THE UNIVERSITY OF THE SOUTH PACIFIC
LIBRARY

Author Statement of Accessibility

Name of Candidate: SHEIKH IZZAL AZID

Degree: DOCTOR OF PHILOSOPHY IN ENGINEERING

Department/School: SCHOOL OF ENGINEERING AND PHYSICS

Institution/University: THE UNIVERSITY OF THE SOUTH PACIFIC

Title of Thesis: ADAPTIVE CONTROL STRATEGIES ROBUST TO
EXTERNAL DISTURBANCES FOR MULTIPROPOR UAV'S: THEORETICAL
ASPECTS AND EXPERIMENTAL VALIDATION

Date of award: OCTOBER, 2018

(Month, year)

Please tick

This thesis may be:

1. consulted in the Library without the author's permission

2. cited without the author's permission provided it is properly acknowledged

3. photocopied in whole without the author's written permission
   If you answered 'No', select the percentage that may be photocopied
   Under 20% ☑ 20-40% ☐ 40-60% ☐ 60-80% ☐ Over 80% ☐

I authorize The University of the South Pacific to:

4. produce a microfilm or microfiche copy for preservation purposes

5. retain a copy in electronic format for archival and preservation purposes

6. make this thesis available on the Internet
   YES ☑ NO ☐

7. make this thesis available on the USP Intranet
   YES ☑
   NO ☐ Give reasons

Signed: [Signature]

Date: 20/07/2020

Contact Information

Email address: shiekhizzal2@gmail.com

Phone Mobile: 9216622

Home: 9216622

Work: 3232859
ADAPTIVE CONTROL STRATEGIES ROBUST TO EXTERNAL DISTURBANCES FOR MULTIROTOR UAV’s: THEORETICAL ASPECTS AND EXPERIMENTAL VALIDATION

by

Sheikh Izzal Azid

A thesis submitted in fulfillment of the requirement for the degree of Doctor of Philosophy

Copyright © 2018 by Sheikh Izzal Azid

School of Engineering and Physics
Faculty of Science, Technology and Environment
The University of the South Pacific
October 2018
Declaration of Originality

Statement by Author

I, Sheikh Izzal Azid, declare that this dissertation is my own work and has not been copied from any other sources. All the sources that have been used are acknowledged by appropriate references.

Signature

Date 26-11-18

Name Sheikh Izzal Azid

Student Id NO. s11012039

Statement by Supervisors

The research in this thesis was performed under our supervision and to our knowledge, it is the sole work of Mr. Sheikh Izzal Azid and has not been plagiarized from any other sources.

Maurizio Cirrincione
Professor of Engineering and Head of School
School of Engineering and Physics
University of the South Pacific (USP)
Lauca Campus, Suva
Fiji

Adriano Fagiolini
Assistant Professor
Co-Supervisor
Mobile and Intelligent Robot Laboratory
Department of Energy, Computer Science, and Mathematical Models (DEIM)
University of Palermo (UNIPA)
Palermo
Italy

Giansalvo Cirrincione
Associate Professor
Co-Supervisor
Department of Electrical Engineering
Université de Picardie Jules Verne (UPJV)
France
Acknowledgements

I would like to take this opportunity to thank all the individuals and organizations who have contributed to the successful completion of this dissertation. Firstly, my sincere appreciation to Professor Maurizio Cirrincione for his continuous guidance and tremendous support during my PhD years and also during my research visit to Palermo, Italy. Moreover, I am very thankful to Dr. Adriano Fagiolini for his continuous guidance throughout the years of my PhD, even though it had to be late night or early morning Skype meets along with my memorable visit to Palermo and his to Fiji.

I would also like to thank The University of the South Pacific and more importantly, The Faculty of Science, Technology and Environment for the PhD grant. My sincere appreciation also goes to the Human Resources SDA Funding which supported my visit to The University of Palermo. A hearty thanks also goes to the Department of Energia, Ingegneria dell’Informazione e Modelli Matematici (DEIM) of The University of Palermo for my visit. A special and hearty thanks also goes to the Acting Vice President, Digital Services & Human Resources, Dr. Anjeela Jokhan and the Dean of the Faculty, Dr. Bibhya Sharma, for their continuous support and words of encouragement. I would also like to acknowledge the support from the School of Engineering and Physics with its outstanding academic and technical staff as well as students with a special thanks to Dr. Ali Mohammadi.

Lastly, I would like to extend my appreciation and acknowledge my family, my wife and my friends; my parents Mr. & Mrs. Sheikh Ajid and brother Mr. Sheikh Zoinal Mushin and sister Zaynah Saheb in Raviravi, Ba, my wife, Mrs. Zahra Nizbat Azid, for her continuous support and understanding towards me over the years while I was working on my PhD. I got married to her in the middle of my PhD studies and came to realize that I started to perform better with studies with her by my side. Also, many thanks to my in-laws in Navu Nadi (Mr. & Mrs. Mohammed Aleem, Mr. Mohammed Shameer and Mr. Mohammed Zubair). A vinaka vakalevu and appreciation also to all my friends to who were there by my side in thick and thin with words of encouragement and support.
Dedication

This thesis is dedicated to my family…

*Abba, Mummy ... Zahra... Zoinal and Zaynah* 

*Dada, Dadi .... Nana Nani*
# Table of Contents

Acknowledgements .......................................................................................................................... iii

Dedication ........................................................................................................................................ iv

List of Figures .................................................................................................................................... viii

List of Tables ...................................................................................................................................... xiii

Abstract ............................................................................................................................................ xiv

Foreword ........................................................................................................................................... xv

Chapter 1 .......................................................................................................................................... 1

1.0 Introduction ................................................................................................................................. 1

1.1 Literature Review ......................................................................................................................... 2

1.1.1 UAV History .......................................................................................................................... 2

1.1.2 A little bit of Quadrotor History [4] ..................................................................................... 3

1.2 Quadrotor Control Methods ....................................................................................................... 4

1.3 State and Input Observers ......................................................................................................... 8

1.4 Thesis Motivation and Outline ................................................................................................. 11

Chapter 2 ........................................................................................................................................... 13

Mathematical Model and PD Control of Quadrotor ....................................................................... 13

2.1 Quadrotor Geometry .................................................................................................................. 13

2.2 Quadrotor Modelling ................................................................................................................ 16

2.3 State-space Model ..................................................................................................................... 20

2.4 Linear Modelling and PD Control ............................................................................................. 21

2.4.1 Attitude Control .................................................................................................................... 21

2.4.2 Position Control .................................................................................................................... 24

Chapter 3 ........................................................................................................................................... 33

Unknown Input Observers for Linear Dynamic System with Partially Unknown Inputs ................. 33

3.1 Linear Unknown Input Observers ............................................................................................. 33
Calculations for Linear UIO................................................................. 114
Appendix 2 ................................................................................................. 122
Calculations for Nonlinear UIO.............................................................. 122
List of Figures

Figure 1.1 (a) ‘Arial Torpedo’, the first UAV and (b) Firebee [3] 2

Figure 1.2 First successful quadrotor [3] 3

Figure 1.3 Diagram of PID controller applied to quadrotor 5

Figure 1.4 Block Diagram of Adaptive controller applied to quadrotor 6

Figure 1.5 Scheme of an observer in a control system 9

Figure 2.1 Earth Frame and Body frame used for Modelling [56] 13

Figure 2.2 Quadrotor Aircraft [56] 14

Figure 2.3 Propeller’s design curve showing the relation of rotor velocities with Forces and Moments [5] 15

Figure 2.4 The MATLAB Simulink Model PD control of the Linearized Model of the Quadrotor 28

Figure 2.5 Linear Position of the quadrotor with target position of \((x = 1m, y = 2m and z = 3m)\) with no wind gusts 29

Figure 2.6 Angular Position of the quadrotor with target position of \((\phi = \frac{\pi}{2}, \theta = 0, \psi = 0)\) with no wind gusts 29

Figure 2.7 Linear Position of the quadrotor with target position of \((x = 1m, y = 2m and z = 3m)\) with wind gusts \([Wx = 2N, Wy = 2N \text{ and } Wz = 2N]\) as step input 30

Figure 2.8 Angular Position of the quadrotor with target position of \((\phi = \frac{\pi}{2}, \theta = 0, \psi = 0)\) with wind gusts \([Wx = 2m/s, Wy = 2m/s \text{ and } Wz = 2m/s]\) 30

Figure: 2.9 shows the time varying wind gusts type 31

Figure 2.10 Linear Position of the quadrotor with target position of \((x = 1m, y = 2m and z = 3m)\) with time varying wind gusts as shown in Figure 2.9 31
Figure 2.11 Angular Position of the quadrotor with target position of \((\phi = \frac{\pi}{2}, \theta = 0, \psi = 0)\) with time varying wind gusts as shown in Figure 2.9

Figure 3.1 Graphic representation of UIO Scheme

Figure 3.2 The MATLAB Simulink Model of UIO

Figure 3.3 Estimated versus actual wind gusts for step input

Figure 3.4 The error graph between estimated and actual wind gusts which is shown in Figure 3.3

Figure 3.5 Estimated versus actual wind gusts for military grade wind model

Figure 3.6 The error graph between estimated and actual wind gusts which is shown in Figure 3.5

Figure 3.7 Estimated versus actual wind gusts for military grade wind model

Figure 3.8 The error graph between estimated and actual wind gusts which is shown in Figure 3.7

Figure 4.1 General Scheme of the Model

Figure 4.2 Results of military grade wind gusts for scenario 1 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded \(\phi\) and \(\theta\); (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory

Figure 4.3 Results of military grade wind gusts for scenario 2 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded \(\phi\) and \(\theta\); (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory

Figure 4.4 Results of military grade wind gusts for scenario 3 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded \(\phi\) and \(\theta\); (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory

Figure 4.5 Results of military grade wind gusts for scenario 4 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded \(\phi\) and \(\theta\); (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory
Figure 4.6 Results of step wind gusts for scenario 1 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory

Figure 4.7 Results of step wind gusts for scenario 2 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory

Figure 4.8 Results of step wind gusts for scenario 3 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory

Figure 4.9 Results of step wind gusts for scenario 4 (a) shows the linear position; (b) shows the orientation; (c) shows the rotor speeds; (d) shows the commanded $\phi$ and $\theta$; (e) shows the actual and estimated wind; (f) shows the state estimation error and (g) shows the 3D trajectory

Figure 4.10 Results of time varying wind showing (a) the wind gusts with estimation; (b) the quadrotor position and (c) the desired spiral trajectory

Figure 4.11 showing the Erlecopter model in Gazebo® software

Figure 4.12 shows the communication between topics and nodes

Figure 4.13 Results of Gazebo® for scenario 1 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the 3D trajectory

Figure 4.14 Results of Gazebo® for scenario 2 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the 3D trajectory

Figure 4.15 Results of Gazebo® for scenario 1 (a) shows the linear position; (b) shows the orientation; (c) shows the rotor speeds; (d) shows the commanded $\phi$ and $\theta$; (e) shows the actual and estimated wind; (f) shows the 3D trajectory

Figure 4.16 The quadrotor flight during experimental testing and on bottom left of the figure is the wind measuring instruments

Figure 4.17 Experimental results (a) the actual and estimated Wx; (b) the actual and estimated Wy
Figure 4.18 Experiment results showing (a) the actual and estimated $Wx$; (b) the actual and estimated $Wy$; (c) the linear position (d) the angular positions (e) the rotor speeds

Figure 5.1 Scheme of the nonlinear unknown input observer

Figure 5.2 Estimated versus actual wind gusts for step input

Figure 5.3 The error graph between estimated and actual wind gusts

Figure 5.4 State estimation error graph

Figure 5.5 Estimated versus actual wind gusts for military grade wind model

Figure 5.6 The error graph between estimated and actual wind gusts

Figure 5.7 State estimation error graph

Figure 5.8 Estimated versus actual wind gusts for military grade wind model

Figure 5.9 The error graph between estimated and actual wind gusts

Figure 5.10 State estimation error graph

Figure 6.1 General scheme of the nonlinear UIO on the nonlinear model

Figure 6.2 Results of military grade wind gusts for scenario 2 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory

Figure 6.3 Results of military grade wind gusts for scenario 3 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory

Figure 6.4 Results of military grade wind gusts for scenario 4 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory
Figure 6.5 Results of step type wind gusts for scenario 1 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory

Figure 6.6 Results of step type wind gusts for scenario 2 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory

Figure 6.7 Results of step type wind gusts for scenario 3 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory

Figure 6.8 Results of step type wind gusts for scenario 4 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory

Figure 6.9 Results of time varying wind showing (a) the wind gusts with estimation; (b) the quadrotor position and (c) the desired spiral trajectory

Figure 6.10 Communication between ROS®, Gazebo® and Ardupilot

Figure 6.11 Results of Gazebo® for scenario 1 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the 3D trajectory

Figure 6.12 Results of Gazebo® for scenario 2 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the 3D trajectory

Figure 6.13 Results of Gazebo® for scenario 3 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the 3D trajectory
List of Tables

Table 1.1 Comparison of quadrotor control algorithms [25] 7
Table 4.1 Parameters of the System in SI Units 47
Table 6.1 Parameters of the System in SI Units 85
Abstract

This thesis presents a contribution to the state-of-the-art control of quadrotors in terms of robustness to the presence of such disturbances as wind gusts and trajectory tracking. Ch.1 presents the state of the art of the linear and nonlinear control of quadrotors, Ch.2 presents the nonlinear model of the system also in terms of state-space variables and the under-actuation nature of the problem is highlighted. Ch.3 presents the first contribution of the thesis, that is the application of Unknown Input Observer to the estimation of the unknown states of the system in presence of wind gusts, which are subsequently estimated so as to compensate them in the control action. Ch.4 gives extensive results of the application of the UIO on the linearized model of the system both in simulation (in Gazebo®/ROS® environment) and experimentally on a real quadrotor aircraft (Erlecopter). Ch. 5 describes the second major contribution of the thesis, the development and application of a nonlinear UIO to the quadrotor system, and its verification in Ch.6 in simulation but also comparatively with the linear UIO. The experimentation and the simulations have been made by assuming either step wind gusts or, more realistically, the military graded wind gusts as well as time varying wind gusts.
Foreword

Over the years the interest to the control of UAV (Unmanned Aerial Vehicles) has been ever increasing in the Region of the Pacific, especially for disaster monitoring in case of natural hazards as well as for security purposes. UAVs can fly up to an altitude of 400 feet and are able to follow the GPS-guided routes in case of emergencies such as Tsunami. Their cameras gather images with normal light, infrared or thermal and still photos or video formats. For these types of operations the UAV’s are more flexible than other methods, less expensive and safer than other methods. In any case, one of the most challenging tasks for a UAV is to achieve every control objective in terms of trajectory in spite of the presence of external disturbances, like sudden wind gusts. Indeed, the dynamical model of a UAV, like a quadrotor, is highly nonlinear and moreover has 2 degrees of under actuation, since it has only 4 inputs and six outputs (the position). In this respect the control action is very hard and it becomes very challenging in presence of wind gusts. This thesis, therefore, wants to address the control of a quadrotor by using a novel observer, the UIO (Unknown Input Observer), capable of estimating all of the states of the quadrotor even in presence of unknown wind gusts, which are subsequently retrieved to synthetize a compensation action to counteract them.

After a thorough literature review in Ch.1 both on the control of quadrotors and the observers which are currently used, Ch. 2 presents the geometry and the mathematical model of the quadrotor together with the traditional PD control of attitude and position. Ch.3 then addresses the issue of the observation of the states of a quadrotor in presence of disturbing wind gusts by presenting a linear UIO applied to the linearized model of the quadrotor around a point of quasi-hovering. This has permitted the Author to familiarize with the technique by using the sound theory of UIO developed in literature and give a contribution to the control of UAV’s, since there are no examples of such application in the literature. In particular both simulation and experimental results have been made extensively and presented in Ch.4 by using a real quadrotor aircraft. Then Ch.5 addresses the more complex challenge of applying a nonlinear UIO (NUIO) to the quadrotor system. This topic has not been at all developed in the area of UAVs as yet in the literature and this PhD thesis wants to give another contribution to the design, development and implementation on NUIO to quadrotors as well as assess the performance of the system. Unfortunately only simulations results have been
presented, even if on a realistic environment as the one offered by the Gazebo®/ROS® software, since the NUIM requires accurate GPS measures that the current available aircraft is not able to give. Despite this, the results which have been obtained have been excellent with accurate estimation of the wind, states and consequent path correction. Also in this case extensive results have been obtained in different realistic scenarios and comparison with the linear UIO has also been made. The wind gusts which have been adopted are the step, military graded and time-varying ones, so as to have scenarios similar to real world operation.

Each chapter is self-contained and can be read independently of the others, so as to drive the reader not only to the full comprehension of the topic as related to its application to quadrotors, but also to the use of UIO to other control schemes which are affected by important disturbances.
Chapter 1

1.0 Introduction

An Unknown Aerial Vehicle (UAV) is defined as a pilot-less aircraft which can be controlled by a pilot at the ground control station or can fly autonomously based on preprogrammed flight plans [1]. Recent advances in sensors, microcontroller technology, control and aerodynamics theory have made UAV a reality. Moreover, the UAV is a good tool for monitoring areas where natural disasters have occurred. UAV’s are specifically of high interest in the Pacific Region where remote areas can be aerially checked after cyclones or floods.

Within UAVs, quadrotors are most popular for their flexibility due essentially to a very low moment of inertia and great stability as well as fewer take-off requirements and better hovering. This has resulted in their use in a number of applications like monitoring marine and agricultural sectors, disaster surveillance and surveying, where UAV’s are required to be autonomous with perfect stability on any flight conditions. Furthermore, the low cost and maneuverability of these systems have made them a potential to change the world in a positive way.

Among the objectives of an autonomous quadrotor great effort has been made to the design of the sensor system. However, for quadrotors which have small mass and inertia and also complex structure, sensors used on board are more affected by noise compared to larger systems. Particularly, for a quadrotor to fly in harsh environmental conditions, such as wind disturbances, is an open challenge; Actually, robustness issues can result from very complicated aerodynamic effects, and this together with inaccurate sensor makes it difficult to control the quadrotor. To overcome these issues of sensors dynamical estimation observers for unknown input and states can be an attractive solution.

The use of observers is therefore justified by the fact that state estimation together with dynamical estimation of the unknown inputs can be used to compensate for the unknown disturbances on the quadrotor. For this purpose The Unknown Input Observers (UIO) can be to control and diagnose. Indeed UIO can tolerate a good degree of model uncertainty and therefore increase the reliability of fault diagnosis.
Although UIO can be tracked back to the 1970’s the problem of designing such observers is still of paramount importance in theoretical and experimental prospects [2]. This thesis focuses on sensorless control of quadrotor vehicles which are required to accurately track position trajectories in the presence of low to moderate unknown wind gusts.

1.1 Literature Review

1.1.1 UAV History

The first UAV’s, as shown in Figure 1.1a, were developed as a long-range armament and are considered as forerunners of cruise missiles[3]. In 1917, the US Navy presented ‘Aerial Torpedo’, a 270Kg and 40 horsepower pilotless bomber, made of wood. This vehicle included gyrostabilizer to keep the aircraft level, an automatic steering to head the vehicle and a barometer to indicate the cruise altitude, allowing the aircraft to level off. Moreover a wind driven electrical generator was used to power the motors.

![Aerial Torpedo](image1)  ![Firebee](image2)

Figure 1.1 (a) ‘Aerial Torpedo’, the first UAV and (b) Firebee [3]

From this era, vast advancements have been made in UAV research. In the 1960’s, UAV’s were used for exploration purposes during wars: Example the ‘Firebee’ carried a still camera whose photos were developed at base after the return of the UAV and this aircraft used a parachute for landing. A variety of designs were developed over the years and to best describe UAV, they can be classified into four categories which are as follows:
1. Fixed wing aircraft with advantage of high cruise speed and long endurance;
2. Flapping-wing vehicles, flying like birds and insects;
3. Bimps, which are long endurance balloons and;
4. Rotary wings, which are also called Vertical Take-Off and Landing (VTOL) vehicles, with hovering capability and high maneuverability.

1.1.2 A little bit of Quadrotor History [4]

Etienne Oehmichen was the first engineer who experimented with rotorcraft designs in 1920. Out of his six designs, one of them had 4 rotors and 8 propellers driven by a single engine, and exhibited a considerable degree of stability during its time. Afterwards, Dr George de Bothezat and Ivan Jerome developed the model with an X shape structure. However, the highest altitude this design ever reached was 5 meters, because this vehicle had to carry the pilot. Another aircraft, Convertawings Model A quadrotor, as shown in Figure 1.2, featured two engines driving four rotors without the tail rotors for varying the thrust: It flew successfully in 1955 and proved the first successful quadrotor design. However, due to lack of commercial orders or military versions, the project was terminated.

![Figure 1.2 First successful quadrotor [3]](image-url)
1.2 Quadrotor Control Methods

Manual flight control system design for aircraft continues to be one of the most demanding problems in the autonomous world. The advantage of this multi-rotor is that each rotor with a smaller diameter, compared to conventional helicopters, stores less kinetic energy during flight. This allows safe maneuvering with lower risk of damaging the environment. Nevertheless, a quadrotor has nonlinear, coupled and under actuated dynamics, which poses serious challenges in control system design. The quadrotor inputs are the four rotor speeds and outputs are the linear and angular positions (x, y, z; roll, pitch, yaw) and velocities (u, v, w, p, q, r). The problem is inherently multivariable, that is, a controller must drive multiple effectors based on the information from multiple states and inputs. A number of control techniques have been proposed for quadrotors. The goal of all these methods are to allow the states to converge to a desired point. For the quadrotor, linear control techniques on the linearized model are more common [5]-[12] due to the simplicity of the quadrotor model when compared to the nonlinear model. The nonlinear model of the quadrotor is linearized around the hovering state with various techniques. The most common methods used are the Taylor series and the derivative rule [5]. [6] has linearized quadrotor model derived from system identification method which is based on collected experimental data. After obtaining quadrotor model, an optimal controller is designed.

There are various linear control techniques, the most common is PID in [7]–[11]. [7] discusses the PID control for the dynamic model of the quadrotor for vertical takeoff and landing (VTOL). The system is under actuated due to fewer inputs then outputs, hence it only works for hovering. Cascaded PI-PID controller is another linear control method where a PI acts as an outer loop controller to control the attitude. The other PID controller acts as an inner loop controller, which reads the output of outer loop controller as set point, controlling a rapid changing parameter, namely the angular velocities [8], [9], [11]. Online self-tuning PID control has also shown good results, the PID controller parameters, proportional gain (Kp), integral gain (Ki) and derivative gain (Kd) which are tuned online while flying [10], [12].
The Linear Quadratic Regulator (LQR) control methods have been used to realize faster stable attitude control of the linear quadrotor system: a simple path following LQR controller was applied on the full dynamic model of the quadrotor, and accurate path following was achieved in simulation [13]. In addition, [14] presented a full linear H∞ controller where the controller showed satisfactory performance in the simulation environment, but was unable to stabilize the vehicle in real experiments. A comparison between LQR and H∞ was made and LQR showed better results.

In general, all linear control methods discussed not be able to work if external disturbances are exposed to it, since they neglect the nonlinearity of the model, which actually should be accounted for, also to ensure its stability.

Nonlinear control methods ought to be used to achieve advanced performance of the quadrotor. Some of the nonlinear techniques include feedback linearization [15], [16]; sliding mode [17] and back stepping [18] which have demonstrated significant achievements in controlling the nonlinear model of the quadrotor. Feedback linearization control algorithm transforms the nonlinear system model into an equivalent linear system through a change in variables. The linearization choice is between states and outputs [15], [16]. Often for feedback linearization, the feedback controller is very sensitive to noise and is not robust when compared to sliding mode controllers [19]. Model Predictive Control is discussed in [20] where it is used on a feedback equivalent system and its control outputs are transformed back into the inputs for the original system. Sliding mode control is a nonlinear control algorithm that works by applying discontinuous control signal to the system to command it to slide...
along a prescribe path [21]. Neural network control [16] and sliding mode control [17], [21], [22] methods are also used to control the quadrotor flight, but the performance of these control methods severely depends on prior information related to the upper bound of uncertainty. Also a fuzzy controller was applied by [23] to control the position and the attitude of the quadrotor with good response in simulation; however, a major limitation of this work was its trial and error approach.

The control methods discussed above are used mainly to stabilize a quadrotor which has been equipped with diverse devices such as camera, measurement instruments and payload, which lead to changes in center of gravity, mass, and inertia properties of the quadrotor system. In other cases, to overcome the uncertainty of the model, adaptive control can be adopted to parameter changes in the system. These parameters, which are either unclear or varying in time[24].

Figure 1.4 Block Diagram of Adaptive controller applied to quadrotor

A review of control algorithms form quadrotor was surveyed by [25]. The performance of each controller on the quadrotor for asymptotic stability is shown in Table 1.1
The issue of robustness is a major concern against the uncertainty of the quadrotor environment. Apart from adaptive control, all other methods mentioned above are used to control and stabilize the quadrotor in a known environment with known parameters. In case of unknown disturbances such as wind, the system becomes unstable and therefore one of the major issues is controlling and stabilizing it in the presence of wind. In [26], a fault tolerant controller is developed for attitude control of quadrotor in presence of actuator fault and wind; here, the wind is the measured noise which occurs perpendicular to the vehicle but in parallel to ground and is compensated. Tilt rotor design is introduced to increase the actuation capability of the quadrotor because the usual 4 inputs to the system increase up to 8 inputs. The ability to tilt rotors of the quadrotor improves its versatility and enables the accomplishment of more critical tasks. The results show that the quadrotor with tilt rotors has good ability to compensate the measured wind gusts [27]. Another method is presented in [28] where the vehicle is controlled with the help of two loops: an inner loop for attitude stabilization and an outer loop, feeding the attitude loop, to ensure trajectory tracking. This technique was successful in simulation with the presence of wind gusts, but no experimentation was made. Cascaded PID [29] and Fuzzy [30] controllers are also used for robustness of the quadrotor, but all these methods use their own compensator.
design to reject the effect of external disturbances. Adaptive methods are also used to achieve robustness where the wind disturbances are assumed as a finite sum of sinusoidal functions with unknown frequencies, amplitudes and phases [31].

Upon looking at the literature mentioned above, there are various successful linear and non-linear controllers. Nonetheless, when faced with external disturbances, robustness is a major concern. To overcome this issue, either more sensors or wind model compensators are used together with these control techniques, which however rely on several assumptions, making them less feasible.

### 1.3 State and Input Observers

As discussed above, control systems are generally used to stabilize almost all types of systems. Most of the concepts used in the control systems are based on having sensors for closed-loop control. Unfortunately, physical sensors have shortcomings that degrade the control systems[32]. Firstly, for the quadrotor, which is very sensitive to weight, sensors including the wiring add extra load to the system. Secondly, the sensors induce errors, such as noise, due to many common reasons like changes in temperature or harsh environmental conditions. Thirdly, it becomes impractical to measure some signals using sensors, since such devices are unavailable. Finally, high costs of the sensors can eventually raise the total cost of the system.

To decrease the number of sensors, observers can be used. They are algorithms based on knowledge of the model and some measured signals to compute unmeasured signals (either states or outputs). These observed signals are less expensive to produce, and more reliable than signals derived from sensors. A typical scheme of a control system with an observer id shown in Figure 1.5., where the estimated state vector \( \hat{x}(t) \), computed on the basis of the input \( u(t) \) and the measured output vector \( y(t) \), is used by the controller.
The theory of observers originated through the work of Luenberger in 1964 [2], [33], [34]. Luenberger’s observer reconstructed the state vector of a linear time invariant system by observing the systems inputs and outputs. In particular, consider a linear dynamic system where \( x(t) \) is a vector of \( n \) components, the input \( u(t) \) has \( m \) components and the output \( y(t) \) has \( p \) components.

\[
x(t) = Ax(t) + Bu(t) \tag{1.1}
\]
\[
y(t) = Cx(t) \tag{1.2}
\]

If the system output are available, the information is used to estimate the all of the system state components. Since matrices \( A, B \) and \( C \) are known, the observer can be as follows, where \( K \) is chosen by the user

\[
\hat{x}(t) = A\hat{x}(t) + Bu(t) + K(y(t) - \hat{y}(t)) \tag{1.3}
\]
\[
\hat{y}(t) = C\hat{x}(t) \tag{1.4}
\]

If the estimation error is \( e(t) = x(t) - \hat{x}(t) \) then

\[
\dot{e}(t) = (A - KC)e(t) \tag{1.5}
\]

Where \( K \) is \( n \times p \) matrix which represents the observer gain that has to be chosen so that \( A-KC \) has negative eigenvalues (Hurwitz Matrix). It is clear that, when \( n = p \), the observer gain matrix \( K \) can be determined and this is known as full order observer design. If some states are known the reduced order observer can be used [35].

Over the years, research in the design of observers has accumulated through literature due to challenges in the requirements of high accuracy, low cost and good prediction performances. In [36] [37] Wang proposed a minimal order unknown input observer (UIO) structure for linear systems with both known and unknown inputs by estimating
the state of the linear time invariant multivariable system for both full and partial state systems with unmeasured disturbances inputs Bhattacharya also used the Luenberger method to evaluate a given set of linear functions of the state observers to extract the unknown input of the system using a geometric approach [38]. The inversion algorithm of Silverman has been used in [39] to show the dynamic portion of the inverse system which gives a partial state observer of the system having completely unknown inputs whereas [40] employed singular value decomposition to formulate a technique for design of Luenberger observers, when there are only known inputs. An algebraic approach was presented in [41] by using a straightforward derivation of the reduced order observers for linear systems. Finally, works on the application of linear observers for fault diagnosis have been ever increasingly developed [42]–[44].

Recently, Sundaram and Hadjicostics constructed a reduced order state observer for discrete time linear system models with unknown inputs. Their approach provides a characterization of observers with delay, which eases the establishment of necessary conditions for existence of UIO with zero delay. By introducing a delay, the observer has the potential to be used in a variety of systems and applications such as feedback control, fault diagnosis and system identification [40] Other papers enlarge the scope of these observers for partial state observers for unknown inputs [45]–[47]. However, all methods mentioned above are restricted to linear systems.

Since the late 1990s, research has shifted from linear to nonlinear system state observers. Earlier class of non-linear Lipschitz system has been studied [48], [49], and in this case the observer design is based on the linear part by imposing certain conditions on the nonlinearity. However, this approach inherited drawbacks that were related to observer convergence conditions which are considered difficult to satisfy for large value of Lipschitz constant [50]. This problem was solved later when [51] proposed to use a Lyapunov function in order to guarantee asymptotic stability. The increasing convergence rate showed good robustness results and, moreover, [52] proposed the application of the $H_\infty$ optimization theory for asymptotic convergence of the observer states.

Researchers in [43] later proposed an algorithm for stable nonlinear observer design for the nonlinear system by placing the eigenvalues far on the left half s-plane so long as the eigenvectors are sufficiently well conditioned for asymptotic stability. Based on
these conditions a computational algorithm has been developed to obtain the observer gain matrix.

Linear Matrix Inequality (LMI) approach has been used for the estimation of states and unknown inputs for Lipschitz nonlinear systems [53], [54]. Actually, the nonlinear unknown input observer (NUIO) condition requires solving a nonlinear matrix inequality which is derived from this method. An advantage of this method is that it permits full order NUIO to be designed using standard commercial software. In general, the NUIO can be applied to systems whose dynamic can be expressed by:

\[
\dot{x} = Ax + Bu + f(x) + Dv \\
y = Cx
\]

Where \( f(x) \) is the nonlinear and \( Ax \) and \( Bu \) are the linear parts of the system, and \( Dv \) is the unknown input. The assumption for \( f(x) \) would be that there exists a positive constant \( \gamma \) which satisfies the Lipschitz condition.

\[
|f(x) - f(\hat{x})| \leq \gamma |x - \hat{x}|
\]

In this direction [55] proposes the design of the nonlinear observer by using the differential mean value (DMV) theorem with an original approach for decoupling the unknown input. In this way, the NUIO design is achieved on the basis of a Lyapunov approach and LMI conditions. This thesis will use this last approach after showing that the quadrotor model can be comprised into this class of dynamical systems.

1.4 Thesis Motivation and Outline

As mentioned in the above paragraphs, the control of quadrotors is an area of research which has been gaining momentum over the last decades in practically all aspects. However, the presence of external disturbances still remains a topic of interest both for the trajectory tracking and the disturbance rejection. One of the motivation of this thesis is to explore both in simulation and experimentally the possibility of improvement in the state estimation and control action by using the UIO presented above. Specifically, this thesis wants to contribute to the application of linear UIO to the linearized model of a quadrotor system and the application of a nonlinear UIO to the nonlinear model of a quadrotor. This has been achieved both in simulation and experimentally for the linear UIO and in simulation for the NUIO, since the accuracy
of GPS signals required by the method could not be obtained with the available quadrotor.

The rest of the thesis is organized as follows

**Chapter 2** outlines the mathematical model of the quadrotor. Together with the model, a PD controller will be designed for the linearized version of the model.

**Chapter 3** describes the Linear Unknown Input Observer and its design for the linearized model of the quadrotor while estimating the wind gust on the quadrotor.

**Chapter 4** presents the Nonlinear Unknown Input Observer applied to the nonlinear model of the quadrotor and the wind gust acting on the quadrotor are estimated.

**Chapter 5** shows the MATLAB-Simulink® simulation results of the linear UIO and NUIO.

**Chapter 6** discusses the Gazebo® and Experimental results of the UIO and NUIO. The test bed of the experiments is also described in detail.

**Chapter 7** draws conclusions and proposes possible future work and development.
Chapter 2
Mathematical Model and PD Control of Quadrotor

2.1 Quadrotor Geometry

The mechanical structure of a quadrotor aircraft consists of a planar cross-shaped rigid chassis, actuated by four independent rotors which are mounted at the ends of the arms of the chassis itself. Before describing the mathematical model, it is necessary to introduce the reference coordinates which describe position and attitude of the quadrotor as illustrated in Figure 2.1. More precisely, the information about the quadrotor’s current position is measured by GPS sensors, thus, making it convenient to describe the center of mass \((x, y, z)\) in the inertial Earth frame \(F_O\). On the other hand, the information about the orientation is obtained from Inertial Measurement Unit (IMU), thus the attitude is described in the body frame \(F_B\) by the set of Euler angles \((\phi, \theta, \psi)\).

![Figure 2.1 Earth Frame and Body frame used for Modelling [56]]

The position and orientation of the quadrotor can be controlled by varying the velocity of the independent rotors. The aircraft attitude is described by the so-called roll \(\phi\), pitch \(\theta\), and yaw \(\psi\) angles. Referring to figure 2.2, roll \(\phi\) indicates a rotation around the x-axis and is obtained with the torques applied by rotors 2 and 4; pitch \(\theta\) is a rotation about the y-axis and is obtained with the torques applied by rotors 1 and 3; yaw \(\psi\) represents a rotation around the z-axis, obtained with simultaneous changes of velocities of rotors 1, 3 and 2, 4. The four brushless DC motors of the quadrotor
generate thrust forces $F_1, F_2, F_3$ and $F_4$, supporting the weight of the aircraft and producing further maneuvering. Additionally, four motor driver boards, also known as Electronic Speed Controllers (ESC), are needed to amplify the power delivered to the motors. Their rotation is then communicated to the propellers. A quadrotor has six degrees of freedom which means it is able to translate in all three directions and rotate in space.

![Figure 2.2 Quadrotor Aircraft [56]](image)

The above mentioned four forces are each proportional to the square of the angular velocities of the respective rotors. The relation between each rotor’s rotation and the produced thrust force $F_i$ is nearly quadratic, that is,

$$F_i = K_F \Omega_i^2,$$  

(2.1)

where $K_F$ is a force constant and $\Omega$ is the speed of the rotor.

In order to spin at desired velocity, each rotor has to overcome drag torque $M_i$, also being quadratic with the velocity of the motor

$$M_i = K_M \Omega_i^2,$$  

(2.2)

where $K_M$ is a Torque constant.
Figure 2.3 Propeller’s design curve showing the relation of rotor velocities with Forces and Moments [5]

The Figure 2.3 above shows that each motor has speed-torque characteristics, indicating the exact torque each motor is able to generate. At hovering, each rotor has to roughly support one fourth of the vehicles weight, thus, the rotational rotor speed required for hovering $\Omega_o$ is

$$\Omega_o = \sqrt{\frac{mg}{4K_F}}. \tag{2.3}$$

Hence, the torque required to support the motor is

$$\tau_o = K_M\Omega_o^2 = \frac{K_M mg}{K_F 4}. \tag{2.4}$$

The constants $K_M$ and $K_F$ are known and depends on the propellers shape and air density.
2.2 Quadrotor Modelling

As discussed, a quadrotor has six degrees of freedom and its position is defined by the three coordinates of the center of mass \((x, y, z)\) with respect to the earth \(F_o\), as well as the three angles \((\phi, \theta, \psi)\) that give the attitude of the aircraft with respect to the center of mass, to which the respective speeds are to be added. This totals up to 12 internal states which are the positions \((x, y, z, \phi, \theta, \psi)\) and velocities \((\dot{x}, \dot{y}, \dot{z}, \dot{\phi}, \dot{\theta}, \dot{\psi})\). This is summarized by the vector \(X\) of the states and the vector \(U\) of the inputs, as follows:

\[
X = (x, y, z, \phi, \theta, \psi, \dot{x}, \dot{y}, \dot{z}, \dot{\phi}, \dot{\theta}, \dot{\psi}) \tag{2.5}
\]

\[
U = \begin{bmatrix} F_z, M_{\phi}, M_{\theta}, M_{\psi} \end{bmatrix}^T. \tag{2.6}
\]

It is also known that the \(i\) \(-\)th rotor applies a force perpendicular to the plane of rotation of its blade (being aligned with the positive z-axis of the body frame \(F_b\)) and proportional to the square of its rotation speed \(K_F \Omega_i^2\). This force generates a torque around the orthogonal axis that is represented by the opposite arm of the vehicles chassis, being \(l K_F \Omega_i^2\), where \(l\) is the arm length. Meanwhile, the \(i\) \(-\)th rotor produces the torque, due to air drag force, that is opposite to its rotation axis and whose absolute value is again proportional to its rotation speed, that is, \(K_M \Omega_i^2\). Therefore, the overall thrust \(F\) and the components of the moment vector are linearly connected with the squares of rotor speed according to the following relations

\[
F_z = K_F (\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2)
\]

\[
\tau_{\phi} = l K_F (\Omega_3^2 - \Omega_1^2)
\]

\[
\tau_{\theta} = l K_F (\Omega_2^2 - \Omega_3^2)
\]

\[
\tau_{\psi} = K_M (-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2)
\]

or in compact form,

\[
\begin{pmatrix}
F \\
\tau_{\phi} \\
\tau_{\theta} \\
\tau_{\psi}
\end{pmatrix}
= 
\begin{pmatrix}
K_F & K_F & K_F & K_F \\
0 & 0 & -l K_F & l K_F \\
-l K_F & l K_F & 0 & 0 \\
-K_M & -K_M & K_M & K_M
\end{pmatrix}
\begin{pmatrix}
\Omega_1^2 \\
\Omega_2^2 \\
\Omega_3^2 \\
\Omega_4^2
\end{pmatrix}. \tag{2.7}
\]
The input vector is made up of $F_Z$ (thrust along the $z$ direction), and of $\tau_\phi, \tau_\theta, \tau_\psi$ that are the torques for roll, pitch and yaw, respectively. Moreover, apart from these 4 known inputs, the unknown action of the wind along the $x$, $y$ and $z$ directions is to be accounted for as another input vector

$$W = \begin{pmatrix} W_x \\ W_y \\ W_z \end{pmatrix}.$$  \hfill (2.9)

Furthermore, the Euler angles are introduced to describe the orientation in 3-dimensional Euclidean space. The proper representation of the quadrotor is firstly introduced to represent the orientation. Among many possibilities, Z-X-Y Euler angles has been chosen for this thesis. More precisely, in order to align the axes of $F_O$ to those of $F_B$, Euler angles represent a sequence of three element rotations. The rotations start from the elementary rotations. The following combinations are used to achieve the rotation matrix.

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi) & \cos(\phi) \end{bmatrix}$$ \hfill (2.10)

$$R_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix}$$ \hfill (2.11)

$$R_z(\psi) = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$ \hfill (2.12)

The rotation matrix $R(\phi, \theta, \psi)$ converting body-frame coordinates into inertial frame coordinates is obtained by combining the above elementary rotations as follows:
\[ R(\phi, \theta, \psi) = \text{Rot}_Z^T(\psi)\text{Rot}_X^T(\phi)\text{Rot}_Y^T(\theta) \]
\[ = \begin{bmatrix}
\cos(\theta) \cos(\psi) - \sin(\theta) \sin(\psi) & -\cos(\phi) \sin(\psi) & \sin(\theta) \cos(\psi) + \sin(\theta) \cos(\theta) \sin(\psi) \\
\cos(\theta) \cos(\psi) + \sin(\theta) \sin(\psi) & \cos(\phi) \sin(\psi) & \sin(\theta) \cos(\psi) - \sin(\theta) \cos(\theta) \sin(\psi) \\
-\cos(\phi) \sin(\theta) & \sin(\theta) & \cos(\phi) \cos(\theta)
\end{bmatrix} \]

(2.13)

The forces acting on the quadrotor’s center of mass are the gravity (which is always oriented along the negative direction of z-axis of \( F_0 \)), the total thrust \( F \) applied by the four rotors (always aligned with the positive z axis of the \( F_B \)), and the wind for \( W = (W_x, W_y, W_z)^T \) (whose components are assumed to be expressed in \( F_0 \) by convention).

Having denoted \( m \) with the vehicle’s mass, and \( g \) with the gravity acceleration, Newton’s equations for the translational motion of center of mass are:

\[ m \begin{pmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{pmatrix} = -m \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix} + R \begin{pmatrix} 0 \\ 0 \\ F \end{pmatrix} + \begin{pmatrix} W_x \\ W_y \\ W_z \end{pmatrix}, \]

(2.14)

which can be expanded as

\[ \begin{pmatrix} m\ddot{x} \\ m\ddot{y} \\ m\ddot{z} \end{pmatrix} = \begin{pmatrix} (\sin(\theta) \cos(\psi) + \sin(\phi) \cos(\theta) \sin(\psi))F + W_x \\ (\sin(\theta) \cos(\psi) - \sin(\phi) \cos(\theta) \sin(\psi))F + W_y \\ \cos(\phi) \cos(\theta) F - mg + W_z \end{pmatrix}. \]

(2.15)

Moreover, as it is known, the vehicles angular velocity vector \((p, q, r)^T\) in the body frame \( F_B \) can be related to Euler angles through a dynamic relation reading, for the ZXY convention, as

\[ \begin{pmatrix} p \\ q \\ r \end{pmatrix} = \begin{pmatrix} 0 \\ \dot{\theta} \\ 0 \end{pmatrix} + \begin{pmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{pmatrix} \begin{pmatrix} \phi \\ 0 \\ \psi \end{pmatrix} + \\
\begin{pmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & \sin(\phi) \\ 0 & -\sin(\phi) & \cos(\phi) \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ \dot{\psi} \end{pmatrix}. \]

(2.16)
When simplified, equation (2.16) becomes
\[
\begin{pmatrix}
\dot{p} \\
\dot{q} \\
\dot{r}
\end{pmatrix} =
\begin{pmatrix}
\cos(\theta) & 0 & -\cos(\phi)\sin(\theta) \\
0 & 1 & 0 \\
\sin(\theta) & 0 & \cos(\phi)\cos(\theta)
\end{pmatrix}
\begin{pmatrix}
\dot{\phi} \\
\dot{\theta} \\
\dot{\psi}
\end{pmatrix}.
\] (2.17)

Furthermore, due to the lean and trim structure of the quadrotor, it is reasonable to assume that the wind momentum is negligible, thereby, implying that the only vector \( \mathbf{T} = (\tau_\phi \quad \tau_\theta \quad \tau_\psi)^T \) acting on the vehicle itself is composed of the torques applied by the rotors spinning (referring again to Figure 2.2). Since \( F_B \) is aligned with the vehicle’s principle inertia axes, Euler’s equation for the angular motion is
\[
\mathbf{T} = \mathbf{I} \begin{pmatrix}
\dot{p} \\
\dot{q} \\
\dot{r}
\end{pmatrix} + \begin{pmatrix}
0 & -r & q \\
r & 0 & -p \\
-q & p & 0
\end{pmatrix} \mathbf{I} \begin{pmatrix}
p \\
q \\
r
\end{pmatrix},
\] (2.18)

where \( \mathbf{I} \) indicates the inertia matrix around the axes of \( F_B \),
\[
\mathbf{I} = \begin{pmatrix}
I_{xx} & 0 & 0 \\
0 & I_{yy} & 0 \\
0 & 0 & I_{zz}
\end{pmatrix},
\] (2.19)

Direct computation of (2.18) and (2.19) leads to
\[
\begin{pmatrix}
l_{xx}\dot{p} \\
l_{yy}\dot{q} \\
l_{zz}\dot{r}
\end{pmatrix} =
\begin{pmatrix}
\tau_\phi - (l_{zz} - l_{xy})qr \\
\tau_\theta(l_{xx} - l_{xz})pr \\
\tau_\psi(l_{yy} - l_{xz})pq
\end{pmatrix}.
\] (2.20)

Equations (2.15), (2.17) and (2.20) represent one possible nonlinear dynamic model of a quadrotor aircraft.
2.3 State-space Model

Having defined the state vector in the following way:

$$X = [x, y, z, \phi, \theta, \psi, \dot{x}, \dot{y}, \dot{z}, \dot{\phi}, \dot{\theta}, \dot{\psi}]^T \in \mathbb{R}^{12},$$

(2.21)

the nonlinear equations of the quadrotor dynamics in the state-space form reads:

$$\dot{x} = u$$
$$\dot{y} = v$$
$$\dot{z} = w$$
$$\dot{\phi} = \cos(\theta) p + \sin(\theta) \psi$$
$$\dot{\theta} = \tan(\phi) \sin(\theta) p + q - \tan(\phi) \cos(\theta) \psi$$
$$\dot{\psi} = -\frac{\sin(\phi)}{\cos(\phi)} p + \frac{\cos(\phi)}{\cos(\phi)} \phi$$
$$\dot{\psi} = (\sin(\theta) \cos(\psi) + \sin(\phi) \cos(\theta) \sin(\psi)) \frac{F}{m} + \frac{W_x}{m}$$
$$\dot{\psi} = (\sin(\theta) \cos(\psi) - \sin(\phi) \cos(\theta) \sin(\psi)) \frac{F}{m} + \frac{W_y}{m}$$
$$\dot{w} = \cos(\phi) \cos(\theta) \frac{F}{m} - g + \frac{W_Z}{m}$$
$$\dot{p} = -\left(\frac{l_{xx} - l_{yy}}{l_{xy}}\right) qr + \frac{\tau_\phi}{l_{xx}}$$
$$\dot{q} = -\left(\frac{l_{xx} - l_{zz}}{l_{xy}}\right) pr + \frac{\tau_\theta}{l_{yy}}$$
$$\dot{r} = -\left(\frac{l_{yy} - l_{xx}}{l_{zz}}\right) pq + \frac{\tau_\psi}{l_{zz}}$$

(2.22)

Furthermore, the nonlinear equations of the quadrotor can be rewritten as follows

$$\dot{x} = f(x) + g(x, u)$$

(2.23)

$$f(x) = \begin{bmatrix}
    u \\
    v \\
    w \\
    \cos(\theta) p + \sin(\theta) \psi \\
    \tan(\phi) \sin(\theta) p + q - \tan(\phi) \cos(\theta) \psi \\
    -\frac{\sin(\phi)}{\cos(\phi)} p + \frac{\cos(\phi)}{\cos(\phi)} \phi \\
    0 \\
    0 \\
    -\left(\frac{l_{xx} - l_{yy}}{l_{xy}}\right) qr \\
    -\left(\frac{l_{xx} - l_{zz}}{l_{yy}}\right) pr \\
    -\left(\frac{l_{yy} - l_{xx}}{l_{zz}}\right) pq
\end{bmatrix}$$

(2.24)
2.4 Linear Modelling and PD Control

The quadrotor has 4 inputs and 6 outputs, which indicates that it has 2 degrees of under actuation. To make the system fully controllable, the quadrotor is considered to be in hovering state. Hovering is when the quadrotor applies equal thrust to the four rotors and it adjusts yawing by applying more thrust to rotors rotating in one direction. Therefore, at hovering, roll $\phi$ and pitch $\theta$ are almost zero. As a result, the quadrotor has 4 inputs $U = [F_x, \tau_\phi, \tau_\theta, \tau_\psi]^T$ and four outputs $Y = [x, y, z, \psi]$, thus making the quadrotor a square system. Hence, for the linear system, the quadrotor is required to reach a desired position $(x_d, y_d, z_d)$ and desired yaw angle $(\psi_d)$.

2.4.1 Attitude Control

For this thesis, Proportional Derivative (PD) method is used to achieve the linear control of the quadrotor. Firstly, the orientation (attitude) of the quadrotor has been addressed. The dynamic equations for the rotations obtained from (2.17) are as follows:

$$
\begin{bmatrix}
\dot{\phi} \\
\dot{\theta} \\
\dot{\psi}
\end{bmatrix} = 
\begin{bmatrix}
\cos(\theta) & 0 & -\cos(\phi) \sin(\theta) \\
0 & 1 & 0 \\
\sin(\theta) & 0 & \cos(\phi) \cos(\theta)
\end{bmatrix}^{-1}
\begin{bmatrix}
p \\
q \\
r
\end{bmatrix},
$$

(2.25)
whilst Euler equations from (2.18) are as follows:

\[
\begin{pmatrix}
  l_{xx} \dot{p} \\
  l_{yy} \dot{q} \\
  l_{zz} \dot{r}
\end{pmatrix}
= -\begin{pmatrix}
  (l_{zz} - l_{yy}) qr \\
  (l_{xx} - l_{zz}) pr \\
  (l_{yy} - l_{xx}) pq
\end{pmatrix}
+ \begin{pmatrix}
  lK_F (\Omega_3^2 - \Omega_1^2) \\
  lK_F (\Omega_4^2 - \Omega_2^2) \\
  KM (-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2)
\end{pmatrix}.
\] (2.26)

At hovering, \( \phi \approx 0 \) and thus, from equations (2.25) and (2.26), it holds \( \dot{\phi} \approx p, \dot{\theta} \approx p \) and \( \dot{\psi} \approx r \). The attitude dynamics simplifies to

\[
\begin{pmatrix}
  \dot{\phi} \\
  \dot{\theta} \\
  \dot{\psi}
\end{pmatrix}
= -\begin{pmatrix}
  (l_{xx} - l_{yy}) \dot{\phi} \psi \\
  (l_{xx} - l_{zz}) \dot{\phi} \theta \\
  (l_{yy} - l_{xx}) \dot{\phi} \phi
\end{pmatrix}
+ \begin{pmatrix}
  iK_F (\Omega_3^2 - \Omega_1^2) \\
  iK_F (\Omega_4^2 - \Omega_2^2) \\
  KM (-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2)
\end{pmatrix}.
\] (2.27)

An attitude controller can be determined based on an approximate linearization of the above equations around hovering configuration, as mentioned in section 2.1, \( \Omega_i \approx \Omega_o = \frac{mg}{4K_F} \) for all \( i \). Having denoted with \( \delta \psi = \psi - \psi_d \) and \( \delta \Omega = \Omega_i - \Omega_o \), the linearized model of the quadrotor’s attitude motion can be obtained using Taylor method expansion, thus yielding:

\[
\begin{pmatrix}
  \ddot{\phi} \\
  \ddot{\theta} \\
  \ddot{\psi}
\end{pmatrix}
= \begin{pmatrix}
  l_{xx} \frac{m g K_F}{l_{xx}} (\delta \Omega_2 - \delta \Omega_4) \\
  l_{yy} \frac{m g K_F}{l_{yy}} (\delta \Omega_3 - \delta \Omega_1) \\
  l_{zz} \frac{m g K_F}{l_{zz}} (\delta \Omega_1 - \delta \Omega_2 + \delta \Omega_3 - \delta \Omega_4)
\end{pmatrix}.
\] (2.28)

Equation (2.28) is then used for PD tuning of the quadroto’s orientation, where \( \delta \phi \equiv \phi - \phi_c, \delta \theta \equiv \theta - \theta_c \) and \( \phi_c \) and \( \theta_c \) are the commanded roll and pitch. Proportional \( k_p \) and Derivative \( k_v \) constants are to be chosen, for each of the rotational states, based on desired eigenvalue locations. Therefore:
\[
\begin{pmatrix}
(\delta \Omega_2 - \delta \Omega_4) \\
(\delta \Omega_3 - \delta \Omega_1) \\
(\delta \Omega_1 - \delta \Omega_2 + \delta \Omega_3 - \delta \Omega_4)
\end{pmatrix} \equiv \begin{pmatrix}
-k_{v,\phi} \dot{\phi} - k_{p,\phi} \phi \\
-k_{v,\theta} \dot{\theta} - k_{p,\theta} \theta \\
-k_{v,\psi} \dot{\psi} - k_{p,\psi} \psi
\end{pmatrix} \equiv \begin{pmatrix}
\delta \Omega_{\phi} \\
\delta \Omega_{\theta} \\
\delta \Omega_{\psi}
\end{pmatrix},
\]

where
\[
k_{v,\phi} = \frac{l_{xx}k_{v,\phi}}{l/mgK_F}; \quad k_{p,\phi} = \frac{l_{xx}k_{p,\phi}}{l/mgK_F};
\]
\[
k_{v,\theta} = \frac{l_{yy}k_{v,\theta}}{l/mgK_F}; \quad k_{p,\theta} = \frac{l_{yy}k_{p,\theta}}{l/mgK_F};
\]
\[
k_{v,\psi} = \frac{l_{zz}k_{v,\psi}}{l/mgK_F}; \quad k_{p,\psi} = \frac{l_{zz}k_{p,\psi}}{l/mgK_F}.
\]

Moreover, in order to determine the variation of each rotor speed, \(\delta \omega_i\) for all \(i\), a strategy to solve this is to introduce a new quantity being proportional to the variation of the total thrust which will be useful in the derivation of the position controller of the quadrotor:

\[
\delta \Omega_z \equiv \delta \Omega_1 + \delta \Omega_2 + \delta \Omega_3 + \delta \Omega_4.
\] (2.30)

Therefore we have:
\[
\begin{pmatrix}
1 & 1 & 1 & 1 \\
0 & 1 & 0 & -1 \\
-1 & 0 & 1 & 0 \\
1 & -1 & 1 & -1
\end{pmatrix} \begin{pmatrix}
\delta \Omega_1 \\
\delta \Omega_2 \\
\delta \Omega_3 \\
\delta \Omega_4
\end{pmatrix} = \begin{pmatrix}
\delta \Omega_z \\
\delta \Omega_{\phi} \\
\delta \Omega_{\theta} \\
\delta \Omega_{\psi}
\end{pmatrix}.
\] (2.31)

Hence, the vector of rotor speed variations is thus given by:
\[
\begin{pmatrix}
\delta \Omega_1 \\
\delta \Omega_2 \\
\delta \Omega_3 \\
\delta \Omega_4
\end{pmatrix} = \frac{1}{4} \begin{pmatrix}
1 & 0 & -2 & 1 \\
1 & 2 & 0 & -1 \\
1 & 0 & 2 & 0 \\
1 & -2 & 0 & -1
\end{pmatrix} \begin{pmatrix}
\delta \Omega_z \\
\delta \Omega_{\phi} \\
\delta \Omega_{\theta} \\
\delta \Omega_{\psi}
\end{pmatrix}.
\] (2.32)
Putting equations (2.29) and (2.32) together and further simplifying, the feedback attitude controller becomes:

\[
\begin{pmatrix}
\Omega_1 \\
\Omega_2 \\
\Omega_3 \\
\Omega_4
\end{pmatrix} = I_{4\times4} \Omega_0 + \frac{1}{\sqrt{mgK_F}} \begin{pmatrix}
\frac{1}{4} & 0 & \frac{I_{xy}}{2l} & 1 \\
\frac{1}{4} & 0 & \frac{I_{xx}}{2l} & -1 \\
\frac{1}{4} & 0 & \frac{I_{yy}}{2l} & 0 \\
\frac{1}{4} & 0 & \frac{I_{zz}}{2l} & -1
\end{pmatrix} \begin{pmatrix}
2mg + \sqrt{mgK_F} \delta \omega_z \\
-k_v,\phi \dot{\phi} - k_{p,\phi} (\phi - \phi_c) \\
-k_v,\theta \dot{\theta} - k_{p,\theta} (\theta - \theta_c) \\
-k_v,\psi \dot{\psi} - k_{p,\psi} (\psi - \psi_d)
\end{pmatrix}.
\]

(2.33)

2.4.2 Position Control

The position of the center of mass of the quadrotor is controlled by inputs of total thrust of the propellers as well as commanded roll and pitch. Recalling from equation (2.15), the accelerations are

\[
\begin{pmatrix}
\ddot{x} \\
\ddot{y} \\
\ddot{z}
\end{pmatrix} = \begin{pmatrix}
\sin(\theta) \cos(\psi) + \sin(\phi) \cos(\theta) \sin(\psi) & \frac{F}{m} + \frac{W_x}{m} \\
\sin(\theta) \cos(\psi) - \sin(\phi) \cos(\theta) \sin(\psi) & \frac{F}{m} + \frac{W_y}{m} \\
\cos(\phi) \cos(\theta) & \frac{F}{m} - g + \frac{W_z}{m}
\end{pmatrix},
\]

(2.34)

where

\[
f \equiv \frac{K_F}{m} (\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2).
\]

(2.35)

Equation (2.34) reads the x, y and z accelerations and we derive feedback controller based around the hovering condition \((x_d, y_d, z_d, \psi_d)\) and \(f = g\). The position error variables are defined as \(\delta x = x - x_d\), \(\delta y = y - y_d\), \(\delta z = z - z_d\) and \(\delta \psi = \psi - \psi_d\).
The linearized model reads

\[
\begin{pmatrix}
\dot{\chi} \\
\dot{\gamma} \\
\dot{\zeta}
\end{pmatrix} = \begin{pmatrix}
g \begin{pmatrix}
\sin(\psi_d) & \cos(\psi_d)
\end{pmatrix} \\
-g \begin{pmatrix}
\cos(\psi_d) & \sin(\psi_d)
\end{pmatrix}
\end{pmatrix} \begin{pmatrix}
\delta \phi \\
\delta \theta
\end{pmatrix},
\]

(2.36)

where \( \delta f \) can be expressed in rotor speed \( \delta \Omega_i \) as follows

\[
\delta f \approx \frac{8K_F \omega_o}{m} (\delta \Omega_1 + \delta \Omega_2 + \delta \Omega_3 + \delta \Omega_4) = \frac{8K_F \omega_o}{m} \delta \Omega_z = 2 \sqrt{\frac{K_F \Omega}{m}} \delta \Omega_z,
\]

(2.37)

where \( \delta \omega_z \) is defined in equation 2.31. In order to stabilize the quadrotor center of mass, the above model is then linearized using first order truncation of Taylor’s expansion.

\[
\begin{pmatrix}
\dot{\chi} \\
\dot{\gamma} \\
\dot{\zeta}
\end{pmatrix} = \begin{pmatrix}
-k_{v_x} \dot{x} - k_{p_x} x \\
-k_{v_y} \dot{y} - k_{p_y} y \\
-k_{v_z} \dot{z} - k_{p_z} z
\end{pmatrix} = \begin{pmatrix}
g \begin{pmatrix}
\sin(\psi_d) & \cos(\psi_d)
\end{pmatrix} \\
-g \begin{pmatrix}
\cos(\psi_d) & \sin(\psi_d)
\end{pmatrix}
\end{pmatrix} \begin{pmatrix}
\delta \phi \\
\delta \theta
\end{pmatrix}.
\]

(2.39)

Finally, the position controller reads:

\[
\begin{pmatrix}
\phi_c \\
\theta_c \\
\hat{f}
\end{pmatrix} = \begin{pmatrix}
\frac{1}{g} \begin{pmatrix}
\sin(\psi_d) & \cos(\psi_d)
\end{pmatrix}(-k_{v_x} \dot{x} - k_{p_x} (x - x_d - W_x)) \\
-g \begin{pmatrix}
\cos(\psi_d) & \sin(\psi_d)
\end{pmatrix}(-k_{v_y} \dot{y} - k_{p_y} (y - y_d - W_y)) \\
\frac{8}{m} (\delta \omega_1 + \delta \omega_2 + \delta \omega_3 + \delta \omega_4) + 2 \sqrt{\frac{K_F \Omega}{m}} (-k_{v_z} \dot{z} - k_{p_z} (z - z_d - W_y))
\end{pmatrix}.
\]

(2.40)

Therefore, the linear model of the controlled quadrotor aircraft can be summarized as follows by the following equations:
\[
\begin{align*}
\delta \ddot{x} &= g \sin(\psi_d) \, \delta \phi + g \cos(\psi_d) \, \delta \theta \\
\delta \ddot{y} &= -g \cos(\psi_d) \, \delta \phi + g \sin(\psi_d) \, \delta \theta \\
\delta \ddot{z} &= 2 \sqrt{\frac{Kg}{m}} \left( \delta \Omega_1 + \delta \Omega_2 + \delta \Omega_3 + \delta \Omega_4 \right) \\
\delta \ddot{\phi} &= \frac{21K_p\omega_0}{l_{xx}} (\delta \omega \Omega_2 - \delta \Omega_4) \\
\delta \ddot{\theta} &= \frac{21K_p\omega_0}{l_{yy}} (\delta \Omega_1 - \delta \Omega_3) \\
\delta \ddot{\psi} &= \frac{21K_p\omega_0}{l_{xx}} (\delta \Omega_1 - \delta \Omega_2 + \delta \Omega_3 - \delta \Omega_4)
\end{align*}
\] (2.41)

where:
\[
\begin{align*}
\delta \omega_x &= -k_{v,x} \dot{x} - k_{p,x} (x - x_d) \\
\delta \omega_y &= -k_{v,y} \dot{y} - k_{p,y} (y - y_d) \\
\delta \omega_z &= (-k_{v,z} \dot{z} - k_{p,z} (z - z_d)) \\
\delta \omega_\phi &= -k_{v,\phi} \dot{\phi} - k_{p,\phi} (\phi - \phi_c) \\
\delta \omega_\theta &= -k_{v,\theta} \dot{\theta} - k_{p,\theta} (\theta - \theta_c) \\
\delta \omega_\psi &= -k_{v,\psi} \dot{\psi} - k_{p,\psi} (\psi - \psi_d)
\end{align*}
\]

The corresponding state space representation of such a linear model is:
\[
\begin{align*}
\dot{x} &= Ax + Bu + B_dW \\ 
\gamma &= Cx
\end{align*}
\] (2.42) (2.43)

where
\[
X = \begin{bmatrix} x, y, z, \phi, \theta, \psi, \dot{x}, \dot{y}, \dot{z}, \dot{\phi}, \dot{\theta}, \dot{\psi} \end{bmatrix}^T \in \mathbb{R}^{12}
\]
\[
U = [\Omega_1, \Omega_2, \Omega_3, \Omega_4] \in \mathbb{R}^{4}
\]
\[
W = [W_x, W_y, W_z] \in \mathbb{R}^{3}
\]

26
\[
A_{d(12\times12)} = \begin{bmatrix}
I_{3\times3} & I_{3\times3} & 0_{3\times3} \\
0_{3\times3} & I_{3\times3} & 0_{3\times3} \\
0_{3\times3} & 0_{3\times3} & I_{3\times3}
\end{bmatrix}
\begin{bmatrix}
g \sin(\psi_d) \\
-g \cos(\psi_d) \\
0
\end{bmatrix}
\begin{bmatrix}
g \cos(\psi_d) \\
g \sin(\psi_d) \\
0
\end{bmatrix}
\begin{bmatrix}
I_{3\times3} & 0_{3\times3} \\
0_{3\times3} & I_{3\times3} \\
0_{3\times3} & 0_{3\times3}
\end{bmatrix}
\]

(2.44)

\[A_d\] is discrete conversion of \(A\)

\[
B_{u12\times4} = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
\sqrt{K_Fm} & \sqrt{K_Fm} & \sqrt{K_Fm} & \sqrt{K_Fm} \\
0 & \frac{IK_F}{l_y} & 0 & \frac{IK_F}{l_x} \\
0 & 0 & \frac{IK_F}{l_y} & \frac{IK_F}{l_x} \\
0 & 0 & 0 & \frac{IK_F}{l_y} \\
\frac{mg/K_F}{l_{zz}} & \frac{mg/K_F}{l_{zz}} & \frac{mg/K_F}{l_{zz}} & \frac{mg/K_F}{l_{zz}}
\end{bmatrix}
\]

(2.45)

\[
B_{D12\times3} = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
1/m & 0 & 0 \\
0 & 0 & 1/m \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 
\end{bmatrix}
\]

(2.46)

\[
C = \begin{bmatrix}
I_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\
0_{3\times3} & I_{3\times3} & 0_{3\times3} & 0_{3\times3} \\
0_{3\times3} & 0_{3\times3} & I_{3\times3} & 0_{3\times3} \\
0_{3\times3} & 0_{3\times3} & 0_{3\times3} & I_{3\times3}
\end{bmatrix}
\]

(2.47)
Figure 2.4 illustrates a Matlab Simulink implementation of such a linear model, while Figure 2.5-2.8 report typical scenarios with and without wind gusts and shows the dynamic behavior of the controlled system. As expected, in the case where wind gusts are active, the desired position is not correctly attained. Wind estimation and further control actions have to be done in order to deal with such an issue, which will be discussed in the next chapter.

Figure 2.4 The MATLAB Simulink Model PD control of the Linearized Model of the Quadrotor

The linear model with PD control was tested in MATLAB Simulink with and without the wind gusts. Figures 2.5 and 2.6 shows the results without wind and Figures 2.7-2.9 show the results with wind gusts.
Figure 2.5 Linear Position of the quadrotor with target position of \((x = 1m, y = 2m \text{ and } z = 3m)\) with no wind gusts.

Figure 2.6 Angular Position of the quadrotor with target position of \((\phi = \frac{\pi}{2}, \theta = 0, \psi = 0)\) with no wind gusts.

The Figures (2.7) - (2.11) shows the results with wind gusts disturbing the positioning of the quadrotor. For Figure 2.7 the target position is \(x = 1m, y = 2m \text{ and } z = 3m\) however the quadrotor goes to the position \(x = 3m, y = 4m \text{ and } z = 5m\) with the wind gusts of \(Wx = 2N, Wy = 2N \text{ and } Wz = 2N\) acting along \(x, y\) and \(z\) axis respectively. While Figure 2.8 shows that there was no effect on angular positioning of the quadrotor with presence of the wind gusts.
Figure 2.7 Linear Position of the quadrotor with target position of \((x = 1m, y = 2m and z = 3m)\) with wind gusts \([W_x = 2N, W_y = 2N and W_z = 2N]\) as step input.

Figure 2.8 Angular Position of the quadrotor with target position of \((\phi = \frac{\pi}{2}, \theta = 0, \psi = 0)\) with wind gusts \([W_x = 2m/s, W_y = 2m/s and W_z = 2m/s]\).
The simulation was further tested with time varying type wind gusts model as shown in Figure (2.10) and (2.11). The target position is \( x = 1m, y = 2m \) and \( z = 3m \) however the quadrotor does not reach its target and waves with the time varying wind gusts (as shown in Figure (2.9)). While Figure 2.8 shows that there was no effect on angular positioning of the quadrotor with presence of the wind gusts.

Figure: 2.9 shows the time varying wind gusts type

Figure 2.10 Linear Position of the quadrotor with target position of \((x = 1m, y = 2m \text{ and } z = 3m)\) with time varying wind gusts as shown in Figure 2.9.
Figure 2.11 Angular Position of the quadrotor with target position of $(\phi = \frac{\pi}{2}, \theta = 0, \psi = 0)$ with time varying wind gusts as shown in Figure 2.9.
Chapter 3

Unknown Input Observers for Linear Dynamic System with Partially Unknown Inputs

This chapter discusses the theory and the application of the Unknown Input Observers (UIO) to estimate wind gusts for the quadrotor aircraft. The dynamic system with known and unknown inputs can be modelled as discussed in Chapter 2. The problem of constructing observers for such systems has received considerable attention in the past few decades. In this chapter, we will discuss the procedure for constructing UIOs for discrete linear time-invariant systems.

3.1 Linear Unknown Input Observers

The theory of linear UIOs presented here builds upon the work in [45]. A class of discrete time linear systems, which is affected by unknown inputs is described by the equations

\[ x[k + 1] = Ax[k] + B_u U[k] + B_d W[k], \]
\[ y[k] = Cx[k] + D_u U[k] + D_d W[k], \]

(3.1)

where \( x[k] \in \mathbb{R}^n \) is a state vector, \( U[k] \in \mathbb{R}^m \) is the known input signal, \( W[k] \in \mathbb{R}^q \) is unknown input disturbance, \( y[k] \in \mathbb{R}^p \) is the output vector and \( A, B_u, B_d, C, D_u \) and \( D_d \) are matrices of appropriate dimensions. A common assumption, which is also done here, is that matrix \( \begin{bmatrix} B_d \\ D_d \end{bmatrix} \) be full rank, which ensures that the system can be inverted and the unknown inputs can be reconstructed.

Furthermore, given a positive time delay \( L \) and having defined the estimated system state \( \hat{x} \), a UIO filter is a dynamic system be designed in such a way that

\[ \hat{x}[k] - x[k] \to 0 \text{ as } k \to \infty \text{ regardless the value of } U[k] \text{ and } W[k]. \]

(3.2)
More precisely, consider a discrete time linear observer of the form
\[
\dot{x}[k + L] = E\dot{x}[k] + Fy[k: k + L].
\] (3.3)

where \( E \) is a Schur dynamic matrix and \( F \) is a suitable output injection matrix to be found. Recall that a matrix \( E \) is said to be Schur if all its eigenvalues are inside the unit circle, i.e., \( max|\lambda_i(E)| < 1 \). First recall that the observability matrix \( O^L \) for the pair \((A, C)\) can be written as
\[
O^L = \begin{bmatrix}
    C \\
    CA \\
    CA^2 \\
    \vdots \\
    CA^{L-1}
\end{bmatrix} = \begin{bmatrix}
    C \\
    O^{L-1}A
\end{bmatrix},
\] (3.4)

and its invertibility matrix \( J^L \) can be described by
\[
J^L = \begin{bmatrix}
    D & 0 & 0 & 0 & \ldots & 0 \\
    CB & D & 0 & 0 & \ldots & 0 \\
    CAB & CB & D & 0 & \ldots & 0 \\
    \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
    CA^{L-1}B & CA^{L-2}B & \ldots & \ldots & \ldots & D
\end{bmatrix} = \begin{bmatrix}
    J_U^L & J_W^L
\end{bmatrix},
\] (3.5)

where \( B = [B_uB_d] \), \( J_U^L = \) invertibility matrix of \((A, B_u, C, D_u)\), and \( J_W^L = \) invertibility matrix of \((A, B_d, C, D_d)\).

The existence and construction of the UIO depend on the system matrices and will be proved in this section. It relates to the concepts of L-step observability and invertibility of a dynamic system. To show this, the output equation is repeated to \( L + 1 \) time-steps as follows
\[
\begin{bmatrix}
    y[0] \\
    y[1] \\
    y[2] \\
    \vdots \\
    y[L]
\end{bmatrix} = \begin{bmatrix}
    C \\
    CA \\
    CA^2 \\
    \vdots \\
    CA^{L-1}
\end{bmatrix} x[0] + \begin{bmatrix}
    D & 0 & 0 & 0 & \ldots & 0 \\
    CB & D & 0 & 0 & \ldots & 0 \\
    CAB & CB & D & 0 & \ldots & 0 \\
    \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
    CA^{L-1}B & CA^{L-2}B & \ldots & \ldots & \ldots & D
\end{bmatrix} \begin{bmatrix}
    u[0] \\
    u[1] \\
    u[2] \\
    \vdots \\
    u[L]
\end{bmatrix}
\] (3.6)

where \( O^L \) is the observability matrix and \( J^L \) is the invertibility matrix of the system.
Therefore, the response of the system over \( L + 1 \) consecutive time-steps is

\[
y[k; k + 1] = O^L x[k] + J^L u[k; k + 1]
\]  

(3.7)

Consider now, from equations (3.1), (3.2) and (3.3), the dynamics of the estimation error \( e[k] = \hat{x}[k] - x[k] \). Given the observer matrices \( E \) and \( F \), the dynamics of such error reads

\[
e[k + 1] = \hat{x}[k + 1] - x[k + 1] \\
e[k + 1] = E \hat{x}[k] + F y[k; k + L] - Ax[k] - B_u U[k] - B_d W[k] \\
e[k + 1] = E e[k] + F y[k; k + L] + (E - A) x[k] - B_u U[k] - B_d W[k]
\]

(3.8)

where \( E \) \( x[k] \) has been added and subtracted in the last line of (3.8). Using equation (3.7) and (3.8), the error can be further expressed as

\[
e[k + 1] = E e[k] + F y[k; k + L] + (E - A) x[k] - B_u U[k] - B_d W[k] \\
e[k + 1] = E e[k] + (E - A + F O^L) x[k] + F J^L U[k; k + L] - B_u U[k] - B_d W[k].
\]

(3.9)

Upon decoupling the error dynamics from system inputs, states and outputs, the dynamic equation obtained is as

\[
e[k + 1] = E e[k],
\]

(3.10)

where \( E \) has to have convergent eigenvalues. In order to enforce such a dynamic for the estimation error, it then must occur that

\[
(E - A + F O^L) x[k] + F J^L U[k; k + L] - B_u U[k] - B_d W[k] = 0,
\]

(3.11)

which is satisfied, regardless of the state value \( x[k] \), if the matrices \( E \) and \( F \) simultaneously meet the following conditions:

\[
F J^L = [B, \ 0, \ \cdots , \ 0]
\]

(3.12)

\[
E = A - F O^L
\]

(3.13)
Furthermore, let the term $J^L U[k:k + L]$ be expressed as a function of $x[k]$ and of the history $y[k]$. To this purpose, the output $y[k]$, $h$ steps in the future, can be iteratively determined as

\[
y[k + h] = CA^h X[k]
\]

\[
+ \sum_{i=0}^{h-1} CA^i B_u U[k + i] + D_u U[k + h] + \sum_{i=0}^{h-1} CA^i B_d d[k + i]
\]

\[
+ D_n d[k + h]
\]

(3.14)

Evaluating the above expression for every $h = 0, \ldots, L$, and rearranging the obtained equations in matrix form yields the following relation:

\[
\]

(3.15)

As evident, equations (3.8) and (3.9) hold for every $x[k]$, $U(k)$ and $W[k]$. By rearranging (3.7), the above expression can be equated to

\[
y[k:L] - O^L x[k] = J^L U[k:L],
\]

where the left side of the equation only contains completely known terms. It is apparent that matrix $J^L$ fully characterizes the ability to recover the inputs of the system. Moreover, it must also satisfy the following theorem:

**Theorem:** With similar reasoning as in (3.7) and under the assumption that $\begin{bmatrix} B_d \\ D_d \end{bmatrix}$ is full column rank, equation (3.12) is solvable if and only if

\[
rank(J^L) = rank(J^{L-1}) = m.
\]

(3.17)

When designing the UIO as described in [Sundaram, 45], the value of the delay $L$ starts is iteratively determined, starting from $L = 0$, and is increased until (3.17) is satisfied. Recall that it also holds $L \leq n$. If these conditions are not satisfied for $L=n$, such an asymptotic estimator of the states is not possible.

To begin with, consider the output of the linear system over $L + 1$ time steps for any positive integer $L$. Moreover, from (3.12), the matrix $F$ must be in left nullspace of the
last \(L_q\) columns of \(J^L\left(\begin{bmatrix} 0 \\ g_{L-1} \end{bmatrix}\right)\). Specifically, let \(\tilde{N}\) be a matrix whose rows form a basis for left nullspace of \(g^{L-1}\), so that \(\begin{bmatrix} I_p & 0 \\ 0 & \tilde{N} \end{bmatrix}\) is a matrix whose rows form a basis for the left nullspace of \(\begin{bmatrix} 0 \\ g_{L-1} \end{bmatrix}\). To further define that,

\[
N = W \begin{bmatrix} I_p & 0 \\ 0 & \tilde{N} \end{bmatrix}
\tag{3.18}
\]

where the rows of \(N\) also form a basis for the left nullspace of \(\begin{bmatrix} 0 \\ g_{L-1} \end{bmatrix}\). Therefore,

\[
N \begin{bmatrix} D_{O^{L-1}A} & 0 \\ g_{L-1} \end{bmatrix} = W \begin{bmatrix} D_{\tilde{N}O^{L-1}} & 0 \\ 0 \end{bmatrix}
\tag{3.19}
\]

Equation (3.17) expresses that the first \(q\) columns of \(J^L\) must be linearly independent of each other and of the remaining \(L_q\) columns. Thus, it can be deduced that the matrix \(\begin{bmatrix} D_{\tilde{N}O^{L-1}} & 0 \\ 0 \end{bmatrix}\) has rank \(q\). Matrix \(W\) is chosen so that the bottom \(q\) rows form a left inverse for the above matrix and the remaining top rows form a basis for the left nullspace of matrix. Hence, \(N\) satisfies

\[
Nj^L = \begin{bmatrix} 0 \\ I_q \\ 0 \end{bmatrix}
\tag{3.20}
\]

Also, from (3.11), \(F\) can be written in the form \(F = \hat{F}N\) for some \(\hat{F} = [\hat{F}_1 \quad \hat{F}_2]\), where \(\hat{F}_2\) has \(q\) columns. As a result, equation (3.12) now becomes

\[
[\hat{F}_1 \quad \hat{F}_2] \begin{bmatrix} 0 \\ I_m \\ 0 \end{bmatrix} = [B \quad 0],
\tag{3.21}
\]

from which it is obvious that \(\hat{F}_2 = B\) and \(\hat{F}_1\) is free matrix. Considering equation (3.12),

\[E = A - FO^L = A - [\hat{F}_1 \quad B]NO^L,\]

and defining
\[
\begin{bmatrix}
S_1 \\
S_2
\end{bmatrix} \equiv NO^L, \quad (3.22)
\]

where \(S_2\) has \(q\) rows. \(\hat{F}_1\) is found such that \(E = (A - BS_2) - \hat{F}_1 S_1\) and is Schur. Since it is required for \(E\) to be a stable matrix, the pair \((A - BS_2, S_1)\) must be detectable.

Finally, by assuming that the estimated state as converged to the actual one, i.e. \(\hat{x}[k] - x[k] \to 0\) as \(k \to \infty\), the reconstruction of the unknown inputs can be easily obtained by rearranging (3.1) as

\[
\begin{bmatrix}
x[k+1] - Ax[k] - Bu[k] \\
y[k] - Cx[k]
\end{bmatrix} = \begin{bmatrix} B_d \\ D_d \end{bmatrix} W[k]. \quad (3.23)
\]

The sought estimated unknown inputs can be extracted as

\[
W[k] = G \begin{bmatrix}
x[k+1] - Ax[k] - Bu[k] \\
y[k] - Cx[k]
\end{bmatrix}, \quad (3.24)
\]

where \(G = \begin{bmatrix} B_d \\ D_d \end{bmatrix}^{-1}\) or \(G \begin{bmatrix} B_d \\ D_d \end{bmatrix} = I_m\). The found UIO can be graphically depicted as in Figure 3.1.

![Figure 3.1 Graphic representation of UIO Scheme](image-url)
3.2 Design of Linear Unknown Input Observers for Quadrotor Model

Recalling from Chapter 2, the linear model of the quadrotor system is firstly discretized with sample time $T_s$ to the following matrices

$$A_d(12 \times 12) = \begin{bmatrix}
I_{3 \times 3} & I_{3 \times 3} & 0_{3 \times 3} & 0_{3 \times 3} \\
0_{3 \times 3} & I_{3 \times 3} & 0_{3 \times 3} & 0_{3 \times 3} \\
0_{3 \times 3} & -g \cos(\psi_d) & g \cos(\psi_d) & 0 \\
0_{3 \times 3} & g \sin(\psi_d) & g \sin(\psi_d) & 0 \\
0_{3 \times 3} & 0_{3 \times 3} & 0_{3 \times 3} & I_{3 \times 3}
\end{bmatrix}$$  (3.25)

$$B_{u12 \times 4} = T_s \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
\sqrt{K_F mg} & \sqrt{K_F mg} & \sqrt{K_F mg} & \sqrt{K_F mg} \\
0 & -lK_F/l_{yy} & 0 & 0 \\
lK_F/l_{yy} & 0 & -lK_F/l_{xx} & 0 \\
-lmg/K_F/l_{xx} & lmg/K_F/l_{xx} & 0 & 0 \\
lmg/K_F/l_{xx} & -lmg/K_F/l_{xx} & 0 & 0
\end{bmatrix}$$  (3.26)

$$B_{D12 \times 3} = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
1/m & 0 & 0 \\
0 & 1/m & 0 \\
0 & 0 & 1/m \\
0 & 0 & 0 \\
0 & 0 & 0
\end{bmatrix}$$  (3.27)
\[ C = \begin{bmatrix} I_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & I_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & I_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & I_{3\times3} \end{bmatrix} \]  \hfill (3.28)

\[ D_U = [0_{6\times4}] \text{and} \ D_D = [0_{6\times3}] \]  \hfill (3.29)

Firstly, by using equation (3.17) the smallest possible delay \( L \) for which the system is invertible is found. In the considered context, the wind gust force is described by its components, \([W_x \ W_y \ W_z]^T\) along the \( x, y \) and \( z \) directions; therefore, the number of unknown inputs is \( q = 3 \). Direct computation shows that the smallest delay by which the condition \( \text{rank}(J^L) - \text{rank}(J^{L-1}) = 3 \) holds is \( L = 2 \), and thus the UIO for the quadrotor system is delayed by \( 2T_s \) seconds. In order to find matrix \( E \) and \( F \), it is essential to determine matrix \( \tilde{N} \) whose row forms a basis of the left nullspace of \( J^1 \) which is given by

\[
\tilde{N} = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix} \]  \hfill (3.30)

and thus matrix \( N \) has the form (\( p=6 \)),

\[ N = W \begin{bmatrix} I_p & 0 \\ 0 & \tilde{N} \end{bmatrix} \]  \hfill (3.31)

where \( W \) has to satisfy the condition

\[
\begin{bmatrix} 0 \\ I_3 \end{bmatrix} = N \begin{bmatrix} D_D & 0 \\ C B_D & C A B_D \end{bmatrix} = W \begin{bmatrix} I_p & 0 \\ 0 & \tilde{N} \end{bmatrix} \begin{bmatrix} D_D & 0 \\ C B_D & C A B_D \end{bmatrix} \]  \hfill (3.32)
A choice for matrix $W$ such that its last $q = 3$ rows are the left inverse of the matrix $H$ and the remainder first $2n - q = 21$ rows form a basis for the left nullspace of the matrix $H$, is:

$$
W = \begin{bmatrix}
I_{12 \times 12} & 0_{3 \times 3} & 0_{3 \times 3} \\
0_{3 \times 3} & 0_{3 \times 3} & I_{3 \times 3} \\
0_{3 \times 3} & m/T_{3 \times 3} & 0_{3 \times 3}
\end{bmatrix}
$$

(3.33)

From equation (3.33)

$$
N = \begin{bmatrix}
I_{12 \times 12} & 0_{3 \times 3} & 0_{3 \times 3} \\
0_{3 \times 3} & 0_{3 \times 3} & I_{3 \times 3} \\
0_{3 \times 3} & m/T_{3 \times 3} & 0_{3 \times 3}
\end{bmatrix}\begin{bmatrix}
I_p \\
0 \\
0 \\
\end{bmatrix}
$$

(3.34)

As a next step the following decomposition is done

$$
\begin{bmatrix} S_1 \\ S_2 \end{bmatrix} = N \begin{bmatrix} C \\ CA_d \\ CA_d^2 \\ \sigma^2 \end{bmatrix}
$$

(3.35)

where $S_2$ is the last $q=3$ rows of the above matrix and $S_1$ is the remainder ones. Finally, $\hat{P}_1$ has to be determined such that $E = A - BS_2 - \hat{P}_1S_1$ is Schur. Since the quadrotor’s eigenvalues are unstable, it is possible to find $\hat{P}_1$ by e.g. the MATLAB command place($(A - BS_2)',S_1,p)'$ in order to allocate all eigenvalues at the locations specified by a vector $p$. Finally, it holds

$$
E = A - BS_2 - \hat{P}_1S_1
$$

(3.36)

$$
F = [\hat{P}_1 \ B]N
$$

(3.37)

and the UIO is given by

$$
x[k + 1] = E\hat{x}[k] + F\hat{y}[k : k + 2]
$$

(3.38)
Once the state estimation error $e_k$ has converged, that is the estimated state $\hat{x}$ coincides with the actual state $x$, the unknown input can be retrieved from the system is modelled as

$$\hat{x}[k + 1] - A\hat{x}[k] - B_u U[k] = B_w W[k]$$
$$y[k] - \hat{y}[k] = y[k] - C\hat{x}[k] = D_w W[k]$$

Hence, the estimates of the inputs of the system now becomes

$$W[k] = G \begin{bmatrix} \hat{x}[k + 1] - A\hat{x}[k] - B_u U[k] \\ y[k] - C\hat{x}[k] \end{bmatrix}$$

and $G = \begin{bmatrix} B_d \\ D_d \end{bmatrix}^{-1}$

A possible Simulink scheme implementing the proposed UIO is described below:

![Figure 3.2 The MATLAB Simulink Model of UIO](image)

Furthermore, the Linear UIO discussed in this chapter is tested with three different scenarios for the quadrotor model with PD controller. The unknown input wind gust observed using MATLAB Simulink designed is shown in Figure 3.2. The two scenarios discussed next show the results of estimation with different wind gusts models.

**Scenario 1: Step Input Wind Gusts**

Figure 3.3 illustrates the results of the Linear UIO with the quadrotor model of the wind gusts as step input. Here, $Wx = 3N$, $Wy = 2N$ and $Wz = 1N$. The solid line
shows the estimated wind gusts while the dotted line shows the actual wind gust. Figure 3.4 shows the error between the estimated and actual wind gusts.

Figure 3.3 Estimated versus actual wind gusts for step input

![Wind Gusts Graph](image)

Figure 3.4 The error graph between estimated and actual wind gusts which is shown in Figure 3.3

*Scenario 2 Military Grade Wind Gusts*

Figure (3.5) illustrates the results of the Linear UIO with the quadrotor model military grade wind gusts model. This wind gust model is more realistic and is usually used to assess airplane responses [56]. The wind gusts are: $W_x = 3N$, $W_y = 2N$ and $W_z = 1N$. The continuous line shows the estimated wind gusts, while the dotted line shows the actual wind gust. Figure 3.6 shows the error between the estimated and actual wind gusts.
Scenario 3: Time varying Wind Gusts

Figure (3.7) illustrates the results of the Linear UIO with the time varying wind gusts model. The wind gusts have the amplitude of: $W_x = 3N$, $W_y = 2N$ and $W_z = 1N$. The continuous line shows the estimated wind gusts, while the dotted line shows the actual wind gust. Figure 3.7 shows the error between the estimated and actual wind gusts.
Figure 3.7 Estimated versus actual wind gusts for military grade wind model

Figure 3.8 The error graph between estimated and actual wind gusts which is shown in Figure 3.7
Chapter 4

Results of the Controlled Aircraft with Linear Unknown Input Wind Observer

This chapter shows the results obtained with the Linear UIO filter discussed in Chapter 3 and the proposed PD control, which is based on the linear model of the quadrotor from Chapter 2, Section 4. The results have been obtained via different validation schemes of increasing complexity and reality: Matlab/Simulink, ROS®-Gazebo® Software-In-The-Loop, and ROS®-based experimental setup. The behaviors of the controlled quadrotor and of the UIO are tested with different types of wind gusts. For each of the results, the desired target outputs are \( r = (x_d, y_d, z_d, \psi_d) \). The general scheme of the entire model is shown in Figure 4.1.

![Figure 4.1 General Scheme of the Model](image)

Using the model of the quadrotor as explained in Chapter 2, the PD controller is divided into two parts: an Attitude Control for orientation in \( \phi, \theta, \psi \) and a Position Control for positioning the quadrotor along \( x, y \) and \( z \). The position controller steers the aircraft position with the information of the estimated wind gusts \( [W_x, W_y, W_z]^T \). As discussed earlier, the quadrotor is in hovering state which implies that \( \phi \approx 0 \) and \( \theta \approx 0 \). In addition, the position controller also focuses on maintaining the orientation for hovering, hence feeding the commanded \( \phi_c \) and \( \theta_c \) to the attitude controller. Together with \( \phi_c \) and \( \theta_c \), desired yaw \( (\psi_d) \) is derived and fed to the
attitude controller. Using all of this information, the rotor velocities of the quadrotor is computed by using equations (2.31)-(2.33) which are the four known inputs to the system. The quadrotor has 12 states but only 6 of them are measurable in experimentation. These outputs are \([x, y, z, \phi, \theta, \psi]\). These outputs are used by the UIO to estimate the 12 states and the unknown wind gusts input. All of this information is then fed back to the PD controller, thus making it a closed loop.

An Erlecopter prototype aircraft is used to carry out the simulation and experimental results. The quadrotor parameters are shown in Table 4.1

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Quadrotor Parameters</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(m)</td>
<td>Mass</td>
<td>1.12m</td>
<td></td>
</tr>
<tr>
<td>(g)</td>
<td>Gravitational Acceleration</td>
<td>9.81m/s²</td>
<td></td>
</tr>
<tr>
<td>(l_{xx})</td>
<td>Inertia about x axis</td>
<td>(34.8 \times 10^{-3} kg/m²)</td>
<td></td>
</tr>
<tr>
<td>(l_{yy})</td>
<td>Inertia about y axis</td>
<td>(45.9 \times 10^{-3} kg/m²)</td>
<td></td>
</tr>
<tr>
<td>(l_{zz})</td>
<td>Inertia about z axis</td>
<td>(97.7 \times 10^{-3} kg/m²)</td>
<td></td>
</tr>
<tr>
<td>(K_F)</td>
<td>Force Constant</td>
<td>(8.55 \times 10^6 Ns²/m²)</td>
<td></td>
</tr>
<tr>
<td>(K_M)</td>
<td>Moment Constant</td>
<td>(0.015 Ns²/m²)</td>
<td></td>
</tr>
<tr>
<td>(l)</td>
<td>Length</td>
<td>0.141m</td>
<td></td>
</tr>
</tbody>
</table>

Euler discretization time \(Ts = 0.01s\) has been used for the implementation of the UIO filter.

4.1 MATLAB Simulink Results

4.1.1 Military Grade Wind Gusts

A more realistic wind gust model is firstly considered, implementing the Military Specification MIL-F-8785C [56], which is commonly used to assess airplane response to large wind disturbances. The MATLAB Simulink simulation results has been obtained for 4 different scenarios, including a Military Grade wind input. The considered military-grade wind gust has a length of \(10 m\) and amplitude \(W_x = 2.5N\), \(W_y = 1.5N\) and \(W_z = 0.5N\). All the starting configuration have \(X(0) = 0_{12 \times 1}\). Following figures summarize the results for all considered scenarios.
**Scenario 1:** For the results presented in Figure 4.2, the quadrotor is required to move to \( x_d=0m, y_d=0m \) and \( z_d=5m \) in the presence of the military grade wind gust. The control parameters are \( K_p = 0.4 \) and \( K_D = 4 \). The quadrotor arrives to the target point and steady state in 3s. The \( \psi \) also stabilizes to zero at 1.5 while \( \phi_C \) and \( \theta_C \) are -0.14 rad and -0.22rad respectively. Figure 4.2(c) illustrates the speeds of the four rotors move the quadrotor to the desired position and the quadrotor eventually maintains its speed at \( \omega_0 \). Figure 4.2(e) shows that the linear UIO correctly estimates the wind gusts with initial small error between 0-1s. The dotted line shows the reference and the solid line shows the estimated wind. The state estimation errors results converges to zero after 10s.
Figure 4.2 Results of military grade wind gusts for scenario 1 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory

**Scenario 2:** For the results presented in Figure 4.3, the quadrotor is required to move in a straight line from the origin to the point of $x_d=1m$, $y_d=2m$ and $z_d=3m$ and $\psi_d = \frac{\pi}{2}$ in the presence of military grade wind gust. The control parameters are $K_p = 0.4$ and $K_D = 4$. The quadrotor reaches the target point in 20s. The $\psi$ also stabilizes to zero at 1.5s while $\phi_c$ and $\theta_c$ are -0.14 rad and -0.22 rad respectively. Figure 4.3(c) illustrates the speeds of the four rotors move the quadrotor to the desired position and the quadrotor eventually maintains its speed at $w_O$. Figure 4.3(e) shows that the linear UIO correctly estimates the wind gusts with initial small error between 0-1s. The dotted line shows the reference and the solid line shows the estimated wind. The state estimation errors are close to zero and few errors are seen when the orientation of the quadrotor reaches the steady state.
Scenario 3: For the results presented in Figure 4.4, the quadrotor is required to move in a circular trajectory of radius 10m and at a height $z_d = 5m$ in the presence of military grade wind gust. The control parameters are changed to $k_p = 0.9$ and $k_d = 9$ so that the quadrotor can reach the desired path more accurately and smoothly in 10s. The $\psi$ also stabilizes to zero at 3s while $\phi_c$ and $\theta_c$ oscillates with amplitude of 0.1 and 0.2 respectively while moving the quadrotor in circular path. Figure 4.4(c) illustrates the speeds of the four rotors move the quadrotor to the desired height and the quadrotor eventually maintains its speed at $w_d$. Figure 4.4(e) shows that the linear
UIO correctly estimates the wind gusts with small error initially. The dotted line shows the reference and the solid line shows the estimated wind. The state estimation errors are close to zero but few errors are seen when the quadrotor rotates in circle because of the oscillations of $\phi_c$ and $\theta_c$.

Figure 4.4 Results of military grade wind gusts for scenario 3 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory.
Scenario 4: For the results presented in Figure 4.5, the quadrotor is required to move at a height \( z_d = 4m \) and then land in spiral path in the presence of military grade wind gust. The control parameters are \( k_p = 0.9 \) and \( k_d = 9 \) so that the quadrotor can reach the desired path more accurately and smoothly in 10s. From Figure 4.5 (a) the total time taken to complete the path and land is 100s. The \( \psi \) also stabilizes to zero very quickly. \( \phi_C \) and \( \theta_C \) are 0.15 rad and -0.21 rad respectively while moving the quadrotor in spiral path. Figure 4.5(c) illustrates that the 4 rotor speeds are higher than \( w_O \), initially, to move the quadrotor at 4m after which it maintains its speed at \( w_O \) for the entire path. Figure 4.5(e) shows that the linear UIO correctly estimates the wind gusts though initially with a small error. The dotted line shows the reference and the solid line shows the estimated wind. The state estimation errors are close to zero but few errors are observed when the quadrotor rotates because of the oscillation of \( \phi_C \) and \( \theta_C \).
Figure 4.5 Results of military grade wind gusts for scenario 4 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory

4.1.2 Step Input Wind Gusts

The simulation results have been obtained for 4 different scenarios with step-shaped wind applied at $t = 0s$. The size of the wind gusts is $W_x = 2.5N$, $W_y = 1.5N$ and $W_z = 0.5N$ in the respective $x$, $y$ and $z$ directions.

Scenario 1: For the results presented in Figure 4.6, the quadrotor is required to move in a straight line to $x_d=0m$, $y_d=0m$ and $z_d=5m$ in the presence of step type wind gust. The quadrotor arrives to the target point in 15s. The $\psi$ stabilizes very quickly to $\frac{\pi}{2}$ while $\phi_C$ and $\theta_C$ are close to zero due to hovering condition. Figure 4.6 (c) illustrates the speeds of the four rotors move the quadrotor to the desired position and the quadrotor eventually maintains its speed at $w_0$. Figure 4.6(e) shows that the linear UIO estimates the wind gusts correctly with minor estimation error at the beginning of the flight. The dotted line shows the reference and the solid line shows the estimated wind. The state estimation errors are close to zero.
Figure 4.6 Results of step wind gusts for scenario 1 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory

**Scenario 2:** For the results presented in Figure 4.7, the quadrotor is required to move in a straight line from the origin to the point $x_d=1\,\text{m}$, $y_d=2\,\text{m}$ and $z_d=3\,\text{m}$ in the presence of step type wind gust. The quadrotor reaches the target point in 10s. $\psi$ stabilizes to $\frac{\pi}{2}$ at 1s while $\phi_c$ and $\theta_c$, are to be zero in hovering condition, become close to 0 at the same time as $\psi$. Figure 4.7(c) shows the speeds of the four rotors move the quadrotor to the desired position and the quadrotor eventually maintains its speed at $w_d$. Figure 4.7(e) shows that the linear UIO correctly estimates the wind gusts with small
estimation error at the beginning of the flight. The dotted line shows the reference and the solid line shows the estimated wind. The state estimation errors converge to zero, few errors are seen between 0s-5s.

Figure 4.7 Results of step wind gusts for scenario 2 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded \( \phi \) and \( \theta \); (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory.
Scenario 3: For the results presented in Figure 4.8, the quadrotor is required to move in a circular path of radius 10m and at a height $z_d = 5m$ in the presence of step wind gust. The control parameters are changed to $k_p = 0.9$ and $k_d = 9$ so that the quadrotor can reach the desired path more accurately and smoothly in 10s. $\psi$ also stabilizes to $\frac{\pi}{2}$ at 1.5s while $\phi_C$ and $\theta_C$ oscillate with amplitude of 0.1rad and 0.2rad respectively while moving the quadrotor in circular path. Figure 4.8(c) illustrates that the 4 rotors to move the quadrotor to the desired height and then it maintains its speed at $w_0$. Figure 4.8(e) shows that the linear UIO correctly estimates the wind gusts with small error between 0-1s. The dotted line shows the reference and the solid line shows the estimated wind. The state estimation errors are close to zero but few errors are seen when the quadrotor rotates in circle because the oscillation of $\phi_C$ and $\theta_C$. 

![Graphs](a), (b), (c), (d), (e), (f)
Figure 4.8 Results of step wind gusts for scenario 3 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory

Scenario 4: Spiral Path \([\text{radius}=1\text{m}, \text{height}=4\text{m}]\)

Scenario 4: For the results presented in Figure 4.9, the quadrotor is required to move at a height $z_d = 4\text{m}$ and then land in spiral path in the presence of step type wind gust. The control parameters are $k_p = 0.9$ and $k_d = 9$ so that the quadrotor can reach the desired path more accurately and smoothly in 100s. The $\psi$ also stabilizes to 0.05 very quickly. $\phi_C$ and $\theta_C$ are 0.16 rad and 0.12 rad respectively while moving the quadrotor in spiral path. Figure 4.9(c) illustrates the speeds of the four rotors move the quadrotor to the desired height and the quadrotor eventually maintains its speed. Figure 4.9(e) shows that the linear UIO correctly estimates the wind gusts with small error between 0-1s. The dotted line shows the reference and the solid line shows the estimated wind. The state estimation errors are close to zero but few errors are observed when the quadrotor rotates because of the oscillation of $\phi_C$ and $\theta_C$. 
Figure 4.9 Results of step wind gusts for scenario 4 (a) shows the linear position; (b) shows the orientation; (c) shows the rotor speeds; (d) shows the commanded $\phi$ and $\theta$; (e) shows the actual and estimated wind; (f) shows the state estimation error and (g) shows the 3D trajectory.
4.1.3 Time-varying Wind Gusts

The MATLAB simulation design is also tested for a large time-varying wind gust as shown in Figure 4.10. The quadrotor is required to move at a height $z_d = 4\text{m}$ and then land in spiral path in the presence of time varying wind gust. The control parameters are $k_p = 0.9$ and $k_d = 9$ so that the quadrotor can reach the desired path more accurately and smoothly in 100s. Figure 4.10(a) shows the estimation of the wind gusts. The wind estimation error is almost 0.

![Figure 4.10](image)

Figure 4.10 Results of time varying wind showing (a) the wind gusts with estimation; (b) the quadrotor position and (c) the desired spiral trajectory
4.1 Gazebo Simulation Results

4.1.1 Gazebo®

Gazebo® is one of the very few available choices for 3D dynamics simulation of multi-robots for any type of environment. It gives the possibility to create realistic worlds for robots in 3D animation rather than do it experimentally outdoors. In addition, Gazebo® is capable of simulating a swarm of robots, sensors and objects in a three dimensional space. In this respect this thesis has simulated the Erlecopter in Gazebo so as to accurately reflect the behavior of its physical counterpart. As a result the ROS® can see, through Gazebo®, the simulated aircraft as if it were real.

Gazebo® is designed to accurately reproduce the dynamic environment of a quadrotor[58]. The simulated prototype has mass, inertia, wind, friction and numerous other attributes that allow it to behave realistically when testing. These actions are also integral parts of an experiment. The aircraft prototype has been developed by Erle Robotics for Gazebo®, and it has a dynamic structure composed of a rigid body with joints, forces and torques to generate propulsion and interaction with the environment. The design environment, in which the aircraft is tested, can incorporate landscapes, buildings and in the case of this thesis, wind gusts. Almost all the aspects of the simulation of the quadrotor in Gazebo® are controllable. However, the sensors remain separate from the dynamics of simulation, since they only collect data or send data via ROS®. The software is best used on Ubuntu-Linux operating system and needs a high performance computer to operate. Figure 4.11 shows some examples of the environment created in Gazebo®.

Figure 4.11 showing the Erlecopter model in Gazebo software
4.1.2 Robot Operating System (ROS®)

ROS® is an open source-software platform designed to support a new generation of mobile robots [59]. The system is composed of reusable libraries that are designed to work independently. The libraries are accessed with a thin message-passing layer that enables them to be used by employing ROS® nodes. A ROS® node is an executable program that allows communication with another nodes. The ROS® community has developed many algorithms that the robots have increasing level of autonomy.

ROS® Topic are the named buses over which nodes exchange messages. Topics have publish/subscribe semantics, which decouples the production of information from its uses. The publish message would send information from the MATLAB/Simulink to the robot and then destruct the topic, on contrary the subscribe message would send information from robot to MATLAB/Simulink. In general the nodes are not aware of which topic they are communicating with [60]. Instead, nodes that are interested in data subscribe to the relevant topic; nodes that generate data publish to the relevant topic. There can be multiple publishers and subscribers of a topic. One of the basic tools of ROS®, rostopic, allows a command-line user to see what is being said about a topic and how frequently messages are being published. In addition to rostopic, ROS® contains many useful tools [60]. There is a set of tools for finding or creating packages, resolving dependencies, and compiling them [61]. There are tools for visualizing the running system and graphing the output of nodes in the system. Figure 4.12 shows the communication between topics and nodes.

Figure 4.12 shows the communication between topics and nodes.

As a further validation step, the proposed solution has been tested within a ROS®/Gazebo®-based framework using the Ardupilot 3.5 controller. This proved that
the approach is valid for any controller steering the drone while remaining in quasi-hovering configurations, provided that rotor speed commands are available to the UIO filter. To ensure complete software compatibility with the Erlecopter Brain 3 interface, ROS® Indigo distribution and Gazebo 4 have been used. More precisely, the architecture implemented is as follows:

1. ROS® provides a middleware layer for the SITL emulation,
2. the Gazebo software provides a reliable simulation platform for the aircraft physics, including wind gust forces, and
3. a Matlab/Simulink scheme implements the proposed UIO filter. Gazebo receives rotor speeds from subscribed ROS® topics and publishes the aircraft’s pose (position and attitude) to ROS®, while the Matlab/Simulink node subscribes and receives such pose topics.

The designed Unknown Input Observer is not limited to work on a specific controller. It can also be used with other types of control systems. This was evident as the UIO was also tested in the Gazebo® simulator. Gazebo® is a robotics simulator which is used to create applications for a physical robot without depending on the actual machine. These applications can be transferred onto the physical robot (or rebuilt) without modifications. ROS® is a software for controlling robotic components from a PC. A ROS® system is comprised of a number of independent nodes, each of which communicates with the other nodes using a publish/subscribe messaging the prototype.

In the Gazebo® simulator, the Erlecopter model was used to carry out tests to validate the Linear UIO. In this, wind disturbance to the system is applied in the form of step input only due to limitations. This is because only step type wind gust is only used for Gazebo® experiments. The UIO was then used to estimate these unknown wind gusts which were applied to the system. This was then compensated by Ardupilot and enabled the quadrotor to reach the desired position. It can be seen from 4.13 that $w_o$ of the quadrotor is below the desired value due to wind $W_Z$ pushing the quadrotor from $Z$ direction.
4.2.1 Step Wind Gusts in Gazebo®

The Gazebo® results have been obtained for 3 different scenarios with the wind gust. First scenario has wind in $X$ and $Y$ directions and second and third scenario are for wind in $X$, $Y$ and $Z$ directions. It is seen that for Figure 4.13 and 4.14, the $w_o$ of the quadrotor is below the desired value due to wind $W_Z$ pushing the quadrotor from $Z$ direction. All the experiments started from configuration $X(0) = 0_{12 \times 1}$. For Gazebo, the $\psi$ is $-1.13$ which has been decided by Ardupilot.

Scenario 1: The proposed UIO is tested with a constant horizontal wind gusts of $W_x = 1.2N$, $W_y = 0.8N$. The quadrotor is required to move to $x_d=0m$, $y_d=0m$ and $z_d=10m$ while $\psi$ is determined the Ardupilot to the value $-1.15$ rad. In the Gazebo environment, $w_0$ is set to $57.7$rad/s and the quadrotor moves to $w_0$ in less than 3s after which it remains at constant speed. The $\phi_c$ and $\theta_c$ are at 0.15 and -0.059 respectively after the quadrotor reaches its desired position. Figure 4.13(e) shows that the linear UIO is able to estimate the wind gusts correctly when the Ardupilot controlling the quadrotor in Gazebo. The estimated winds have initial errors and then settled to the reference wind gust amplitude.
Figure 4.13 Results of Gazebo® for scenario 1 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the 3D trajectory.

**Scenario 2**

The UIO is tested with constant wind gusts of $W_x = -0.5N$, $W_y = 1.2N$ and $W_z = 2N$. The quadrotor is required to move to $x_d=0m$, $y_d=0m$ and $z_d=10m$ and the $\psi$ determined by the Ardupilot is -0.84 rad. In the Gazebo® environment the $w_0$ is set to 57.7rad/s but the quadrotor rotor speeds are 49rad/s, less than $w_0$ because of the wind gust acting along the positive Z direction hence pushing the quadrotor. The $\phi_c$ and $\theta_c$ are at 0.155 and -0.055 respectively after the quadrotor reaches its desired position. It is seen from Figure 4.14(e) that the linear UIO is able to estimate the wind gusts correctly even when the Ardupilot controls the quadrotor in Gazebo®. $\hat{W}_x$ and $\hat{W}_y$ settles down to the references quite quickly after the initial transient. The estimated wind along the Z direction, $\hat{W}_z$, has an initial oscillation which dies out after 4s.
Figure 4.14 Results of Gazebo® for scenario 2 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the 3D trajectory.

Scenario 3: The linear UIO is tested with constant wind gusts of $W_x = 2.5 N$, $W_y = 1.5 N$ and $W_z = 0.5 N$. The quadrotor is required to move from the origin to $x_d = 1.2 m$, $y_d = 3.5 m$ and $z_d = 4 m$ along a straight line and the $\psi$ determined by the Ardupilot. In the Gazebo® environment the $w_0$ is 57.7 rad/s but the quadrotor rotor speeds are 55 rad/s, less than $w_0$, because of the wind gust acting along the positive Z directions pushing the quadrotor. The $\phi_c$ and $\theta_c$ are at 0.29 and -0.14 respectively after the quadrotor reaches its desired position. It is seen from Figure 4.15(c) that the linear UIO is able to estimate the wind gusts correctly even when the Ardupilot controls the quadrotor in Gazebo®. The estimated wind have initial error but settles down to the reference wind gust amplitude.
Figure 4.15 Results of Gazebo® for scenario 1 (a) shows the linear position; (b) shows the orientation; (c) shows the rotor speeds; (d) shows the commanded $\phi$ and $\theta$; (e) shows the actual and estimated wind; (f) shows the 3D trajectory.

4.3 Experimental Results

As the final step in the validation process, experiments on a real Erlecopter platform have been carried out with time varying horizontal wind. The Erlecopter platform has a built-in uBlox Neo-8M GPS for center of mass position measurement, an IMU sensor for attitude measurement, and a WIFI dongle to communicate with the base station. For the scope of the validation, actual pose information is extracted through the APM planner interface running on the base station. Reference positions for the experiment are given through a compiled Matlab/Simulink ROS®-based node, which is loaded onboard the Erle-Brain and can be activated from the base station. Ground truth information about the wind speed is provided by a Campbell Scientific CR1000 data logger (as shown in Figure 4.18), connected to the base station via RS232 port and receiving data from an R30M 3-cup rotor sensor (measuring wind speed norm) and a wind vane (measuring wind direction). More precisely, given the measured wind speed components, $v_x$ and $v_y$, the wind speed force components, $W_x$ and $W_y$, can be obtained according to the known formula

$$W_i = \rho S_i v_i^2, \text{ for } i \in \{x, y\} \tag{4.1}$$

where $\rho \approx 1.225\, \text{kg/m}^3$ is the air density at sea level and the quadrotor lateral sections, for small roll and pitch, can be approximated as $S_x = S_y = 9.88 \times 10^{-3}\, \text{m}^2$.

The drone is required to move along a series of waypoints. Rotor speeds commanded by Ardupilot and pose information are collected by the base station. The APM
(Ardupilot Mega) mission planner logs all the flight data including linear positions, angular positions and rotor speeds. All this data is processed offline by the proposed UIO filter.

Figure 4.16 The quadrotor flight during experimental testing. On bottom left of the figure are wind measuring instruments during testing at The University of the South Pacific Campus.

For the results in Figure 4.17, the quadrotor is required to move to \( x = 0 \, m, y = 0 \, m \) and \( z = 3 \, m \). The experimental work was chosen to be taken outdoor at The University of the South Pacific, Laucala Campus due to better GPS signal than indoor testing. The flight time to reach the desired target is very little in which wind speed and direction fluctuates highly. The experimental results in Figure 4.17 results proves that UIO shows sufficient level of accuracy in estimating wind gusts. The small errors are due to GPS resolution and the complex structure of the quadrotor and the area of the quadrotor is not very precise. The area of the propeller is also neglected as it is impossible to estimate the area in flight. Moreover the trend of estimated wind is similar to actual wind.
Figure 4.17 Experimental results (a) the actual and estimated $W_x$; (b) the actual and estimated $W_y$.

Figure 4.18 shows the plotted results of the logged data by APM. There is a large GPS error while doing the experimentation. The manufacturer stated that the GPS has an error of $\pm 5m$ and 5 satellites should be on line of sight [84]. This resulted in a lot of errors of $x$ and $y$ as shown in Figure 4.18(a) and 4.18(b). Hence, this also affected the UIO estimation for experimentation. Figure 4.18(c) shows the 4 rotor speeds.
Figure 4.18 Experiment results showing (a) the linear position (b) the angular positions (c) the rotor speeds.
Chapter 5

Unknown Input Observers for Nonlinear System with Partially Unknown Inputs

This chapter discusses the nonlinear model of the quadrotor investigated by a nonlinear unknown input observer (NUIO) to retrieve the unknown wind gusts. The proposed method uses the Linear Matrix Inequality (LMI) approach for the observer design. The method presented in [55] is used.

5.1 Problem Formulation

Consider a class of nonlinear system whose state description is

\[
\begin{align*}
\dot{x}(t) &= Ax(t) + g(u(t), y(t)) + f(x(t)) + Dw(t), \\
y(t) &= Cx(t),
\end{align*}
\]

(5.1)

where \(x(t) \in \mathbb{R}^n\) represents the state vector, \(u(t) \in \mathbb{R}^m\) represents the known input vector, \(w(t) \in \mathbb{R}^k\) represents the unknown input vector and \(y(t) \in \mathbb{R}^p\) represents the output vector. Matrices \(A, C\) and \(D\) are appropriate dimension matrices and \(g(u(t))\) is a function \(\mathbb{R}^{m+p} \to \mathbb{R}^n\). \(D\) is assumed to be full column rank, without the loss of generality.

\(f(x(t))\) is a nonlinear function \(\mathbb{R}^n \to \mathbb{R}^n\) that satisfies the following assumption of Lipschitz

\[
|f(x) - f(\hat{x})| \leq \gamma |x - \hat{x}| \quad \text{for all } x, \hat{x} \quad \text{for } \gamma > 0.
\]

(5.2)

The observer is constructed so that it can asymptotically estimate the states of the nonlinear system without the knowledge of the unknown input \(w(t)\).

Consider the proposed scheme of observer as proposed in [55]

\[
\dot{z}(t) = Nz(t) + Ly(t) + MG(u(t), y(t)) + Mf(\hat{x}(t)),
\]

(5.3)

and

\[
\hat{x}(t) = z(t) - Ey(t),
\]

(5.5)

where \(z(t) \in \mathbb{R}^n\) and \(\hat{x}(t) \in \mathbb{R}^n\).
The matrices $N, L, M$ and $E$ are unknown matrices of appropriate dimensions which must be determined such that $\hat{x}(t)$ will asymptotically converge to $x(t)$. $N, L$ and $M$ are:

$$
N = MA - LC \\
L = K(I_p + CE) - MAE \\
M = I_n + CE
$$

(5.6)

$E$ and $K$ will be selected by the observer design.

By introducing

$$
e(t) = x(t) - \hat{x}(t) = x(t) - z(t) + Ey(t).
$$

(5.7)

The derivative of this observer error is

$$
\dot{e}(t) = M\dot{x}(t) - \dot{z}(t).
$$

(5.8)

Thus, by using (5.7) and (5.8), the derivative of error can also be stated as

$$
\dot{e}(t) = Ne(t) + (MA - LC - NM)x(t) + M(fx(t) - f\hat{x}(t)) + MDw(t)
$$

(5.9)

by employing the term $MA - LC - NM = 0$, so

$$
\dot{e}(t) = Ne(t) + M(fx(t) - f\hat{x}(t)) + MDw(t)
$$

(5.10)

As it is known

$$
fx(t) - f\hat{x}(t) \approx \begin{bmatrix}
\frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\
\vdots & \ddots & \vdots \\
\frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n}
\end{bmatrix}(x(t) - \hat{x}(t))
$$

(5.11)

Therefore, by using differential mean value theorem (DMVT)

$$
f(x) - f(\hat{x}) = J_f(c)(x - \hat{x}), \text{where } c(t) \in [x(t), \hat{x}(t)]
$$

(5.12)
Thus
\[ M(fx(t) - f\hat{x}(t)) = MJ_f(c)e. \quad (5.13) \]

Now, if there exists an E matrix such that
\[ ECD = -D \rightarrow (EC + I_n)D = 0 \rightarrow MD = 0 \quad (5.14) \]
Then (5.10) becomes
\[ \dot{e}(t) = Ne + MJ_f(c)e = (N + MJ_f(c))e \quad (5.15) \]

since \( N = MA - KC \),
\[ \dot{e}(t) = (MA - KC)e(t) + MJ_f(c)e \quad (5.16) \]
\[ \dot{e}(t) = \left(M \left( A + J_f(c) \right) - KC \right)e(t) \quad (5.17) \]

Let \( A + J_f(c) = A(c(t)) \)
Thus \( \dot{e}(t) = (MA(c(t) - KC)e(t) \) (5.18)

To proceed further, the Lyapunov function is chosen to converge the error to zero
\[ V(t) = e^T(t)Pe(t) \text{ where } P = P^T > 0; \text{ then} \]
\[ \dot{V} = e^T(t)[(PMA(c(t)) - PKC + A^T(c(t))M^TP - C^TK^TP)]e(t) \quad (5.19) \]

Hence, the estimation error will converge to zero if (5.19) is satisfied
\[ (PMA(c(t)) - PKC + A^T(c(t))M^TP - C^TK^TP) < 0 \quad (5.20) \]

The following assumption is made
\[ \left| \frac{\partial f_i}{\partial x_j} c(t) \right| < +\infty \]
That is, for all the components of the Jacobian matrix, all $t$ are bounded. Each $h_{ij} = \frac{\partial f_i}{\partial x_j} c(t)$ has a minimum $h_{ij}$ and maximum $\overline{h}_{ij}$, so $h_{ij} \in [h_{ij}, \overline{h}_{ij}]$. So all elements of $h_{ij}$ of the matrix are contained in parallelepipedon $H$ whose faces are limited by $\overline{h}_{ij}$ and $h_{ij}$. This parallelepipedon $H$ has $2^{n \times n}$ vertices $\alpha$ given by $\{\alpha_{ij}\}$.

If (5.20) is satisfied in the vertices $\{\alpha_{ij}\}$, then it is satisfied also for all values of $h(t) = J_{f}(c)$ lying inside parallelepipedon $H$, then error will converge to zero.

Hence

$$\begin{align*}
(PMA(\alpha) - PKC + A^T(\alpha)MT - CTKTP) &< 0 \\
\forall \alpha \in V_{H_{q,n}}
\end{align*} \tag{5.21}$$

where $A(\alpha) = A + J_{f}(\alpha)$, $\alpha$ is the set of all vertices $\alpha_{ij}$ and $i, j = 1 \ldots n$.

If (5.20) and (5.21) are satisfied then $(t) \to 0; t \to \infty$. In order to give a sufficient condition for the existence of the observer, (5.19) is transformed into a set of Linear Matrix Inequalities (LMI) which is easier to solve.

Firstly, $E$ is to be found so that $MD = 0$. Hence, the unknown input is decoupled so as to satisfy (5.14), $(EC + I_{n})D = 0$. This means that $ECD = -D$ This matrix equation gives solutions if

$$\begin{align*}
\text{rank} \left[ \begin{array}{c} CD \\ D \end{array} \right] &= \text{rank}[CD] \\
\text{rank}[CD] &= \begin{pmatrix} \text{rank}[CD] \end{pmatrix} \\
\text{rank}[CD] &= \begin{pmatrix} (CD)^+ \end{pmatrix} \tag{5.22}
\end{align*}$$

All the possible solutions of $E$ are given as

$$E = -D(CD)^+ + S(I_p - (CD)(CD)^+) \tag{5.23}$$

Where $(CD)^+ = ((CD)^T(CD)^{-1}(CD)^T)$ is the pseudo matrix inverse of $CD$.

Since $(CD)^+(CD) = I_k$, both sides of the equation can be multiplied by , therefore, the required equality $ECD = -D$ is obtained as in (5.23). $S$ is an arbitrary matrix of the size $p$.

Let

$$U = -D(CD)^+ \quad \text{and} \quad V = I_p - (CD)(CD)^+ \quad \text{then} \quad E = U + SV \tag{5.24}$$
By substituting $M$ and $E = U + SV$ in equation (5.21)

$$
((I_n + UC)A(\alpha))^T P + P(I_n + UC)A(\alpha) + (VCA(\alpha))^T S^T P + PS(VCA(\alpha))
- C^T K^T P - PKC < 0 \quad \forall \alpha \in V_{H,q,n}
$$

(5.25)

If $PS = \tilde{S}$ and $= \tilde{R}$, then the set of nonlinear matrix inequalities in (5.25) will become

$$
((I_n + UC)A(\alpha))^T P + P(I_n + UC)A(\alpha) + (VCA(\alpha))^T \tilde{S}^T + \tilde{S}(VCA(\alpha)) - C^T \tilde{K}^T - \tilde{K} C < 0
$$

\forall \alpha \in V_{H,q,n}

(5.26)

which is a set of linear inequalities in $P, \tilde{S}, \tilde{R}$ and resolvable by using LMI.

If there exist $\tilde{S}, \tilde{R}$ and symmetric matrix $P > 0$ such that (5.26) has a solution in the vertices $\alpha_{i,j}$, then the observer has $e(t) \to 0$ as $t \to \infty$ for any initial condition $e(0)$.

In order to compute the unknown input $w$ it is more convenient to use the discretised form of the observer (5.3), where it is shown that it is given by (k is the discrete time and $Ad$ is the matrix after discretization):

$$
w(k) = [D^T D]^{-1}D^T [\tilde{x}(k + 1) - A_d \tilde{x}(k) - f(\tilde{x}(k)) - g(u(k), y(k))]
$$

(5.27)

Furthermore, Figure 5.1 illustrates the nonlinear UIO scheme which has been derived from equation (5.27).
5.2 Design of Non-Linear Unknown Input Observer for Quadrotor Model

Recall from Chapter 2, the nonlinear model of the quadrotor reads

\[ \dot{x} = f(\bar{x}) + g(\bar{x})\bar{u} \]  

(5.28)

\[
f(\bar{x}) = \begin{bmatrix}
    u \\
    v \\
    w \\
    0 \\
    0 \\
    -g \\
    \cos(\theta) p + \sin(\theta) \\
    \tan(\phi) \sin(\theta) p + q - \tan(\phi) \cos(\theta) r \\
    -\frac{\sin(\theta)}{\cos(\phi)} p + \frac{\cos(\theta)}{\cos(\phi)} r \\
    -\left( \frac{l_{xx} - l_{yy}}{l_{xx}} \right) qr \\
    -\left( \frac{l_{xx} - l_{zz}}{l_{yy}} \right) pr \\
    -\left( \frac{l_{yy} - l_{xx}}{l_{zz}} \right) pq
\end{bmatrix}
\]  

(5.29)

\[
g(\bar{x})\bar{u} = \begin{bmatrix}
    0 & 0 & 0 & 0 & 0 & 0 \\
    0 & 0 & 0 & 0 & 0 & 0 \\
    0 & 0 & 0 & 0 & 0 & 0 \\
    0 & 0 & 0 & 0 & 0 & 0 \\
    0 & 0 & 0 & 0 & 0 & 0 \\
    1 & 0 & 0 & 0 & 0 & 0 \\
    \cos(\phi) \cos(\theta) & \frac{1}{m} & 0 & 0 & 0 & 0 \\
    0 & 0 & 0 & 0 & 0 & 0 \\
    0 & 0 & 0 & 0 & 0 & 0 \\
    0 & 0 & 0 & 0 & 0 & 0 \\
    0 & 0 & 0 & 0 & 0 & 0 \\
    0 & 0 & 1/l_{xx} & 0 & 0 & \end{bmatrix}
\]
Moreover, when the model is converted to the form of equation 5.1, this can be written as

\[
\dot{x}(t) = Ax(t) + g_1(u(t), y(t)) + f_1(x(t)) + Dw(t)
\]

\[
y(t) = Cx(t)
\]

\[
A_{12\times12} = \begin{bmatrix}
0_{3\times3} & I_{3\times3} & 0_{3\times3} & 0_{3\times3} \\
0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\
0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\
0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3}
\end{bmatrix}
\]

\[
g_1(u(t), y(t)) = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
\sin(\theta) \cos(\psi) + \sin(\phi) \cos(\theta) \sin(\psi) \frac{F}{m} & 0 & 0 & 0 \\
\sin(\theta) \cos(\psi) - \sin(\phi) \cos(\theta) \sin(\psi) \frac{F}{m} & 0 & 0 & 0 \\
\cos(\phi) \cos(\theta) \frac{F}{m} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 1/L_{xx} & 0 & 0 \\
0 & 0 & 1/L_{yy} & 0 \\
0 & 0 & 0 & 1/L_{zz}
\end{bmatrix}
\]
$$f_1(x) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -g \\ \cos(\theta) p + \sin(\theta) \\ \tan(\phi) \sin(\theta) p + q - \tan(\phi) \cos(\theta) r \\ -\frac{\sin(\theta)}{\cos(\phi)} p + \frac{\cos(\theta)}{\cos(\phi)} r \\ -\frac{l_{xx}^2 - l_{yy}^2}{l_{xx}} qr \\ -\frac{l_{xx}^2 - l_{zz}^2}{l_{yy}} pr \\ -\frac{l_{yy}^2 - l_{zz}^2}{l_{zz}} pq \end{bmatrix}$$ \hspace{1cm} (5.33)

And the unknown wind gust matrix is

$$Dw = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1/m \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$ \hspace{1cm} (5.34)

To satisfy equation (5.22), the measurable outputs of $C$ are $[x, y, z, \phi, \theta, \psi, u, v, w]$, hence $ECD = -D$ has the appropriate solution of $E$. As discussed in the earlier chapters, the measurable outputs of the Linear UIO are only the positions, but the same 6 outputs for Nonlinear UIO does not satisfy equation (5.20), so the following matrix $C$ has to be used for 9 outputs.

$$C = [I_{9 \times 9} \quad 0_{9 \times 3}]$$ \hspace{1cm} (5.35)

On the basis of $C$ and $D$, matrices $U$ and $V$ are found as discussed in (5.21) and (5.22).
The next step is to identify all the vertices \( \alpha_{ij} \) and \( i, j = 1 \ldots n \) and compute the LMI for (5.26). For this purpose \( A(\alpha) \) must be computed for all \( \alpha = \alpha_{ij} \) as stated before. The Jacobian matrix has been developed for the hovering condition of the quadrotor, so that \( \phi \) and \( \theta \approx 0 \). In this way the quadrotor has 4 outputs and 4 inputs and the system is fully manageable and moreover, the pair \( (A(\alpha) \text{ and } C) \) are observable.

Therefore

\[
J_f = \begin{bmatrix}
0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\
0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\
0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\
0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3}
\end{bmatrix}
\begin{bmatrix}
0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\
0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\
0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\
0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3}
\end{bmatrix}
\begin{bmatrix}
0 & r & 0 & 0 \\
-\frac{(l_{yx} - l_{zz})}{l_{xx}} & 1 & 0 & 0 \\
-\frac{(l_{zx} - l_{yy})}{l_{yy}} & 0 & 1 & 0 \\
q & \frac{(l_{zx} - l_{yy})}{l_{xx}} & p & 0
\end{bmatrix}
\begin{bmatrix}
0 & r & 0 & 0 \\
-\frac{(l_{yx} - l_{zz})}{l_{xx}} & 1 & 0 & 0 \\
-\frac{(l_{zx} - l_{yy})}{l_{yy}} & 0 & 1 & 0 \\
q & \frac{(l_{zx} - l_{yy})}{l_{xx}} & p & 0
\end{bmatrix}
\]

Hence, \( A(\alpha) = A + J_f \)
Using $A(\alpha)$, the equation (5.24) is then solved using LMI technique in MATLAB. The LMI solves and give $\tilde{S}$ and $\tilde{R}$ for systematic matrix for which $P > 0$. Using this, $S$ and $K$ are computed as $S = P^{-1}\tilde{S}$, $K = P^{-1}\tilde{R}$. The matrix inequalities $PMA(\alpha) - PKC + A^T(\alpha)M^TP - C^TK^TP < 0$ is verified for all $\alpha_{ij} i = 1 ... n$ and $j = 1 ... n$ to see if the condition is met, which means that the eigenvalues are less than zero. In this case the error of the system will converge to zero. Also the matrices $N = MA - KC$ and $L = K(I_p + CE) - MAE$ are computed, since they are required by the observer design as expressed in (5.1). The Matlab/Simulink implementation is shown in the appendix. In addition, $E = U + SV$ and $M = I_n + EC$ as discussed earlier.

The nonlinear UIO discussed in this chapter was tested to estimate the wind gusts in two different conditions which are step type wind gusts and military grade wind gusts for the quadrotor model with the PD controller. The wind gusts observed are implemented in Matlab/Simulink.

Scenario 1: Step Input Wind Gusts

Figure 5.2 illustrates the results of the nonlinear UIO with the quadrotor model of the wind gust force as step input, where, $Wx = 3N$, $Wy = 2N$ and $Wz = 1N$. The continuous line shows the estimated wind gusts while the dotted line shows the actual wind gust. Moreover, Figure 5.3 shows the error between the estimated and actual wind gusts and Figure 5.4 shows the state estimation error.
Figure 5.2 Estimated versus actual wind gusts for step input

Figure 5.3 The error graph between estimated and actual wind gusts
Scenario 2 Military Grade Wind Gusts

Figure 5.5 illustrates the results of the nonlinear UIO with the quadrotor model military grade wind gusts model. The wind gusts are $W_x = 3N$, $W_y = 2N$ and $W_z = 1N$. The continuous line shows the estimated wind gusts while the dotted line shows the actual wind gusts. In addition, Figure 5.6 shows the error between the estimated and actual wind gusts and Figure 5.7 shows the state estimation error.
Figure 5.6 The error graph between estimated and actual wind gusts

Figure 5.7 State estimation error graph

Scenario 3 Time Varying Wind Gusts

Figure 5.8 illustrates the results of the nonlinear UIO with the quadrotor model time varying wind gusts model. The wind gusts have the amplitude of $Wx = 3N$, $Wy = 2N$ and $Wz = 1N$. The continuous line shows the estimated wind gusts while the dotted line shows the actual wind gusts. In addition, Figure 5.9 shows the error between the estimated and actual wind gusts and Figure 5.10 shows the state estimation error.
Figure 5.8 Estimated versus actual wind gusts for military grade wind model

Figure 5.9 The error graph between estimated and actual wind gusts

Figure 5.10 State estimation error graph
Chapter 6
The results of the Nonlinear Unknown Input Observer for Wind Gusts Estimation

This chapter shows the results obtained with the nonlinear Unknown Input Observer model discussed in Chapter 4 and the PD control on the nonlinear Model of the quadrotor from Chapter 2 Section 4. The results are computed in MATLAB Simulink® and ROS®-Gazebo®. The quadrotor is tested with different types of wind gust: step and military grade. For each result, the desired target outputs are indicated as \( r = (x_d, y_d, z_d, \dot{\psi}_d)^T \). The general scheme of the entire model is shown in Figure 6.1.

Like the Linear Model of the quadrotor, the same PD controller is used for the nonlinear model. The Attitude Control is used for the orientation in \( \phi, \theta, \psi \) while the Position Control is used to position the quadrotor along the axes \( x, y \) and \( z \) and by employing the information of estimated wind gusts \( \hat{\omega}_x, \hat{\omega}_y, \hat{\omega}_z \)^T coming from the UIO. As discussed earlier, the quadrotor is in hovering state which implies that \( \phi \approx 0 \) and \( \theta \approx 0 \). Using all of the control information, the rotor velocities of the quadrotor are then computed using equations (2.31)-(2.33), which are based on the 4 known inputs to the system.
For the nonlinear UIO, only 9 out of the 12 states are measurable in experimentation, as explained in Chapter 5: that is \([x, y, z, \phi, \theta, \psi, u, v, w]\). These are used by the nonlinear UIO to estimate the 3 remaining states and the unknown wind gust vector. The parameters of the Erlecopter prototype aircraft have been used to carry out the MATLAB Simulink® and ROS®-Gazebo® simulation are the same as those shown in Table 4.1 and repeated here in Table 6.1 for clarity.

Table 6.1 Parameters of the System in SI Units

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
<th>Quadrotor Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>(m)</td>
<td>Mass</td>
<td>1.12(m)</td>
</tr>
<tr>
<td>(g)</td>
<td>Gravitational Acceleration</td>
<td>9.81(m/s^2)</td>
</tr>
<tr>
<td>(I_{xx})</td>
<td>Inertia about x axis</td>
<td>(34.8 \times 10^{-3} kg/m^2)</td>
</tr>
<tr>
<td>(I_{yy})</td>
<td>Inertia about y axis</td>
<td>(45.9 \times 10^{-3} kg/m^2)</td>
</tr>
<tr>
<td>(I_{zz})</td>
<td>Inertia about z axis</td>
<td>(97.7 \times 10^{-3} kg/m^2)</td>
</tr>
<tr>
<td>(K_F)</td>
<td>Force Constant</td>
<td>(8.55 \times 10^6 Ns^2/m^2)</td>
</tr>
<tr>
<td>(K_M)</td>
<td>Moment Constant</td>
<td>0.015 (Ns^2/m^2)</td>
</tr>
<tr>
<td>(l)</td>
<td>Length</td>
<td>0.141(m)</td>
</tr>
</tbody>
</table>

6.1 MATLAB Simulink Results

6.1.2 Military Grade Wind Gusts

The MATLAB Simulink® results are obtained for 4 different scenarios with the military grade wind gust with a length of 10\(m\) and final amplitude of \(W_x = 2.5N\), \(W_y = 1.5N\) and \(W_z = 0.5N\). The starting state vector is assumed null, i.e., \(X(0) = 0_{12 \times 1}\).

Figures 6.1 to 6.9 shows the results for all considered scenarios.

Scenario 1: For the results presented in Figure 6.1, the quadrotor is required to move to \(x_d=0m\), \(y_d=0m\) and \(z_d=5m\) in the presence of military grade wind gust. The control parameters are \(K_p = 0.4\) and \(K_D = 4\). The quadrotor arrives to the target point in 5s, however, it reaches the steady state in 10s. The \(\psi\) also stabilizes to zero at 10s while \(\phi_c\) and \(\theta_c\), which are to be zero in hovering condition, become 0 at the same time as \(\psi\). Figure 6.1(c) illustrates the 4 rotor speeds are higher than \(w_o\) (hovering speed see...
Chapter 2) to move the quadrotor to the desired position and then it maintains its speed at $w_D$. Figure 6.1(c) shows that the nonlinear UIO correctly estimates the wind gusts with initial small error between 0-2s. The dotted line shows the reference and the solid line shows the estimated wind. The state estimation errors results are practically zero.

Figure 6.1 Results of military grade wind gusts for scenario 1 showing (a) the linear position; (b) shows the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the state estimation error and (g) shows the 3D trajectory.
Scenario 2: For the results presented in Figure 6.2, the quadrotor is required to move in a linear path from the origin to the point of \( x_d = 1 \text{m}, y_d = 2 \text{m} \) and \( z_d = 3 \text{m} \) in the presence of the military grade wind gust. The control parameters are \( K_p = 0.4 \) and \( K_D = 4 \). The quadrotor reaches the target point in 5s, and settles down to the steady state in 10s. \( \psi \) also stabilizes to zero at 10s while \( \phi_C \) and \( \theta_C \), which are to be zero in hovering conditions, become 0 at the same time as \( \psi \). Figure 6.2(c) illustrates the speeds of the four rotors move the quadrotor to the desired position and the quadrotor eventually maintains its speed at \( w_D \). Figure 6.2(e) shows that the nonlinear UIO correctly estimates the wind gusts with small error initially. The dotted line shows the reference and the solid line shows the estimated wind. The state estimation errors are close to zero and few errors are seen when the orientation of the quadrotor reaches the steady state.
Figure 6.2 Results of military grade wind gusts for scenario 2 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory.

**Scenario 3:** For the results presented in Figure 6.3, the quadrotor is required to move in a circular trajectory of radius 10m and at a height $z_d = 5m$ in the presence of military grade wind gust. The control parameters are changed to $k_p = 0.9$ and $k_d = 9$ so that the quadrotor can reach the desired path more accurately and smoothly in 10s. The $\psi$ also stabilizes to zero at 10s while $\phi_c$ and $\theta_c$ oscillates with amplitude of 0.2 while moving the quadrotor in circular path. Figure 6.3(c) illustrates the speeds of the four rotors move the quadrotor to the desired position and the quadrotor eventually maintains its speed at $w_d$. Figure 6.3(d) shows that the nonlinear UIO correctly estimates the wind gusts with small error initially. The dotted line shows the reference and the solid line shows the estimated wind. The state estimation errors are close to zero but fewer errors are seen when the quadrotor rotates in circle because of the oscillations of $\phi_c$ and $\theta_c$. 

88
Figure 6.3 Results of military grade wind gusts for scenario 3 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory.
Scenario 4: For the results presented in Figure 6.4, the quadrotor is required to move at a height \( z_d = 4 \text{m} \) and then land in spiral path in the presence of military grade wind gust. The control parameters are \( k_p = 0.9 \) and \( k_d = 9 \) so that the quadrotor can reach the desired path more accurately and smoothly in 10s. From Figure 6.4 (a) the total time taken to complete the path and land is 300s. The \( \psi \) also stabilizes to zero at 10s together with \( \phi_c \) and \( \theta_c \). Afterwards \( \phi_c \) and \( \theta_c \) oscillate at amplitude of 0.1 while moving the quadrotor in spiral path. Figure 6.4(c) illustrates the 4 rotor speeds are higher than \( w_o \), initially, to move the quadrotor at 4m and then the quadrotor eventually maintains its speed close to \( w_o \) for the entire path. Figure 6.4(e) shows that the nonlinear UIO correctly estimates the wind gusts with small error initially. The dotted line shows the reference and the solid line shows the estimated wind. The state estimation errors are close to zero but fewer errors are observed when the quadrotor rotates because of the oscillation of \( \phi_c \) and \( \theta_c \).
Figure 6.4 Results of military grade wind gusts for scenario 4 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory

6.1.3 Step Wind Gusts

The MATLAB Simulink® results are obtained for four different scenarios with the step type wind gust. The step type wind gusts have the size of $W_x = 2.5 N$, $W_y = 1.5 N$ and $W_z = 0.5 N$. The starting state vector is assumed null, i.e., $X(0) = 0_{12 \times 1}$.

Figures 6.5 to 6.8 show the results for all considered scenarios.

Scenario 1: For the results presented in Figure 6.5, the quadrotor is required to move in a straight line to $x_d = 0m$, $y_d = 0m$ and $z_d = 5m$ in the presence of step type wind gust. The quadrotor arrives to the target point in 5s, however, it reaches the steady state in 10s. The $\psi$ also stabilizes to zero at 10s while $\phi_c$ and $\theta_c$, are to be zero due to hovering condition, become 0 at the same time as $\psi$. Figure 6.1(c) illustrates the speeds of the four rotors move the quadrotor to the desired position and the quadrotor eventually maintains its speed at $w_D$. Figure 6.5(e) shows that the nonlinear UIO correctly estimates the wind gusts with a small estimation error initially between 0-1s. The dotted line shows the reference and the solid line shows the estimated wind. The state estimation errors are close to zero.
Figure 6.5 Results of step type wind gusts for scenario 1 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory.

**Scenario 2:** For the results presented in Figure 6.6, the quadrotor is required to move in a line path from the origin to the point $x_d=1\text{m}$, $y_d=2\text{m}$ and $z_d=3\text{m}$ in the presence of the step type wind gust. The quadrotor reaches the target point in 5s and settles.
down to the steady state in 10s. $\psi$ also stabilizes to zero at 10s while $\phi_c$ and $\theta_c$, are to be zero in hovering condition, become 0 at the same time as $\psi$. Figure 6.6(c) illustrates the speeds of the four rotors move the quadrotor to the desired position and the quadrotor eventually maintains its speed at $w_0$. Figure 6.6(e) shows that the nonlinear UIO correctly estimates the wind gusts small estimation error initially between 0-1s. The dotted line shows the reference and the solid line shows the estimated wind. The state estimation errors converges to zero.
Figure 6.6 Results of step type wind gusts for scenario 2 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory.

**Scenario 3:** For the results presented in Figure 6.7, the quadrotor is required to move in a circular path of radius 10m and at a height $z_d = 5m$ in the presence of step wind gust. The control parameters are changed to $k_p = 0.9$ and $k_d = 9$ so that the quadrotor can reach the desired path more accurately and smoothly in 10s. $\psi$ also stabilizes to zero at 10s while $\phi_c$ and $\theta_c$ oscillate with amplitude of 0.2 while moving the quadrotor in circular path. Figure 6.7(c) illustrates the speeds of the four rotors move the quadrotor to the desired position and the quadrotor eventually maintains its speed close to $w_o$. Figure 6.7(e) shows that the nonlinear UIO correctly estimates the wind gusts with small error initially between 0-1s. The dotted line shows the reference and the solid line shows the estimated wind. The state estimation errors are close to zero but few errors are seen when the quadrotor rotates in circle because the oscillation of $\phi_c$ and $\theta_c$. 
Figure 6.7 Results of step type wind gusts for scenario 3 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded \( \phi \) and \( \theta \); (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory.

**Scenario 4:** For the results presented in Figure 6.8, the quadrotor is required to move at a height \( z_d = 4m \) and then land in spiral path in the presence of the step type wind gust. The control parameters are \( k_p = 0.9 \) and \( k_d = 9 \) so that the quadrotor can reach the desired path more accurately and smoothly in 10s. From Figure 6.8 (a) the total time taken to complete the path and land is 300s. The \( \psi \) also stabilizes to zero at 10s together with \( \phi_c \) and \( \theta_c \). Afterwards \( \phi_c \) and \( \theta_c \) oscillate at amplitude of 0.1 while moving the quadrotor in spiral path. Figure 6.8(c) illustrates the 4 rotor speeds are higher than \( w_o \) , initially, to move the quadrotor at 4m and then it maintains close to
$w_d$ for the entire path. Figure 6.8(e) shows that the nonlinear UIO correctly estimates the wind gusts with small error between 0-1s. The dotted line shows the reference and the solid line shows the estimated wind. The state estimation error is close to zero but some errors are observed when the quadrotor rotates because of the oscillation of $\phi_c$ and $\theta_c$. 

(a) (b) (c) (d) (e) (f)
Figure 6.8 Results of step type wind gusts for scenario 4 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the state estimation error and (g) the 3D trajectory

4.1.3 Time-varying Wind Gusts

The MATLAB simulation design is also tested for a large time-varying wind gust as shown in Figure 6.9. The quadrotor is required to move at a height $z_d = 4m$ and then land in spiral path in the presence of time varying wind gust. The control parameters are $k_p = 0.9$ and $k_d = 9$ so that the quadrotor can reach the desired path more accurately and smoothly in 100s. Figure 6.9(a) shows the estimation of the wind gusts. The wind estimation error is almost 0.
6.2 Gazebo® Simulation Results

Similar to Chapter 4, the proposed solution is tested with ROS®/Gazebo®-based framework using Ardupilot 3.5 controller. The Ardupilot has its own nonlinear controller, however the nonlinear UIO design is valid for any type of controller, provided the rotor speeds and the outputs are available to the nonlinear UIO for estimation of wind gusts. The ROS nodes communicates to the Simulink and Gazebo® software. Figure 6.10 shows the communication links between Gazebo®, ROS® and Ardupilot.
In the Gazebo® simulator, the Erlecopter model is used to carry out the testing of the nonlinear UIO. ROS® receives the 9 outputs and the 4 rotor speeds from Gazebo® through MAVROS and sends it to Simulink for computation of the unknown wind gusts. The Gazebo results is tested for 3 different scenarios similar to Gazebo® experimentations of the linear UIO in Chapter 4.

### 6.2.1 Step Wind Gusts in Gazebo®

Only the step type wind gust model is tested because of the limitation issues of wind gust type in the Gazebo® software. Scenario 1 has a wind in $X$ and $Y$ directions and scenarios 2 and 3 show the results for the wind in $Z$ direction as well as along $X$ and $Y$ directions. It is noticed that the rotor speeds are below $w_o$ for scenario 2 and 3 because the wind action along the $Z$ direction pushes the quadrotor. All the experimentations start from the null state vector configurations $X_0 = 0_{12 \times 1}$. For Gazebo®, the $\psi$ is determined by Ardupilot.

**Scenario 1**: The proposed nonlinear UIO is tested with a constant horizontal wind gusts of $W_x = 1.2N$, $W_y = 0.8N$. The quadrotor is required to move to $x_d=0m$, $y_d=0m$ and $z_d=10m$ while $\psi$ is determined the Ardupilot to the value -0.5 rad. In the Gazebo environment, $w_0$ is set to $57.7$rad/s and the quadrotor moves to $w_0$ in less than 5s and then remains at constant speed. The $\phi_c$ and $\theta_c$ are at 0.05 and -0.15 respectively after the quadrotor reaches its desired position. Figure 6.11(e) shows that the nonlinear UIO is able to estimate the wind gusts correctly also when the Ardupilot controls the quadrotor in Gazebo®. The estimated winds had initial error but then settled to the reference wind gust amplitude.
Figure 6.11 Results of Gazebo® for scenario 1 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the 3D trajectory

Scenario 2: The nonlinear UIO is tested with constant wind gusts of $W_x = -0.5N$, $W_y = 1.2N$ and $W_z = 2N$. The quadrotor is required to move to $x_d=0m$, $y_d=0m$ and $z_d=10m$ and the $\psi$ determined by the Ardupilot is $-0.75$ rad. In the Gazebo® environment the $\omega_0$ is set to 57.7rad/s but the quadrotor rotor speeds are 49rad/s, less than $\omega_0$ because of the wind gust acting along the positive Z direction hence pushing the quadrotor. The $\phi_C$ and $\theta_C$ are at 0.05 and -0.15 respectively after the quadrotor reaches its desired position. It is seen from Figure 6.12(e) that the nonlinear UIO is able to estimate the wind gusts correctly even when the Ardupilot controls the quadrotor in Gazebo®. The $\hat{W}_x$ and $\hat{W}_y$ settles down to the references quite quickly after the initial transient. The estimated wind along the Z direction, $\hat{W}_z$, has an initial oscillation which dies out after 6s.
Figure 6.12 Results of Gazebo® for scenario 2 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the 3D trajectory.

Scenario 3: The nonlinear UIO is tested with constant wind gusts of $W_x = 2.5N$, $W_y = 1.5N$ and $W_z = 0.5N$. The quadrotor is required to move from the origin to $x_d = 1.2m$, $y_d = 3.5m$ and $z_d = 4m$ along a straight line and the $\psi$ determined by the Ardupilot is -1.4 rad. In the Gazebo® environment the $w_0$ is 57.7rad/s but the quadrotor rotor speeds are 55rad/s, less than $w_0$, because of the wind gust acting along the positive Z directions pushing the quadrotor. The $\phi_c$ and $\theta_c$ are at 0.29 and -0.1 respectively after the quadrotor reaches its desired position. It is seen from Figure 6.13(c) that the nonlinear UIO is able to estimate the wind gusts correctly even when the Ardupilot controls the quadrotor in Gazebo®. The estimated wind had initial error and then settles down to the reference wind gust amplitude.
Figure 6.13 Results of Gazebo® for scenario 3 showing (a) the linear position; (b) the orientation; (c) the rotor speeds; (d) the commanded $\phi$ and $\theta$; (e) the actual and estimated wind; (f) the 3D trajectory.
Chapter 7

Conclusion

In this thesis, firstly, the mathematical model of the quadrotor dynamics has been developed by using Newton’s and Euler’s laws to which the effect of wind gust along the x, y and z directions on the quadrotor aircraft has been added. Moreover, from these nonlinear equations the linearized model has been derived by using the Taylors series. The entire system has been controlled by using PD controller and then tested in Simulink to verify the goodness of the model as well as the behavior of the quadrotor in the presence of wind gusts as disturbances.

The Linear Unknown Input Observer (UIO) is designed for the discrete time linear quadrotor system which has been affected by the unknown wind gusts. The design has required the knowledge of the input of the system which are the 4 rotor speeds of the quadrotor as well as 6 states out of 12 states of the model. These 6 states \([x, y, z, \phi, \theta, \psi]\) were used by the linear UIO, which would estimate the other 6 unknown states \([u, v, w, p, q, r]\) which are the represented as the speeds together with the 3 unknown wind gusts \([w_x, w_y, w_z]\). The robustness of the observation system is also discussed as well as the sufficient conditions for the reconstruction of unknown wind gusts with no steady state error as shown in Chapter 3. The design has been tested in Chapter 4 with a few types of wind gust models: the step type, the military grade type and the time varying type. The Linear UIO system correctly estimates all types of wind gusts with a delay of \(2T_e\) (0.02s). The estimated wind gusts are feedback to the PD controller to compensate for the wind and consequently correct the path of the quadrotor by adjusting the rotor speeds. The entire model has been tested using MATLAB/Simulink®, Gazebo® and also experimentally through the use of Robot Operating System (ROS®) and an Erlecopter quadrotor aircraft. Gazebo® is a simulation software for robots which uses the robust physics of the quadrotor and implements the quadrotor in high-quality graphics, with the help of a simple programmatic and graphical interfaces. Both Gazebo® and ROS® have been used by using Linux® operating system, and the interface between Gazebo® and Simulink has been done through ROS®. Gazebo® simulation results have resulted to be of high precision almost as similar as the experimental ones. Finally, the experimental tests
have been carried outdoors in the test field. An anemometer and wind vane were stationed preliminary in a test field at the desired heights to measure the wind speed and directions which have been subsequently compared to the estimated wind gusts of the quadrotor aircraft. The Erlecopter which is a programmable aircraft ROS® nodes were used to program the UIO. The Ardupilot controller has been used for the aircraft and the internal GPS to gather the information on the positioning of the quadrotor. (The quadrotor’s internal IMU could be accessed for other information such as orientation. ROS® nodes also subscribed the rotor speeds.) The quadrotor position, orientation, rotor speeds, real wind gusts, estimated wind gusts and state estimation error have been presented in Chapter 4 and the results show the effectiveness and the goodness of the method.

In Chapter 5, the nonlinear model of the quadrotor has been used to develop a nonlinear UIO to extract the unknown wind gusts. For this purpose, differential mean value theorem (DMVT), together with a Lyapunov approach requiring the solution of linear matrix inequality (LMI) conditions for feasibility of the observer design. The LMI solutions provided the observer design matrices has been derived in Chapter 5. Similar to the Linear UIO, only some states and the inputs have been given to the nonlinear UIO to estimate the non-measured states and inputs. For this case, 9 states \([x, y, z, \phi, \theta, \psi, u, v, w]\) have been given togeteher with the 4 rotor speeds. The nonlinear UIO observer design has then estimated the other 3 states and the 3 unknown wind gusts \([w_x, w_y, w_z]\) with different types of wind gusts models. The estimated wind gusts have been feedback to the PD controller for compensation of the wind and to correct the path of the quadrotor. Although the PD controller is more appropriate for the linear model, it was used when employing the nonlinear model for its good results. The control has been achieved by varying the rotor speeds of the quadrotor with respect to the estimated wind. The entire nonlinear design has been tested by using the MATLAB/Simulink® and Gazebo® using Erlecopter quadrotor aircraft. Again, ROS® has been used to extract the rotor speeds and the 9 outputs which have been fed to the UIO for the estimation of the wind gust in Gazebo® environment. Finally, the quadrotors orientation, position, rotor speeds have been all extracted from Gazebo® using ROS®. Similar to linear UIO, the quadrotor’s position, orientation, rotor speeds, real wind gusts, estimated wind gusts and state estimation error have been presented in Chapter 6. It has been observed that the estimation of wind for the nonlinear UIO is
much faster when compared to the linear UIO and the nonlinear model designed has resulted in better performance in terms of robustness and settling time than the case of the linear system.

In conclusion the successful implementation of unknown wind gusts observers for both linear and nonlinear model of the quadrotor has been achieved in this thesis, together with the robust control of the quadrotor in the presence of wind gusts.

The claims of originality of the thesis are the following:

- Application of the linear unknown input observer to the linear model of the quadrotor aircraft. Compensation of the observed wind gusts to correct the path of the quadrotor by changing the rotor speeds in presence of unknown wind in x, y and z directions.

- Application of the nonlinear unknown input observer to the nonlinear model of the quadrotor aircraft. Compensation of the observed wind gusts to correct the path of the quadrotor by changing the rotor speeds in presence of unknown wind in x, y and z directions.

- Implementation of both linear and nonlinear system with UIO in the above points in MATLAB/Simulink®

- Implementation of both linear and nonlinear system with UIO in the above points in Gazebo® using ROS®.

- Implementation of with UIO to linear system as discussed in the first point and its experimental verification by the Erlecopter quadrotor aircraft and ROS®.
References


[34] D. G. Luenberger, “An Introduction to Observers,” IEEE Transactions on


[71] P. Velez, N. Certad, and E. Ruiz, “Trajectory Generation and Tracking Using the AR.Drone 2.0 Quadcopter UAV,” in Proceedings - 12th LARS Latin American Robotics Symposium and 3rd SBR Brazilian Robotics Symposium,


[82] R. Mahony, V. Kumar, and P. Corke, “Multirotor Aerial Vehicles: Modeling,


http://docs.erlerobotics.com/brains/discontinued/erle-brain
2/sofware/apm/log_files/gps
Appendix 1

Calculations for Linear UIO

The parameters for the Erlecopter model are the same as in Table 4.1 and is repeated here.

Table A1 Parameters of the Erlecopter Aircraft in SI Units

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Quadrotor Parameters</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m$</td>
<td>Mass</td>
<td></td>
<td>$1.12m$</td>
</tr>
<tr>
<td>$g$</td>
<td>Gravitational Acceleration</td>
<td></td>
<td>$9.81m/s^2$</td>
</tr>
<tr>
<td>$I_{xx}$</td>
<td>Inertia about x axis</td>
<td></td>
<td>$34.8 \times 10^{-3}kg/m^2$</td>
</tr>
<tr>
<td>$I_{yy}$</td>
<td>Inertia about y axis</td>
<td></td>
<td>$45.9 \times 10^{-3}kg/m^2$</td>
</tr>
<tr>
<td>$I_{zz}$</td>
<td>Inertia about z axis</td>
<td></td>
<td>$97.7 \times 10^{-3}kg/m^2$</td>
</tr>
<tr>
<td>$K_F$</td>
<td>Force Constant</td>
<td></td>
<td>$8.55 \times 10^6Ns^2/m^2$</td>
</tr>
<tr>
<td>$K_M$</td>
<td>Moment Constant</td>
<td></td>
<td>$0.015Ns^2/m^2$</td>
</tr>
<tr>
<td>$l$</td>
<td>Length</td>
<td></td>
<td>$0.141m$</td>
</tr>
</tbody>
</table>

Therefore the discrete time linear state space model is:

$$x[k + 1] = A_d x[k] + B_u U[k] + B_D W[k]$$

$$Y[k] = C x[k]$$

With the sampling time $T_s=0.01$.

$$A_d(12 \times 12) = \begin{bmatrix}
I_{3 \times 3} & 0.01 \times I_{3 \times 3} & 0_{3 \times 3} & 0_{3 \times 3} \\
0_{3 \times 3} & I_{3 \times 3} & 0.01 \times I_{3 \times 3} & 0_{3 \times 3} \\
0_{3 \times 3} & (-0.098 \ 0 \ 0) & I_{3 \times 3} & 0.01 \times I_{3 \times 3} \\
0_{3 \times 3} & 0_{3 \times 3} & 0_{3 \times 3} & I_{3 \times 3}
\end{bmatrix}$$

(A1.1)
Firstly, the rank of the following equation was calculated, to ensure the unknown input can be reconstructed, hence, the rank should be full (rank = 3)

\[
\text{Rank} \begin{bmatrix} B_d \\ D_d \end{bmatrix} = 3
\]  

(A1.7)

The discrete time observer is described by

\[
\hat{x}[k + L] = E\hat{x}[k] + Fy[k:k + L]
\]  

(A1.8)
The smallest $L$ is found such that $\text{rank}(J^L) = \text{rank}(J^{L-1}) = 3$.

$$\text{rank} \begin{bmatrix} D_D & 0 & 0 \\ CB_D & D_D & 0 \\ CAB_D & CAB_D & D_D \end{bmatrix}_{j^2} - \text{rank} \begin{bmatrix} D_D & 0 \\ CB_D & D_D \end{bmatrix}_{j^1} = 3$$  \hspace{1cm} (A1.9)

Next is to calculate $\bar{N}$ so that equation (3.20) is satisfied. $\bar{N}$ is found whose rows form a basis of the left nullspace of $J^1$ which is given by

$$\bar{N} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$  \hspace{1cm} (A1.10)

Thus:

$$N = W \begin{bmatrix} I_p & 0 \\ 0 & \bar{N} \end{bmatrix} =$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{M_{1}}$$  \hspace{1cm} (A1.11)
Therefore:

\[ H = M_1 \times M_2 \]  \quad (A1.13)

W is chosen such that its last \( q = 3 \) rows are the left inverse of the matrix \( H \) and the remainder first \( 2n - q = 21 \) rows form a basis for the left nullspace of the matrix \( H \), is:

\[
W = \begin{bmatrix}
\text{left nullspace of } H \\
\text{left inverse of } H
\end{bmatrix}
\]
Therefore:

\[
N = W \times M_1 =
\]

(A1.15)
Next, we calculate

$$
\begin{bmatrix}
S_1 \\
S_2
\end{bmatrix} = N \begin{bmatrix}
C \\
CA_d \\
CA_d^2
\end{bmatrix}_{\sigma^2}
$$

\[\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.2 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.2 & 0 \\
112 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 22.4 & 0 & 0 \\
0 & 112 & 0 & 10.99 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 22.4 & 0 & 0 \\
0 & 0 & 112 & 0 & 0 & 0 & 0 & 0 & 0 & 22.4 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}
\]

(A1.16)

Where $S_2$ is the last 3 rows of the above matrix and $S_1$ is the remainder rows.

$$
S_1 = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.2 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.2 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.2 & 0
\end{bmatrix}
$$

(A1.17)
\[
S_2 = \begin{bmatrix}
112 & 0 & 0 & 0 & -10.99 & 22.4 & 0 & 0 & 0 & 0 \\
0 & 112 & 0 & 10.99 & 0 & 0 & 22.4 & 0 & 0 & 0 \\
0 & 0 & 112 & 0 & 0 & 0 & 0 & 0 & 22.4 & 0 \\
0 & 0 & 0 & 112 & 0 & 0 & 0 & 0 & 0 & 22.4 \\
\end{bmatrix}
\]

(A1.18)

Now, to find \( \hat{F}_1 \) such that \( E_1 = A - BS_2 - \hat{F}_1S_1 \) is shur.

\[
E_1 = A - BS_2 = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0.1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0.1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0.1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0.1 & 0 & 0 & 0 \\
-10 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & -10 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & -10 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}
\]

(A1.19)

Since, it is unstable, after placing the poles and \( \hat{F}_1 \) is obtained as:

\[
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0.33 & 0 & 0 & 0 & 0.3 & 0 & 0 \\
0 & 0 & 0 & 0.33 & 0 & 0 & 0 & 0.3 & 0 & 0 \\
0 & 0 & 0 & 0 & 0.33 & 0 & 0 & 0.3 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -10 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -10 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -10 & 0 & 0 & 0 \\
0 & 0 & 0 & -5 & 0 & 0 & 0 & 0 & 5 & 0 \\
0 & 0 & 0 & -5 & 0 & 0 & 0 & 0 & 5 & 0 \\
0 & 0 & 0 & 0 & -5 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -5 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}
\]

(A1.20)

Therefore after placing the poles as per equation (3.35) \( E = A - BS_2 - \hat{F}_1S_1 \) is

\[
E = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}
\]
And $F$ is

\[
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0.33 & 0 & 0 & 0 & 0 & 0 & 0.33 & 0 \\
0 & 0 & 0 & 0 & 0.33 & 0 & 0 & 0 & 0 & 0 & 0.33 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -10 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -10 & 0 & 0 & 0 \\
0 & 0 & 0 & -5 & 0 & 0 & 0 & 0 & 0 & 5 & 0 \\
0 & 0 & 0 & -5 & 0 & 0 & 0 & 0 & 0 & 0 & 5 \\
0 & 0 & 0 & 0 & -5 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}
\]

And $G$ is computed as

\[
G = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 11.2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 11.2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 11.2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}
\]

The unknown wind is extracted as:

\[
W[k] = G \left[ x[k+1] - Ax[k] - Bu[k] \right]
\]

Figure A1 shows the MATLAB/Simulink of the Linear UIO system
Appendix 2
Calculations for Nonlinear UIO

The nonlinear model is written in the form,
\[
\dot{x}(t) = Ax(t) + g_1(u(t), y(t)) + f_1(x(t)) + Dw(t) \\
y(t) = Cx(t)
\]  \hspace{1cm} \text{(A2.1)}

Where;
\[
A_{12 \times 12} = \begin{bmatrix}
0_{3 \times 3} & I_{3 \times 3} & 0_{3 \times 3} & 0_{3 \times 3} \\
0_{3 \times 3} & 0_{3 \times 3} & 0_{3 \times 3} & 0_{3 \times 3} \\
0_{3 \times 3} & 0_{3 \times 3} & 0_{3 \times 3} & \left(\begin{array}{c}
0 \\
0 \\
0 \\
0
\end{array}\right)
\end{bmatrix}
\]  \hspace{1cm} \text{(A2.2)}

\[
g_1(u(t), y(t)) = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
\left(\frac{\sin(\theta) \cos(\psi) + \sin(\phi) \cos(\theta) \sin(\psi)}{F_m}\right) & 0 & 0 & 0 \\
\left(\frac{\sin(\theta) \cos(\psi) - \sin(\phi) \cos(\theta) \sin(\psi)}{F_m}\right) & 0 & 0 & 0 \\
\cos(\phi) \cos(\theta) & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 1/l_{xx} & 0 & 0 \\
0 & 0 & 1/l_{yy} & 0 \\
0 & 0 & 0 & 1/l_{zz}
\end{bmatrix}
\]  \hspace{1cm} \text{(A2.3)}
\[
f_1(x) = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
\end{bmatrix}
\begin{bmatrix}
-g \\
\cos(\theta)p + \sin(\theta) \\
\tan(\phi) \sin(\theta)p + q - \tan(\phi) \cos(\theta)r \\
-\sin(\phi)p + \cos(\theta) \cos(\phi)r \\
-(l_{zz} - l_{yy}) q^r \\
\frac{l_{xx}}{(l_{xx} - l_{zz})} pr \\
\frac{l_{yy}}{(l_{yy} - l_{xx})} pq \\
\end{bmatrix}
\]  

(A2.4)

And the unknown wind gust matrix is

\[
Dw = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
1/m & 0 & 0 \\
0 & 1/m & 0 \\
0 & 0 & 1/m \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
\end{bmatrix}
\]  

(A2.5)

\[
C = [l_{9 \times 9} \quad 0_{9 \times 3}]
\]  

(A2.6)

Firstly \(U\) and \(V\) are calculated

\[
U = -D(CD)^+\]

(A2.7)
The Jacobian of \( A \) is:

\[
U = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]  \( \text{(A2.8)} \)

And

\[
V = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]  \( \text{(A2.9)} \)

The LMI is uses \( A(\alpha) = A + J_f \) and equations 5.25 and 5.26 to calculate \( P \) and \( S \) with upper boundaries and lower boundaries of \( p, q \) and \( r \).
\[
S = U = \begin{bmatrix}
-3.35 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -3.35 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 5.91 & 0 & 0 & 0 & 0 & 0 \\
-1.19 \times 10^6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1.19 \times 10^6 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 127.6 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 127.6 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1.79 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1.75 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1.79 \\
\end{bmatrix}
\]

(A2.11)

Once \( S \) and \( P \) are found

\[
\begin{align*}
N &= MA - LC \\
L &= K(I_p + CE) - MAE \\
M &= I_n + CE
\end{align*}
\]

(A2.12)

\[
N = \begin{bmatrix}
-82.8 & 0 & 0 & -5.7 & 0 & 0 & 0 & 0 & 0 \\
0 & -82.8 & 0 & 0 & -5.7 & 0 & 0 & 0 & 0 \\
0 & 0 & -82.8 & 0 & 0 & -10.8 & 0 & 0 & 0 \\
6.7 & 0 & 0 & -82.8 & 0 & 0 & 0 & 0 & 0 \\
0 & 6.7 & 0 & 0 & -82.8 & 0 & 0 & 0 & 0 \\
0 & 0 & 11.8 & 0 & 0 & -82.8 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -89 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -95.6 & 79.3 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -79.3 & -82.8 \\
0 & 0 & 0 & 0 & 0 & 0 & -127.5 & 17.94 \\
0 & 0 & 0 & 0 & 0 & 0 & -17.4 & -127.56 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -17.94 & 0 & 0 \\
\end{bmatrix}
\]

(A2.13)
\[ L = \begin{bmatrix} -6.8 & 0 & 0 & -2.35 & 0 & 0 & 0 & 0 \\ 0 & -6.8 & 0 & 0 & -2.35 & 0 & 0 & 0 \\ 0 & 0 & -1.29 & 0 & 0 & -2.35 & 0 & 0 \\ -9.87 & 0 & 0 & -1.19 \times 10^6 & 0 & 0 & 0 & 0 \\ 0 & -9.87 & 0 & 0 & -1.19 \times 10^6 & 0 & 0 & 0 \\ 0 & 0 & -9.87 & 0 & 0 & -1.19 \times 10^6 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1.14 \times 10^4 & -0.0369 \\ 0 & 0 & 0 & 0 & 0 & 0 & -0.0369 & 1.25 \times 10^4 & -79.3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1.02 \times 10^4 & 82.8 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1.64 \times 10^4 & -2.31 \times 10^4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2.24 \times 10^3 & 1.64 \times 10^4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2.31 \times 10^3 & 0 \end{bmatrix} \]  
(A2.14)

\[ M = \begin{bmatrix} -2.35 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -2.35 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -2.35 & 0 & 0 & 0 & 0 & 0 \\ -1.19 \times 10^6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1.19 \times 10^6 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1.19 \times 10^6 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 128.6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 128.6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 127.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -17.4 & -1.75 \\ 0 & 0 & 0 & 0 & 0 & 0 & -17.4 & -1.75 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \]  
(A2.15)

\[ [D^T D]^{-1} D^T = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1.3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \]  
(A2.16)

Hence

\[ w(k) = [D^T D]^{-1} D^T \left[ \hat{x}(k + 1) - A_d \hat{x}(k) - f(\hat{x}(k)) - g(u(k), y(k)) \right] \]  
(A2.17)
Figure A2 shows the MATLAB/Simulink of the nonlinear UIO system