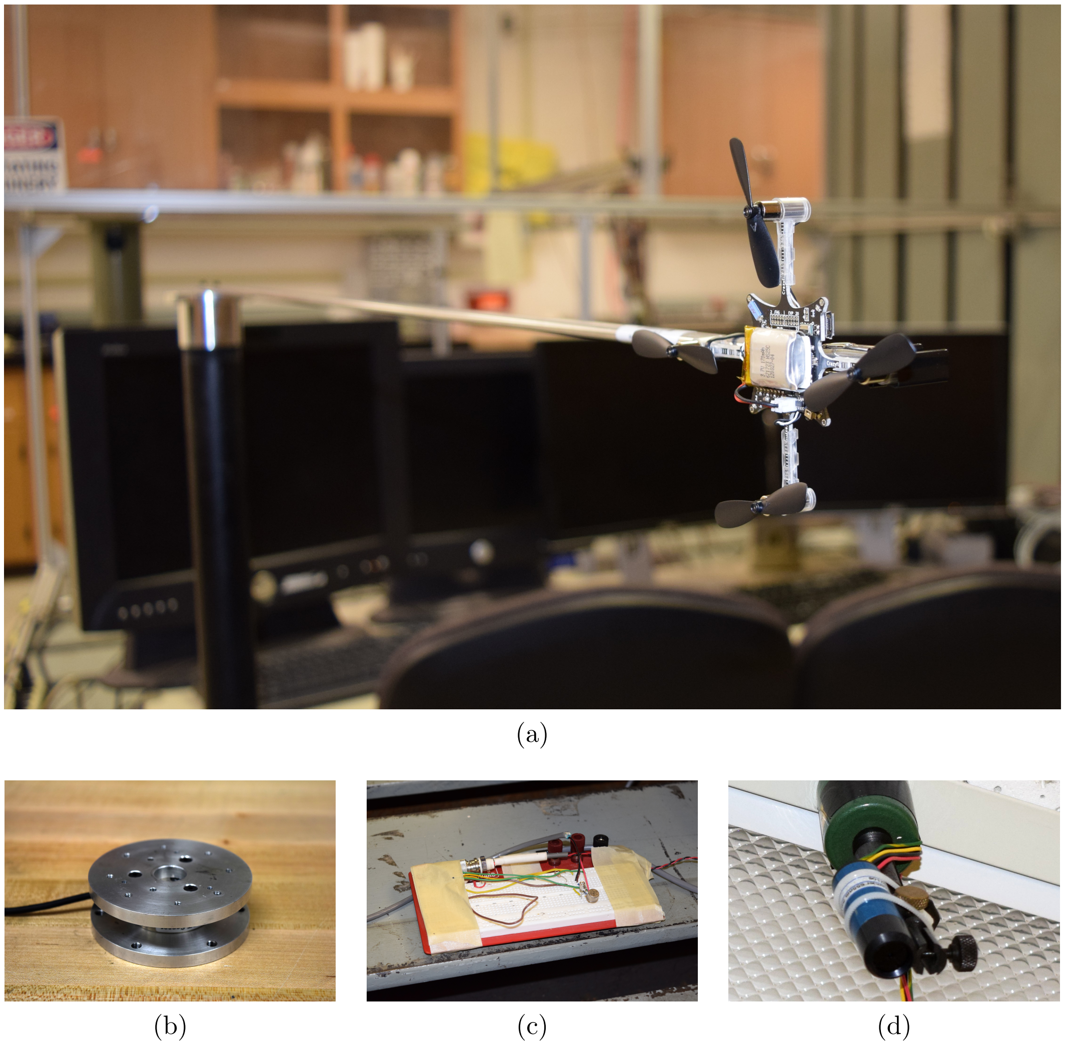


Figure 3.7: The thrust characterization test setup. (a) The quadrotor attached to the setup (b) The load cell used for force and torque measurements (c) The photo-diode (d) The laser pointer

The mean thrust force vs thrust input is curve fitted using least squares, and this is presented in Figure 3.8. Although the goodness of linear fit is sufficient with a coefficient of determination of $R^2 = 0.9944$, the second degree

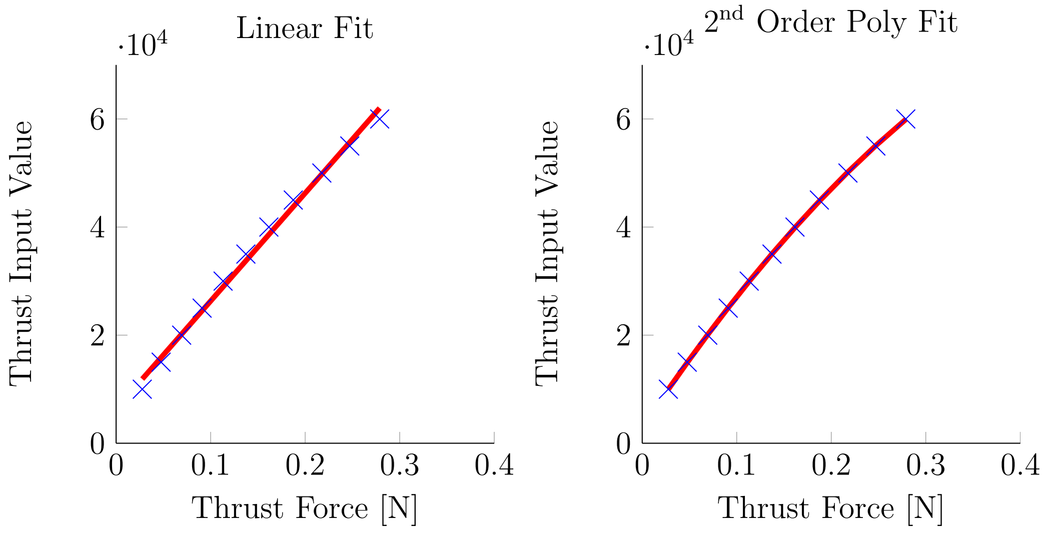


Figure 3.8: Curve fitted thrust input

polynomial fit is preferred as it provides an almost perfect fit. The thrust input is determined using the following equation of the fitted curve:

$$Thrust = -209780(m_{quad}\|U_d\|)^2 + 263090(m_{quad}\|U_d\|) + 2870. \tag{3.11}$$

3.6 Desired Thrust Direction to Attitude Input Conversion

Up until this point, the optimal trajectory generation algorithm, the motion tracking and velocity estimation and the trajectory tracking controller were all based on a lumped mass assumption. The validity of this assumption lies in the agility of a quadrotor. A quadrotor can perform the attitude maneuvers very quickly with minimal impact to the translational dynamics. Another important assumption is the thrust force the quadrotor produces is always in the body-z direction. In reality there are many aerodynamic forces acting on the propellers such as hub force and rolling moment [15]. Moreover the propeller blades are not rigid and depending on the angular momentum of the entire body, they can produce thrust forces in a direction slightly off-centered from the body-z direction. These forces and moments have little effect on translational dynamics and can be regarded as small disturbances in attitude. The quadrotor has a lot of attitude control authority and any small disturbance can be negated.

In order to produce the desired thrust, the orientation of the quadrotor should be adjusted such that the body-z axis is aligned with the desired acceleration vector. Using the rotation matrices for yaw, pitch and roll a single rotation matrix is formed consistent with most aeronautical applications [22].

$$R = \begin{bmatrix} \cos\alpha\cos\beta & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \sin\alpha\cos\beta & \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & \sin\alpha\sin\beta\cos\gamma - cos\alpha\sin\gamma \\ -\sin\beta & \cos\beta\sin\gamma & \cos\beta\cos\gamma \end{bmatrix},$$

where α is yaw, β is pitch and γ is roll. The order of operations is roll, pitch and yaw.

In order to determine the necessary orientation the following equation should be solved for given desired acceleration direction $\hat{U^d}$, where the body-z axis of the quadrotor is aligned with the desired force vector:

$$\hat{U^d} = \frac{1}{\|U^d\|} \begin{bmatrix} U_x^d \\ U_y^d \\ U_z^d \end{bmatrix} = R(\alpha, \beta, \gamma) \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}.$$

The subsequent equations are:

$$\frac{U_x^d}{\|U^d\|} = \hat{u}_x^d = \cos\alpha \sin\beta \cos\gamma + \sin\alpha \sin\gamma, \tag{3.12}$$

$$\frac{U_y^d}{\|U^d\|} = \hat{u}_y^d = \sin\alpha \sin\beta \cos\gamma - cos\alpha \sin\gamma, \tag{3.13}$$

$$\frac{U_z^d}{\|U^d\|} = \hat{u}_z^d = \cos\beta \cos\gamma. \tag{3.14}$$

This is an undetermined system and there are infinitely many solutions. For an aircraft with a fixed thrust direction relative to the body frame, the rotations around the axis of thrust have no effect on the representation of the thrust vector in the inertial frame. This can be seen physically when the quadrotor body-z axis is pointing towards a direction, it is free to rotate around its body-z axis thus changing its orientation without effecting the direction of the net thrust force vector. When switching between two orientations an axis of rotation and a rotation angle can always be found, also known as the quaternion. However rotating around this axis to change the orientation is not necessarily the optimal solution in terms of time. For example a quadrotor can generate more torque in X-mode compared to the plus mode and the moment of inertia around these two axes are relatively close. This combined with gyroscopic effects the choice of the rotation configuration is not trivial.

A number of strategies exist to find a solution and are readily implementable. One of them is to fix the heading of the quadrotor such that a constant yaw, $\alpha = 0$ is always maintained. With zero yaw, pitch and roll can be determined. From equation 3.13 for $\alpha = 0$:

$$\gamma = \arcsin(-\hat{u}_y^d), \tag{3.15}$$

and from equations 3.12 and 3.14

$$\hat{u}_x^d = \sin\beta\cos\gamma, \quad \hat{u}_z^d = \cos\beta\cos\gamma,$$

so the pitch can be determined as

$$\beta = \arctan 2(\hat{u}_x^d, \hat{u}_z^d). \tag{3.16}$$

Using equations 3.15, 3.16 and controlling the yaw to be 0, the desired pitch and roll values for the desired orientation can be determined. Unlike for roll and pitch, the gyros and accelerometers are not able to determine the yaw reliably and the integrated yaw measurement will drift. Therefore another onboard sensor like a magnetometer or an external attitude determination is necessary.

Another way of determining the orientation is finding the appropriate pitch and roll values for the current yaw and letting the onboard controller stabilize the yawrate. The yaw measurement drifts slowly but the motion tracking system is able to determine the current yaw reliably when the control input is updated. Now for variable α, the equations 3.12 and 3.13 is manipulated by multiplying with $\sin\alpha$ and $\cos\alpha$ respectively.

$$\hat{u}_x^d \sin\alpha = \sin\alpha\cos\alpha\sin\beta\cos\gamma + \sin^2\alpha\sin\gamma,$$

$$\hat{u}_y^d \cos\alpha = \sin\alpha\cos\alpha\sin\beta\cos\gamma - cos^2\alpha\sin\gamma.$$

Now we can take the difference of the two and cancel the first terms on the right sides as well as the square terms and determine the roll value.

$$\gamma = \arcsin(\hat{u}_x^d \sin\alpha - \hat{u}_y^d \cos\alpha). \tag{3.17}$$

For the pitch we can manipulate equations 3.12 and 3.13 by multiplying with $\frac{\cos\alpha}{\cos\gamma}$ and $\frac{\sin\alpha}{\cos\gamma}$ respectively.

$$\hat{u}_x^d \frac{\cos\alpha}{\cos\gamma} = \cos^2\alpha \sin\beta + \sin\alpha\cos\alpha,$$

$$\hat{u}_y^d \frac{\sin\alpha}{\cos\gamma} = \sin^2\alpha \sin\beta - \sin\alpha\cos\alpha.$$

Adding the two together and canceling the square terms we can determine $\sin\beta$ as:

$$\sin\beta = \hat{u}_x^d \frac{\cos\alpha}{\cos\gamma} + \hat{u}_y^d \frac{\sin\alpha}{\cos\gamma}. \tag{3.18}$$

To avoid singularity we can get the $\cos\beta$ term from equation 3.14,

$$\cos\beta = \frac{\hat{u}_z^d}{\cos\gamma}, \tag{3.19}$$

and use equations 3.18 and 3.19 together to determine the pitch value,

$$\beta = \arctan 2(\hat{u}_x^d \cos\alpha + \hat{u}_y^d \sin\alpha, \hat{u}_z^d). \tag{3.20}$$

It is possible to use the second method in multiple scenarios where the yaw value is prescribed and the respective pitch and roll values are determined. First the yaw can be set to be a constant value and controlled it to be so. Secondly, the yaw can be set such that the heading is in the same direction as the translational motion. This is beneficial especially if the body of the vehicle is streamlined in a way to reduce aerodynamic drag in one direction. Finally like previously mentioned, the yaw control can be left to the on board controller and the drift can be handled by updating the current yaw at the same time as the translational control input.

Chapter 4

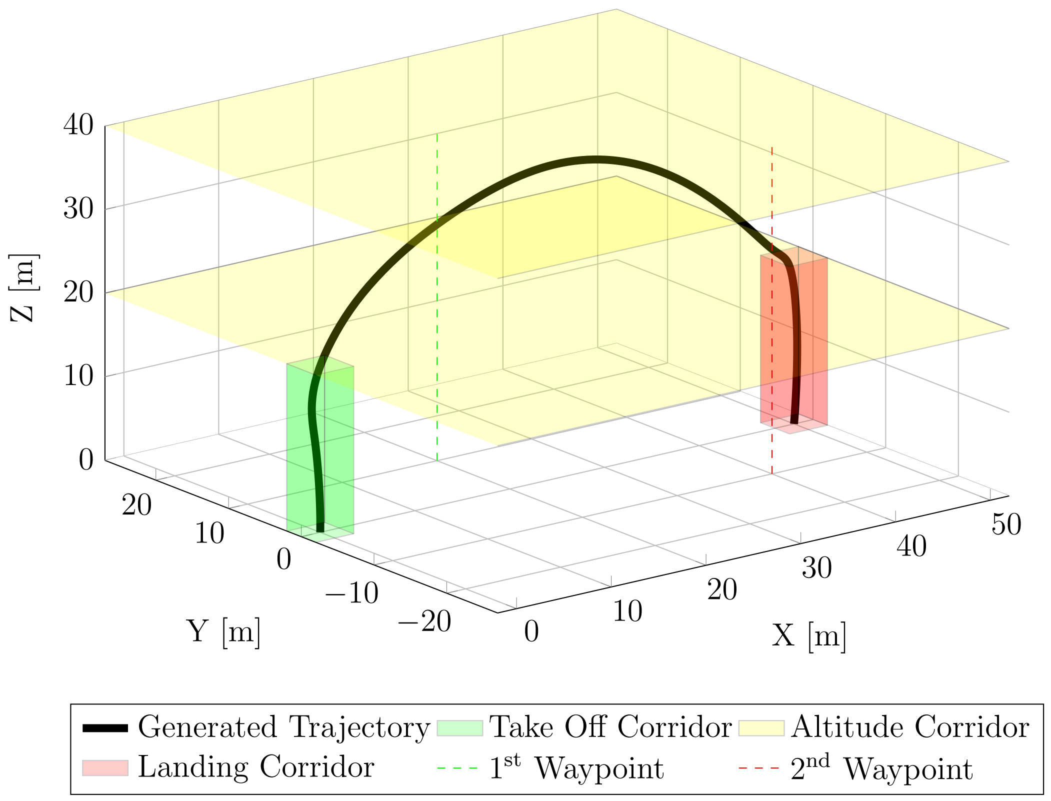
Results

In this chapter, the generated optimal trajectories and the results of the flight tests are presented. As mentioned earlier there are two path constraining strategies developed from the waypoint and the corridor path constraints. With the guidance algorithm, it is possible to create as many waypoints and corridors along the trajectory as desired. The combination of these strategies produce interesting applications and results which are elaborated in the following sections.

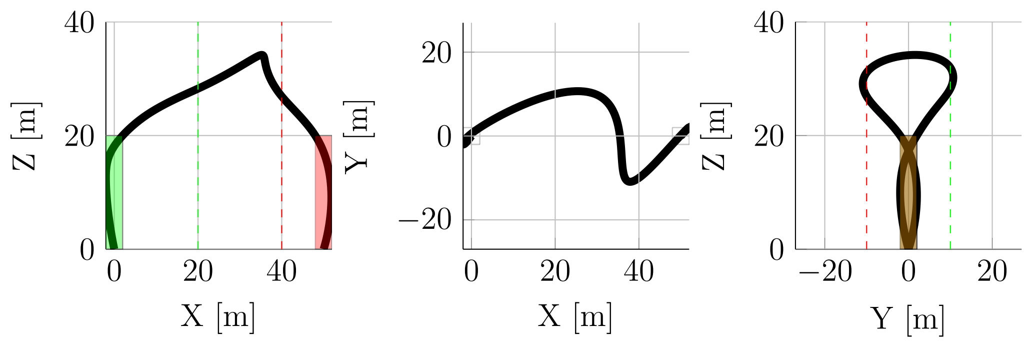
4.1 Trajectory Generation

Table 4.1: Flight parameters for the combined constraint case

State$[\mathbf{m}, \mathbf{m/s}]$	$x(\hat{t}) = \xi_i$
Initial	$\xi_0 = [0, 0, 0, 0, 0, 0]$
Final	$\xi_f = [0, 0, 50, 0, 0, 0]$
Corridor$[\mathbf{m}]$	$\beta \leq x \leq \gamma$
Take Off	$\beta = [-2, -2, \sim] \quad \gamma = [2, 2, 40]$
Altitude	$\beta = [\sim, \sim, 20] \quad \gamma = [\sim, \sim, 40]$
Landing	$\beta = [48, -2, \sim] \quad \gamma = [52, 2, 40]$
Waypoint$[\mathbf{m}]$	$x(\hat{t}) = \xi_i$
First	$\xi_{m1} = [20, 10, \sim]$
Second	$\xi_{m2} = [40, -10, \sim]$



(a)



(b)

Figure 4.1: (a) The generated optimal trajectory with the initial and final positions and the waypoint for path 1. (b) The projection of the generated trajectory to the xy xz and yz planes

Table 4.2: Flight parameters for the waypoint constraint case

$\mathbf{State}[\mathbf{m}, \mathbf{m/s}]$	$x(\hat{t}) = \xi_i$
Initial	$\xi_0 = [-1, -1, 1, 0, 0, 0]$
Final	$\xi_f = [1, 1, 2, 0, 0, 0]$
$\mathbf{Waypoint}[\mathbf{m}]$	$x(\hat{t}) = \xi_i$
First	$\xi_{m1} = [0, -1, , 1.5]$

The guidance algorithm can be used to create many different trajectories for different purposes. For example for an outdoor implementation, it is possible to define take off, altitude and landing corridors as well as many waypoints or other corridor constraints to avoid any obstacles. Such a flight scenario is simulated and the flight parameters are presented in Table 4.1. In this case, the initial and final states are stationary. The corridor constraints are applied in succession. The take off and landing corridors are the only areas where the quadrotor can touch the ground and any other time it has to stay inside an altitude range. The waypoints have no constraint on altitude.

The trajectory generated for the take-off and landing flight scenario in Table 4.1 is presented in Figure 4.1. Note that as this is an optimal trajectory it pushes the limits of the constraints. This is especially noticeable in the take off and landing corridors.

4.2 Flight Tests

The flight tests are conducted indoors with limited space and the trajectories are simpler than the previous scenario. The simulated situation is having an obstacle in the line of sight from the initial state to the desired position. Both waypoint and corridor strategies are used to handle similar situations.

4.2.1 Waypoint Strategy

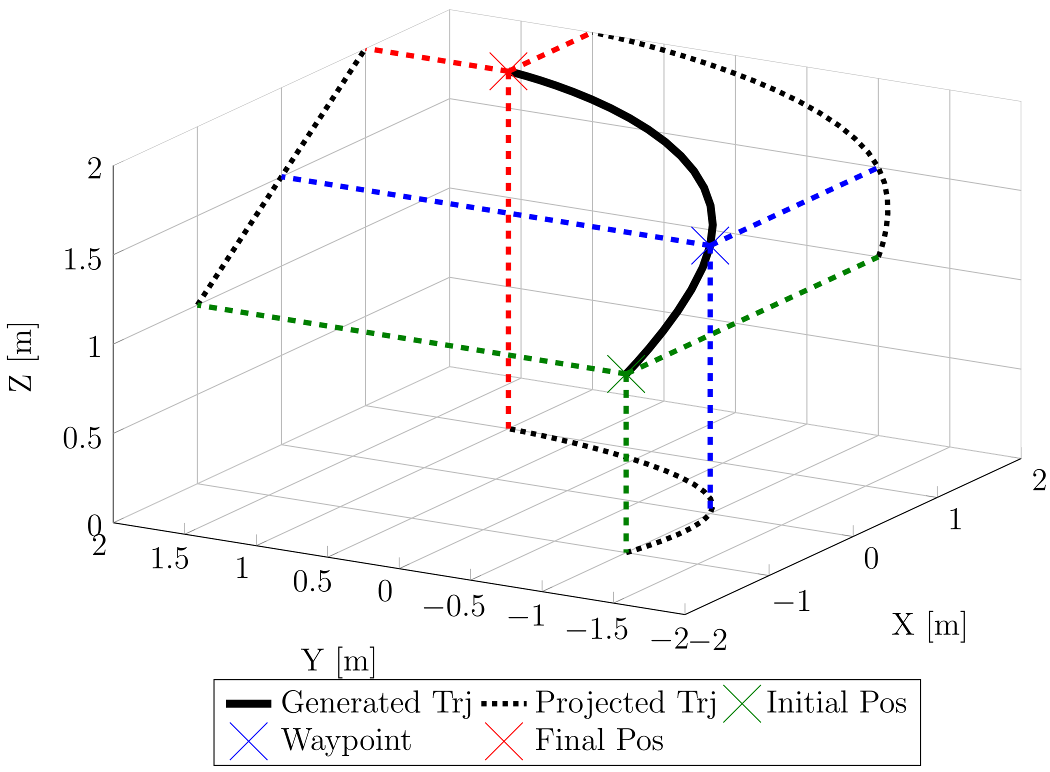


Figure 4.2: The generated optimal trajectory with the initial and final positions and the waypoint for path 1. The projections of the generated trajectory to the xy xz and yz planes are also plotted

The flight case considered for the waypoint demonstration is a forward altitude gain maneuver with an obstacle in the middle. The waypoint is chosen such that the obstacle will be avoided. The flight parameters are presented in Table 4.2 and the generated trajectory is presented in Figure 4.2. A more

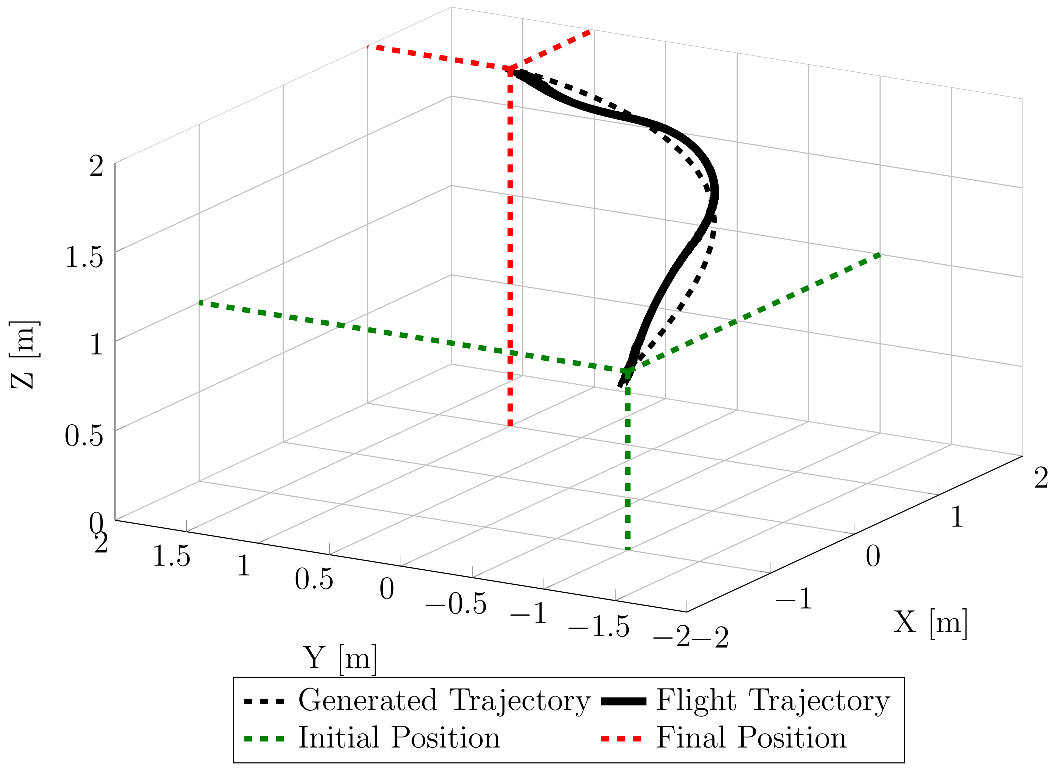


Figure 4.3: The actual trajectory and the generated trajectory of flight test of path 1

detailed break down of the desired states and input is presented in Figure 4.4. After the trajectory is created the flight tests are conducted.

The actual trajectory of the flight test is presented in Figure 4.3. The actual positions the desired positions, and the errors vs. time are presented in Figure 4.5. The actual velocities, the desired velocities, and the errors vs. time are presented in Figure 4.6. The errors are quite small and the maneuver was successful.

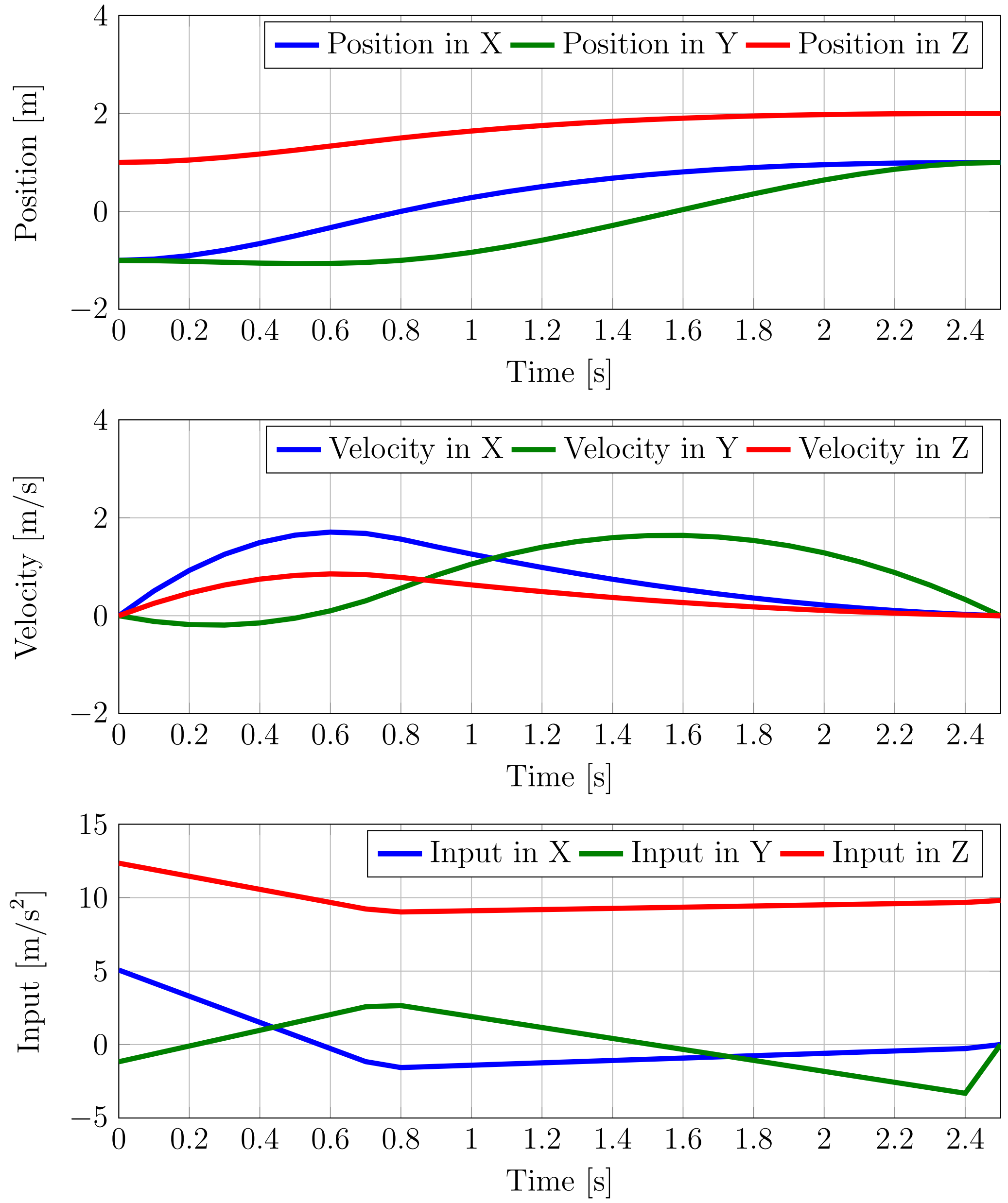


Figure 4.4: The desired states and input vs. time for path 1

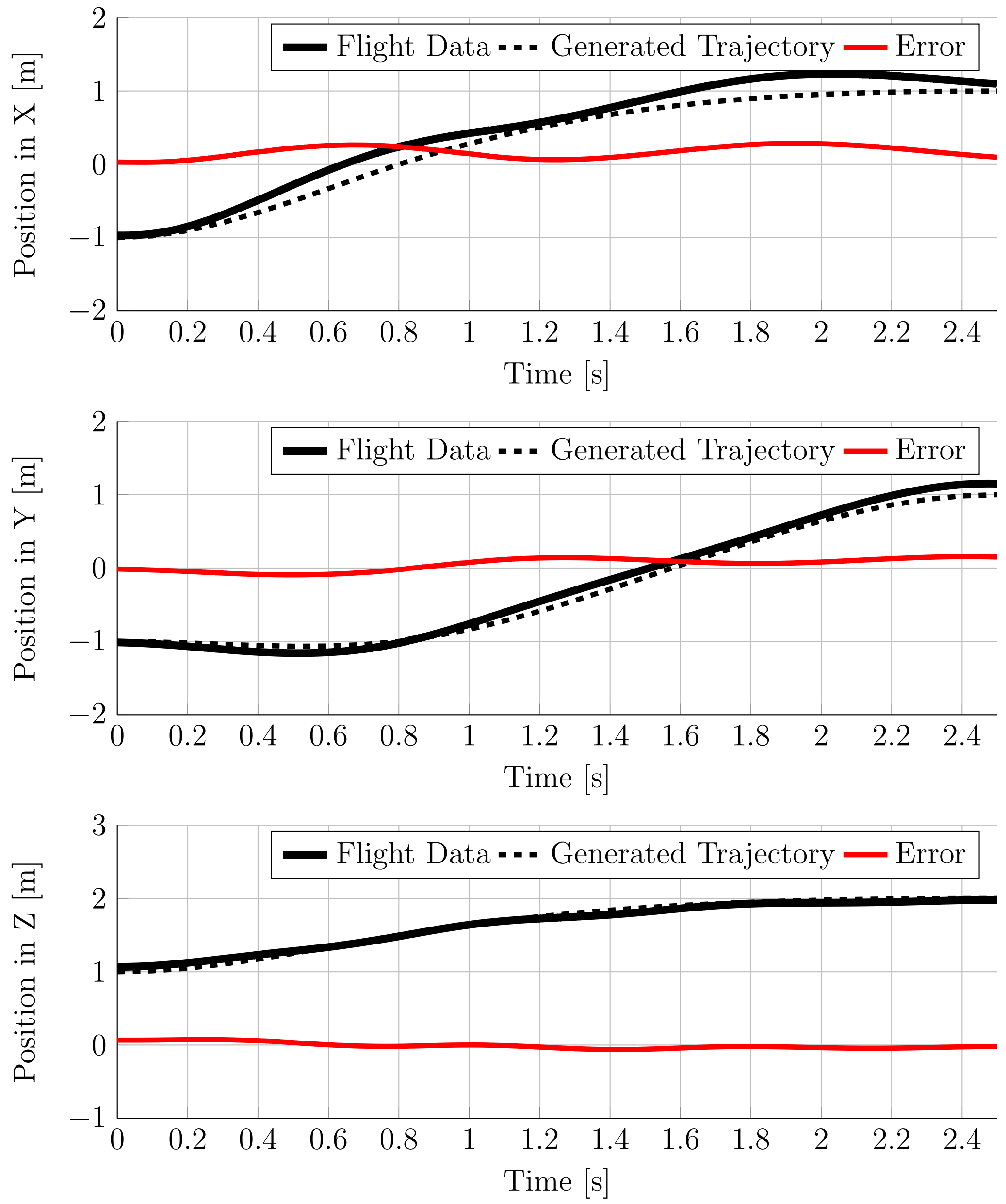


Figure 4.5: The actual positions from the flight test of path 1 with the desired positions and the errors vs. time

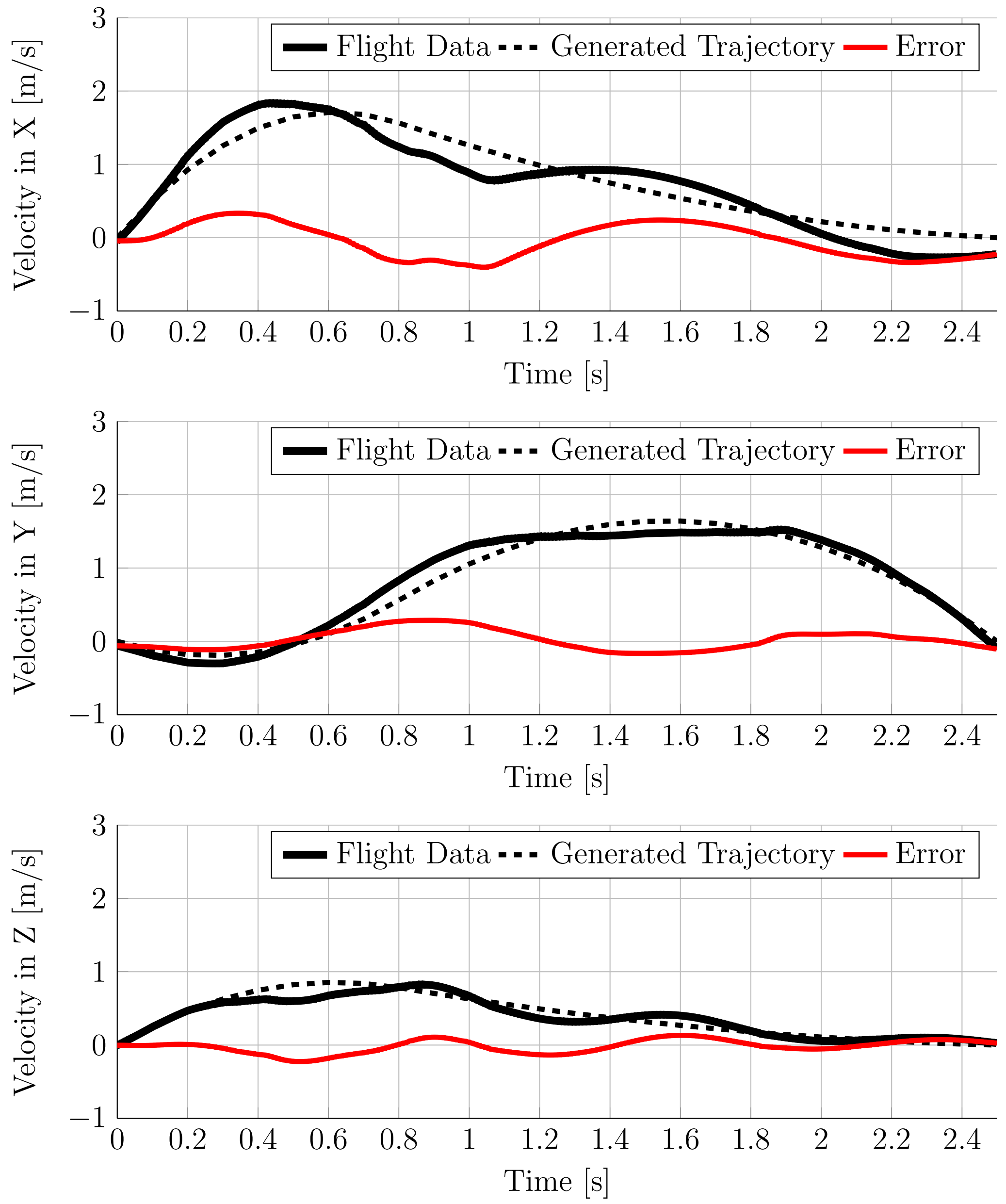


Figure 4.6: The actual velocities from the flight test of path 1 with the desired velocities and the errors vs. time

4.2.2 Corridor Strategy

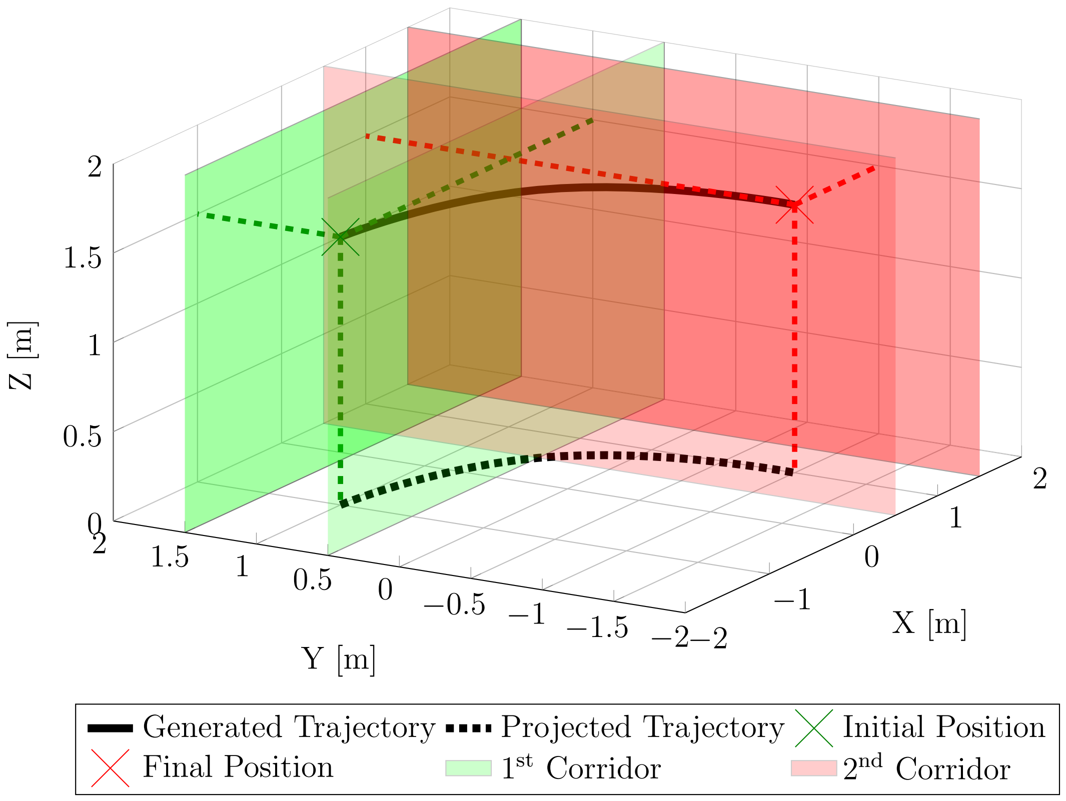


Figure 4.7: The generated optimal trajectory with the initial and final positions and the waypoint for path 2. The projection of the generated trajectory to the xy plane

The flight case considered for the corridor demonstration is a forward constant altitude maneuver with an obstacle in the middle. The corridors are chosen such that the obstacle will be avoided. The flight parameters are presented in Table 4.3 and the generated trajectory is presented in Figure 4.7. A more detailed break down of the desired states and input is presented in Figure 4.9. After the trajectory is created the flight tests are conducted.

Table 4.3: Flight parameters for the corridor constraint case

State$[\mathbf{m}, \mathbf{m/s}]$	$x(\hat{t}) = \xi_i$
Initial	$\xi_0 = [-1, -1, 1, 0, 0, 0]$
Final	$\xi_f = [1, 1, 2, 0, 0, 0]$
Waypoint$[\mathbf{m}]$	$x(\hat{t}) = \xi_i$
First	$\xi_{m1} = [0, -1, , 1.5]$

The actual trajectory of the flight test is presented in Figure 4.8. The actual positions the desired positions, and the errors vs. time are presented in Figure 4.10. The actual velocities, the desired velocities, and the errors vs. time are presented in Figure 4.11. Similar to the waypoint case the errors are quite small and the maneuver was successful.

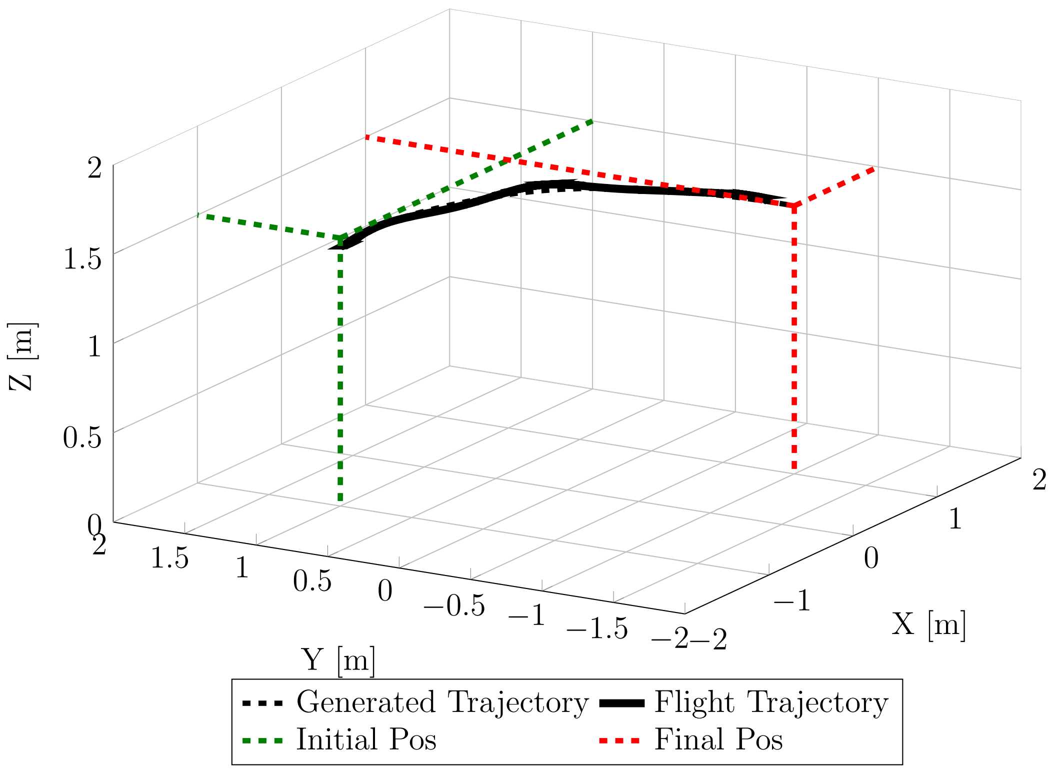


Figure 4.8: The actual trajectory and the generated trajectory of flight test of path 2

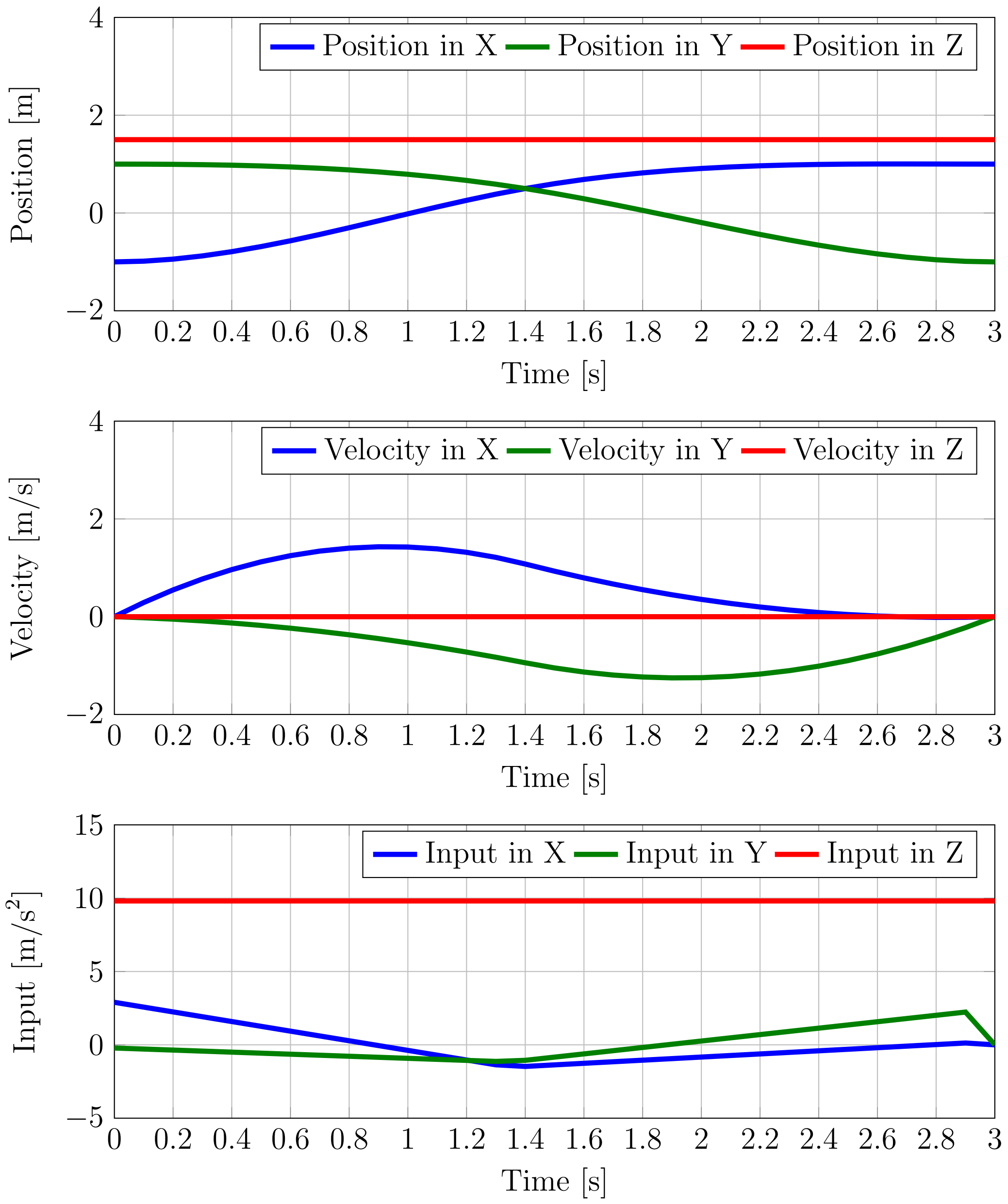


Figure 4.9: The desired states and input vs. time for path 2

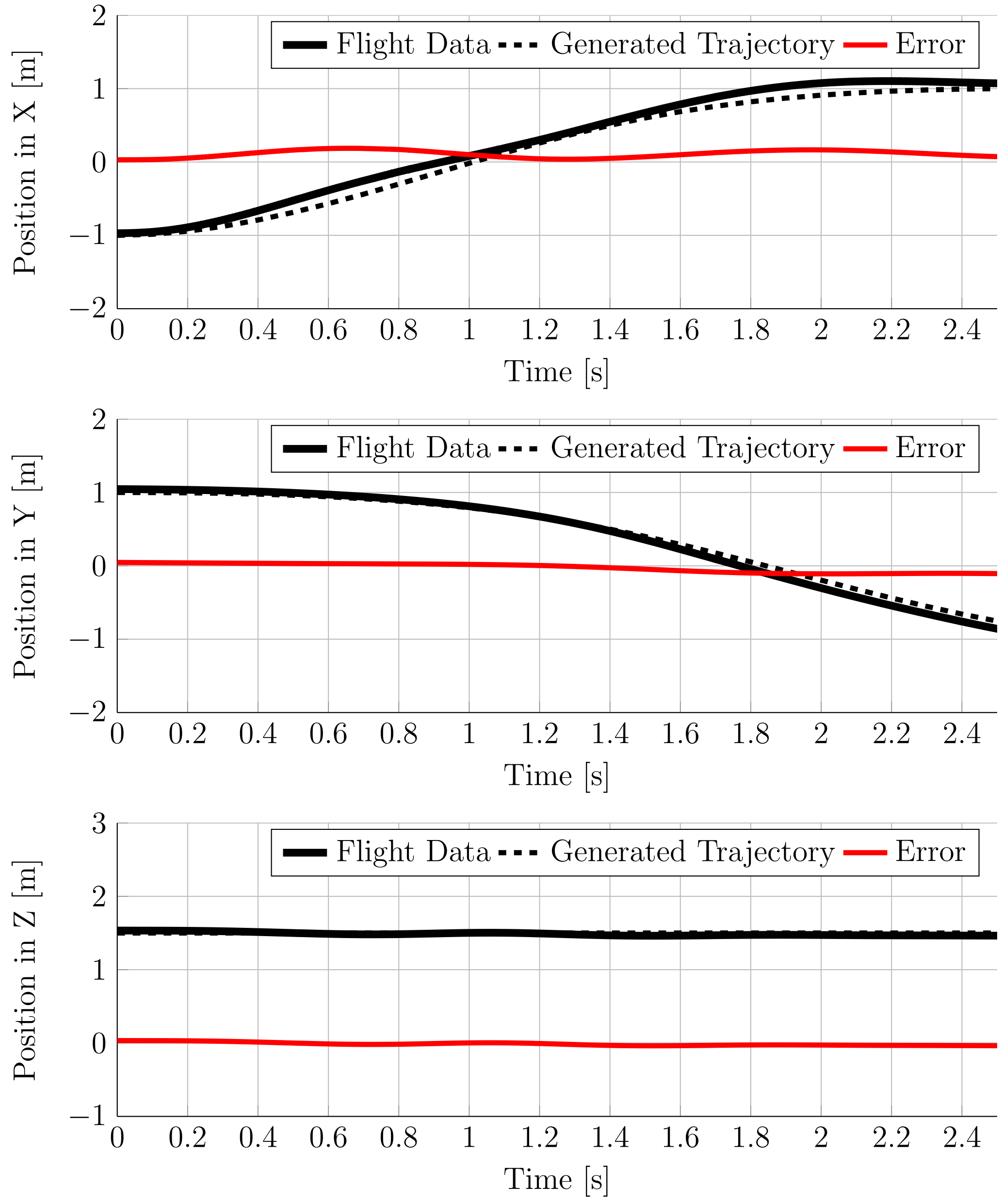


Figure 4.10: The actual positions from the flight test of path 2 with the desired positions and the errors vs. time

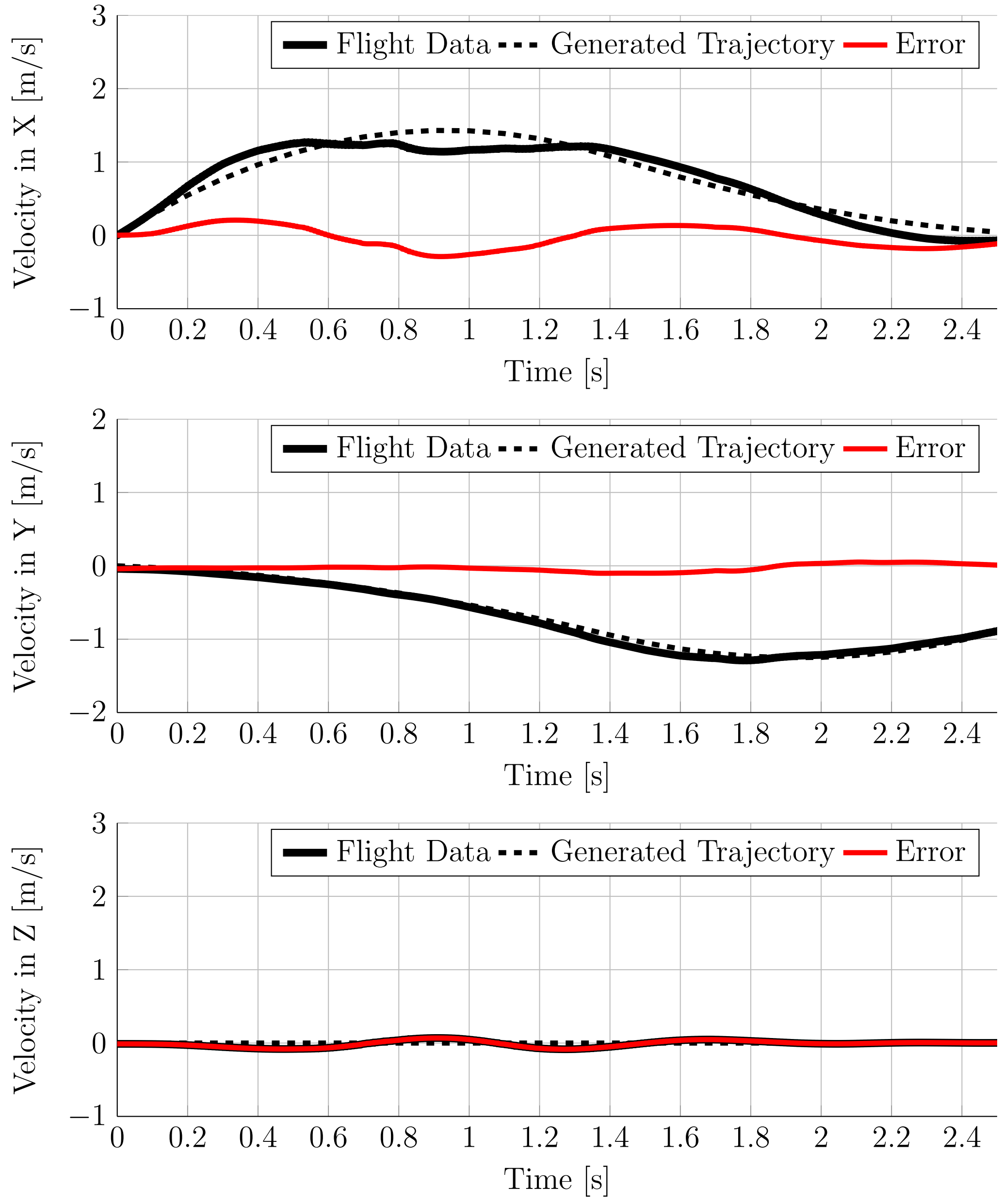


Figure 4.11: The actual velocities from the flight test of path 2 with the desired velocities and the errors vs. time

Chapter 5

Conclusion

A fuel and time optimal quadrotor guidance algorithm to autonomously generate optimal trajectories given initial and final positions and multiple physical and geometric constraints was successfully developed through this work. The algorithm is successfully used to generate optimal trajectories. An experimental setup is created around a nano quadrotor with motion tracking, velocity estimation and an external trajectory controller. Flight tests for indoor obstacle avoidance scenarios are conducted. The trajectories are tracked successfully and with high accuracy and precision.

5.1 Future Work

There are two major improvements underway. First the algorithm will be customized and an in-house developed solver will be implemented[21]. The customization of the algorithm will enable very fast solutions. With a fast algorithm, initial states with velocity components will become viable. The speed of calculations of trajectories will enable the quadrotor to update or change trajectories in mid flight. Updating trajectories will increase the overall performance. There are interesting applications that requires frequent trajectory changes, such as following a target. Additionally, if there are dynamic path constraints present, such as other moving vehicles, very fast trajectory generation is a major benefit.

The second improvement will be putting the algorithm onboard a more capable quadrotor. This will liberate the quadrotor from the ground station for guidance purposes. Combined with GPS, it will enable long distance outdoor flights. Long range and autonomy in guidance is crucial in many applications. Currently an inhouse quadrotor with high computation capabilities is being developed. A fast, reliable, robust, and onboard guidance algorithm will be an important advantage the inhouse quadrotor will have over its counterparts.

The quadrotors will be used in many projects both in single-quad and swarm formations. The swarm applications require trajectory generation for individual units. Once the general distribution of the tasks are accomplished, the members of the swarm will decide their paths individually. All in all, the guidance algorithm will become a fundamental part of future projects.

Bibliography

[1] Amazon prime air. `http://www.amazon.com/b?node=8037720011`. Accessed: 2014-07-22.

[2] Crazyflie kit electronics explained. `http://wiki.bitcraze.se/projects:crazyflie:hardware:explained`. Accessed: 2014-07-24.

[3] The crazyflie nano quadcopter. `http://www.bitcraze.se/crazyflie/`. Accessed: 2014-07-24.

[4] Crazyradio. `http://wiki.bitcraze.se/projects:crazyradio:index`. Accessed: 2014-07-24.

[5] Did amazon just pull off the best pr stunt ever? `http://www.cnbc.com/id/101239524#.` Accessed: 2014-07-22.

[6] Nbc news: Domino's 'domicopter' drone can deliver two large pepperonis. `http://www.nbcnews.com/tech/innovation/dominos-domicopter-. -drone-can-deliver-two-large-pepperonis-f6C10182466` , Accessed: 2014-07-22.

[7] Overall firmware architecture. `http://wiki.bitcraze.se/projects:crazyflie:firmware:arch`. Accessed: 2014-07-24.

[8] The real reason amazon announced delivery drones last night: $3 million in free advertising on cyber monday. `http://www.businessinsider.com/why-amazon-announced-delivery-drones-2013-12`. , Accessed: 2014-07-22.

[9] Vayu, aerial solutions serving humanity. `https://angel.co/vayu`. Accessed: 2014-08-1.

[10] B. Acikmese, M. Aung, J. Casoliva, S. Mohan, A. Johnson, D. Scharf, D. Masten, J. Scotkin, A. Wolf, and M. Regehr. Flight testing of trajectories computed by g-fold: Fuel optimal large divert guidance algorithm for planetary landing. In *23rd AAS/AIAA Spaceflight Mechanics Meeting*, Kauai, USA, To appear 2013.

[11] B. Acikmese and L. Blackmore. Lossless convexication of a class of non-convex optimal control problems for linear systems. In *American Control Conference*, Baltimore, USA, 06/2010 2010.

[12] B. Acikmese and L. Blackmore. Lossless convexification for a class of optimal control problems with nonconvex control constraints. *Automatica*, 47, 2011.

[13] B. Acikmese, J. M. Carson, and L. Blackmore. Lossless convexification of non-convex control bound and pointing constraints of the soft landing optimal control problem. *IEEE Transactions on Control Systems Technology*, 21:2104–2113, 2013 2013.

[14] Behcet Acikmese and David S. Bayard. Markov chain approach to probabilistic guidance for swarms of autonomous agents. *Invited paper, In Press, Asian Journal of Control*, 2014.

[15] S. Bouabdallah and R. Siegwart. Full control of a quadrotor. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 153–158, Oct 2007.

[16] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization.* Cambridge University Press, New York, NY, USA, 2004.

[17] Chi-Tsong Chen. *Linear System Theory and Design.* Oxford University Press, Inc., New York, NY, USA, 3rd edition, 1998.

[18] P. Cheng, Z. Shen, and S. M. Lavalle. RRT-Based Trajectory Design for Autonomous Automobiles and Spacecraft. *Archives of Control Sciences*, 11(3-4):167–194, 2001.

[19] N. Demir and B. Açıkmeşe. Density control for decentralized autonomous agents with conflict avoidance. In *IFAC World Congress*, August 2014 2014.

[20] N. Demir, B. Açıkmeşe, and M. Harris. Convex optimization formulation of density upper bound constraints in markov chain synthesis. In *American Control Conference*, June, 2014 2014.

[21] D. Dueri, J. Zhang, and B. Açıkmeşe. Automated custom code generation for embedded real-time second order cone programming. In *IFAC World Congress*, August 2014 2014.

[22] Dzul Lpez A.E. Lozano R. Pgard C. Garca Carrillo, L.R. *Quad Rotorcraft Control*. Advances in Industrial Control. Springer London, 2013.

[23] GJ Grenzdrffer, A. Engel, and B. Teichert. The photogrammetric potential of low-cost UAVs in forestry and agriculture. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 31(B3):1207–1214, 2008.

[24] Matthew W. Harris and Behcet Acikmese. Maximum divert for planetary landing using convex optimization. *Journal of Optimization Theory and Applications*, Published online, 2014.

[25] M. Hehn and R. DAndrea. Quadrocopter trajectory generation and control. In *Proceedings of the IFAC world congress*, pages 1485–1491, 2011.

[26] Amanda Hodgson, Natalie Kelly, and David Peel. Unmanned aerial vehicles (uavs) for surveying marine fauna: A dugong case study. *PLoS One*, 8(11):e79556, 2013.

[27] Matthew A. Burgess H. Franklin Percival Daniel E. Fagan Beth E. Gardner Joel G. Ortega-Ortiz Peter G. Ifju Brandon S. Evers Thomas J. Rambo

Julien Martin, Holly H. Edwards. Estimating distribution of hidden objects with drones: From tennis balls to manatees. *PLoS ONE*, (6), 2012.

[28] K. Kanistras, G. Martins, M.J. Rutherford, and K.P. Valavanis. A survey of unmanned aerial vehicles (uavs) for traffic monitoring. In *Unmanned Aircraft Systems (ICUAS), 2013 International Conference on*, pages 221–234, May 2013.

[29] James Keller, Dinesh Thakur, Vladimir Dobrokhodov, Kevin Jones, Mihail Pivtoraiko, Jean Gallier, Isaac Kaminer, and Vijay Kumar. *Unmanned Systems.*

[30] Yu-Ju Kuo and Hans D. Mittelmann. Interior point methods for second-order cone programming and or applications. *Comput. Optim. Appl.*, 28(3):255–285, September 2004.

[31] Li-Chun Lai, Chi-Ching Yang, and Chia-Ju Wu. Time-optimal control of a hovering quad-rotor helicopter. *J. Intell. Robotics Syst.*, 45(2):115–135, February 2006.

[32] Xinfu Liu. *Autonomous Trajectory Planning by Convex Optimization.* PhD thesis, Iowa State University, 2013.

[33] J. Löfberg. YALMIP : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004.

[34] J. Mattingley and S. Boyd. Real-time convex optimization in signal processing. *Signal Processing Magazine, IEEE*, 27(3):50–61, May 2010.

[35] Daniel Mellinger, Nathan Michael, and Vijay Kumar. Trajectory generation and control for precise aggressive maneuvers with quadrotors. *The International Journal of Robotics Research*, 31(5):664–674, 2012.

[36] Y. Nesterov and A. Nemirovskii. *Interior Point Polynomial Algorithms in Convex Programming.* Studies in Applied Mathematics. Society for Industrial and Applied Mathematics, 1994.

[37] Katsuhiko Ogata. *System Dynamics.* Prentice Hall, 1998.

[38] Lesley Evans Ogden. Drone ecology. *BioScience*, 63(9):776, 2013.

[39] Charles Richter, Adam Bry, and Nicholas Roy. Polynomial trajectory planning for quadrotor flight. In *IEEE International Conference on Robotics and Automation*, 2013.

[40] M Turpin, N Michael, and V Kumar. Computationally efficient trajectory planning and task assignment for large teams of unlabeled robots. In *ICRA*, May 2013.

[41] R. H. Tutuncu, K. C. Toh, and M. J. Todd. Solving semidefinite-quadratic-linear programs using sdpt3. *Mathematical Programming*, 95(2):189–217, 2003.

[42] W. Van Loock, G. Pipeleers, and J. Swevers. Time-optimal quadrotor flight. In *Control Conference (ECC), 2013 European*, pages 1788–1792, July 2013.

[43] Cé dric Vermeulen, Philippe Lejeune, Jonathan Lisein, Prosper Sawadogo, and Philippe Bouché ;. Unmanned aerial survey of elephants. *PLoS One*, 8(2):e54700, 2013.

[44] D. Verscheure, B. Demeulenaere, J. Swevers, J. De Schutter, and M. Diehl. Time-optimal path tracking for robots: A convex optimization approach. *Automatic Control, IEEE Transactions on*, 54(10):2318–2327, Oct 2009.

[45] Wei Wu. Dc motor parameter identification using speed step responses. *Model. Simul. Eng.*, 2012:30:30–30:30, January 2012.

[46] Gouqing Zhou, Chaokui Li, and Penggen Cheng. Unmanned aerial vehicle (uav) real-time video registration for forest fire monitoring. In *Geoscience and Remote Sensing Symposium, 2005. IGARSS '05. Proceedings. 2005 IEEE International*, volume 3, pages 1803–1806, July 2005.