

Modelling, Simulation, and Implementation of Linear Control for Asymmetric Multirotor Unmanned Aerial Vehicles

Akinola Alexander Dada

Reg No.160140802

*May*17, 2020

Contents

1	Introduction	4
1.1	Background	4
1.2	Aims and Objectives	5
1.2.1	Basic Objectives	5
1.2.2	Advanced Objectives	6
1.3	Project Management	6
2	Review Of Literature	9
2.1	Introduction	9
2.2	Mathematical Modelling and System Identification	9
2.2.1	UAV	9
2.2.2	Dynamics	10
2.2.3	Actuator Dynamics	12
2.3	Control Systems Design, Simulation and Implementation	12
2.3.1	LQG	12
2.3.2	LQ-MPC	12
2.4	summary	13
3	Mathematical Modelling	14
3.1	Introduction	14
3.2	UAV	15
3.3	Kinematics	16
3.4	Kinetics	19
3.4.1	Newton-Euler	20
3.4.2	Asymmetry	21

3.5	Actuator Dynamics	22
3.6	Non-Linear Model	26
3.7	Linearisation	28
3.7.1	Jacobian Method	29
3.8	Summary	32
4	Linear Quadratic Gaussian and Model Predictive Control	34
4.1	Introduction	34
4.2	Discretization	35
4.3	Model Analysis	35
4.3.1	Stability	36
4.3.2	Reachability	37
4.3.3	Observability	37
4.4	LQG	39
4.4.1	Full-State Feedback	39
4.4.2	LQR	39
4.4.3	Reference Tracking and Integral Action	40
4.4.4	State Estimation	41
4.4.5	Estimator Modification	43
4.4.6	Results	45
4.5	LQ-MPC	48
4.5.1	Limitations of LQG control	48
4.5.2	Derivation of LQ-MPC	48
4.6	Summary	48
5	Embedded Control Systems Implementation	49
5.1	Introduction	49
5.2	Hardware	50
5.2.1	MARG and Barometer	51
5.3	Software and UML	51
5.4	Summary	52
6	Conclusions	54

List of Figures

1.1	Initial Timetable	6
1.2	Final Timetable	7
3.1	Y6 Hexarotor UAV	15
3.2	Coordinate Frames	16
3.3	UAV with Body Frame	22
3.4	A2212/13T 1000KV BLDC Motor	23
3.5	Motor Test Stand	23
3.6	Motor Input Step Data	24
3.7	First Order Model Step Response	25
3.8	Thrust against RPM	26
3.9	Torque against RPM	26
4.1	Open-Loop Eigenvalues	36
4.2	LQR Simulink Implimentation	41
4.3	KF Error	44
4.4	Limited EKF Error	44
4.5	Closed-Loop Eigenvalues	46
4.6	System Input PWM Signal	46
4.7	System Estimated Outputs	47
4.8	System Measured Outputs	47
5.1	BeagleBone Blue SBC	50
5.2	Embedded Component Structure	52

List of Tables

3.1	Table of Model Parameters	33
4.1	Table of Step Response values	45
5.1	Table of Control and Implementation Parameters	53

Abstract

Unmanned Aerial Vehicles (UAVs) are aerial systems not directly controlled by onboard personnel. There are many types of UAV platforms, chiefly defined by the characteristics of their mechanical construction. One such class of UAVs are multirotors which possess multiple rotor wing actuators. These types of UAVs have a wide range of applications due to their ability to perform vertical/Short take-off and landing (VSTOL), stationary and low speed flight, coupled with their relatively simple mechanical designs when compared to single rotor constructions, such as more traditional helicopters. The major challenge in dealing with these types of UAVs is their inherent instability in flight, save for the intervention of complex control systems. Therefore, in order to understand control systems capable of producing guaranteed requirement meeting performance, this project first seeks to investigate the physical characteristics and responses of UAVs through mathematical modelling, then proceeds to, design, develop, implement and compare the performance of 2 advanced linear control schemes on an embedded microprocessor, where this level of insight is requisite. These control schemes are, Linear Quadratic Gaussian (LQG) and Linear Quadratic Model Predictive Control (LQ-MPC).

Executive Summary

The first chapter provides a project introduction containing background information pertinent to the investigation, briefly discussing the nature and potential uses of UAVs. This section includes the systems engineering approaches taken, such as breaking down the project aims and objectives and setting them up as a set of task requirements with achievable milestones. This section provides information on project planning and management which includes the use of Gantt charts amongst others. This section will also detail all considerations and changes made to the project due to the SARSCOVID-2 pandemic.

The second chapter reviews and provides an analysis of the relevant supporting literature collected, which is referenced throughout the writing. The supporting research materials take the form of scientific journals articles, independent research publications, exerts from university lectures, Master's/ PhD thesis publications and published books.

The third chapter briefly discusses the mechanical design and construction of the specific asymmetric Y6 'hexarotor' UAV being used as the case study. This Section goes through and details the processes and results of analysing the UAV and deriving a mathematical representation of the physical characteristic through a combination of first principles dynamic modelling and hardware-in-the-loop (HIL) testing. This section explores and expatiates on different modelling techniques while discussing the various assumptions made during the modelling process and the reasoning behind the adopting of the techniques chosen. This section then converts the derived model into various forms useful for control design, analysis and simulation while presenting rigorous analyses of the system characteristics along with a dynamic simulation using industry standard applications.

The forth chapter details the process behind the design and development of an LQG feedback controller, then discusses and analyses the simulation results. This section then discusses the limitations of LQG controllers. This section details the design and development of LQ-MPC feedback controllers. Extending from the previous section, this section presents and justifies the reasoning behind transitioning to LQ-MPC as well as analysing the simulation results of the LQ-MPC feedback controller. This section then compares the performance of the LQG and LQ-MPC feedback controllers using appropriate metrics.

The fifth chapter discusses the characteristics of the embedded Linux single board computers (SBCs), using the BeagleBone Blue platform. This section discusses and details the processes and considerations behind implementing UAV control systems on the platform. The section then presents a Unified Modelling Language (UML) Class diagram detailing an implementation proposal for an embedded LQG Controller in the C Language.

The sixth and final chapter concludes, summarises results and finding then presents possibilities for future work and development.

Chapter 1

Introduction

1. Background.
2. Aims and Objectives.
3. Project Management.

1.1 Background

UAVs are aerial systems which are not directly controlled by a human onboard the vehicle. There are many types of UAV platforms which are chiefly defined by the characteristics of their mechanical constructions [3],[4]. These defining criteria break down to include:

- The position, number and type of their actuators.
- Their mode/modes of flight.
- Their use cases specific features.

One such type of UAVs are multirotors, defined as such due to their multiple rotor wing actuators. UAVs of this description were developed in response to the mechanical complexity of single or dual rotor winged aircraft such as the various helicopter and auto-gyros variants. The greater simplicity of multirotor craft could potentially lead to reduced development, construction and maintenance costs as well as greater reliability in use. Both classes of UAVs possess the ability to perform operations requiring VTOL, stationary and low speed flight [12].

These features enable them to be utilised in many wide ranging applications, where great dexterity in motion is required, and as such would not be possible using fixed winged platforms,

such as operations required during; directed precision administration of substances to plants in precision agriculture, frequent and or detailed inspection of standing structures, precision wide area surveillance, amongst many others [6].

However, the major challenge in dealing with multirotor UAVs is their inherent instability in real world flight situations, where disturbances are present, making it impossible to operate these craft with direct open loop control of their actuators, as is possible and frequently done with fixed winged craft. Therefore, the need for the development of control systems is apparent.

In order for the developed control systems to behave in a guaranteed predictable manner and be able to meet defined performance specifications, it becomes necessary to understand the behaviour of the system to be controlled and thus derive a mathematical behavioural representation to be used in guiding and informing the design process, which is done through mathematical systems modelling. This level of insight is also required in the development of advanced control schemes.

1.2 Aims and Objectives

The framework of this project was set around the design and development of control software for VSTOL model aircraft. From this broad theme, the aims of modelling an asymmetric multirotor platform, developing multiple control schemes and then implementing these schemes on a multirotor platform via an embedded SBC were derived.

These aims break down into multiple objectives and milestones which must each be achieved to fulfil the full scope of the aims outlined. These objectives can be broadly classified into 2 categories:

1.2.1 Basic Objectives

1. Develop a mathematical model representing the dynamics of the multirotor aircraft.
2. Develop a dynamic simulation of the crafts behaviour.
3. Develop LQG feedback control.
4. Investigate feedback control laws in simulation with the mathematical model to achieve behavioural targets.

5. Develop flight control software to interface with sensors and implement control laws.
6. Implement the flight control software on an embedded microprocessor unit(**).
7. Discuss the results of performance comparisons between the simulation and hardware implementations(**).

1.2.2 Advanced Objectives

1. Investigate the application of LQ-MPC schemes in simulation.
2. Incorporate and implement LQ-MPC schemes unto the flight control software(**).
3. Discuss the differences between the LQG and LQ-MPC implementations.

1.3 Project Management

		Name	Duration	Start	Finish	Predecessors
1		Development of VSTOL MultiRotor	189 days	21/10/19 08:00	27/04/20 08:00	
2		S1 Week 4 Aims and Objectives	0 days	21/10/19 08:00	21/10/19 08:00	
3		S1 Week 6 Progress Review	0 days	04/11/19 09:00	04/11/19 09:00	
4		S1 Week 10 Second Reviewer	0 days	02/12/19 09:00	02/12/19 09:00	
5		S1 Week 11 Interim Report	0 days	09/12/19 09:00	09/12/19 09:00	
6		S1 Week 12 Second Reader Meeting	0 days	16/12/19 09:00	16/12/19 09:00	
7		S2 Week 1 Interme Report Feedback	0 days	03/02/20 09:00	03/02/20 09:00	
8		S2 Week 6 Progress Review	0 days	02/03/20 09:00	02/03/20 09:00	
9		S2 Week 11 Final Report	0 days	20/04/20 08:00	20/04/20 08:00	
10		S2 Week 12 Presentation	0 days	27/04/20 08:00	27/04/20 08:00	
11		Modelling	68 days	21/10/19 08:00	28/12/19 08:00	
12		Obtain Parts	14 days	21/10/19 08:00	04/11/19 08:00	
13		Complete Multirotor Platform	14 days	04/11/19 08:00	18/11/19 08:00	12
14		Investigate Literature	14 days	21/10/19 08:00	04/11/19 08:00	
15		Develop Dynamical Model	40 days	04/11/19 08:00	14/12/19 08:00	14
16		Develop Model Simulation	14 days	14/12/19 08:00	28/12/19 08:00	15
17		Control Systems Design	85 days	04/11/19 08:00	28/01/20 08:00	
18		Investigate Literature	30 days	04/11/19 08:00	04/12/19 08:00	14
19		PID Control Design	14 days	28/12/19 08:00	11/01/20 08:00	16;18
20		LQG Control Design	21 days	28/12/19 08:00	18/01/20 08:00	16;18
21		Obtain results in Simulation	10 days	18/01/20 08:00	28/01/20 08:00	20
22		Flight Software Development	150 days	21/10/19 08:00	19/03/20 08:00	
23		Obtain key Libraries	7 days	21/10/19 08:00	28/10/19 08:00	
24		Discretize Control Laws	7 days	28/01/20 08:00	04/02/20 08:00	20;19;21
25		Modify/Develop Existing Software	30 days	04/02/20 08:00	05/03/20 08:00	23;24
26		Implement on Embedded Hardware	14 days	05/03/20 08:00	19/03/20 08:00	25
27		Implementation and Testing	21 days	19/03/20 08:00	09/04/20 08:00	13;15
28		Implement/Test On platform	14 days	19/03/20 08:00	02/04/20 08:00	26
29		Obtain Results and Analyse Performance	7 days	02/04/20 08:00	09/04/20 08:00	28
30		Advanced Objectives	176 days	21/10/19 08:00	14/04/20 08:00	
31		Investigate Predictive control	120 days	21/10/19 08:00	18/02/20 08:00	
32		Implement in Simulation	14 days	18/02/20 08:00	03/03/20 08:00	16;31;21
33		Modify Flight Software	21 days	03/03/20 08:00	24/03/20 08:00	32
34		Implement/test on Hardware	5 days	09/04/20 08:00	14/04/20 08:00	33;29

Figure 1.1: Initial Timetable

		Name	Duration	Start	Finish	Predecessors
1		Development of V/STOL MultiRotor	213 days	21/10/19 08:00	21/05/20 08:00	
2		S1 Week 4 Aims and Objectives	0 days	21/10/19 08:00	21/10/19 08:00	
3		S1 Week 6 Progress Review	0 days	04/11/19 09:00	04/11/19 09:00	
4		S1 Week 10 Second Reviewer	0 days	02/12/19 09:00	02/12/19 09:00	
5		S1 Week 11 Interim Report	0 days	09/12/19 09:00	09/12/19 09:00	
6		S1 Week 12 Second Reader Meeting	0 days	16/12/19 09:00	16/12/19 09:00	
7		S2 Week 1 Intereme Report Feedback	0 days	03/02/20 09:00	03/02/20 09:00	
8		S2 Week 6 Progress Review	0 days	16/03/20 09:00	16/03/20 09:00	
9		S2 Week 11 Final Report	0 days	18/05/20 08:00	18/05/20 08:00	
10		S2 Week 12 Presentation	0 days	21/05/20 08:00	21/05/20 08:00	
11		Modelling	68 days	21/10/19 08:00	28/12/19 08:00	
12		Obtain Parts	14 days	21/10/19 08:00	04/11/19 08:00	
13		Complete Multirotor Platform	14 days	04/11/19 08:00	18/11/19 08:00	12
14		Investigate Literature	14 days	21/10/19 08:00	04/11/19 08:00	
15		Develop Dynamical Model	40 days	04/11/19 08:00	14/12/19 08:00	14
16		Develop Model Simulation	14 days	14/12/19 08:00	28/12/19 08:00	15
17		Control Systems Design	137.375 days	04/11/19 08:00	20/03/20 17:00	
18		Investigate Literature	30 days	04/11/19 08:00	04/12/19 08:00	14
19		LQG Control Design	73.375 days	28/12/19 08:00	10/03/20 17:00	16,18
20		Obtain results in Simulation	10 days	10/03/20 17:00	20/03/20 17:00	19
21		Flight Software Development	7 days	21/10/19 08:00	28/10/19 08:00	
22		Obtain key Libraries	7 days	21/10/19 08:00	28/10/19 08:00	
23		Advanced Objectives	208 days	21/10/19 08:00	16/05/20 08:00	
24		Investigate Predictive control	180 days	21/10/19 08:00	18/04/20 08:00	
25		Implement in Simulation	28 days	18/04/20 08:00	16/05/20 08:00	16,24;20

Figure 1.2: Final Timetable

In order to properly execute on the project Aims and Objectives, project activities were coordinated and managed by further breaking down or grouping task then laying them out in tabulations, Figures 1.1 and 1.2, which were used to generate the project Gantt charts.

The project has by in large proceeded along on the initial schedule proposed with only one major non SARSCOVID-2 related change made to tasks and execution dates from the initial proposal.

Due to the effects and responses to the global SARSCOVID-2 pandemic, The University of Sheffield was forced into closure/Cancellation of all on-site and in-person teaching, meetings and instruction from March of 2020. This coupled with the government mandated nationwide quarantine and lock down in the United Kingdom made access to the necessary facilities for the testing and evaluation of any developed UAV software impossible. As such, activities and work towards meeting the objectives succeeded by a Double-Asterix in the previous aims and objectives section are not being undertaken. Project planning has been adjusted accordingly. This has also led to changes in term dates which resulted in adjustments being made to task timelines.

Regarding other aspects of the project, all material purchases were made on time, acquiring

key hardware such as the main flight computer, the flight radio control system, which would have served as the main communication unit sending control commands to the multirotor, electronics speed controllers, and load cells which will be used to develop the lab-bench platforms to identify the actuator dynamics. All Ad hoc minor changes to task completion periods are reflected in timeline changes from the initial proposal table, Figure 1.1, to the final proposal table, Figure 1.2.

The only major change to the project scope, not enforce by SARSCOVID-2, from its initial proposal was the dropping of a PID control design section and all such tasks, which occurred before, and is reflected in, the S2 week project proposal review. This change was made to re-focus the project objectives solely on model-based control design outcomes.

All changes to task completion dates for further objectives are displayed in the final proposal table, Figure 1.2, which was used to produce the final project Gantt chart available as an Appendix.

Chapter 2

Review Of Literature

2.1 Introduction

In preparing to execute the project tasks, various supporting and reference materials were acquired. The various literature take the form of articles published in scientific journals, research publications, exerts from university lectures, Masters and PhD thesis publications and published books, all of which together broadly cover and present 3 major themes that are relevant to the project's investigation:

1. Mathematical Modelling and Systems Identification.
2. Control Systems Design.
3. Simulation and Implementation.

2.2 Mathematical Modelling and System Identification

2.2.1 UAV

The UAV is actuated by 6 1000Kv brushless direct current (BLDC) motors [14], attached to 6 10x45inch propellers and is powered by a 4-cell lithium-polymer battery with an operational voltage range of 16.8 volts to 12.8 volts. The UAV has dimensions of, 45 cm in length from rear motor centres to front motor centres, 44 cm in length from left motor centres to right motor centres, and a mass of 1.3 kilograms. The exact details involved with the design, development and

construction of UAVs is beyond the scope of the project investigation but material, mechanical and electronic consideration when designing such craft are explored in [12].

2.2.2 Dynamics

In modelling systems, it is first necessary to state all assumptions made about its characteristics. These assumptions simplify the process of systems model abstraction to ensure minimum viability standards are met. Following that, multirotors can be defined as rigid-bodies free to move in three-dimensional (3-D) space, therefore, they move with 6 degrees of freedom (6-DOF). With this modelling assumption, the motion of the vehicles is subject to the laws of rigid body Kinematics and Kinetics. All the acquired material takes this stance.

Vehicle motion is defined in terms of 2 coordinate frames moving relative to one another where the physical quantities that change with time, states, change with respect to one frame or the other. They are the Earth-Frame, fixed to the earth and the Body-Frame, attached to the vehicle body. Each frame consists of 3 orthogonal axes, whose nomenclature varies from source to source, about and along which rotational and translational motion can occur respectively [1],[2],[4],[15].

The sources obtained explore 2 ways for representing this rotational motion of the reference frames:

- Sources [1],[2],[3],[4],[5],[6],[7],[9] present their models using Euler angles which describe arbitrary orientation in the 3-dimensional Euclidean space using three parameters which represent a sequence of three elemental rotations about each axes of the coordinate system. Any orientation in 3D space can be achieved by composing these 3 elemental rotations.
- Sources [1],[8],[16] present representations through Quaternions which solve issues present with computing Euler angles such as computational expense and numerical singularities which occur when certain orientations are reached, as it does not require the calculation of sin and cos when certain angles go to 0 or 90 degrees and their multiples. Quaternions represent any arbitrary orientation by constructing rotations as that of a single angle about an imaginary axis.

The rigid body assumptions also allow for the utilisation of several Mechanical Modelling techniques and conventions which are used to derive the non-linear dynamics of the body frame

mechanical systems considering both kinematics and Kinetics:

- The Newton-Euler convention which derives representations of systems dynamics through the application of first principles using newtons 2nd law of motion showing the effects of forces applied to the rigid body by its actuators and environment. Utilised in [2],[3],[5],[6],[7],[8],[9].
- The Euler-Lagrange convention which derives representations of systems dynamics through the study of energy differentials between kinetic and gravitational potential generated by motion. Utilised in [4].
- The Newton-Hamiltonian.
- Pure Hardware-in-the-Loop system identification, which decouples each axes of motion and determines approximate model frameworks for each then populates said framework with experimentally obtained elements. Such was the process taken by [28]

This investigation utilises the Euler-angle representation and the Newton-Euler convention due to its intuitive familiarity and relative simplicity.

Due to the UAV's layout, it is asymmetric, this creates a system where its mass is not geometrically evenly distributed about the centre of mass leading to where inertia is not represented by a diagonal inertial tensor but instead one with non-trivial off diagonal elements and thus creating transposed reference frames. However, for the purpose of this investigation, it is assumed that the system is symmetrical as the off-diagonal terms in the inertia matrix are far lower in magnitude than the diagonal terms, thus can be reasonably neglected. Otherwise, the entire coordinates system would be adjusted and transformed to produce a true diagonal matrix as presented in [10].

In order to design the controllers, the non-linear model obtained via the Newton-Euler convention needs to be linearised. This is accomplished through the application of Jacobi's linearisation method as presented in [11]. This process produces a full linear state space representation of systems dynamics around stable operating points. This model is then discretized for implementation in digital embedded systems.

2.2.3 Actuator Dynamics

As stated in the UAV subsection, the multi-rotor is actuated by BLDC motors [14], To fully develop a system model, these must be modelled. This can either be done via first principles calculations as presented in [2],[8],[17] or via experimental systems identification as presented in [9],[12],[14],[15],[16].

The systems identification approach involves the derivation of a lumped parameter linear input-output model, encompassing the electronic speed control unit, motor and propeller dynamics, between input Pulse width modulated (PWM) signal duty cycles or pulse width and the output angular velocity, torque and thrust forces. This has the advantage of reducing complex dynamics enabling the utilisation of a minimum viable model and as such, is being implemented in this investigation.

2.3 Control Systems Design, Simulation and Implementation

2.3.1 LQG

The Linear Quadratic Gaussian makes use of the full state of a system, obtainable from a limited set of measurements through the application of a Gaussian estimator, in this case a Kalman filter [21],[22],[23],[24] to obtain the full state. It is a form of optimal control where its objective is to find a set of control actions which minimize some cost or performance function to bring the system's state to a desired set of values. Sources [18],[19],[20],[25] present derivations of state feedback control leading to different implementation of LQG control. Source [19] in particular details implementation of LQG control in embedded systems.

2.3.2 LQ-MPC

Like LQG, MPC is also a kind of optimal control, however, this technique's cost or performance function is limited to looking at future states over some fixed finite horizon and control signals can be determined so that they implicitly taking into account and determine control action so as to meet physical constraints on the system. MPC is also known as receding horizon control (RHC) as it only acts on the first step in the horizon of states before recalculating and performing this action over and over until the target is achieved.

MPC does not refer to any one algorithm or method but instead a series of methodological philosophy which implements certain key concepts and ideas in various ways as stated by[28]. Thus, this investigation makes a limiting delineation by specifying LQ-MPC which limits the scope to MPC consisting of linear systems and quadratic cost functions.

Sources [25],[26],[27] flush out and give step by step guides and best practices and considerations to follow when designing Linear predictive control techniques. Source [25] in particular discusses and reasons through the limitations of LQG and the derivation of LQ-MPC from the perspective of transitioning from an LQG formulation. one source [28] also discusses various implementations methods including utilising machine leaning techniques to estimate certain optimisation parameters, as well as proposing and implementing such a controller.

The developed Control systems will be implemented on a Beagle Bone Blue Linux based single board computer (SBC) extending from the Robot Control Library (RCL) [31]. Sources [19],[29] presents consideration for the implementation of flight control systems and source [30] presents considerations for designing software in general looking at systems engineering approaches.

2.4 summary

This Chapter:

- Described the Material available from the reference source material gathered.
- Highlighted the major take aways from each source and put them in the context of the larger project objectives.
- Presented the reasoning behind the selective application of source material.

Chapter 3

Mathematical Modelling

3.1 Introduction

Mathematical models are used to describes the characteristics of physical systems. They are derived from assumptions made about a systems behaviour which are represented abstractly using mathematics. In modelling systems, it is first necessary to state all assumptions made about its characteristics. These assumptions simplify the process of model abstraction to ensure minimum viability standards are met. This allows for a mapping of a multirotor movements and behaviour with the respect to it inputs and external influences meaning mathematical models can be considered as a series of functions that map inputs onto outputs while determining all the important time dependant elements of the system. In order to successfully accomplish the development of an appropriate mathematical model for both simulation and control design, the following topics will be explored:

1. UAV
2. Kinematics
3. Kinetics
4. Actuator Dynamics
5. Linearisation



Figure 3.1: Y6 Hexarotor UAV

3.2 UAV

The UAV under consideration throughout this project is an asymmetrical Y6 hexarotor. This kind of configuration has actuators in 3 counter-rotating co-axial pairs and 3 motor groups. A coaxial Y6 hexarotor is mechanically simpler than a classical helicopter since it has propellers with constant pitch and without a swash plate [6]. The advantages of Y6- hexarotors with respect to other multirotor configurations and classical helicopters is that the double drive configuration increases the thrust without increasing the frame size, while naturally eliminating the loss of efficiency due to torque compensation, another advantage of this configuration is better stability. However, the disadvantages are the higher energy consumption for six propulsion units and inefficiency due to the aerodynamic interaction present with co-axial actuator combinations[9]. The UAV is actuated by 6 1000Kv BLDC motors [14], attached to 6 10x45inch propellers and is powered by a 4-cell lithium-polymer battery. The exact details involved with the design, development and construction of UAVs is beyond the scope of the project investigation but material, mechanical and electronic consideration when designing such craft are explored in [12].

Following that, initially, multirotors can be defined as; rigid-bodies free to move in 3-D space with 6-DOF, with all motion restricted to being either rotational or translational.

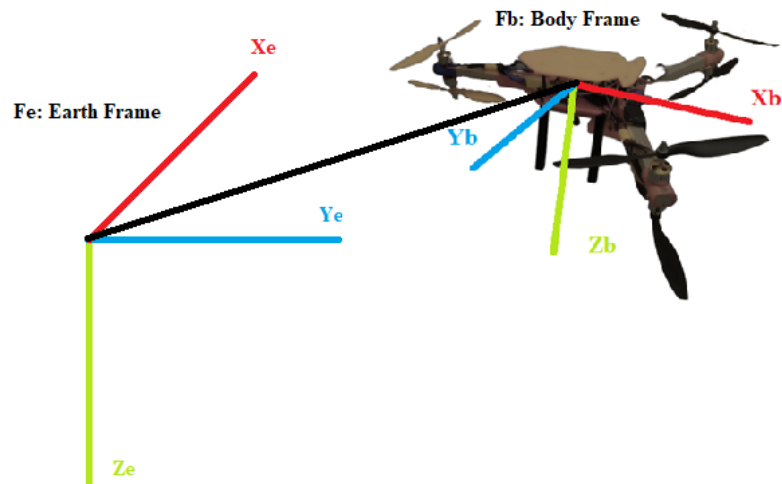


Figure 3.2: Coordinate Frames

More detailed mathematical models may describe certain multirotor characteristics more accurately, but they also require more computational resources which can lead to, longer simulation times, the inability to successfully complete simulations [5], and more complex than necessary control design processes and solutions.

This project will derive and present 2 models, a Non-linear model utilised in simulation, and a Linear model utilised in analysis and control design.

3.3 Kinematics

Multirotor mathematical models have to describe attitude and position according to the geometry of the UAV [6]. one of the most important parts of multirotor modelling is understanding the geometric and kinematic relationships between the reference frames [8]. Kinematics is the study of motion in terms of positions and velocities without regard to the forces causing the motion. Multirotor motion can be described by a number of variables called states, that are related and change with suitably chosen axes systems or reference frames [1]. These reference frames are:

- The Earth Frame: Fe
- The Body Frame: Fb

Each frame consists of 3 orthogonal axes, X_e, Y_e, Z_e and X_b, Y_b, Z_b respectively, about and along which rotational and translational motion respectively can occur. Where F_e is fixed to the earth near the multirotor such that a flat earth assumption can be made and sustained [2] and F_b is attached to the vehicle, centred at the UAV's centre of mass, which is free to move with respect to F_e . The usual convention for the axes representation is to have a fixed right-handed axes system with the positive Z axes pointing downwards, towards the earth when levelled or hovering, the positive X axis pointing forward and the positive y axis pointing towards right or starboard side. This convention is referred to as the NORTH-EAST-DOWN Right-handed coordinate system [1],[2],[8]. Given the definition of the coordinate reference frames, system variables can now be defined in terms of these reference frames.

In the F_e , we define vectors:

$$\vec{E}l = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad (3.1)$$

which represent the linear translational positions of F_b 's centre with respect to (w.r.t) F_e .

$$\vec{E}r = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} \quad (3.2)$$

which represent the angular rotation of F_b about all 3 axes respectively, w.r.t F_e . Which are also known as (roll, pitch, yaw) respectively.

In the F_b , we define vectors:

$$\vec{B}l = \begin{bmatrix} u \\ v \\ w \end{bmatrix} \quad (3.3)$$

which represents the linear velocity in each F_b axis respectively.

$$\vec{B}r = \begin{bmatrix} p \\ q \\ r \end{bmatrix} \quad (3.4)$$

which represents the angular velocity in each Fb axis respectively.

From these 12 elemental variables and 4 elemental vectors all, subsequent states and motion can be derived and represented [1],[2],[4].

Taking the ϕ, θ, ψ representation, the rotational motions of Fb w.r.t Fe can be derived by looking at rotations about each axis individually. Decomposing any arbitrary motion and orientation in 3-D Euclidean space in this manner presents a sequence of three elemental rotations of each axes of Fb w.r.t its counterpart in Fe. These elemental rotation angles are known as *Euler angles* and rotations are represented as direction cosine matrices [1],[2]. These are:

$$R_{\psi} = \begin{bmatrix} C_{\psi} & S_{\psi} & 0 \\ -S_{\psi} & C_{\psi} & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3.5)$$

$$R_{\theta} = \begin{bmatrix} C_{\theta} & 0 & -S_{\theta} \\ 0 & 1 & 0 \\ S_{\theta} & 0 & C_{\theta} \end{bmatrix} \quad (3.6)$$

$$R_{\phi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & C_{\phi} & S_{\phi} \\ 0 & -S_{\phi} & C_{\phi} \end{bmatrix} \quad (3.7)$$

These elemental rotations can be combined using the ZYX convention which multiplies them together in the specific sequence:

$$R^{-1} = R_{\psi} \cdot R_{\theta} \cdot R_{\phi}$$

to produce a matrix, the inverse of which, R, is able to map the translational motion of Fb unto Fe give any set of arbitrary ϕ, θ, ψ values. [1],[2],[3],[4],[5],[6],[7],[8],[9].

$$R = \begin{bmatrix} C_{\theta}C_{\psi} & -C_{\theta}S_{\psi} + S_{\phi}S_{\theta}C_{\psi} & -S_{\phi}S_{\psi} + C_{\phi}S_{\theta}C_{\psi} \\ C_{\theta}S_{\psi} & C_{\phi}C_{\psi} + S_{\phi}S_{\theta}S_{\psi} & -S_{\phi}C_{\psi} + C_{\phi}S_{\theta}S_{\psi} \\ -S_{\theta} & S_{\phi}C_{\theta} & C_{\phi}C_{\theta} \end{bmatrix} \quad (3.8)$$

Therefore:

$$\vec{E}l = R \cdot \vec{B}l$$

Which maps the linear velocities and accelerations in Fb to linear velocities and accelerations with respect to Fe.

Similarly, to map the angular velocities and accelerations in Fb w.r.t Fe requires another transformation matrix, the inverse of which, T, is also derived from the manipulation of Euler angles [1],[4].

$$T = \begin{bmatrix} 1 & S_\phi T_\theta & C_\phi T_\theta \\ 0 & C_\phi & -S_\phi \\ 0 & \frac{S_\phi}{C_\theta} & \frac{C_\phi}{C_\theta} \end{bmatrix} \quad (3.9)$$

Therefore:

$$\vec{E}r = T \cdot \vec{B}r$$

Where S, C, T, represent Sin, Cosine and Tan respectively.

3.4 Kinetics

Kinetics is the study of motion considering the forces and torques which cause the motion. As stated in the previous UAV subsection, the project looks at a Y6 hexarotor UAV which possesses 6 BLDC motors with propellers in a coaxial motor layout. Each motor propeller unit produces a thrust force, the collective effect of which can be summed and lumped together as a single force F, which means:

$$\sum_{n=1}^{n=6} F_n$$

where n denotes the motor index.

Each motor propeller unit or propulsive unit, also produces a reaction torque. When a motor

turns, in overcoming air resistance, a reactive force acts on the propeller in the direction opposite to the motor's rotation which produce a torque acting on the UAV body. These torques, τ , can then also be summed together, meaning:

$$\sum_{n=1}^{n=6} \tau_n$$

The co-axial configuration also ensures single point torque balancing [6]. meaning, so long as all rotors pair each produce the same torque, the net reactive torque produced is zero. The translational and angular motion of the Y6 hexarotor is controlled by thrust forces and torques produced by each motor. The main thrust is the sum of all rotors thrust, and rotational movement is generated by differences in individual motor thrusts and torques [9].

3.4.1 Newton-Euler

The Multirotor 6-DOF rigid body kinetics and takes into account the mass, and the inertia of the body. These are described by differential equations, which are derived through the utilization of the Newton-Euler modelling convention which derives representations of systems dynamics through the application of first principles via newtons 2nd law of motion aggregating all forces and torques applied to the rigid body by its actuators and environment while observing the resultant accelerations produced.

Resolving the Forces acting linearly on the UAV:

$$M \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = M \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} - \begin{bmatrix} C_\theta C_\psi & -C_\theta S_\psi + S_\phi S_\theta C_\psi & -S_\phi S_\psi + C_\phi S_\theta C_\psi \\ C_\theta S_\psi & C_\phi C_\psi + S_\phi S_\theta S_\psi & -S_\phi C_\psi + C_\phi S_\theta S_\psi \\ -S_\theta & S_\phi C_\theta & C_\phi C_\theta \end{bmatrix} \begin{bmatrix} \ddot{u} \\ \ddot{v} \\ \ddot{w} \end{bmatrix} - \begin{bmatrix} D_{lxx} & 0 & 0 \\ 0 & D_{lyy} & 0 \\ 0 & 0 & D_{lzz} \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} \quad (3.10)$$

$$M \cdot \ddot{\vec{E}l} = R \cdot \ddot{\vec{B}l} - M \cdot \vec{g} - \text{diag}(D_l) \cdot \dot{\vec{E}l}$$

where M is mass, g is acceleration due to gravity acting in the vertical Ze axis plane, and D_l is a diagonal matrix of drag coefficients with aerodynamic drag acting directly proportional to velocity in the Fe frame.

Given that all thrust forces from the motor act along the same Zb axis plane, and there are no other force acting in the other axes therefore, the resultant equation can be rewritten as:

$$M \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = M \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} - \begin{bmatrix} C_\theta C_\psi & -C_\theta S_\psi + S_\phi S_\theta C_\psi & -S_\phi S_\psi + C_\phi S_\theta C_\psi \\ C_\theta S_\psi & C_\phi C_\psi + S_\phi S_\theta S_\psi & -S_\phi C_\psi + C_\phi S_\theta S_\psi \\ -S_\theta & S_\phi C_\theta & C_\phi C_\theta \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ \sum F_n \end{bmatrix} - \begin{bmatrix} D_{lxx} & 0 & 0 \\ 0 & D_{lyy} & 0 \\ 0 & 0 & D_{lzz} \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} \quad (3.11)$$

Resolving the torques and forces acting angularly on the UAV in Fb:

$$\begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} A_p \\ A_q \\ A_r \end{bmatrix} - \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} - \begin{bmatrix} D_{rxx} & 0 & 0 \\ 0 & D_{ryy} & 0 \\ 0 & 0 & D_{rzz} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \quad (3.12)$$

$$I \cdot \dot{\vec{B}}r = A - C - \text{diag}(D_r) \cdot \vec{B}r$$

Where A is the vector of torques generated by the sum of motor forces and torques in each axis, C is the Coriolis matrix formed of the cross product between the angular velocity vector and the diagonal inertia Tensor I, and D_r is the diagonal matrix of drag coefficients acting against motion in each axis..

The A vector breaks down to:

$$A_p = L_1((F_1 + F_2) - (F_3 + F_4)) \quad (3.13)$$

$$A_q = L_2((F_1 + F_2 + F_3 + F_4) - L_3(F_5 + F_6)) \quad (3.14)$$

$$A_r = \sum_{n=1}^{n=6} \tau_n \quad (3.15)$$

Where L is the perpendicular distance from each motor group to the centre of mass.

3.4.2 Asymmetry

Due to the asymmetry of the Y6 hexarotor, mass is not even distributed across the UAV frame. This creates a particular imbalance of inertia about the centre of mass, which is the centre of all rotation in Fb. This leads to a case where I has off diagonal elements[10]. i.e:

$$\begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix}$$

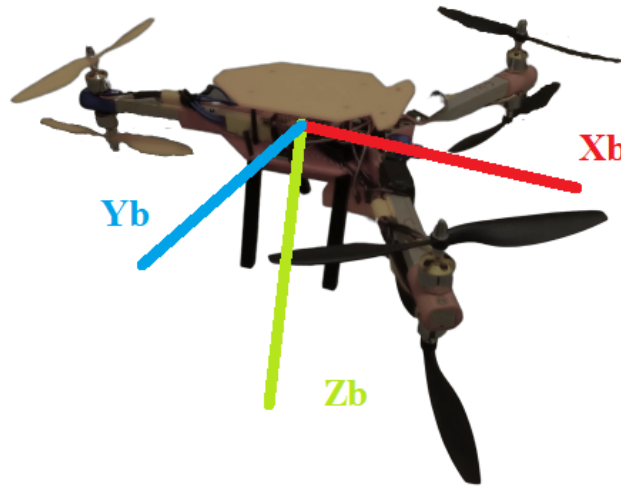


Figure 3.3: UAV with Body Frame

This creates a situation whereby F_b as shown in Figure 3.3, is inaccurate and instead, is transposed to some other orientation, meaning rotation about the centre mass will occur in an oblong manner.

To correct this, it is possible to adjust the inertia tensor by deriving some diagonalising operation such as performing an Eigen vector decomposition [10], this transformation will then need to be applied to all Inertia dependant terms in the angular motion calculation.

However, for the purpose of this investigation, it is possible to operate under the assumption that the system is symmetrical, as presented by a purely diagonal inertial tensor. This is due to the off-diagonal terms in the inertia matrix being far lower in magnitude than the diagonal terms themselves, meaning that the transposed reference frame does not differ too significantly from assumed symmetrical axes, thus any deviations can be reasonably neglected [10].

3.5 Actuator Dynamics

The thrust forces and torques acting on the multirotor UAV are primarily generated by propulsion units consisting of a BLDC motor, shown in Figure 3.4, an ECS and propeller. In order to produce a useful and fully developed systems model, the characteristics of the propulsive units must be known. This can either be done via first principles analysis looking at the mechanical, electrical and aerodynamic properties of each individual component, before calculating and



Figure 3.4: A2212/13T 1000KV BLDC Motor

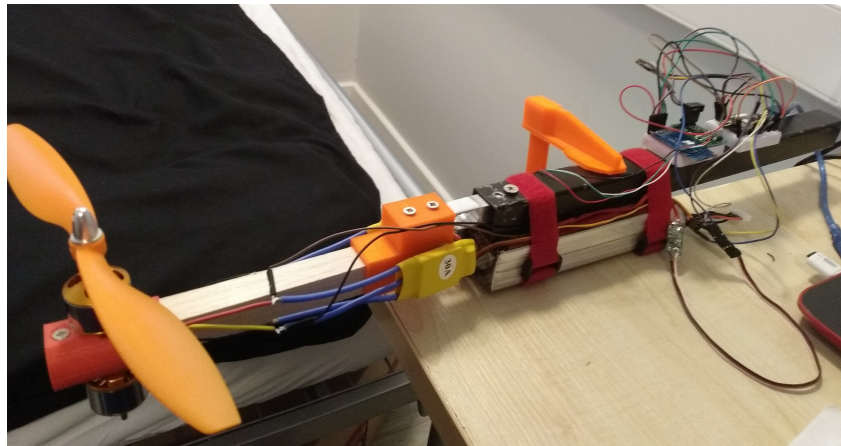


Figure 3.5: Motor Test Stand

then applying the results together in series to produce a high order, high fidelity model as is presented in [17].

However taking this approach in many cases is unnecessary. This is due to being able to derive a lumped parameter model, which views the sum total of all the individual components as a single system with singular dynamics. Therefore, using system's response analysis and computational methods such as data fitting, it is possible via experimental systems identification to generate a lower order, minimum realisation model of sufficient fidelity [9],[12],[15],[16]. The systems identification approach involves the derivation of lumped parameter linear input-output models, between input PWM signal width and the output angular velocity, torque and thrust forces respectively using a motor test stand, shown in Figure 3.5, which consists of a load

cell, for measuring forces and torques as well as an electronic revolutions per minute (RPM) sensor.

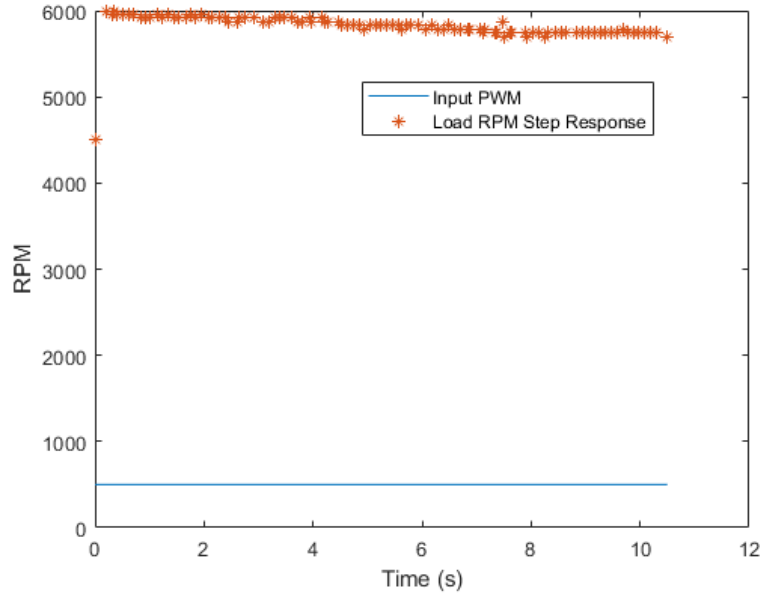


Figure 3.6: Motor Input Step Data

Using the Test stand, input-output PWM to RPM data is collected. Shown in Figure 3.6, is the response of the propulsive unit to a step input of $500\mu s$. This data was then passed through the MATLAB system identification application where in order to account for potential delays in the input response, a first order transfer function model of the form:

$$\frac{K_u}{\tau_m s + 1}$$

This transfer function can then be reformulated into a differential equation

$$\dot{\omega}_n = -\frac{1}{\tau_m} \omega_n + \frac{K_u}{\tau_m} \mu_n \quad (3.16)$$

This fit to the data where, K_u , is motor gain and τ_m is the response time constant, μ_n is the input PWM signal width and ω_n is the angular velocity. Figure 3.7 shows the response of the identified model to the same step of $500\mu s$.

It is important to note that due to filtration present in the ESCs, the response time constant is sensitive to the frequency of the input PWM signal, therefore, throughout the investigation a frequency of 400 Hz is maintained. Similarly, using the test stand, as shown in Figure 3.8

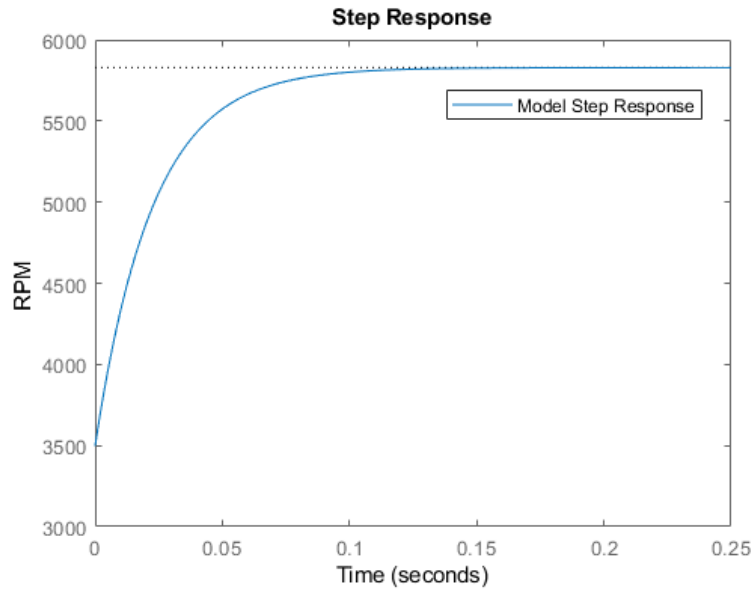


Figure 3.7: First Order Model Step Response

and Figure 3.9, data showing the relationship between the force and torque produced by the propulsion unit against generated angular velocity is shown. 2 quadratic models of the form:

$$F_n = K_a \omega_n^2 + K_b \omega_n \quad (3.17)$$

$$\tau_n = K_\tau \omega_n^2 \quad (3.18)$$

respectively are then fit to the data where ω is the angular velocity of the motor, K_a , K_b and K_τ are the force and torque coefficients respectively derived from the quadratic models. Due to the complex aerodynamic interactions from the overlapping air flow between the coaxial pairs, the actual thrust force produced is lower than the sum of the two propulsion units if taken individually [14]. This loss is represented by a gain K_w that is less than 1 and applied to all the propulsion units on top.

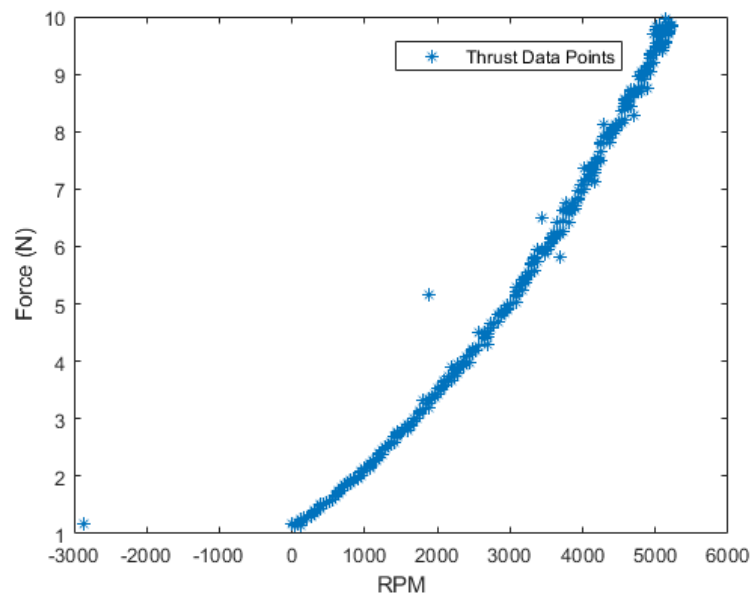


Figure 3.8: Thrust against RPM

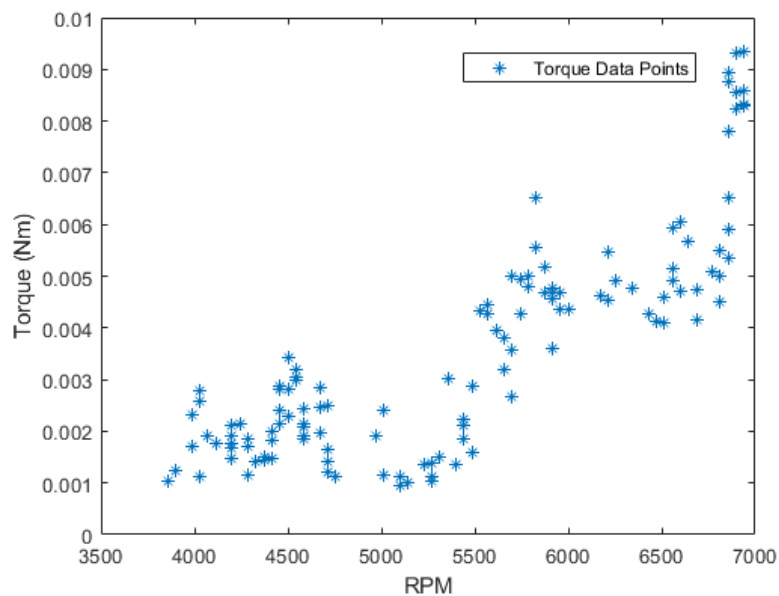


Figure 3.9: Torque against RPM

3.6 Non-Linear Model

Going through the modelling process presents the following non-linear mathematical model describing the dynamics of the Y6 hexarotor UAV, which will be used to produce a simulation of the system in MATLAB/Simulink.

Translational Motion:

$$M \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = M \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} - \begin{bmatrix} C_\theta C_\psi & -C_\theta S_\psi + S_\phi S_\theta C_\psi & -S_\phi S_\psi + C_\phi S_\theta C_\psi \\ C_\theta S_\psi & C_\phi C_\psi + S_\phi S_\theta S_\psi & -S_\phi C_\psi + C_\phi S_\theta S_\psi \\ -S_\theta & S_\phi C_\theta & C_\phi C_\theta \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ \sum F_n \end{bmatrix} - \begin{bmatrix} D_{lxx} & 0 & 0 \\ 0 & D_{lyy} & 0 \\ 0 & 0 & D_{lzz} \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix}$$

Rotational Motion:

$$\begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} A_p \\ A_q \\ A_r \end{bmatrix} - \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} - \begin{bmatrix} D_{rxx} & 0 & 0 \\ 0 & D_{ryy} & 0 \\ 0 & 0 & D_{rzz} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

Where:

$$A_p = L_1((F_1 + F_2) - (F_3 + F_4))$$

$$A_q = L_2((F_1 + F_2 + F_3 + F_4) - L_3(F_5 + F_6))$$

$$A_r = \sum_{n=1}^{n=6} \tau_n$$

and motor propulsion unit dynamics are:

$$\dot{\omega}_n = -\frac{1}{\tau_m} \omega_n + \frac{K_u}{\tau_m} \mu_n$$

$$F_n = K_a \omega_n^2 + K_b \omega_n$$

$$\tau_n = K_\tau \omega_n^2$$

Then applying $\vec{E}r = T \cdot \vec{B}r$ to convert angular velocity in Fb to angular velocity w.r.t Fe:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & S_\phi T_\theta & C_\phi T_\theta \\ 0 & C_\phi & -S_\phi \\ 0 & \frac{S_\phi}{C_\theta} & \frac{C_\phi}{C_\theta} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

3.7 Linearisation

As with the UAV dynamics presented in the previous subsection, the majority of real world systems are non-linear, in that the states of the systems evolve in some fashion that does not satisfy the criteria for linearity. However, the majority of techniques developed over the decades in systems and control theory have been developed for application to and analysis of linear systems. Therefore in order to utilise these techniques in the presence of Non-linearities it is pertinent to devise some means, a least in a limited sense, of operating with Non-linear systems as if they were linear.[11] one means of accomplishing this is through the application of the Jacobian process linearisation of non-linear systems about specific operating or equilibrium points.[11],[18]. First however, several assumptions need to be made to reduce the model complexity making it more amicable to linearisation.

1. The UAV is operating in stationary hover conditions
2. The effects of drag are negligible
3. Motor dynamics are purely quadratic

Under these assumptions:

- $\ddot{E}r \approx \dot{B}r$
- $C, D_r, \text{ and } D_l = 0$

This reduces the model down to:

$$M \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = M \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} - \begin{bmatrix} C_\theta C_\psi & -C_\theta S_\psi + S_\phi S_\theta C_\psi & -S_\phi S_\psi + C_\phi S_\theta C_\psi \\ C_\theta S_\psi & C_\phi C_\psi + S_\phi S_\theta S_\psi & -S_\phi C_\psi + C_\phi S_\theta S_\psi \\ -S_\theta & S_\phi C_\theta & C_\phi C_\theta \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ \sum F_n \end{bmatrix} \quad (3.19)$$

$$\begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} L_1((F_1 + F_2) - (F_3 + F_4)) \\ L_2((F_1 + F_2 + F_3 + F_4) - L_3(F_5 + F_6)) \\ \sum_{n=1}^{n=6} \tau_n \end{bmatrix} \quad (3.20)$$

$$\dot{\omega}_n = -\frac{1}{\tau_m} \omega_n + \frac{K_u}{\tau_m} \mu_n \quad (3.21)$$

$$F_n = K_a \omega_n^2 \quad (3.22)$$

$$\tau_n = K_\tau \omega_n^2 \quad (3.23)$$

Multiplying out the matrices and collecting terms produces 6 distinct 2nd order non-linear Differential and 1 linear differential equation replicated for each motor representing the motor dynamics:

$$\ddot{x} = \frac{\sum_{n=1}^{n=6} K_a \omega_n^2}{M} C \phi S \theta C \psi + \frac{\sum_{n=1}^{n=6} K_a \omega_n^2}{M} S \phi S \psi \quad (3.24)$$

$$\ddot{y} = \frac{\sum_{n=1}^{n=6} K_a \omega_n^2}{M} C \phi S \theta C \psi - \frac{\sum_{n=1}^{n=6} K_a \omega_n^2}{M} S \phi C \psi \quad (3.25)$$

$$\ddot{z} = g - \frac{\sum_{n=1}^{n=6} K_a \omega_n^2}{M} C \phi C \theta \quad (3.26)$$

$$\ddot{\phi} = \frac{L_1}{I_{xx}} ((K_a \omega_1^2 + K_a \omega_2^2) - (K_a \omega_3^2 + K_a \omega_4^2)) \quad (3.27)$$

$$\ddot{\theta} = \frac{L_2}{I_{yy}} ((K_a \omega_1^2 + K_a \omega_2^2 + K_a \omega_3^2 + K_a \omega_4^2) - \frac{L_3}{I_{yy}} (K_a \omega_5^2 + K_a \omega_6^2)) \quad (3.28)$$

$$\ddot{\psi} = \frac{\sum_{n=1}^{n=6} K_\tau \omega_n^2}{I_{zz}} \quad (3.29)$$

$$\dot{\omega}_n = -\frac{1}{\tau_m} \omega_n + \frac{K_u}{\tau_m} \mu_n \quad (3.30)$$

Here we can define the full state vector:

$$X = [x, \dot{x}, y, \dot{y}, z, \dot{z}, \phi, \dot{\phi}, \theta, \dot{\theta}, \psi, \dot{\psi}, \omega_1, \omega_2, \omega_3, \omega_4, \omega_5, \omega_6]$$

However for the purposes of this project the full state isn't utilised as feedback control authority in x and y are not required to maintain a stable hover. therefore, a reduced state can be defined:

$$X = [z, \dot{z}, \phi, \dot{\phi}, \theta, \dot{\theta}, \psi, \dot{\psi}, \omega_1, \omega_2, \omega_3, \omega_4, \omega_5, \omega_6]$$

3.7.1 Jacobian Method

In order to linearise the system equations, equilibrium points must be defined. These are the set of states and input values for which the system is stationary i.e:

$$\dot{X} = f(X_e, \mu_e) = 0$$

where f represents the systems dynamics, therefore, the following equilibrium input and state vectors can be defined:

$$X_e = [1.5, 0, 0, 0, 0, 0, 0, 0, \omega_{e1}, \omega_{e2}, \omega_{e3}, \omega_{e4}, \omega_{e5}, \omega_{e6}]$$

$$\mu_e = [\mu_e1, \mu_e2, \mu_e3, \mu_e4, \mu_e5, \mu_e6]$$

Following this, the system states are redefined in terms of deviations from these equilibrium points:

$$\bar{X} = X - X_e$$

$$\bar{\mu} = \mu - \mu_e$$

These deviation variables are then substituted back into the dynamics equations and a first order Taylor's series expansion is performed. where:

$$\dot{\bar{X}} = f(\bar{X}, \bar{\mu}) + \left. \frac{\partial f}{\partial X} \right|_{\mu=\mu_e}^{X=X_e} \bar{X} + \left. \frac{\partial f}{\partial \mu} \right|_{\mu=\mu_e}^{X=X_e} \bar{\mu} \quad (3.31)$$

and:

$$f(\bar{X}, \bar{\mu}) = 0$$

These differential equations approximately govern the deviation variables, and as long as they remain small, this presents a linear, time-invariant, differential equation, as the derivatives of \bar{X} are now linear combinations of the deviation states and the deviation inputs, $\bar{\mu}$ [11].

Finally, the Linear Continuous-Time State-Space model emerges following:

$$A = \left. \frac{\partial f}{\partial X} \right|_{\mu=\mu_e}^{X=X_e}, \in R^{n \times n} \quad (3.32)$$

$$B = \left. \frac{\partial f}{\partial \mu} \right|_{\mu=\mu_e}^{X=X_e}, \in R^{n \times m} \quad (3.33)$$

Where n is the number of states, m the number of inputs, A the n-by-n dynamics matrix and B the n-by-m input matrix.

Thus:

A =

$$\begin{bmatrix}
 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{2K_a\omega_e1}{M} & \frac{2K_wK_a\omega_e2}{M} & \frac{2K_a\omega_e3}{M} & \frac{2K_wK_a\omega_e4}{M} & \frac{2K_wK_a\omega_e5}{M} & \frac{2K_wK_a\omega_e6}{M} \\
 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{2L_1K_a\omega_e1}{I_{xx}} & \frac{2L_1K_wK_a\omega_e2}{I_{xx}} & \frac{-2L_1K_a\omega_e3}{I_{xx}} & \frac{-2L_1K_wK_a\omega_e4}{I_{xx}} & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{2L_1K_a*\omega_e}{I_{yy}} & \frac{2L_2K_wK_a\omega_e2}{I_{yy}} & \frac{2L_2K_a\omega_e3}{I_{yy}} & \frac{2L_2K_wK_a\omega_e4}{I_{yy}} & \frac{-2L_3K_a*\omega_e5}{I_{yy}} & \frac{-2L_3K_wK_a\omega_e6}{I_{yy}} \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{-2K_\tau\omega_e1}{I_{zz}} & \frac{2K_\tau\omega_e2}{I_{zz}} & \frac{2K_\tau\omega_e3}{I_{zz}} & \frac{-2K_\tau\omega_e4}{I_{zz}} & \frac{-2K_\tau\omega_e5}{I_{zz}} & \frac{2K_\tau\omega_e6}{I_{zz}} \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{-1}{\tau_m} & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{-1}{\tau_m} & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{-1}{\tau_m} & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{-1}{\tau_m} & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{-1}{\tau_m} & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{-1}{\tau_m}
 \end{bmatrix}
 \tag{3.34}$$

B =

$$\begin{bmatrix}
 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 \\
 \frac{K_u}{\tau_m} & 0 & 0 & 0 & 0 & 0 \\
 0 & \frac{K_u}{\tau_m} & 0 & 0 & 0 & 0 \\
 0 & 0 & \frac{K_u}{\tau_m} & 0 & 0 & 0 \\
 0 & 0 & 0 & \frac{K_u}{\tau_m} & 0 & 0 \\
 0 & 0 & 0 & 0 & \frac{K_u}{\tau_m} & 0 \\
 0 & 0 & 0 & 0 & 0 & \frac{K_u}{\tau_m}
 \end{bmatrix}
 \tag{3.35}$$

Where $n = 14$ and $m = 6$.

3.8 Summary

This Chapter:

- Introduced and defined the UAV system under consideration.
- Discussed and derived the mathematical equations describing the non-linear behaviour of the UAV, from both the kinematics and kinetics representations while implementing Euler angles, the Newton-Euler convention, and describing their relevance to modelling.
- Presented the means by which through experimental identification of the key aspects that define the actuation forces may be derived and discussed the relevance of mass distribution considerations.
- Presented a means of converting the derived non-linear dynamics into a form which could be utilised in the development of linear control systems, then developed and presented said linearised model.

Model Parameters		
Character	Name	Value
M	Mass	1.455882 Kg
I _{xx}	Inertia about Xb	0.014N/m ²
I _{yy}	Inertia about Yb	0.028N/m ²
I _{zz}	Inertia about Zb	0.038N/m ²
D _{lxx}	Translational Drag Coefficient in Xe	0.01212
D _{lyy}	Translational Drag Coefficient in Ye	0.01212
D _{lzz}	Translational Drag Coefficient in Ze	0.0648
D _{rxx}	Angular Drag Coefficient in Xb	0
D _{ryy}	Angular Drag Coefficient in Yb	0
D _{rzx}	Angular Drag Coefficient in Zb	0
L ₁	Lateral Length From Motor 1 To Center Of Mass	0.19 m
L ₂	Longitudinal Length From Motor 1 To Center Of Mass	0.18 m
L ₃	Longitudinal Length From Motor 5 To Center Of Mass	0.30 m
K _u	Motor RPM Gain	515.5 × τ_m
τ_m	Motor RPM Time Constant	$\frac{1}{44.22}$
K _w	Quadratic RPM to Force Gain	0.85
K _a	Quadratic RPM to Force Gain	1.812e-07
K _b	Linear RPM to Force Gain	0.0007326
K _{τ}	Quadratic RPM to Torque Gain	7.708e-10
μ_e	Equilibrium Input values	(176.1, 178.5, 177.2, 177.6, 202.2, 202.4)
ω_e	Equilibrium Motor Angular Velocities	$\mu_e \times K_u$
g	Acceleration Due To Gravity	9.81

Table 3.1: Table of Model Parameters

Chapter 4

Linear Quadratic Gaussian and Model Predictive Control

4.1 Introduction

The objective of this investigation is to devise control systems capable of taking the UAV to a state of stable hovering conditions. The purpose of designing and implementing control systems is to reject or minimise the effects of disturbance to the system, as well as change the characteristics of the system so as to meet performance requirements. Therefore in order to accomplish this with the UAV model derived in chapter 3, this investigation seeks to develop both LQG and an LQ-MPC control systems. These are both advanced linear model based controller that are extremely powerful in their application and are able to nominally provide stability guarantees with wide margins. In order to develop and then implement these the following themes will be explored:

1. Discretization
2. Model Analysis
3. LQG
4. LQ-MPC

4.2 Discretization

The Linearised model obtained in chapter 3 is derived in the continuous-time domain. In order to design control systems which can be implemented onto a digital embedded microprocessor, it must be in discrete-time domain.

This can be done either by designing the control systems in the continuous-time domain then converting the resulting continuous control laws into discrete ones, or by transforming the entire continuous-time system model to the discrete-time domain then designing the control law using the discrete-time system model [18]. This investigation utilises the latter method.

To convert a state space model of form:

$$\dot{X}(t) = AX(t) + B\mu(t)$$

Produces:

$$X(k+1) = A_{dt}X(k) + B_{dt}\mu(k) \quad (4.1)$$

where:

$$A_{dt} = e^{AT} \quad (4.2)$$

$$B_{dt} = A^{-1}(e^{AT} - I)B \quad (4.3)$$

T is the discretization sample time, k is the time increment, and I an identity matrix of sufficient dimension.

4.3 Model Analysis

Model-Based feedback control systems design utilises insights gained from having the system model on hand to inform the design process and determine how best adjustments should be made. In order to accomplish this, the system is first thoroughly analysed and understood. Given the linear state state-space model developed at the end of chapter 3, the discretization performed in previous section, along with the values presented in the Table 3.1, it is possible to use linear system analysis techniques to arrive at key insight such as:

- Stability
- Reachability
- Observability

4.3.1 Stability

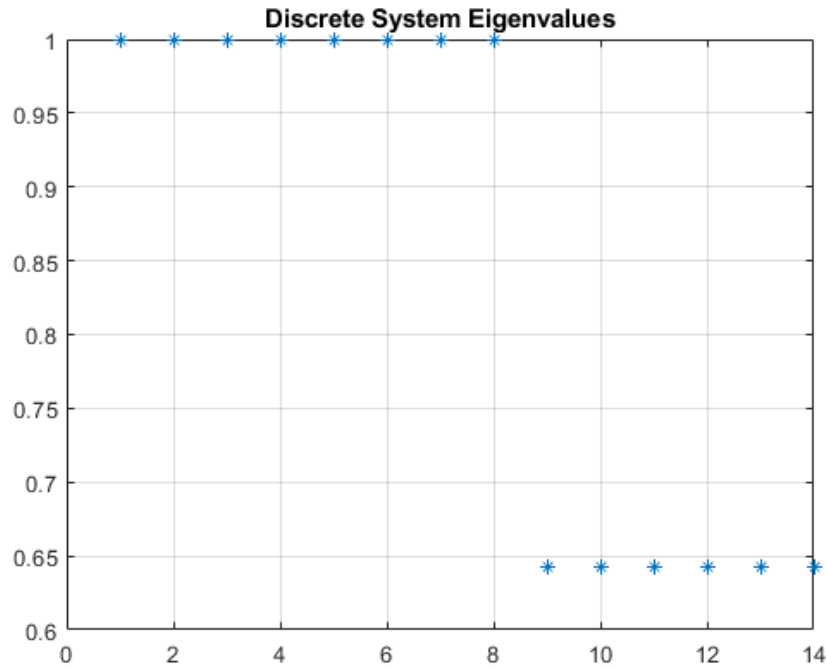


Figure 4.1: Open-Loop Eigenvalues

Stability theory plays a key role in the design of control. There are different kinds of stability problems that arise in the study of dynamical systems. The stability of systems about equilibrium operating points is usually characterised in the Lyapunov sense where an equilibrium point is said to be stable where all solutions starting nearby the equilibrium points stay nearby for all time, otherwise the system is said to be unstable [13],[18].

This definition of stability can be further specified to define stability in the asymptotic sense, where a system is said to be asymptotically stable if all solutions starting nearby the equilibrium points not only stays nearby, but also tend to the equilibrium point as time goes to infinity [13],[18].

For linear state-space models, stability can be ascertained from the locations of the eigenvalues, λ , of the A_{dt} matrix, This is know as Lyapunov's indirect method [13] where:

$$\det|A_{dt} - \lambda I| = 0 \quad (4.4)$$

Figure 4.1 reveals the presence 6 eigenvalues with a value less than 1, and 8 eigenvalues with a value of 1. For discrete-time systems, stability can be said to be asymptotic if all eigenvalues are strictly within the unit circle, Lyapunov if all eigenvalues are within or on the unit circle,

and unstable otherwise [18]. Therefore the linear systems model can be said to be stable in the sense of Lyapunov. This implies also that the non-linear systems model is locally stable in the sense of Lyapunov about the equilibrium points X_e .

However due to real world disturbances such as model uncertainty, sensor noise and external wind disturbances, the system will not remain at the point of equilibrium save for the intervention of some control law to reject the effect of these disturbances. Hence the system although statically stable is dynamically unstable.

4.3.2 Reachability

Reachability is a property which is determined by coupling the inputs into the system and the states, a system is said to be reachable if for any initial set of state values X_0 , and any final set of state value, X_f , there is a control signal $\mu(k)$ that can take the system from X_0 at $k = 0$, to X_f with some finite time [18].

A system is said to be reachable if the reachability matrix:

$$W_r = \begin{bmatrix} B_{dt} & A_{dt}B_{dt} & \dots & A_{dt}^{n-1}B_{dt} \end{bmatrix} \quad (4.5)$$

is full rank equalling n .

Thus, as the rank of W_r is equals to n , the system can be said to be reachable.

The less stringent property of stabilisability can also be defined where only the unstable states need be reachable.

4.3.3 Observability

Like reachability, Observability is a property which is determined by coupling, in this case, that between the system outputs and the system states. Therefore the system outputs, Y , must be defined where Y is a subset of the state X :

$$Y \in X$$

The measured systems outputs are:

$$Y = [z, \phi, \theta, \psi] \quad (4.6)$$

There for a matrix to perform the state selection is defined as:

$$Y(k) = CX(k) \quad (4.7)$$

$$C \in R^{p \times n}$$

where p is the number of outputs.

Thus:

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (4.8)$$

$$p = 4$$

where:

$$C_{dt} = C$$

A system is said to be observable if for all time, it is possible to, from a limited set of state measurements $Y(k)$, and the inputs $\mu(k)$, to *uniquely* obtain the full state [18].

A system is said to be observable if the observability matrix:

$$W_o = \begin{bmatrix} C_{dt} \\ C_{dt}A_{dt} \\ \vdots \\ C_{dt}A_{dt}^{n-1} \end{bmatrix} \quad (4.9)$$

is full rank equalling n .

Again as with reachability, a less stringent property, detectability, can also be defined where only the unstable states need be observable.

Interestingly, for this combination of dynamics A_{dt} and C_{dt} , it is found that the W_o is not full rank, and therefore the system in its current form is not observable. Upon further analysis, it is discovered that this is as a result of a phenomenon created due to the Coaxial propulsion unit configuration, as there are numerous combinations of thrust force values that each individual propulsion unit can generate which produce the same resultant torque about the centre of mass making it impossible to uniquely obtain the full state. One possible solution is to modify the UAV design to include RPM sensors to measure at least 2 non-coaxial motors. However, in the upcoming LQG section, a more elegant solution is proposed.

4.4 LQG

4.4.1 Full-State Feedback

Given a discrete-time linear state-space model of form:

$$X(k+1) = A_{dt}X(k) + B_{dt}\mu(k)$$

$$Y(k) = C_{dt}X(k)$$

Full-state feedback is a control technique which makes use of the full-state, $X(k)$ of a system, to modify the system's eigenvalue so as to change its characteristic responses[18]. This is done by determining some set of gains K which when applied to the system states and the resultant of which being fed back into the system, adjust the dynamics of the system, where:

$$\mu(k) = -KX(k) \quad (4.10)$$

This leads to a modified closed loop system:

$$X(k+1) = (A_{dt} - B_{dt}K)X(k) \quad (4.11)$$

Where the eigenvalue are now determined by:

$$\det|(A_{dt} - B_{dt}K) - \lambda I| = 0 \quad (4.12)$$

4.4.2 LQR

LQR is a technique for finding the set of gains K , in a manner which is in some wise optimal [18],[19],[20],[25]. In order to accomplish this, LQR seeks to find a control sequence μ to minimise a *quadratic* cost function by formulating the control problem as optimisation problem $P_{\infty}(X(0))$, where :

$$P_{\infty}(X(0)) =$$

$$\min_{\mu} V_{\infty}(X(0), \mu) = \sum_{k=0}^{k=\infty} X^T Q X + \mu^T R \mu \quad (4.13)$$

s.t

$$X(k+1) = A_{dt}X(k) + B_{dt}\mu(k), k = 1, 2, 3 \dots$$

Where Q n-by-n, and R m-by-m, are 2-norm weighing matrices on the states and inputs respectively.

Such an optimisation over an infinite horizon, as $k \rightarrow \infty$, would be intractable, however through the application of the Bellman's principle of optimality, the solution emerges as a result of the derivation of the Discrete Algebraic Riccati Equation (DARE) [25] where:

DARE =

$$\begin{aligned} K &= -(R + B_{dt}^T \Pi B_{dt})^{-1} B_{dt}^T \Pi A_{dt} \\ \Pi &= Q + A_{dt}^T A_{dt} - A_{dt}^T \Pi B_{dt} (R + B_{dt}^T \Pi B_{dt})^{-1} B_{dt}^T \Pi A_{dt} \end{aligned} \quad (4.14)$$

In order for the optimisation problem to be feasible, certain criteria must be satisfied [18],[25]:

- Q be at least Positive Semi-definite (P.S.D) and R be strictly Positive Definite (P.D).
- (A_{dt}, B_{dt}) must be at least stabilisable.
- (C_{dt}, A_{dt}) must be at least detectable.

4.4.3 Reference Tracking and Integral Action

The Control Problem as set up in the previous subsections is on capable of realising a control system with the goal of driving all system states to the equilibrium. However, in order for the control system to be usefully implemented on a UAV, it must be able to track and set outputs Y to, a desired reference target signals, r , preferably with no errors asymptotically.

Therefore, to accomplish this, the system's model must be augmented to include behaviour which seeks to minimise errors between a reference target and UAV output. hence:

$$X_{aug} = \begin{bmatrix} X \\ X_I \end{bmatrix} \quad (4.15)$$

Where $X_I = r - Y$:

$$X_I = \begin{bmatrix} Z_I \\ \phi_I \\ \theta_I \\ \psi_I \end{bmatrix} \quad (4.16)$$

This Leads to augmented discrete state-space Matrices:

$$A_{aug} = \begin{bmatrix} A_{dt} & 0 \\ -C_{dt}A_{dt} & I \end{bmatrix}, B_{aug} = \begin{bmatrix} B_{dt} \\ -C_{dt}B_{dt} \end{bmatrix}, C_{aug} = \begin{bmatrix} C_{dt} & 0 \end{bmatrix} \quad (4.17)$$

Where 0 is a square zeros matrix of dimension P and I is an identity matrix of dimension p . Solving the LQR optimisation problem with, $A_{aug}, B_{aug}, C_{aug}$ and Q $n+p$ -by- $n+p$, given this augmented system, produces a control law [19]:

$$\mu(k) = -KX(k) + K_I X_I(k) \quad (4.18)$$

Where:

$$X_{aug}(k+1) = (A_{aug} - B_{aug}K_y)X_{aug}(k) + B_{aug}K_I r(k) \quad (4.19)$$

$$K_y = \begin{bmatrix} K & -K_I \end{bmatrix}$$

This is implemented in the MATLAB-Simulink environment while taking into account the SBC dynamics such the effects of quantisation as well as digital to analog and vise-versa conversions which are always present in digital controllers which interface with the analog continuous-time systems:

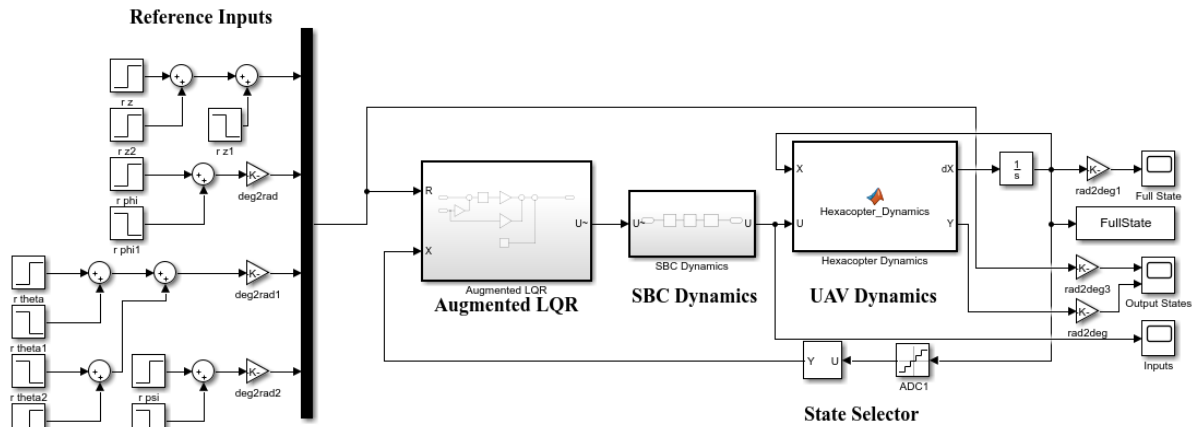


Figure 4.2: LQR Simulink Implimentation

4.4.4 State Estimation

The control law devised in LQR subsection was produced under the assumption that the full state of the system, X , is available for full-state feedback, however, this is not the case as only a limited number of state measurements are available and these form the measured outputs, Y .

Therefore, in order to utilise a full-state feedback control law, the full-state of the system must be estimated from the limited measurements present. This is performed through the implementation of a State Observer/Estimator. State estimators accomplish this by apply knowledge of the system dynamics through the utilisation of the system's model and its observability property [18],[21],[22],[23],[24].

The estimator mimics the system dynamics giving:

$$\begin{aligned}\hat{X}(k+1) &= A_{dt}\hat{X} + B_{dt}\mu(k) + Le \\ \hat{Y}(k) &= C_{dt}\hat{X}\end{aligned}\tag{4.20}$$

where \hat{X} and \hat{Y} are the estimated system states and outputs, L is the estimator gain with e being the output error, which is the difference between the estimator model and the system outputs.

$$e = Y - \hat{Y}$$

As with the systems model, the behaviour of estimator model and thus the quality of the estimation is determined by the state estimation error dynamics [18], where:

$$\hat{X}_e = X - \hat{X}$$

The open-loop estimator dynamics are given by:

$$\hat{X}_e(k+1) = A_{dt}(X - \hat{X})\tag{4.21}$$

Due to the implementation of the observer gain L, the closed-loop estimator dynamics are give by:

$$\hat{X}_e(k+1) = (A_{dt} - LC_{dt})\hat{X}_e\tag{4.22}$$

If the eigenvalues of the estimator exist outside the unit circle, then the estimator is unstable and the state estimation is be divergent, and if the system is observable, then it is always possible to derive some gain L, to produce any set of desired eigenvalues [18]

4.4.5 Estimator Modification

In the earlier observability subsection, it was found that the system is not observable due to propulsion unit configuration, and the emergent thrust and torque combinations making the estimation of a unique full state impossible.

This can be rectified simply by making the system observable by modifying the hardware and measuring the value of some of the motor states via an RPM sensor.

There is another way this can be solved without the need to obtain additional sensors, that is by virtually augmenting the output, Y to include terms for at least 2 non coaxial motor pairs, the returning and error vector where the error value for those 2 motors is 0, therefore:

$$C_y = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}, Y_y = \begin{bmatrix} Z \\ \phi \\ \theta \\ \psi \\ \omega_1 \\ \omega_6 \end{bmatrix}, e = \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \\ 0 \\ 0 \end{bmatrix} \quad (4.23)$$

This augmentation has the effect of using the model to completely determine the value of the motor by assuming it is a perfect model of the actual system. The augmentation is done solely for the purpose of state estimation so there are still only 4 reference input to the controller and only 4 measured outputs, but with this, the system is now fully observable.

The Kalman Filter

Kalman Filtering encompasses a series of techniques for obtaining an estimator gain L , which is in some wise optimal. This is accomplished by constructing an optimisation problem with the objective of minimising a set of noise measurements covariances which arise from disturbances to the system such those due to model uncertainty and noise present in the output sensor readings, these sources of noise are modelled as Gaussian noise distributions. Thus, when used in combination with the LQR, the whole system is called the Linear Quadratic Gaussian.

The Kalman Filter problem is a corollary problem to the LQR where optimisation is also performed over infinite horizon, requiring the application of Bellman's optimality principle and is

thus derived resulting in another DARE [18],[21].

There are 2 major implementations of the Kalman filter which are popularly employed.

- The Linear Kalman Filter (LKF)
- The Extended Kalman Filter (EKF)

The process described above only fully pertains to the design of an LKF. The EKF is utilised in estimating the states of non-linear systems, utilising a the non-linear system model directly but then implementing an online lineariser that linearises the system at each time interval, k , to produce a linear model which is then used to solve for the estimator gain L [21],[22],[243].

However, through experimentation with the estimator model , it was found that a the systems states could be estimated by modifying the LKF to use the non linear model derived in chapter 3, similar to the EKF. This modified LKF would then use the linear model to derive the estimator gain L and the full state estimations would be derived from applying this gain to the discretised non-linear model.

This is possible due to the limited operational scope where the system is not expected to deviate far enough from the equilibrium operating points about which the non-linear model was linearised as presented in chapter 3. Therefore this method could be called a Limited EKF of sorts.

This Limited EKF modification results in more accurate state and output predictions than the standard LKF implementation, as shown when the errors between the actual non-linear model outputs and estimator outputs are compared for both, as shown in Figures 4.3 and 4.4 during a 100 second simulation.

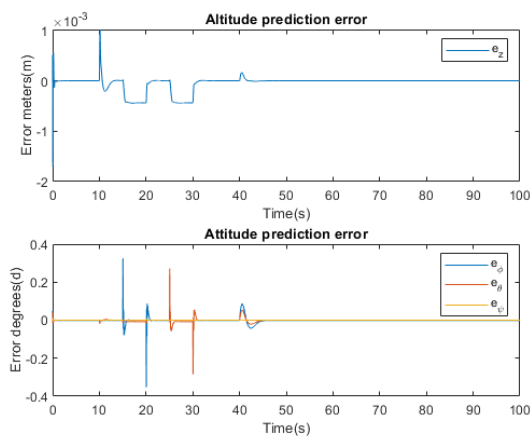


Figure 4.3: KF Error

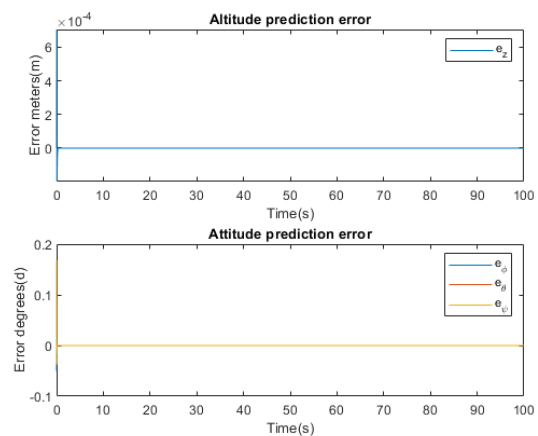


Figure 4.4: Limited EKF Error

4.4.6 Results

Figures 4.5 shows the closed-loop system eigenvalues , where all values are now in within the unit circle. Figures 4.6 to 4.8 show the response plots of the system inputs, and both the estimated and measured outputs respectively, as the simulation is run over an interval of 100 seconds. The system is subjected to reference step inputs where the results are displayed in Table 4.1:

Step Responses					
Output	Reference	Overshoot (%)	Settling Time (s)	Rise Time (s)	Steady State Error
Z	1 meter	< 5	3	< 2	0
ϕ	30 degrees	< 5	< 1	< 0.5	0
θ	30 degrees	< 5	< 1	< 0.5	0
ψ	90 degrees	0	< 5	< 3	0

Table 4.1: Table of Step Response values

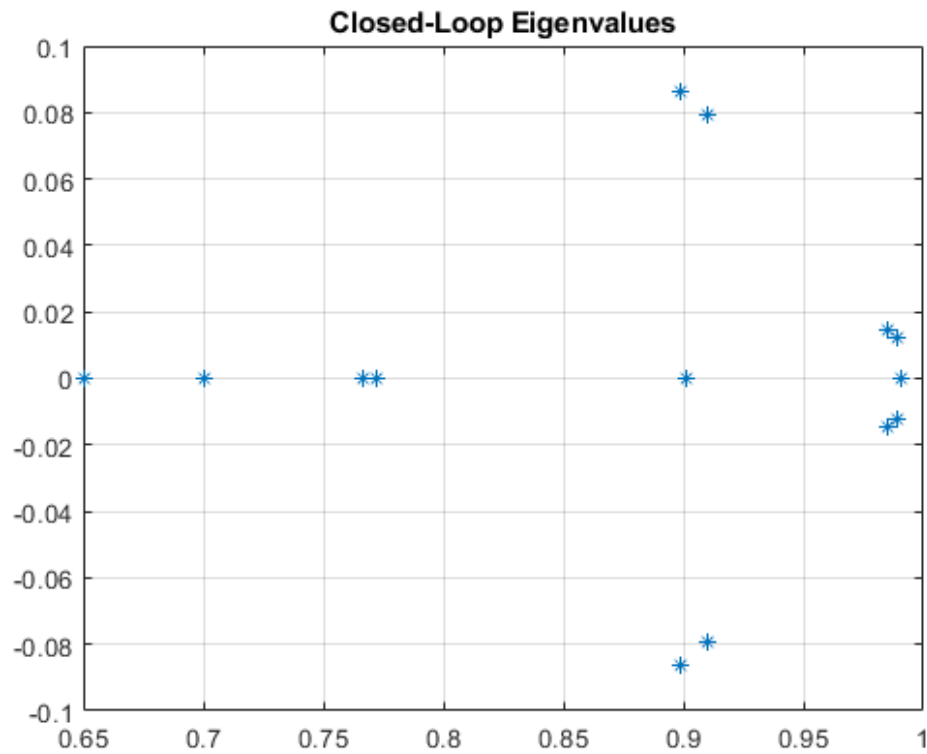


Figure 4.5: Closed-Loop Eigenvalues

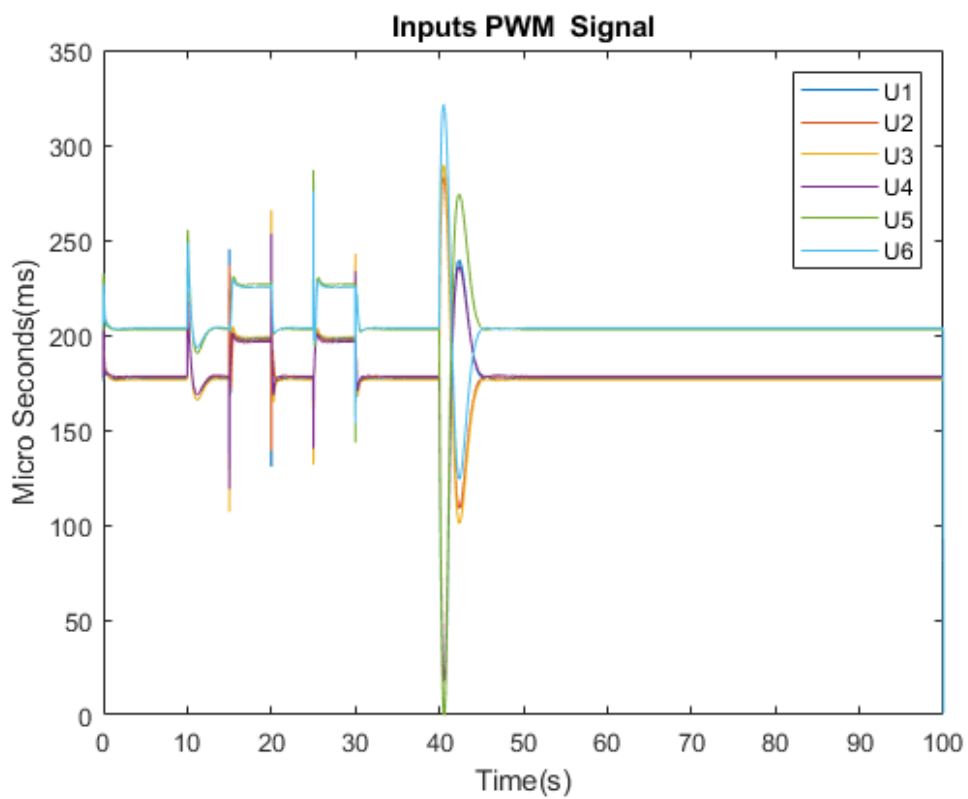


Figure 4.6: System Input PWM Signal

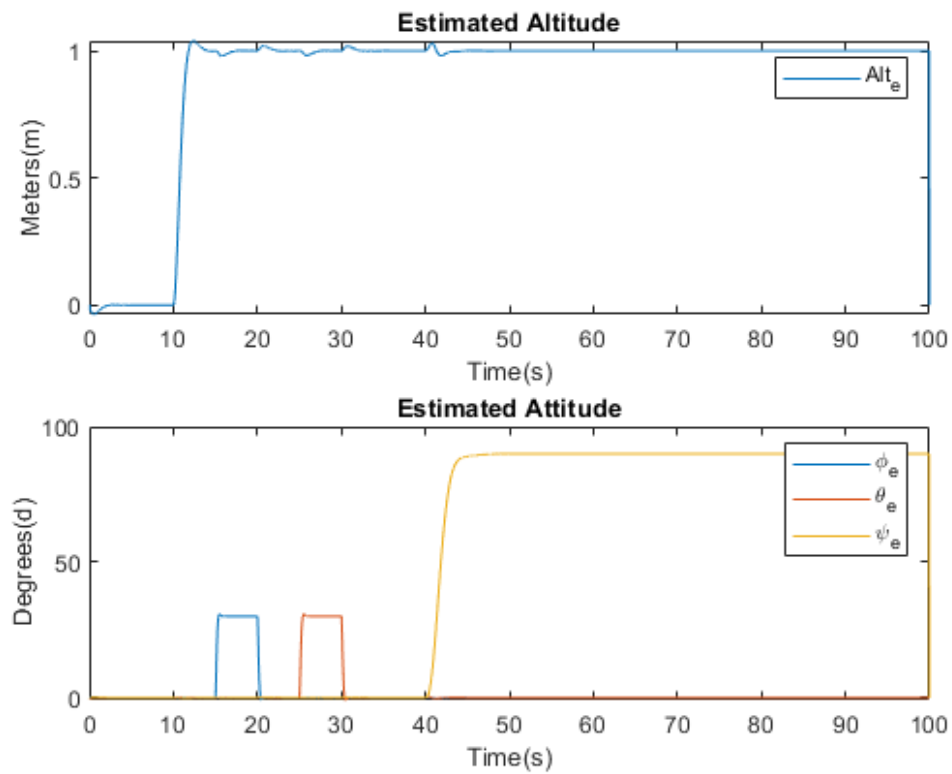


Figure 4.7: System Estimated Outputs

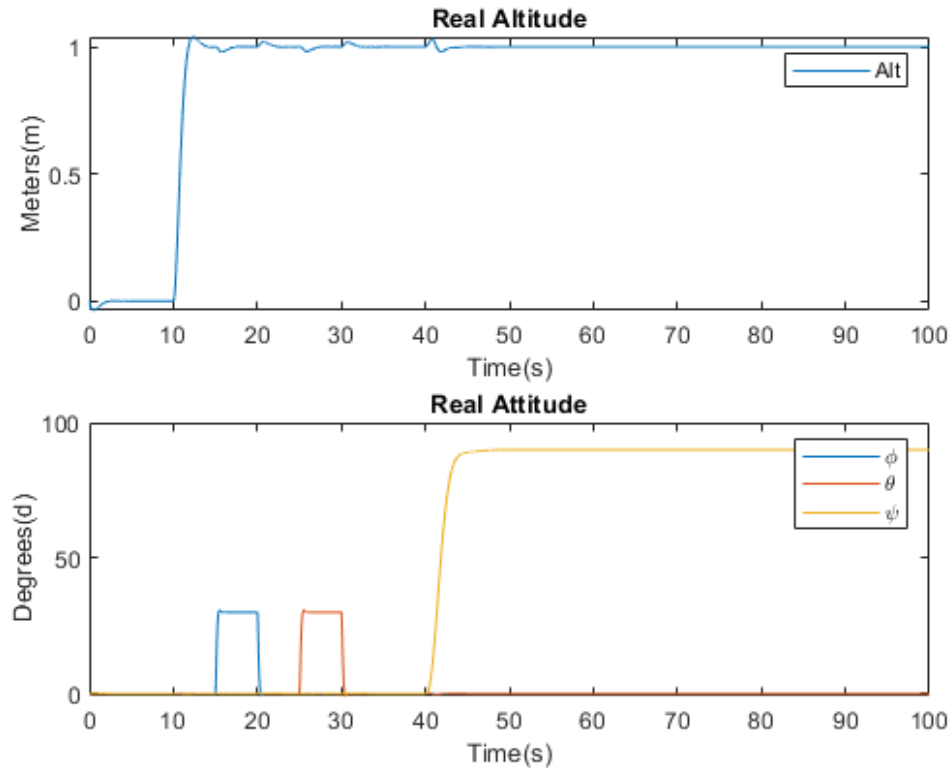


Figure 4.8: System Measured Outputs

4.5 LQ-MPC

4.5.1 Limitations of LQG control

Like LQG, MPC is also a kind of optimal control, however, this technique's cost or performance function is limited to looking at future states over some fixed finite horizon and control signals can be determined so that they implicitly taking into account and determine control action so as to meet physical constraints on the system. MPC is also known as receding horizon control (RHC) as it only acts on the first step in the horizon of states before recalculating and performing this action over and over until the target is achieved.

MPC does not refer to any one algorithm or method but instead a series of methodological philosophy which implements certain key concepts and ideas in various ways [28]. Thus, this investigation makes a limiting delineation by specifying LQ-MPC which limits the scope to MPC consisting of linear systems and quadratic cost functions.

4.5.2 Derivation of LQ-MPC

4.6 Summary

This Chapter:

- Introduced and defined the UAV system under consideration.
- Discussed and derived the mathematical equations describing the non-linear behaviour of the UAV, from both the kinematics and kinetics representations while implementing Euler angles, the Newton-Euler convention, and describing their relevance to modelling.
- Presented the means by which through experimental identification of the key aspects that define the actuation forces may be derived and discussed the relevance of mass distribution considerations.
- Presented a means of converting the derived non-linear dynamics into a form which could be utilised in the development of linear control systems, then developed and presented said linearised model.

Chapter 5

Embedded Control Systems Implementation

5.1 Introduction

Advanced control systems need to be implemented using digital computers. In an application such as with a UAV, this computation need to occur on a device as physically small as possible, due to design considerations such as minimising total vehicle mass. On such class of applicably small computers are SBCs. These are usually fully capable computers in their own right running light weight operating systems, such as watered down Linux distributions, that typically have smaller physical dimensions than a smart phone. Such devices have become possible and prevalent due to the advent of smartphone technology driving the trend towards cost effective minimisation of processing units.

Therefore this chapter will go through the considerations and processes involved in implementing control systems and proposes an implementation via a systems engineering framework. Which are:

1. Hardware
2. Sensors
3. Software Libraries
4. UML Implementation

5.2 Hardware

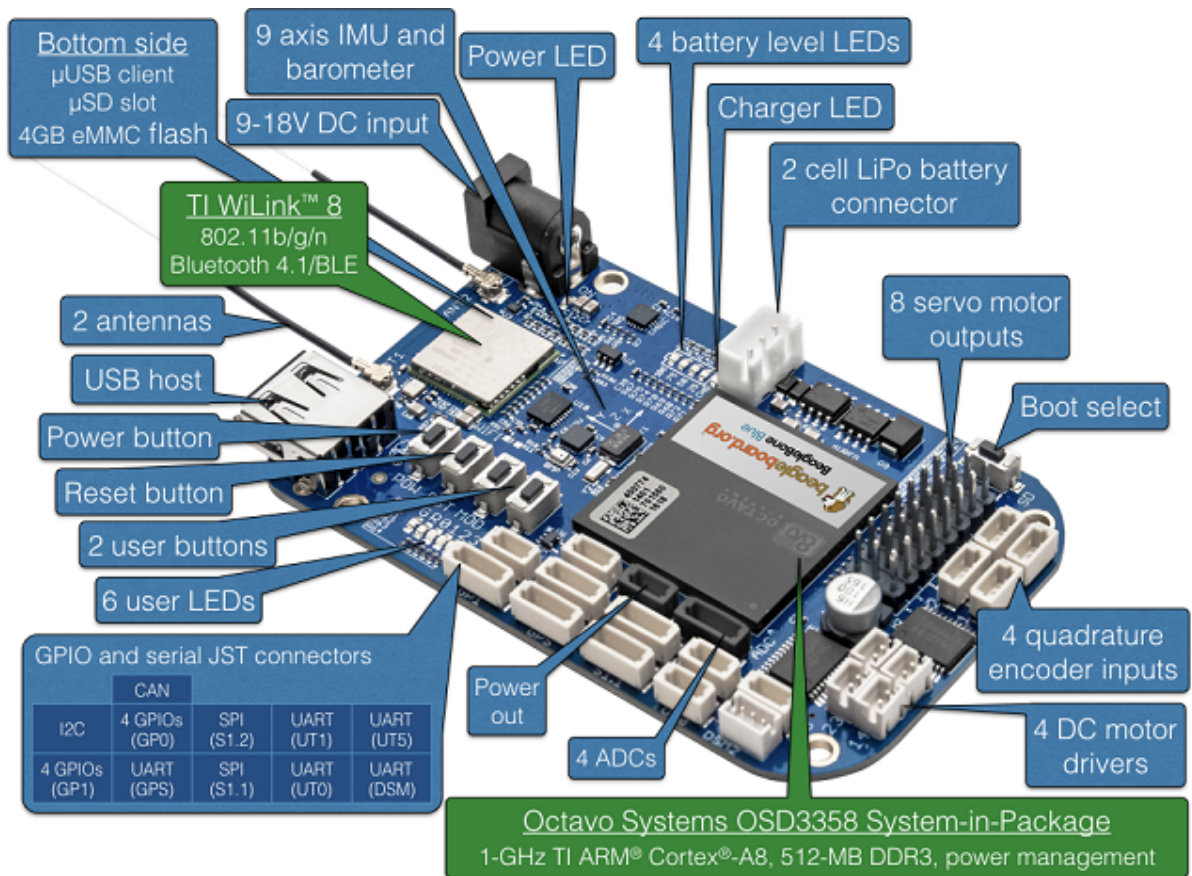


Figure 5.1: BeagleBone Blue SBC

The SBC under consideration for the project is Beagle Bone Blue (BBB) from the Beagle-Board group, shown in Figure 5.1, which, in addition to the computation provided by all SBCs, comes equipped with multiple, peripheral communication interfaces such as:

- Inter-Integrated Circuit (I^2C)
- Serial Peripheral interface (SPI)
- General Purpose Input-Outputs (GPIOs)
- Universal Asynchronous receiver-Transmitter (UART)

The BBB possesses a 1 Gigahertz ARM Cortex-A8 micro-processor unit, supported by 512 Megabytes of DDR3 random access memory (RAM). This SBC is also capable of wireless

communication through various Bluetooth and Wi-Fi standards. However for the purposes of this investigation all communication to the SBC will occur via a radio transmitter to a waiting receiver onboard the UAV. This transmitter will effectively be sending the reference command, r , commands to the controller.

This comes equipped with multiple sensors, the addition of which are the most pertinent hardware inclusions with regards to this investigation:

- Magnetic, Angular rate, Gravity Sensor (MARG)
- Barometric Pressure Sensor (Barometer)

5.2.1 MARG and Barometer

A MARG sensor consists of a 3-axis gyroscope, which measures angular rate, a 3-axis accelerometer which measures linear acceleration, and a 3-axis magnetometer which measures the strength of the earth's magnetic field.

Barometers are sensors which measure atmospheric pressure. Their use on UAVs is key as pressure on the surface of the earth is a function of the altitude where it is being measured, therefore the pressure readings from a barometer can be used to estimate the altitude above sea-level of a body [20],[29].

When combined, the sensors can be used to estimate the angular position and altitude of the body they are attached to. In the form factor able to be implemented on an SBC, MARGs come as micro electro-mechanical systems (MEMS) sensors which take the form of integrated circuits. MEMS devices contain moving masses and are able to convert the movement of these masses into readable data [20],[29]. The MARG implemented here is the MPU-9650 from InvenSense and the Barometer is the BMP-280 from Bosch.

In order to be used these devices they must be calibrated to remove bias and offset errors due to the effect of external disturbances, especially on MEMS devices [29].

5.3 Software and UML

and even software additions [31] presents consideration for the implementation of flight control systems Sources [20] extending from the Robot Control Library (RCL)

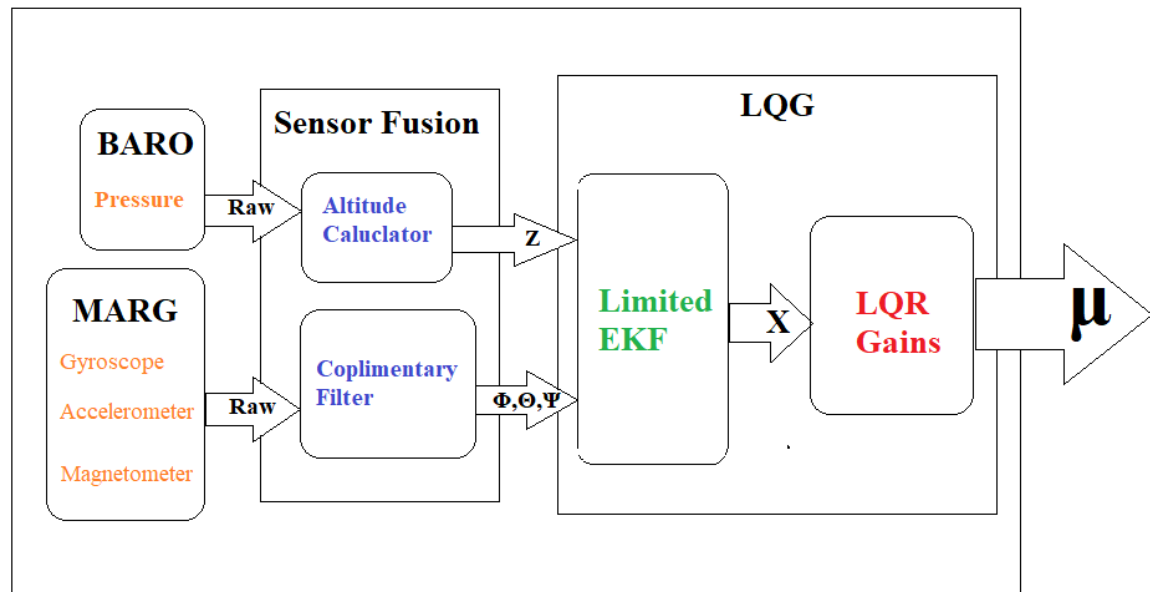


Figure 5.2: Embedded Component Structure

[30] presents considerations for designing software in general looking at systems engineering approaches.

5.4 Summary

Control and Implementation Parameters		
Character	Definition	Value
T	Discrete Sample Time	0.01 s
DAC	Digital to Analog converter resolution	10 bits
ADC	Analog to Digital converter resolution	10 bits
V_{op}	SBC Operating Voltage	3.3 v

Table 5.1: Table of Control and Implementation Parameters

Chapter 6

Conclusions