Chapter 1 Introduction
~~do~~ about quad copters / UAVs
1.1 Motivation
1.2 Prior work
1.3 Organization of thesis

Chapter 1

Chapter 1 – introduction needs more stuff.

Introduction

⟹ talk about advances, research in the area, papers published, applications for ~~papers~~. about a page

1.1 Motivation

The technology surrounding unmanned aerial vehicles (UAV's) and in particular, quad-rotors has seen tremendous development in recent years. Likewise, the creative application of this technology has expanded into many contexts.

One pervasive engineering problem that still exists in general with UAV technology is that of managing the energy usage. Quad-rotors specifically are plagued by very high energy demand. This work is to develop an energy optimization technique that can operate on a near real-time schedule. Two different methods are discussed. We compare a classical optimal control technique with a simpler heuristic approach involving PID controller tuning.

expand on this - what is current focus among research efforts? how is energy optimization currently tackled?

make this a chapter.

this goes at the end, when you talk about chapters

1.2 Problem Statement

We wish to find a set of control expressions for a quad rotor UAV which minimizes the energy expended in flying between two known 3D points. In order to maintain

focus on a tractable problem, some mathematical assumptions are made about the scenario. First, we assume that the flight path that will be optimized is free of obstacles. Second, we assume ~~no un-~~modeled environmental variables. We use a mathematical model of the system derived from a Euler-Lagrange formulation as in [1] and [2].

In the classical optimal control approach,[3] [Kirk] [4], the control of the system and the optimization are represented in a single mathematical formulation. Solving the optimal control problem is ~~reduced to~~ solving a boundary value problem. For a highly non-linear system such as a quad rotor, this becomes ~~extremely involved.~~ For ~~our~~ heuristic approach, full control of the UAV is attained by using PD attitude controllers in conjunction with PID controllers for position. This provides a platform for simulating the UAV as it flies from a known initial position to a desired set point location. The optimization procedure evaluates the results of these simulations for optimallity as a function of the PID gains used in the position control expressions.[5] [6]

It is pertinent to define what is meant by 'near' real-time in our somewhat sterile mathematical context. We assume that the set of initial and final locations of the quad-rotor are defined ~~perhaps~~ by a user on a 'human' time scale. ~~One can~~ imagine a graphical user interface in which the desired location of the UAV is programmed. The quad-rotor then physically traverses the optimal path without more than a second or two of computation before the flight begins. ~~The algorithm which implements the heuristic approach described above comes close to this goal.~~ The classical optimal control approach is shown to be too computationally intensive for a real-time implementation because the result is a monstrous two point boundary value problem. Solving the theoretical optimal control problem would likely produce accurate results, but only after an inordinent ~~amount of number crunching on a laptop~~.

1.2

1.3 Relation to Prior Work

In this section, we describe our work in relation to the body of published work that we intend to build upon.

Excellent referecnces for the background material for understanding the dynamical model of the quad rotor as given by the Euler-Lagrange formulation are [7] and [8]. These references provided detailed derivations and discusstion of the Euler - Lagrange equations of motion as well as related topics like Hamiltonian mechanics and the calculus of variations.Also, [8] provides an in depth review of the historical context surrounding the development of classical mechanics.

The derivation of the quad rotor dynamical model as well as attitude and position control via PD or PID controllers is discussed in [9], [2], [1]. These papers provide derivations of both the Euler - Lagrange and Newtonian formulations for the quad rotor.

Several books on optimal control were used as reverences for the derivations in Chapter 3 and 4 : [4] , [3], [10], [11].

Varous numerical techniques were found in [6], [5],and [12].

There is no info on prior research, what was their conclusion, what you plan to use the research for. Just quoting references won't do.

1.3 organization

Chapter 2 deals with problem Statement, chapter 3 deals with - - - 4 - - - 5 - - - etc. till what the appendices deal with.

This chapter needs a full re-write.

- Introduction
- 3.1 what is a quad copter? Mathematical model
- 3.2 Optimal control fundamentals - E-L formulation for a generic optimization with references

Chapter 2 (3)

Quadrotor Dynamic Model

In this chapter, we will derive the QDM, and explain the set up of the E-L formulation.

2.1 Euler-Lagrangian Formulation

As in Luukkonen [1] , we start by deriving a mathematical model for the quadrotor. In order to implement a control algorithm, we must understand the mathematical relationships between the control input and the resulting dynamics of the system. Using the Euler-Lagrange formulation from classical mechanics, we can obtain a nonlinear, deterministic dynamical model. Define the translational and rotational coordinates.

ψ is the yaw angle around the z-axis

θ is the pitch angle around the y-axis

ϕ is the roll angle around the x-axis

$$\boldsymbol{\xi} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \boldsymbol{\eta} = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix}, \boldsymbol{q} = \begin{bmatrix} \xi \\ \eta \end{bmatrix}$$

- 3.3 E-L for model you are looking at (Quad copter)
- 3.4 Relation between motor speeds & System torque
- 3.5 Co-State, Coriolis matrix
- 3.6 Complete math model with states & co-states for use with the optimization problem

The coordinates $q_i = \{x, y, z, \psi, \theta, \phi\}$ are in reference to a ground based inertial coordinate system. The system states and control inputs must be mapped from the quad-rotor frame of reference to the inertial frame in order to express the dynamics of the system. The matrix below expresses an arbitrary rotation transformation from the body frame to the inertial frame is:

$$\boldsymbol{R} = \begin{bmatrix} c(\psi)c(\theta) & c(\psi)s(\theta)s(\phi) - s(\psi)c(\phi) & c(\psi)s(\theta)c(\phi) + s(\psi)s(\phi) \\ s(\psi)c(\theta) & s(\psi)s(\theta)s(\phi) + c(\psi)c(\phi) & s(\psi)s(\theta)c(\phi) - c(\psi)s(\phi) \\ -s(\theta) & c(\theta)s(\phi) & c(\theta)c(\phi) \end{bmatrix}$$

For simplicity, cos is denoted by 'c' and sin is denoted by 's' in the above expression.

The rotor angular velocities are related to the forces they produce by $f_i = s\omega_i^2$ where s is the constant of proportionallity. The torques due to the rotors about the rotors' axes of rotation are given by $\tau_{M_i} = b\omega_i^2 + I_M\dot{\omega}_i$. The parameter b is a drag coefficient. Note the effect of $\dot{\omega}_i$ is considered to be negligible because the rotational inertia of the rotor itself is negligible.

In the quad-rotor frame of reference, the motors produce torques on the system.

$$\boldsymbol{\tau}_B = \begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = \begin{bmatrix} ls(-\omega_2^2 + \omega_4^2) \\ ls(-\omega_1^2 + \omega_3^2) \\ \sum_{i=1}^{4} b\omega^2 \end{bmatrix} \tag{2.1}$$

The combined thrust of the rotors in the direction of the quad-rotor frame z axis is

$$\boldsymbol{T}_B = \begin{bmatrix} 0 \\ 0 \\ T \end{bmatrix} \text{ where,}$$

$$T = \sum_{i=1}^{4} f_i \tag{2.2}$$

The mass of the quad-rotor is m. Each of the moments of inertia i_{xx} and i_{yy} are assumed to be composed of a rod of length L which accounts for half of the mass of the quad-rotor. Assume the mass is evenly distributed along the two perpendicular rods. Let $\beta = \frac{1}{12}ml^2$. The inertia matrix for the quad-rotor is then:

$$I = \begin{pmatrix} \frac{1}{12}\left(\frac{m}{2}\right)l^2 & 0 & 0 \\ 0 & \frac{1}{12}\left(\frac{m}{2}\right)l^2 & 0 \\ 0 & 0 & \frac{1}{12}ml^2 \end{pmatrix} = \begin{pmatrix} \frac{\beta}{2} & 0 & 0 \\ 0 & \frac{\beta}{2} & 0 \\ 0 & 0 & \beta \end{pmatrix}. \tag{2.3}$$

In the inertial frame, the kinetic and potential energy of the system are given by

$$T_{\text{trans}} = \frac{1}{2}m\dot{\xi}^T\dot{\xi} \tag{2.4}$$

$$T_{\text{rot}} = \frac{1}{2}\dot{\eta}^T J\dot{\eta} \tag{2.5}$$

$$U = mgz. \tag{2.6}$$

The Lagrangian is formed as the difference between kinetic and potential energy.

$$L = \frac{1}{2}m\dot{\xi}^T\dot{\xi} + \frac{1}{2}\dot{\eta}^T J\dot{\eta} \; - mgz \tag{2.7}$$

Where J is the jacobian for transforming the angular velocities from the quad-rotor body frame to the inertial frame.

$$J = W_\eta^T I W_\eta \tag{2.8}$$

$$\boldsymbol{W}_\eta = \begin{bmatrix} 1 & 0 & -sin(\theta) \\ 0 & cos(\phi) & cos(\theta)sin(\phi) \\ 0 & -sin(\phi) & cos(\theta)cos(\phi) \end{bmatrix} \tag{2.9}$$

The matrix $\boldsymbol{W}_\eta$ is ...need a good explaination of the W matrix, there is some subtle math here that needs to be clarified

$$J = \begin{pmatrix} \frac{\beta}{2} & 0 & -\frac{\beta}{2}sin\theta \\ 0 & \frac{\beta}{2}cos^2\phi + \beta sin^2\phi & \frac{-\beta}{2}cos\phi\ sin\phi\ cos\theta \\ -\beta sin\theta & \frac{-\beta}{2}cos\phi\ sin\phi\ cos\theta & \frac{\beta}{2}sin^2\theta + \frac{\beta}{2}sin^2\phi\ cos^2\theta + \beta\ cos^2\phi\ cos^2\theta \end{pmatrix} \tag{2.10}$$

The dynamics of the system are represented by the Euler - Lagrange differential equations of motion.

$$\frac{d}{\mathrm{dt}}\left(\frac{\delta L}{\delta \dot{q}}\right) - \frac{\delta L}{\delta q} = F \tag{2.11}$$

$q = \{x, y, z, \psi, \theta, \phi\} = \{\xi, \eta\}$.

f is the generalized force vector representing the linear external force acting on the system. τ_b are the torques acting on the system due to the rotors.

$$\begin{pmatrix} f \\ \tau \end{pmatrix} = \frac{d}{\mathrm{dt}}\left(\frac{\delta L}{\delta \dot{q}}\right) - \frac{\delta L}{\delta q} \tag{2.12}$$

The linear components of the generalized forces produce the following equations.

$$f = RT_B = m\ddot{\xi} - G \tag{2.13}$$

$$m\begin{pmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{pmatrix} - \begin{pmatrix} 0 \\ 0 \\ -mg \end{pmatrix} = u\begin{pmatrix} \cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi \\ \sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi \\ \cos\theta\cos\phi \end{pmatrix} \tag{2.14}$$

The angular conponents are expressed as

$$\tau = \tau_b = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\eta}}\right) - \frac{\partial L}{\partial \eta} \tag{2.15}$$

$$\tau_b = \frac{d}{dt}(J\dot{\eta}) - \frac{1}{2}\frac{\partial}{\partial \eta}(\dot{\eta}^T J\dot{\eta}) \tag{2.16}$$

$$\tau_b = J\ddot{\eta} + \frac{d}{dt}(J)\dot{\eta} - \frac{1}{2}\frac{\partial}{\partial \eta}(\dot{\eta}^T J\dot{\eta}) \tag{2.17}$$

$$\tau_b = J\ddot{\eta} + C(\eta,\dot{\eta})\dot{\eta} \tag{2.18}$$

$$\ddot{\eta} = J^{-1}\left(\tau_b - C(\eta,\dot{\eta})\dot{\eta}\right) \tag{2.19}$$

The Coriolis matrix is defined as follows. NEED A REFERECED DOCUMENT WITH THE DERIVATION

$$\mathfrak{C}(\eta,\dot{\eta}) = \begin{pmatrix} \mathfrak{C}_{(11)} & \mathfrak{C}_{(12)} & \mathfrak{C}_{(13)} \\ \mathfrak{C}_{(21)} & \mathfrak{C}_{(22)} & \mathfrak{C}_{(23)} \\ \mathfrak{C}_{(31)} & \mathfrak{C}_{(32)} & \mathfrak{C}_{(33)} \end{pmatrix} \tag{2.20}$$

$$\mathfrak{C}_{(11)} = 0$$

$$\mathfrak{C}_{(12)} = (I_{yy} - I_{zz})(\dot{\theta}C_\phi S_\phi + \dot{\psi}C_\theta S_\phi^2) + (I_{zz} - I_{yy})\dot{\psi}C_\phi^2 C_\theta - I_{xx}\dot{\psi}C_\theta$$

$$\mathfrak{C}_{(13)} = (I_{zz} - I_{yy})\dot{\psi}C_\phi S_\phi C_\theta^2$$

$$\mathfrak{C}_{(21)} = (I_{zz} - I_{yy})(\dot{\theta}C_\phi S_\phi + \dot{\psi}S_\phi C_\theta) + (I_{yy} - I_{zz})\dot{\psi}C_\phi^2 C_\theta + I_{xx}\dot{\psi}C_\theta$$

$$\mathfrak{C}_{(22)} = (I_{zz} - I_{yy})\dot{\phi}C_\phi S_\phi$$

$$\mathfrak{C}_{(23)} = -I_{xx}\dot{\psi}S_\theta C_\theta + I_{yy}\dot{\psi}S_\phi^2 S_\theta C_\theta + I_{zz}\dot{\psi}C_\phi^2 S_\theta C_\theta$$

$$\mathfrak{C}_{(31)} = (I_{yy} - I_{zz})\dot{\psi}C_\theta^2 S_\phi C_\phi - I_{xx}\dot{\theta}C_\theta$$

$$\mathfrak{C}_{(32)} = (I_{zz} - I_{yy})(\dot{\theta}C_\phi S_\phi S_\theta + \dot{\phi}S_\phi^2 C_\theta) + (I_{yy} - I_{zz})\dot{\phi}C_\phi^2 C_\theta + I_{xx}\dot{\psi}S_\theta C_\theta - I_{yy}\dot{\psi}S_\phi^2 S_\theta C_\theta - I_{zz}\dot{\psi}C_\phi^2 S_\theta C_\theta$$

$$\mathfrak{C}_{(33)} = (I_{yy} - I_{zz})\dot{\phi}C_\phi S_\phi C_\theta^2 - I_{yy}\dot{\theta}S_\phi^2 C_\theta S_\theta - I_{zz}\dot{\theta}C_\phi^2 C_\theta S_\theta + I_{xx}\dot{\theta}C_\theta S_\theta$$

In general, the Coriolis term is a mathematical result of the rotational motion of one coordinate system with respect to another. Since we are considering abitrary three dimensional motion, there are three orthogonal axes of rotation and the resulting matrix is quite complicated. According to the expression for the angular acceleration (equation (2.19)), the physical units of the Coriolis term must be torque to maintain algebraic continuity. It is important to keep in mind that the Coriolis term does not represent a real force or torque acting on the system. It is only an artifact which is needed to account for the relative rotation of one coordinate frame with respect to another.

A complete mathematical representation of the quad-rotor is as follows.

$$\begin{pmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix} + \frac{T}{m} \begin{pmatrix} C_\psi S_\theta C_\phi + S_\psi S_\phi \\ S_\psi S_\theta C_\phi - C_\psi S_\phi \\ C_\theta C_\phi \end{pmatrix} \tag{2.21}$$

$$\begin{pmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{pmatrix} = J^{-1} \left[\begin{pmatrix} ls(-\omega_2^2 + \omega_4^2) \\ ls(-\omega_1^2 + \omega_3^2) \\ \sum_{i=1}^{4} b\omega_i^2 \end{pmatrix} - \mathfrak{C} \begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} \right] \tag{2.22}$$

This system model is admittedly rather involved and non-linear. Even in this form there are many important aspects of quadrotor flight dynamics and environmental variables which are omitted. The general problem of designing a system that is able to understand and adapt to varying goals and circumstances is a huge one.

Despite the apparent shortcomings of this model, it will proove very useful as a core component to our work here. Although the assumed (controlled) mathematical environment of our simulations is somewhat of a departure from reallty, it will allow us to maintain a firm theoretical gounding which validates the development of the control system and optimization scheme.

Chapter 3 4

Classical Optimal Control Formulation

→ In this chapter we will - - - -

3.1 Derivation of the Optimality Conditions

In section 2.3 of Bryson and Ho [3] the conditions for the optimal control of a continuous time system are derived. In this derivation, it is presumed that the system is presented as a set of first order differential equations. For a quadrotor, it is more convenient to leave the system equations as a set of second order differential equations. The motivation for this is as follows. With the finite difference method for solving two point boundary value problems, there are a set of simultaneous algebraic equations that are defined for each point in time for which a solution is desired. If we were to express the system of differential equations that govern the dynamics of a quadrotor in a first order form, the number of algebraic equations defined by the finite difference method would be effectively doubled. Here, we derive the analogous conditions for optimality for a second order system.

The dynamic equations of motion are appended to the performance index as follows.

$$0 = F - \begin{pmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \\ \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{pmatrix} \tag{3.1}$$

Note that F is the vector function representation of the quadrotor system equations. The components' physical units are linear and angular acceleration, not force.

$$J = \nu\Psi(q(t_f), t_f) + \int_{t_0}^{t_f} \left[L(q(t), u(t), t) + \lambda^T \big(F(q(t), u(t), t) - \ddot{q}\big)\right] dt \tag{3.2}$$

The Lagrangian for our problem is defined as $L[q(t), u(t), t] = u^T I u$. I is the $4x4$ identity.

Note the names 'Lagrangian' and 'Hamiltonian' are used here in an optimal control context. The ~~re-use~~ of these names in reference to specific types of expressions is an artifact of the pervasive work of Lagrange and Hamilton. We rely on a contextual and conceptual separation in our understanding. Specifically, the 'Lagrangian', ~~which~~ is formed as the difference of the expressions for kinetic and potential energies, is unique to the context of classical mechanics. Likewise, the 'Lagrangian' in the optimal control context is a term in the integrand of our objective function which represents a vector of performance metrics.

need some elaboration on what each of the quantities in the performance index really represent. need to explain why the Lagrange multipliers work , also why λ

$$J = \nu\Psi(q(t_f), t_f) - (\lambda^T \dot{q})|_{t_0}^{t_f} + (\dot{\lambda}^T q)|_{t_0}^{t_f} + \int_{t_0}^{t_f} H(q(t), u(t), t) - \ddot{\lambda}^T q \; dt \tag{3.8}$$

$$J = \nu\Psi(q(t_f), t_f) + \left[\dot{\lambda}^T q - \lambda^T \dot{q}\right]_{t_0}^{t_f} + \int_{t_0}^{t_f} \left(H(q(t), u(t), t) - \ddot{\lambda}^T q\right) \; dt \tag{3.9}$$

need to explain what a variation is and how it's different that a total derivative of a function

The first variation in J is given by

$$\delta J = \frac{\partial J}{\partial q}\delta q + \frac{\partial J}{\partial \dot{q}}\delta \dot{q} + \frac{\partial J}{\partial u}\delta u. \tag{3.10}$$

$$\begin{aligned} \delta J = {} & \nu^T \frac{\partial \Psi}{\partial q}\delta q|_{t_f} + \nu^T \frac{\partial \Psi}{\partial \dot{q}}\delta \dot{q}|_{t_f} + \left[\dot{\lambda}^T \delta q - \lambda^T \delta \dot{q}\right]_{t_0}^{t_f} \\ & + \int_{t_0}^{t_f} \left[\frac{\partial H}{\partial q}\delta q + \frac{\partial H}{\partial \dot{q}}\delta \dot{q} + \frac{\partial H}{\partial u}\delta u - \ddot{\lambda}^T \delta q\right] \; dt \end{aligned} \tag{3.11}$$

$$\begin{aligned} \delta J = {} & (\nu^T \frac{\partial \Psi}{\partial q} + \dot{\lambda}^T)\delta q|_{t_f} + (\nu^T \frac{\partial \Psi}{\partial \dot{q}} - \lambda^T)\delta \dot{q}|_{t_f} + [\lambda^T \delta \dot{q} - \dot{\lambda}^T \delta q]_{t_0} \\ & + \int_{t_0}^{t_f} \left[(\frac{\partial H}{\partial q} - \ddot{\lambda}^T)\delta q + \frac{\partial H}{\partial \dot{q}}\delta \dot{q} + \frac{\partial h}{\partial u}\delta u\right] \; dt. \end{aligned} \tag{3.12}$$

The optimality conditions are found by setting $\delta J = 0$ and asserting that each of the added terms must therefore go to 0.

becomes a function of time when optimizing a dynamical system that evolves in time.

The Hamiltonian can be written as

$$H = L(q(t), u(t), t) + \lambda^T \big(F(q(t), u(t), t)\big) \tag{3.3}$$

The performance index is simplified as:

$$J = \nu\Psi(q(t_f), t_f) + \int_{t_0}^{t_f} H(q(t), u(t), t) - \lambda^T \ddot{q} dt \tag{3.4}$$

The second term in the integrand is integrated by parts.

$$J = \nu\Psi(q(t_f), t_f) + \int_{t_0}^{t_f} H(q(t), u(t), t)\ dt - \int_{t_0}^{t_f} \lambda^T \ddot{q}\ dt \tag{3.5}$$

In general: $\int u\ dv = (uv)|_{t_0}^{t_f} - \int v\ du$

$$\int_{t_0}^{t_f} \lambda^T \ddot{q}\ dt = (\lambda^T \dot{q})|_{t_0}^{t_f} - \int_{t_0}^{t_f} \dot{\lambda}^T \dot{q}\ dt \tag{3.6}$$

This gives us:

$$J = \nu\Psi(q(t_f), t_f) + \int_{t_0}^{t_f} H(q(t), u(t), t)\ dt - (\lambda^T \dot{q})|_{t_0}^{t_f} + \int_{t_0}^{t_f} \dot{\lambda}^T \dot{q}\ dt \tag{3.7}$$

The last term is integrated by parts again.

Will augment in later after this rewrite

need an in depth explanation/interpretation of each of the optimality conditions...

From my Bryson & Ho, we can define the problem as follows.

or, from sec. 4.2, etc.

The Co state equations are

$$\frac{\partial H}{\partial q} = \ddot{\lambda} \tag{3.13}$$

$$\ddot{\lambda} = (\frac{\partial L}{\partial q})^T + (\frac{\partial F}{\partial q})^T \lambda. \tag{3.14}$$

The Stationarity Conditions are

$$\frac{\partial H}{\partial u} = 0 \tag{3.15}$$

$$\frac{\partial L}{\partial u} + (\frac{\partial F}{\partial u})^T \lambda = 0 \tag{3.16}$$

Secondary algebraic Co state condition

$$\frac{\partial H}{\partial \dot{q}} = 0 \tag{3.17}$$

$$(\frac{\partial F}{\partial \dot{q}})^T \lambda = 0 \tag{3.18}$$

Terminal Boundary conditions:

$$\nu^T \frac{\partial \Psi}{\partial q}|_{t_f} + \dot{\lambda}(t_f)^T = 0 \tag{3.19}$$

References

$$\nu^T \frac{\partial \Psi}{\partial \dot{q}}|_{t_f} - \lambda(t_f)^T = 0 \tag{3.20}$$

Initial Co state conditions

$$(\lambda^T \delta\dot{q} - \dot{\lambda}^T \delta q)|_{t_0} = 0 \tag{3.21}$$

$$\lambda(t_0) = 0 \tag{3.22}$$

$$\dot{\lambda}(t_0) = 0 \tag{3.23}$$

Together, the state equations, the costate equations, the stationarity equations, the secondary algebraic constraints, and the boundary conditions form a complete two-point boundary value problem. Two general ways to solve two-point boundary value problems are the shooting method and the finite difference method [12]. Both have limitations. The shooting method is a relatively straightforward combination of a time marching algorithm (Runga-Kutta or the like...) to solve a set of differential equations and an error minimization technique. The shooting method works by iterativly solving the set of differential equations as an initial value problem and then measureing the error in the final state of the system compaired to the desired final state. The shooting method is subject to the stability of the differential equations in question. If the time marching algorithm does not converge, the method will not work. Unfortunately, the boundary value problem for the quad rotor that is formulated in the next chapter falls into this catagory. The quad rotor system model and the coupled optimallity conditions are simply too unstable to be solved with the shooting method.

The finite difference method poses another possibility.[13] It involves creating a system of algebraic equations to be solved at each instance in time where the solution is desired. For a simulation like ours, this means at least hundreds if not thousands of timesteps. The unknowns for each set of equations for each time step are defined as the vector of unknown functions which solve the differential equations evaluated at each time step. The derivitives in the differential equations are expressed as finite differences involving variables at adjacent time steps. This makes a system of equations in (number of coupled differential equations) x (number of time steps) equations and unknowns. For a linear system this is not so bad because the problem is reduced to the inversion of a sparce matrix. For this there are efficient numerical algorithms that can be used. Since the quad rotor boundary value problem is non linear, it must be solved with a gradient descent technique or something similar.

In the next section we derive the set of differential equations which form the boundary value problem defined by our goal of optimizing the energy useage of a quad rotor.

Chapter 4

Optimal Control Boundary Value Problem Statement

4.1 Applying the Optimallity Conditions to the Quadrotor

In this section we use the expressions for the dynamic model of the quadrotor and the hamiltonian to derive the explicit optimallity conditions for our specific optimization problem.

NEED EXPLANATIONS OF EACH OF THE OPTIMALLITY CONDIONS.
NEED MORE CONTEXT IN THIS CHAPTER...

4.1.1 The Co-State equations

The Co-State equations are expressed as follows where F is our set of system equations and L is the lagrangian defined in our performance index.

$$\ddot{\lambda} = -(\frac{\partial F}{\partial q})^T \lambda - (\frac{\partial L}{\partial q})^T \tag{4.1}$$

$$\ddot{\lambda} = -(\frac{\partial F}{\partial q})^T \lambda \tag{4.2}$$

This state transition matrix of partials is tremendous, but there are some simplifications to be made. → why? non dependence of some eqns on translational elements.

$$\frac{\partial F}{\partial q} = \begin{pmatrix} \frac{\partial F_{(1)}}{\partial x} & \frac{\partial F_{(1)}}{\partial y} & \frac{\partial F_{(1)}}{\partial z} & \frac{\partial F_{(1)}}{\partial \phi} & \frac{\partial F_{(1)}}{\partial \theta} & \frac{\partial F_{(1)}}{\partial \psi} \\ \frac{\partial F_{(2)}}{\partial x} & \frac{\partial F_{(2)}}{\partial y} & \frac{\partial F_{(2)}}{\partial z} & \frac{\partial F_{(2)}}{\partial \phi} & \frac{\partial F_{(2)}}{\partial \theta} & \frac{\partial F_{(2)}}{\partial \psi} \\ \cdot & & & & & \\ \cdot & & & & & \\ \cdot & & & & & \\ \frac{\partial F_{(6)}}{\partial x} & \frac{\partial F_{(6)}}{\partial y} & \frac{\partial F_{(6)}}{\partial z} & \frac{\partial F_{(6)}}{\partial \phi} & \frac{\partial F_{(6)}}{\partial \theta} & \frac{\partial F_{(6)}}{\partial \psi} \end{pmatrix} \tag{4.3}$$

$$\frac{\partial F}{\partial q} = \begin{pmatrix} 0 & 0 & 0 & \frac{\partial F_{(1)}}{\partial \phi} & \frac{\partial F_{(1)}}{\partial \theta} & \frac{\partial F_{(1)}}{\partial \psi} \\ 0 & 0 & 0 & \frac{\partial F_{(2)}}{\partial \phi} & \frac{\partial F_{(2)}}{\partial \theta} & \frac{\partial F_{(2)}}{\partial \psi} \\ 0 & 0 & 0 & \frac{\partial F_{(3)}}{\partial \phi} & \frac{\partial F_{(3)}}{\partial \theta} & 0 \\ 0 & 0 & 0 & \frac{\partial F_{(4)}}{\partial \phi} & \frac{\partial F_{(4)}}{\partial \theta} & \frac{\partial F_{(4)}}{\partial \psi} \\ 0 & 0 & 0 & \frac{\partial F_{(5)}}{\partial \phi} & \frac{\partial F_{(5)}}{\partial \theta} & \frac{\partial F_{(5)}}{\partial \psi} \\ 0 & 0 & 0 & \frac{\partial F_{(6)}}{\partial \phi} & \frac{\partial F_{(6)}}{\partial \theta} & \frac{\partial F_{(6)}}{\partial \psi} \end{pmatrix} \tag{4.4}$$

Each of the elements must be computed numerically.

$$\frac{\partial F_i}{\partial q_j} \approx \frac{f_i(q_j + \alpha) - f_i(q_j)}{\alpha} \tag{4.5}$$

$$\ddot{\lambda} = \begin{pmatrix} \frac{\partial F_{(1)}}{\partial \phi} & \frac{\partial F_{(1)}}{\partial \theta} & \frac{\partial F_{(1)}}{\partial \psi} \\ \frac{\partial F_{(2)}}{\partial \phi} & \frac{\partial F_{(2)}}{\partial \theta} & \frac{\partial F_{(2)}}{\partial \psi} \\ \frac{\partial F_{(3)}}{\partial \phi} & \frac{\partial F_{(3)}}{\partial \theta} & 0 \\ \frac{\partial F_{(4)}}{\partial \phi} & \frac{\partial F_{(4)}}{\partial \theta} & \frac{\partial F_{(4)}}{\partial \psi} \\ \frac{\partial F_{(5)}}{\partial \phi} & \frac{\partial F_{(5)}}{\partial \theta} & \frac{\partial F_{(5)}}{\partial \psi} \\ \frac{\partial F_{(6)}}{\partial \phi} & \frac{\partial F_{(6)}}{\partial \theta} & \frac{\partial F_{(6)}}{\partial \psi} \end{pmatrix} \begin{pmatrix} \lambda_4 \\ \lambda_5 \\ \lambda_6 \end{pmatrix} \tag{4.6}$$

4.1.2 Secondary Algebraic Co-state Equations

$$\frac{\partial H}{\partial \dot{q}} = 0 \tag{4.7}$$

$$0 = (\frac{\partial F}{\partial \dot{q}})^T \lambda + (\frac{\partial L}{\partial \dot{q}})^T \tag{4.8}$$

$$0 = (\frac{\partial F}{\partial \dot{q}})^T \lambda \tag{4.9}$$

$$0 = \begin{pmatrix} \frac{\partial F_{(1)}}{\partial \dot{x}} & \frac{\partial F_{(1)}}{\partial \dot{y}} & \frac{\partial F_{(1)}}{\partial \dot{z}} & \frac{\partial F_{(1)}}{\partial \dot{\phi}} & \frac{\partial F_{(1)}}{\partial \dot{\theta}} & \frac{\partial F_{(1)}}{\partial \dot{\psi}} \\ \frac{\partial F_{(2)}}{\partial \dot{x}} & \frac{\partial F_{(2)}}{\partial \dot{y}} & \frac{\partial F_{(2)}}{\partial \dot{z}} & \frac{\partial F_{(2)}}{\partial \dot{\phi}} & \frac{\partial F_{(2)}}{\partial \dot{\theta}} & \frac{\partial F_{(2)}}{\partial \dot{\psi}} \\ \frac{\partial F_{(3)}}{\partial \dot{x}} & \frac{\partial F_{(3)}}{\partial \dot{y}} & \frac{\partial F_{(3)}}{\partial \dot{z}} & \frac{\partial F_{(3)}}{\partial \dot{\phi}} & \frac{\partial F_{(3)}}{\partial \dot{\theta}} & \frac{\partial F_{(3)}}{\partial \dot{\psi}} \\ \frac{\partial F_{(4)}}{\partial \dot{x}} & \frac{\partial F_{(4)}}{\partial \dot{y}} & \frac{\partial F_{(4)}}{\partial \dot{z}} & \frac{\partial F_{(4)}}{\partial \dot{\phi}} & \frac{\partial F_{(4)}}{\partial \dot{\theta}} & \frac{\partial F_{(4)}}{\partial \dot{\psi}} \\ \frac{\partial F_{(5)}}{\partial \dot{x}} & \frac{\partial F_{(5)}}{\partial \dot{y}} & \frac{\partial F_{(5)}}{\partial \dot{z}} & \frac{\partial F_{(5)}}{\partial \dot{\phi}} & \frac{\partial F_{(5)}}{\partial \dot{\theta}} & \frac{\partial F_{(5)}}{\partial \dot{\psi}} \\ \frac{\partial F_{(6)}}{\partial \dot{x}} & \frac{\partial F_{(6)}}{\partial \dot{y}} & \frac{\partial F_{(6)}}{\partial \dot{z}} & \frac{\partial F_{(6)}}{\partial \dot{\phi}} & \frac{\partial F_{(6)}}{\partial \dot{\theta}} & \frac{\partial F_{(6)}}{\partial \dot{\psi}} \end{pmatrix}^T \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \\ \lambda_5 \\ \lambda_6 \end{pmatrix} \tag{4.10}$$

This matrix can simplify considerably. → why? (as in 5.1.1, non-dependence etc)

$$0 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{\partial F_{(4)}}{\partial \dot{\phi}} & \frac{\partial F_{(4)}}{\partial \dot{\theta}} & \frac{\partial F_{(4)}}{\partial \dot{\psi}} \\ 0 & 0 & 0 & \frac{\partial F_{(5)}}{\partial \dot{\phi}} & \frac{\partial F_{(5)}}{\partial \dot{\theta}} & \frac{\partial F_{(5)}}{\partial \dot{\psi}} \\ 0 & 0 & 0 & \frac{\partial F_{(6)}}{\partial \dot{\phi}} & \frac{\partial F_{(6)}}{\partial \dot{\theta}} & \frac{\partial F_{(6)}}{\partial \dot{\psi}} \end{pmatrix}^T \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \\ \lambda_5 \\ \lambda_6 \end{pmatrix} \tag{4.11}$$

$$0 = \begin{pmatrix} \frac{\partial F_{(4)}}{\partial \dot{\phi}} & \frac{\partial F_{(4)}}{\partial \dot{\theta}} & \frac{\partial F_{(4)}}{\partial \dot{\psi}} \\ \frac{\partial F_{(5)}}{\partial \dot{\phi}} & \frac{\partial F_{(5)}}{\partial \dot{\theta}} & \frac{\partial F_{(5)}}{\partial \dot{\psi}} \\ \frac{\partial F_{(6)}}{\partial \dot{\phi}} & \frac{\partial F_{(6)}}{\partial \dot{\theta}} & \frac{\partial F_{(6)}}{\partial \dot{\psi}} \end{pmatrix}^T \begin{pmatrix} \lambda_4 \\ \lambda_5 \\ \lambda_6 \end{pmatrix} \tag{4.12}$$

partial derivatives

...Again the partials in this matrix must be computed numerically as

$$\frac{\partial F_i}{\partial \dot{q}_j} \approx \frac{F_i(\dot{q}_j + \alpha) - F_i(\dot{q}_j)}{\alpha} \tag{4.13}$$

4.1.3 Stationarity Conditions

$H = \lambda^T F + L$

$\frac{\partial H}{\partial u} = \lambda^T \frac{\partial F}{\partial u} + \frac{\partial L}{\partial u}$

Taking the Transpose!

never just write equations the must be some words as well, like: we know,

$$(\frac{\partial H}{\partial u})^T = (\frac{\partial F}{\partial u})^T \lambda + (\frac{\partial L}{\partial u})^T = 0 \tag{4.14}$$

$$\frac{\partial F}{\partial q} = \begin{pmatrix} \frac{\partial F_{(1)}}{\partial u_1} & \frac{\partial F_{(1)}}{\partial u_2} & \frac{\partial F_{(1)}}{\partial u_3} & \frac{\partial F_{(1)}}{\partial u_4} \\ \frac{\partial F_{(2)}}{\partial u_1} & \frac{\partial F_{(2)}}{\partial u_2} & \frac{\partial F_{(2)}}{\partial u_3} & \frac{\partial F_{(2)}}{\partial u_4} \\ \frac{\partial F_{(3)}}{\partial u_1} & \frac{\partial F_{(3)}}{\partial u_2} & \frac{\partial F_{(3)}}{\partial u_3} & \frac{\partial F_{(3)}}{\partial u_4} \\ \vdots & & & \\ \frac{\partial F_{(6)}}{\partial u_1} & \frac{\partial F_{(6)}}{\partial u_2} & \frac{\partial F_{(6)}}{\partial u_3} & \frac{\partial F_{(6)}}{\partial u_4} \end{pmatrix} \tag{4.15}$$

why?

$$\frac{\partial L}{\partial u} = 2(u_1, u_2, u_3, u_4) \tag{4.16}$$

$$0 = \begin{pmatrix} \frac{\partial F_{(1)}}{\partial u_1} & \frac{\partial F_{(2)}}{\partial u_1} & \frac{\partial F_{(3)}}{\partial u_1} & \frac{\partial F_{(4)}}{\partial u_1} & \frac{\partial F_{(5)}}{\partial u_1} & \frac{\partial F_{(6)}}{\partial u_1} \\ \frac{\partial F_{(1)}}{\partial u_2} & \frac{\partial F_{(2)}}{\partial u_2} & \frac{\partial F_{(3)}}{\partial u_2} & \frac{\partial F_{(4)}}{\partial u_2} & \frac{\partial F_{(5)}}{\partial u_2} & \frac{\partial F_{(6)}}{\partial u_2} \\ \frac{\partial F_{(1)}}{\partial u_3} & \frac{\partial F_{(2)}}{\partial u_3} & \frac{\partial F_{(3)}}{\partial u_3} & \frac{\partial F_{(4)}}{\partial u_3} & \frac{\partial F_{(5)}}{\partial u_3} & \frac{\partial F_{(6)}}{\partial u_3} \\ \frac{\partial F_{(1)}}{\partial u_4} & \frac{\partial F_{(2)}}{\partial u_4} & \frac{\partial F_{(3)}}{\partial u_4} & \frac{\partial F_{(4)}}{\partial u_4} & \frac{\partial F_{(5)}}{\partial u_4} & \frac{\partial F_{(6)}}{\partial u_4} \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \\ \lambda_5 \\ \lambda_6 \end{pmatrix} + 2 \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{pmatrix} \tag{4.17}$$

As before, the partials are numerically computed as:

$$\frac{\partial F_i}{\partial u_j} \approx \frac{F_i(u_j + \alpha) - F_i(u_j)}{\alpha} \tag{4.18}$$

4.2 Discretization

The continuous system equations were shown in chapter 3 as:

$$\begin{pmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix} + \frac{T}{m} \begin{pmatrix} C_\psi S_\theta C_\phi + S_\psi S_\phi \\ S_\psi S_\theta C_\phi - C_\psi S_\phi \\ C_\theta C_\phi \end{pmatrix} \tag{4.19}$$

Same eq no as ch. 3

$$\begin{pmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{pmatrix} = J^{-1} \left[\begin{pmatrix} ls(-\omega_2^2 + \omega_4^2) \\ ls(-\omega_1^2 + \omega_3^2) \\ \sum_{i=1}^{4} b\omega_i^2 \end{pmatrix} - \mathfrak{C} \begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} \right] \tag{4.20}$$

The discrete system equations using Euler's two sided approximation, are:

$$\begin{pmatrix} \frac{x[k+1]-2x[k]+x[k-1]}{h^2} \\ \frac{y[k+1]-2y[k]+y[k-1]}{h^2} \\ \frac{z[k+1]-2z[k]+z[k-1]}{h^2} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix} + \frac{T[k]}{m} \begin{pmatrix} C_{\psi[k]} S_{\theta[k]} C_{\phi[k]} + S_{\psi[k]} S_{\phi[k]} \\ S_{\psi[k]} S_{\theta[k]} C_{\phi[k]} - C_{\psi[k]} S_{\phi[k]} \\ C_{\theta[k]} C_{\phi[k]} \end{pmatrix} \tag{4.21}$$

use \displaystyle before fractions

$$\begin{pmatrix} \frac{\phi[k+1]-2\phi[k]+\phi[k-1]}{h^2} \\ \frac{\theta[k+1]-2\theta[k]+\theta[k-1]}{h^2} \\ \frac{\psi[k+1]-2\psi[k]+\psi[k-1]}{h^2} \end{pmatrix} = J^{-1}[k] \left[\begin{pmatrix} ls(-\omega_2[k]^2 + \omega_4[k]^2) \\ ls(-\omega_1[k]^2 + \omega_3[k]^2) \\ \sum_{i=1}^{4} b\omega_i[k]^2 \end{pmatrix} - \mathfrak{C}[k] \begin{pmatrix} \frac{\phi[k]-\phi[k-1]}{h} \\ \frac{\theta[k]-\theta[k-1]}{h} \\ \frac{\psi[k]-\psi[k-1]}{h} \end{pmatrix} \right] \tag{4.22}$$

Assign a change of variables:

$$q = \begin{pmatrix} x \\ y \\ z \\ \phi \\ \theta \\ \psi \end{pmatrix} = \begin{pmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \\ q_5 \\ q_6 \end{pmatrix} \tag{4.23}$$

results?

~~Comp~~

5.3 → Computational results
- shooting method, finite difference method.

5.4 → Conclusions
→ all the negatives, and so we can't use this

Your voice is changing active to passive / passive to active in paragraphs. They should be the same!

Chapter 5 → 6

PID/PD Control

What is done in this chapter?
→ Introduce PID control!!

5.1 → 6.2 Deriving the Control Expressions

The control of the quadrotor requires three independent PID conrollers for the x, y, and z directions. In addition, the attitude stabillity of the aircraft is accomplished by three independent PD controllers for each of the Euler angles (ϕ, θ, ψ) . It is assumed for the purpose of simulation that the input to the control expressons includes precise knowledge of the system state. In other words, it is assumed that the process noise and the measurement noise are zero. Given the natural complexity of the system, inclusion of stochastic processes into the model is left for further work. The control algorithm procedes as follows.

Algorithm 6.1

Step 1 ↓

- The position control expressions give the 'commanded' linear accelerations that are required to drive the system to the desired state.
- Given the commanded linear accelerations, the nesscary total thrust, pitch, and roll are determined
- The commanded torqes about the three axes of the quad rotor are given by PD controllers using the commanded pitch and roll as angular set points.

- Given the commanded total thrust and the commanded torques, the motor speeds can be determined.
- Once the motor speeds are known, the system model can be used to obtain the ubdated state of the system
- repeat...

references

need to go figure out what paper this came from (Z. Zau??)
also need a picture showing the linear axes and each motor and angular direction that it corresponds to...

Our goal in the following derivation is to arrive at expressions for the motor speeds that are required to drive the system to the desired state.

$$P_c = \begin{pmatrix} x_c \\ y_c \\ z_c \end{pmatrix} = \textit{desired (commanded) set point location}$$

$$P = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \textit{actual position at time step } k$$

The descrete-time PID control expssions are formulated using these vectors.

$$\ddot{P}_c = k_p(P_c - P) + k_i \sum_k (P_c - P) + k_d(\dot{P}_c - \dot{P}) \tag{5.1}$$

$\ddot{P}_c$ is the vector of commanded accelerations.

$$\ddot{P}_c = \begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix} \tag{5.2}$$

$$\ddot{P}_c = -ge_{inz} + (\frac{1}{m})(Te_{qrz})R \tag{5.3}$$

$$e_{inz} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \quad \text{(in the inertial reference frame)}$$

$$e_{qrz} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \quad \text{(in the quad rotor reference frame)}$$

ϕ and θ, and the total thrust T can be determined algebraically, assuming we know $\ddot{P}_c$ and ψ.

$$R^T(\ddot{P}_c + ge_{inz}) = (\frac{1}{m})(Te_{qrz}) \tag{5.4}$$

$$\begin{pmatrix} c(\psi)c(\theta) & s(\psi)c(\theta) & -s(\theta) \\ c(\psi)s(\theta)s(\phi) - s(\psi)c(\phi) & s(\psi)s(\theta)s(\phi) + c(\psi)c(\phi) & c(\theta)s(\phi) \\ c(\psi)s(\theta)c(\phi) + s(\psi)s(\phi) & s(\psi)s(\theta)c(\phi) - c(\psi)s(\phi) & c(\theta)c(\phi) \end{pmatrix} \begin{pmatrix} a_x \\ a_y \\ a_z + g \end{pmatrix}$$
$$= \frac{1}{m} \begin{pmatrix} 0 \\ 0 \\ T \end{pmatrix} \tag{5.5}$$

The matrix equation above is then written as three independent scalar expressions.

$$a_x c(\psi)c(\theta) + a_y s(\psi)c(\theta) - s(\theta)(a_z + g) = 0 \tag{5.6}$$

try to extend it to fit a line

$$\begin{aligned} & a_x(c(\psi)s(\theta)s(\phi) - s(\psi)c(\phi)) \\ & + a_y(s(\psi)s(\theta)s(\phi) + c(\psi)c(\phi)) \\ & + (a_z + g)c(\theta)s(\phi) = 0 \end{aligned} \tag{5.7}$$

$$\begin{aligned} & a_x(c(\psi)s(\theta)c(\phi) + s(\psi)s(\phi)) \\ & + a_y(s(\psi)s(\theta)c(\phi) - c(\psi)s(\phi)) \\ & + (a_z + g)c(\theta)c(\phi) = (\frac{T}{m}) \end{aligned} \tag{5.8}$$

Next, we divide (5.6) by $c(\theta)$ and solve for θ.

$$a_x c(\psi) + a_y s(\psi) + (a_z + g)(-\tan(\theta)) = 0 \tag{5.9}$$

$$\theta_c = \arctan(\frac{a_x c(\psi) + a_y s(\psi)}{a_z + g}) \tag{5.10}$$

Next: (5.7) x $s(\phi)$ - (5.8) x $c(\phi)$.... The result is

$$\phi = \arcsin(\frac{a_x s(\psi) - a_y c(\psi)}{T/m}). \tag{5.11}$$

Square both sides of (5.4) and note that $R^T = R^{-1}$.

$$a_x^2 + a_y^2 + (a_z + g)^2 = (\frac{T}{m})^2 \tag{5.12}$$

$$(\frac{T}{m}) = \sqrt{a_x^2 + a_y^2 + (a_z + g)^2} \tag{5.13}$$

This result is then substituted back in (eq19) to give

$$\phi_c = \arcsin\left(\frac{a_x s(\psi) - a_y c(\psi)}{\sqrt{a_x^2 + a_y^2 + (a_z + g)^2}}\right) \tag{5.14}$$

Using θ_c and ϕ_c as set points, we can write the PD angular control laws. The subscript 'c' stands for 'commanded'.

$$\tau_{\phi c} = [k_{p\phi}(\phi_c - \phi) + k_{d\phi}(\dot{\phi}_c - \dot{\phi})]I_x \tag{5.15}$$

$$\tau_{\theta c} = [k_{p\theta}(\theta_c - \theta) + k_{d\theta}(\dot{\theta}_c - \dot{\theta})]I_y \tag{5.16}$$

$$\tau_{\psi c} = [k_{p\psi}(\psi_c - \psi) + k_{d\psi}(\dot{\psi}_c - \dot{\psi})]I_z \tag{5.17}$$

Given the commanded torques and the commanded total thrust, the commanded motor speeds can be obtained by inverting the expressions from chapter 2 (need equation referencing....

$$\omega_{1c} = \sqrt{\frac{T_c}{4k} - \frac{\tau_{\theta c}}{2kL} - \frac{\tau_{\psi c}}{4b}} \tag{5.18}$$

$$\omega_{2c} = \sqrt{\frac{T_c}{4k} - \frac{\tau_{\phi c}}{2kL} + \frac{\tau_{\psi c}}{4b}} \tag{5.19}$$

$$\omega_{3c} = \sqrt{\frac{T_c}{4k} + \frac{\tau_{\theta c}}{2kL} - \frac{\tau_{\psi c}}{4b}} \tag{5.20}$$

$$\omega_{4c} = \sqrt{\frac{T_c}{4k} + \frac{\tau_{\phi c}}{2kL} + \frac{\tau_{\psi c}}{4b}} \tag{5.21}$$

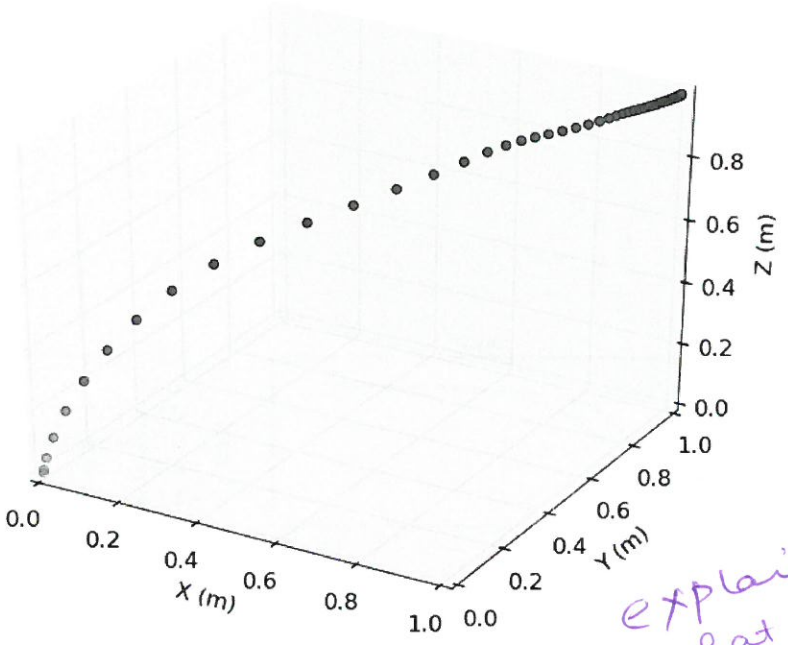
With the above results, we can summerize the control loop. The PID control expressions prescribe linear accelerations in each direction (x, y, z) which will drive the system toward the desired position. The linear accelerations are used to calculate the angles ϕ and θ, and the total thrust T. Given the angles and their time derivatives, the prescribed torques about the quad rotor center of mass are given by PD control laws. Given the torques and the total thrust, the vector of motor speeds can be calculated.

In a physical implementation, after the motor speeds are updated, the state of the system would be estimated from whatever sensor data is available. The environmental context would dictate which type of sensor hardware would be appropriate. In a simulation context, we use the dynamical system model from chapter 2 to evaluate the resulting motion of the system. In a sense, this process is just the inverse of the control loop. For further work, a random process could be included here to model sensor noise. This would give a nice simulation platform for evaluating the performance of a Kalman filter for estimating the state of the quad rotor.

5.2 Testing the Control Scheme

Add a picture and output dictionary for a typical run ...NEED TO TALK ABOUT STOPPING CONDITIONS A BIT

With experimentally tuned control expressions, arbitrarily shaped, sub-optimal paths can be formed by updating the desired location periodically. Figures 5.3 through 5.6 show the utility of the control scheme. It is important to note that



Explain what each plot represents & for what values - in title as well.

FIGURE 5.1: A typical run - the 3D path

in our simulations, the desired velocity at each of the ordered set points is zero. In words, the control algorithm is saying to the quadrotor: 'Go to the desired location and hover until the set point is updated'. To design a path that includes set points with a non-zero desired velocity vector would require modification of the algorithm.

More details. What were the values of m, l, g, etc that you have used here?

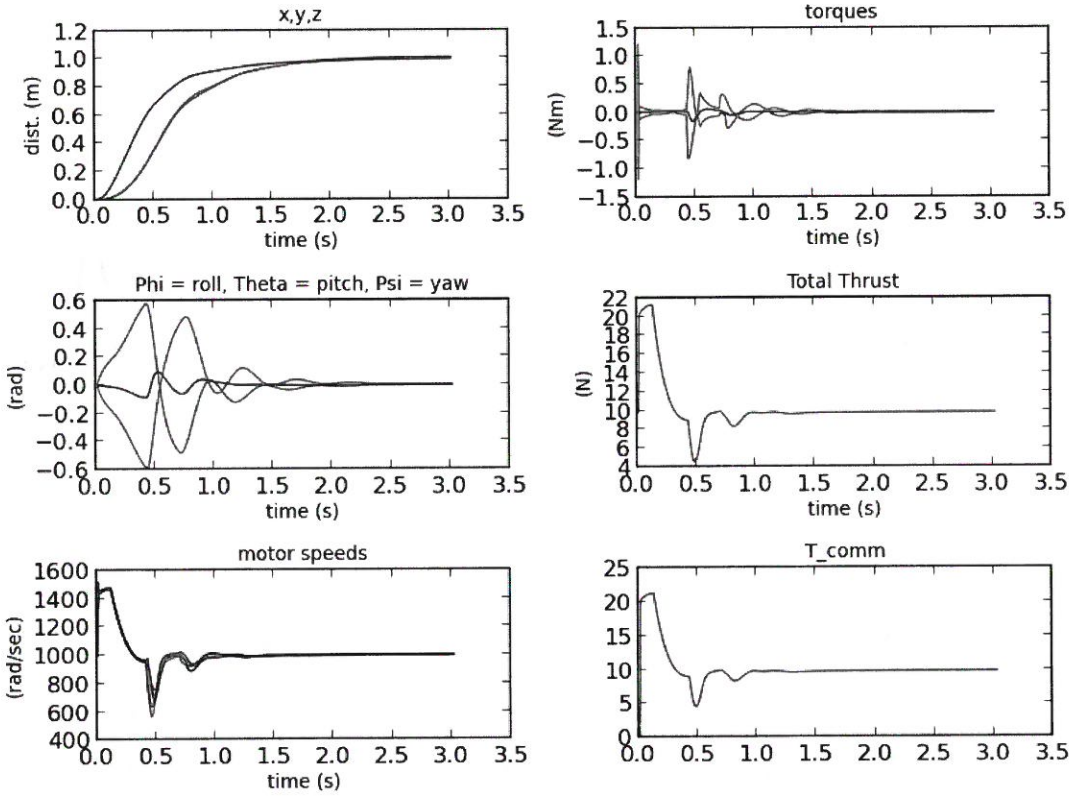


FIGURE 5.2: A typical run - time domain

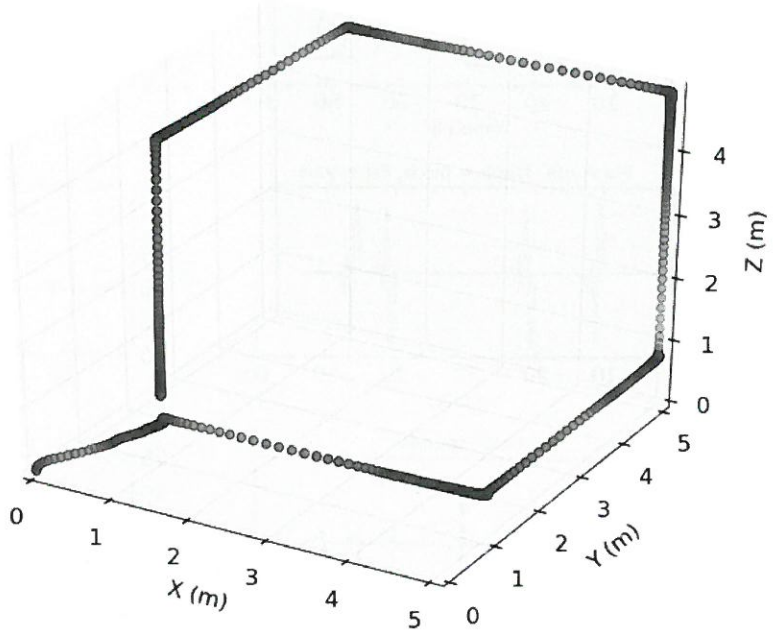


FIGURE 5.3: Tracing some of the edges of a 4m cube - the 3D path

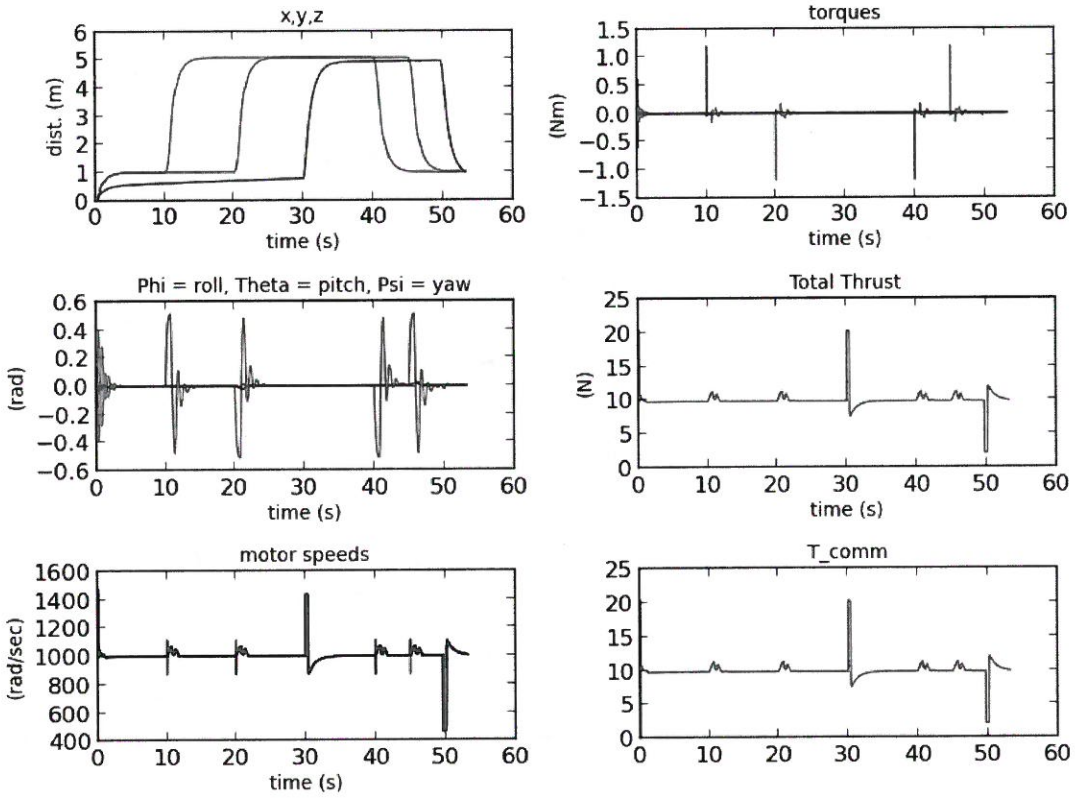


FIGURE 5.4: Tracing some of the edges of a 4m cube - time domain plots

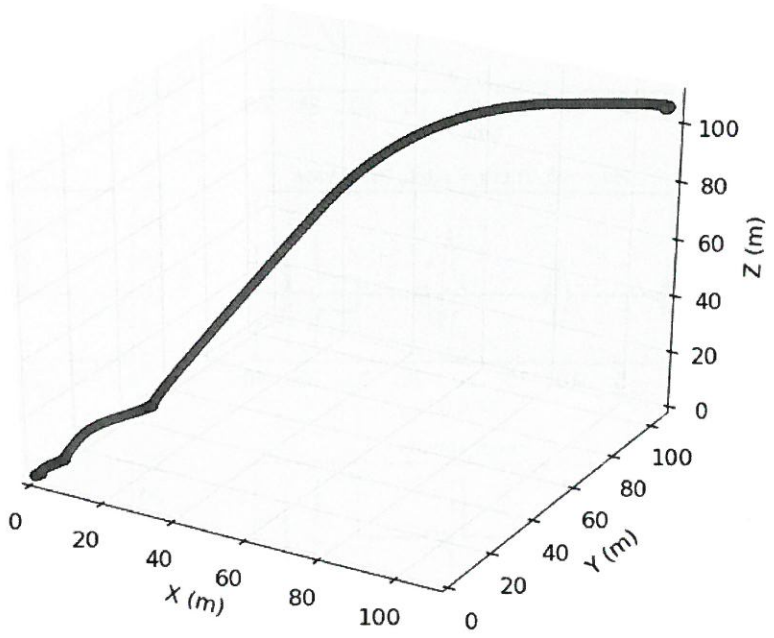


FIGURE 5.5: Testing the control with larger set points - the 3D path

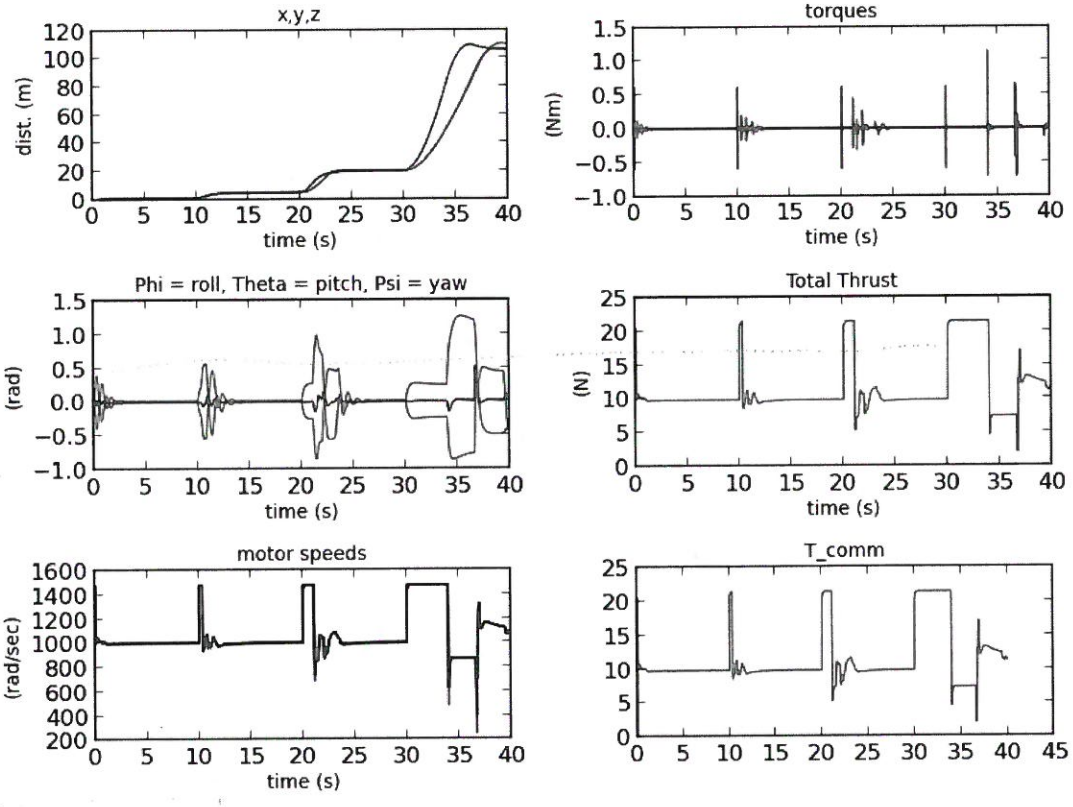


FIGURE 5.6: Testing the control with larger set points - time domain plots

This chapter has practically NO references.

Chapter 6

→ In this chapter . . . .

PID Gain Optimization

7.1 Intro

7.2 Problem Statement

In order to arrive at a solution to the quad-rotor energy optimization problem that is closer to running in real-time, a heuristic approach has been adopted. Recall that our general aim is to effectively control the vector position of the quad rotor and additionally use the least energy in doing so. The relative simplicity of a PID controller makes it a good choice instead of the full nonlinear classical optimal control formulation. Also, if the PID control expressions are tuned well and that tuning is not changed, the control algorithm performs well. Conversely, in an optimization scheme, the viability of the PID controllers is questionable. This is due to the fact that the PID control expressions are linear and they are used to control an naturally unstable non-linear system. (would be nice to show this with some kind of stability analysis...) → I will check to see if I can find something!

Specifically we wish to find: $argmin\left[\sum_{k,i}\omega_i[k] \mid K_p, K_i, K_d\right]$, where $\omega_i[k]$ is the ith rotor speed at the kth time step. Also, K_p , K_i and K_d are the vectors of proportional, integral, and derivative gains defined as:

$$K_p = \begin{bmatrix} k_{px} \\ k_{py} \\ k_{pz} \end{bmatrix}, K_i = \begin{bmatrix} k_{ix} \\ k_{iy} \\ k_{iz} \end{bmatrix}, K_d = \begin{bmatrix} k_{dx} \\ k_{dy} \\ k_{dz} \end{bmatrix}$$

In our simulations, the time integral of all four motor speeds is proportional to the total energy used. Calculation of the actual energy used by the UAV in traversing a flight path would require a model for the motor. This is seen as unnecessary for our purposes since the time integral of the motor speeds and the total energy used will have the same effective minimum. Since the control input is calculated as part of the control algorithm anyway, it is used as a performance metric.

MAYBE TALK HERE ABOUT THE PERFORMANCE OF THE CONTROLLER IN GENERAL (OVER SHOOT, TIME OF FLIGHT TOTAL THRUST)
ALSO TALK ABOUT WHY THE TAKE OFF PART IS DIFFERENT THAN THE HOVER- TO - HOVER PART OF THE FLIGHT

6.1 ~~Barak mentioned hope at some point...~~

needs rewriting see sheet

Evidence for the lack of robustness of the PID control in this case is shown by analyzing the relationship between the measured total thrust and the PID gains. Ideally, a gradient descent method would be used to minimize the thrust as a function of the control gains. The basic flow of the algorithm that we would really like to implement is as follows.

Algorithm 7.1

Step 1

- choose a set of proportional and derivative gains for each vector direction (x,y, and,z),
- perform a simulation that controls the quad rotor from an initial vector position to a desired vector position
- calculate the sum of the four motor speeds over the duration of the simulation
- appropriately change the PID gains such that the sum of the motor speeds decreases

- repeat until the sum of the motor speeds is found to be a minimum.

In reality, is has been found that the relationship between the measured total thrust and PID gains is not well behaved. After many days of trying to debug a gradient descent algorithm, and observing inconsistent behavior, it was decided to run a simulation for every possible gain vector for the set point $(0, 0, 1)$. A deeper understanding of the objective function was desired. By choosing this set point,the motion of the quad rotor is intentionally limited to the z-direction which limits the number of possible gain vectors for this test. This makes the tuning of the x and y direction controllers irrelevant. To further limit the number of simulations required, the Ziegler-Nichols PID tuning method was used. This makes each of the PID gains a function of a single gain variable, k_u , which is allowed to range from 1 to 100. The results of these simulations are shown in Figure 6.1.

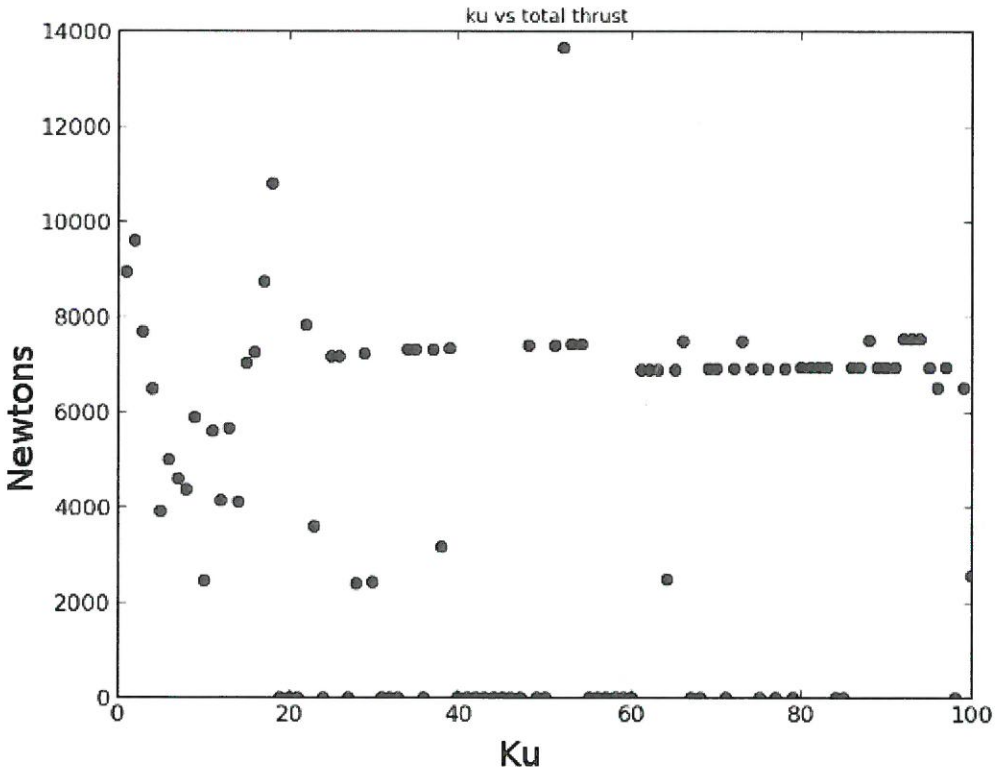


FIGURE 6.1: The relationship between Ku and the measured total thrust.

The relationship between Ku and the total measured thrust depicted in Figure 6.1 is rather disappointing to say the least. The values of zero total thrust are the result of simulations that failed to converge to the set point. Without an objective function that is at least marginally well behaved we have no hope to employ a gradient descent minimization technique. The relationship shown in Figure 6.1 is indeed a product of a deterministic system but displays little to no continuity. There are a few outliers in the data which are substantially lower in measured total thrust but the reason for their presence in the data as outliers is glib.

Another detail which cannot be ignored is that the total thrust alone is not sufficient as a performance criteria. Additionally, for appropriate control of the UAV, the overshoot and oscillations which show up in improperly tuned PID controllers must be accounted for. Specifically, as the proportional gain is increased to drive the system to the desired location more quickly, the total thrust for the simulation will decrease but eventually the overshoot and oscillations grow to unacceptable levels. In the opposite fashion, if the differential gain is increased, the overshoot and oscillations will be suppressed but the time required to reach the set point will increase along with the total thrust. The competitive nature of these three (necessary) performance criteria further complicate the relationship between the PID gain vectors and the objective function making a gradient descent minimization technique even less viable.

6.2 So where do we go from here?

Given that a standard mathematical optimization technique is out of the question, what options do we have left? With another year of research a nonlinear control law could be implemented and would perhaps make the optimization possible. Instead I opted for a naive, brute force approach. Sadly this is actually *MORE* time efficient when compared to another year of research! For the sake of learning,

a brute force algorithm is a something of a cop out but for the sanctity of my mortal soul, another year of research is simply not an option.

The decision was thus made to simply try many (many) possible gain vectors and have an appropriate objective function to characterize each of the simulations. Given that I have eight CPU's in my laptop and the brute force algorithm is easy to write (and massively parallel-izable), I was able to churn out around 6000 simulations over a period of several hours. The computation was delegated two six independent instantiations of a python script, each one taking on a subset of the possible gain vectors.

A separate script was used to parse through all the results of the simulations which were stored in many time-stamped files. Of the roughly 6000 simulations, about 1000 of them ran to completion without a numerical explosion. For these, the set point was reached and the stopping criteria for the simulation were satisfied. Of the 1000 or so good runs, about 80 simulations satisfied the maximum overshoot and oscillation criteria. Only the gain vectors which produced less than ten percent overshoot were accepted. Likewise, only simulations which crossed the desired set point in each direction fewer than four times were deemed acceptable. The remaining 80 simulations were sorted lowest to highest by total measured thrust. The optimal run that was found is detailed below.

The Optimal Run	
kpx	15
kpy	15
kpz	40
kix	0.8
kiy	0.8
kiz	15
kdx	10
kdy	10
kdz	50
ending iteration	987
descrete timestep	0.01 (s)
flight time	9.87 (s)
cpu runtime	11.41 (s)
return value	1 (great success)
initial position	[0,0,1] (m)
set point	[1, 1, 2] (m)
total thrust	4969.8 (Newton seconds)
x crossings	3
x overshoot	0.0249 (m)
y crossings	1
y overshoot	0.0185 (m)
z crossings	1
z overshoot	0.0992 (m)

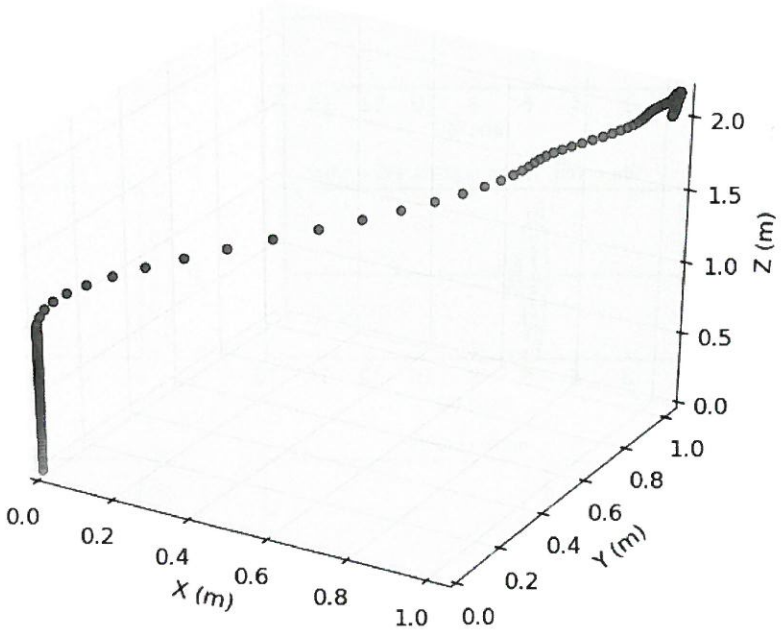


FIGURE 6.2: The Optimal Run - 3D path

explain!!

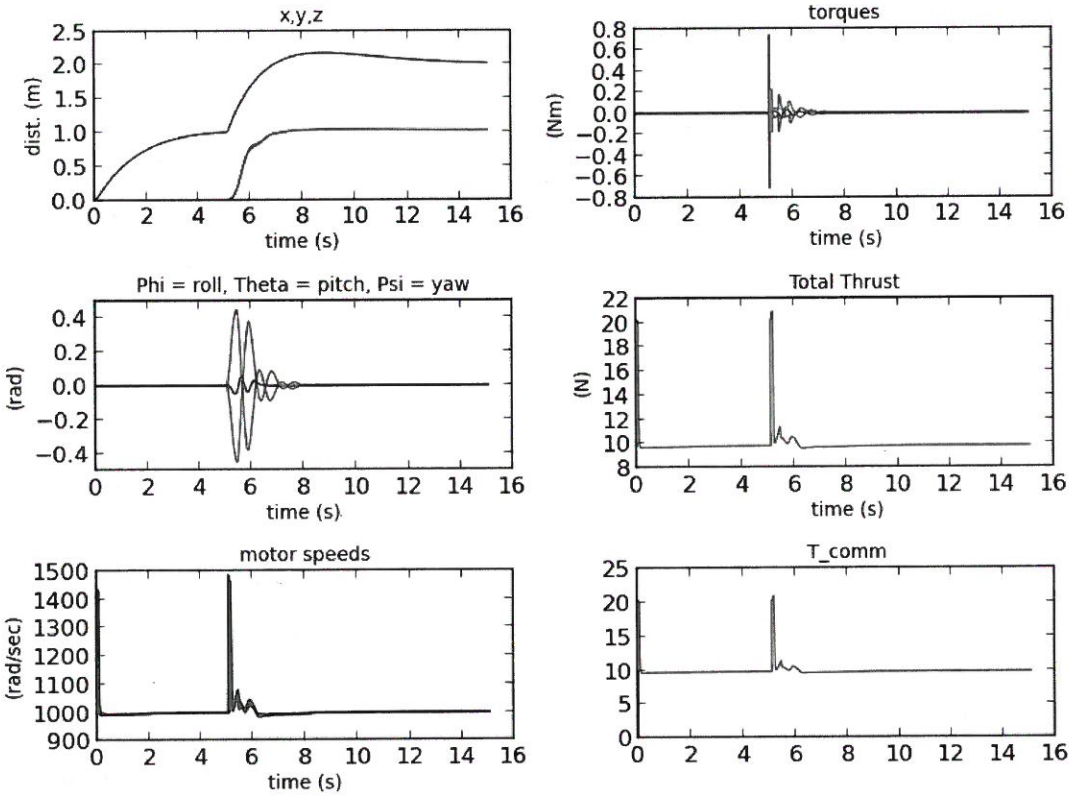


FIGURE 6.3: The Optimal Run - time domain

Conclusion