FAULT TOLERANT CONTROL OF MULTI-ROTOR UNMANNED AERIAL VEHICLES USING SLIDING MODE BASED SCHEMES

PhD Thesis
Thesis submitted for the degree of
Doctor of Philosophy
at the University of Exeter

by

Ahmed Khattab
Structures, Dynamics and Control Research Group
College of Engineering, Mathematics and Physical Sciences,
University of Exeter

September 17, 2020

FAULT TOLERANT CONTROL OF MULTI-ROTOR UNMANNED AERIAL VEHICLES USING SLIDING MODE BASED SCHEMES

Submitted by Ahmed Khattab to the University of Exeter
as a thesis for the degree of
Doctor of Philosophy in Engineering
In September 17, 2020

Signature: ....

Abstract

FAULT TOLERANT CONTROL OF MULTI-ROTOR UNMANNED AERIAL VEHICLES USING SLIDING MODE BASED SCHEMES

Ahmed Khattab

This thesis investigates fault-tolerant control (FTC) for the specific application of small multirotor unmanned aerial vehicles (Unmanned Aerial Vehicle (UAV)s). The fault-tolerant controllers in this thesis are based on the combination of sliding mode control with control allocation where the control signals are distributed based on motors' health level. This alleviates the need to reconfigure the overall structure of the controllers. The thesis considered both the over actuated (sufficient redundancy) and under-actuated UAVs. Three multirotor UAVs have been considered in this thesis which includes a quadrotor (4 rotors), an Octocopter (8 rotors) and a spherical UAV. The non-linear mathematical models for each of the UAVs are presented. One of the main contributions of this thesis is the hardware implementation of the sliding mode Fault Tolerant Control (FTC) scheme on an open-source autopilot microcontroller called Pixhawk for a quadrotor UAV. The controller was developed in Simulink and implemented on the microcontroller using the Matlab/Simulink support packages. A gimbal- based test rig was developed and built to offer a safe test bed for testing control designs. Actual flight tests were done which showed sound responses during fault-free and faulty scenarios. This work represents one of successful implementation work of sliding mode FTC in the literature. Another key contribution of this thesis is the development of the mathematical model of a unique spherical UAV with highly redundant control inputs. The use of novel 8 flaps and 2 rotors configuration of the spherical UAV considered in this thesis provides a unique fault-tolerant capability, especially when combined with the sliding mode-based FTC scheme. A key development in the later chapters of the thesis considers fault-tolerant control strategy when no redundancy is available. Unlike many works which consider FTC on quadrotors in the literature (which can only handle faults), the proposed schemes in the later chapters also include cases when failures also occur converting the system to an under actuated system. In one chapter, a bespoke Linear Parameter Varying (LPV) based controller is developed for a reduced attitude dynamics system by exploiting non-standard equation of motions which relates to position acceleration and load factor dynamics. This is unique as compared to the typical Euler angle control (roll, pitch and yaw angle control). In the last chapter, a fault-tolerant control scheme which can handle both the over and under actuated system is presented. The scheme considers an octocopter and can be used in fault-free, faulty and failure conditions up to two remaining motors. The scheme exploits

the differential flatness property, another unique property of multirotor UAVs. This allows both inner loop and outer loop controller to be designed using sliding mode (as opposed to many sliding mode FTC in the literature, which only considers sliding mode for the inner loop control).

Acknowledgements

Praise be to Allah for his graces and merits that filled the earth and the sky, and I ask him to grant me sincerity and acceptance in my work.

I would like to sincerely acknowledge and express my gratitude to my advisor, Dr Halim Alwi for his heartfelt support and guidance and the utmost patience. Without his able guidance, this thesis would not have been possible and I shall forever be beholden to him for his assistance.

My father and My Mother, I am sorry I would not be able to really thank you. Without your unconditional support, your sincere prayers, your patience on being away from you all that time and your trust on me more than myself, I would be lost. I thank them for being there where I felt stumped, goading me to the good. Any appreciations in my life are own to them and would not have been possible without their unwavering and unselfish love and their support that is given to me at all times.

I would like to thank my wife for her continuous support, her selflessness and utmost care.

I would like to thank my sisters and their husbands for their eternal encourage and for great times.

Contents

List of Figures

List of Tables

Abbreviations and nomenclature

Abbreviations

AFTC Active Fault Tolerant Control. 17, 18, 23, 25

CA Control Allocation. 10, 25, 60, 79

ESC Electric Speed Controller. 74, 78

FDI Fault Detection and Isolation. 10, 13, 17–20, 23–25, 62, 91, 104, 112, 114, 156

FTC Fault Tolerant Control. i, ii, vii, 9–11, 13, 15, 17–20, 23–26, 44, 57, 59, 70, 78, 79, 100–103, 120, 131, 142, 153–156

GA Genetic Algorithm. 90

LPV Linear Parameter Varying. i, iv, 11, 20, 24, 101, 102, 108, 112, 117, 154

LQR Linear Quadratic Regulator Control. 10, 23, 24, 29, 30, 42, 50, 55, 62, 102, 121

LTI Linear Time Invariant. 85, 108

PFTC Passive Fault Tolerant Control. 17, 18, 23–25

PSP Pixhawk Support Package. iii, 11, 59, 68, 76, 77, 154

PWM Pulse Width Modulation. viii, 56, 57, 68, 74–77

SMC Sliding Mode Control. 10, 23, 25, 27, 117, 120, 131, 133, 137, 142, 155

UAV Unmanned Aerial Vehicle. i–iii, vii, ix, 1–11, 16–18, 23–26, 59, 68, 78–80, 82, 84, 89, 92, 100, 101, 103, 120, 153–158

Nomenclature

I_{xx}, I_{yy}, I_{zz} Inertias around body axes. 66, 158

Q LQR states weighting matrix. 30, 50, 55, 70

R_b^i direction cosine matrix. 106, 122, 159

T_h Total thrust magnitude of motors. 132, 134

W Effectiveness of the motors. 61, 91, 114, 127, 128, 131, 136

Ω rotor speed (RPM). 107

δ small positive scalar in the sigmoidal approximation. 33, 112

ℓ multirotor arm length. 112, 114, 123

ϕ, θ, ψ Euler Angles (roll, pitch, yaw angles). 67, 158

ρ sliding mode modulation gain. 33, 35, 36

ρ_{air} air density. 82

ϱ LPV varying parameter. 108

b rotor thrust coefficient. 73, 112, 114

d rotor torque coefficient. 73, 112, 114

g the gravitational acceleration. 81, 160

m_{kg} mass. 66, 73, 89, 103, 112, 122, 158, 160, 163

p, q, r Angular velocities in body axes(roll, pitch, yaw rates). 50, 158

$s(t)$ Sliding Surface Function (Switching Function). 31, 32, 39, 70, 88, 92

$u(t)$ control input. 32, 36, 63, 64, 74, 81, 88

u, v, w Linear velocities in body (x,y,z) axes. 81, 82, 89, 91, 158

x_e, y_e, z_e position in inertial/earth axes. 91

$\mathcal{L}, \mathcal{M}, \mathcal{N}$ moments in body (x,y,z) axes. 122, 158

Chapter 1

Introduction

In this chapter, the importance and the growth in using unmanned aerial vehicles will be discussed. The chapter will discuss the safety concerns of using unmanned aerial vehicles in the civil application and the need for fault tolerant control schemes. The chapter concludes by discussing the structure and the work done in this thesis.

1.1 Motivation

In the last decades, (UAVs) or typically called 'drone' had gained a highly important role in both military and civil applications. UAV could go for missions that are considered too dull, dirty or dangerous for manned aircraft. UAV could cross the pilot safety and physical limits in high manoeuvres or places with a dangerous environment such as nuclear power plant or polluted environments such as chemical pollution or forest fires [1]. Figure 1.1 shows an excellent example where a small UAV is used to transfer live TV for an active volcano. UAVs also have certain benefit as compared to large aircraft, for example, UAVs are typically lightweight, cheaper and more simple. Hence, there has been a surge in the use of UAV for many civil and commercial applications.

It can be seen by various studies, for example in [2] as shown in Figure 1.2 that the use of UAV in civil and commercial applications have seen a surge in the last few years. Recently, large companies such as Amazon, DHL have started to consider UAV for package delivery [5, 6](see for example Figure 1.6), while Facebook is considering (High Altitude Long Endurance) HALE UAV (see for example Figure 1.7) to provide wide coverage of internet connections especially in rural areas [7].

The fact that these well-established companies have seriously started to look at UAV, indicate its importance in civil and commercial applications. The UAV market for civil applications is increasing every year and expected to increase more than the double in the coming few years [2], as seen in Figures 1.4. and 1.2. The importance of UAV for

Figure 1.1: A UAV flies over a volcano for live TV [1]

civil applications are also highlighted by the exponential increase in the number of patents related to UAV [3], as seen in Figure 1.3. UAV typically used for photography and film making, for both for leisure and professional use. For example, in the application of real estate, drone photography could provide a better overview of the property and the surrounding area [18], as seen in Figure 1.16.

However, the applications of UAV also extend beyond photography and film making. For example, in an emergency situation, medical aid can be delivered using UAV to places in crisis, quickly than a traditional ambulance. Figure 1.9 shows how UAV can be used as an air ambulance and to deliver blood and urgent medical supplies [8, 9, 10, 11]. UAV also can be used for search and rescue, natural disasters relief [13] (see Figure 1.11). Recently in Exeter, UAV has been used to monitor and coordinate fire and rescue services during the fire that engulfed the Royal Clarence Hotel in Exeter [14], as seen in Figure 1.12.

UAVs also have been used for civil aircraft visual inspection as demonstrated in Figure 1.13. In construction sites, as seen in Figure 1.14, aerial view using a small UAV could save time and cost, where UAV could monitor progress, conduct site surveys, provide precise elevation data, contour lines, and fast 3D modelling [16]. For Oil and gas production, continuous inspection of pipelines and gas could be easily done using UAV. The UAV can also be used for the geophysical survey of oil and gas where measurements of the varying magnetic field strength of the earth are used to calculate the nature of the magnetic rock structure. This could help to predict the location of mineral deposits [17]. For pipelines, routine thermal images using UAV could observe the thermal heat capacity of the ground around the pipeline just after sunset or after sunrise as seen in Figure 1.15. This could detect the leakage in pipelines joints accurately and is cost effective [17].

1.2 Fault tolerant control motivation for multirotor UAVs

The various applications of UAV described in the previous sections show a list of non-exhaustive examples of applications for UAV in civil and commercial applications. One of

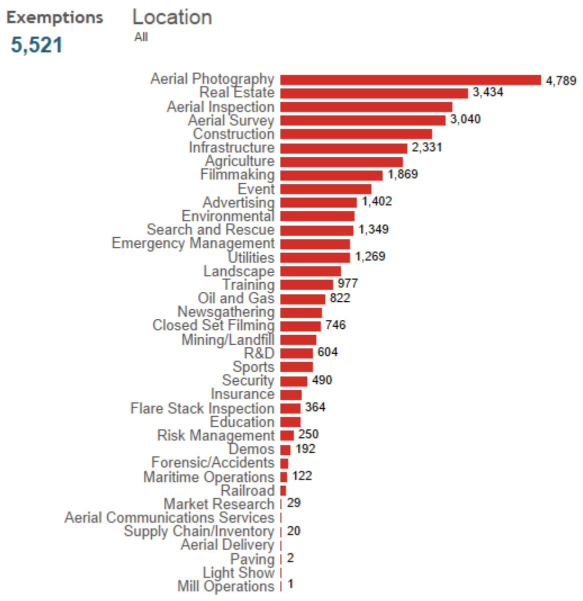


Figure 1.2: U.S. commercial UAV market by application, 2012-2022, (USD Million) [2]

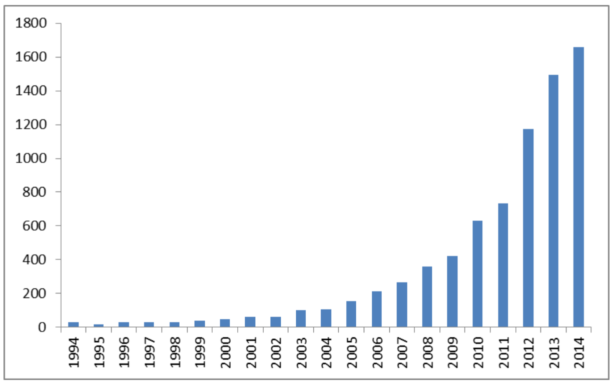


Figure 1.3: Evolution of patent publication on UAVs [3]

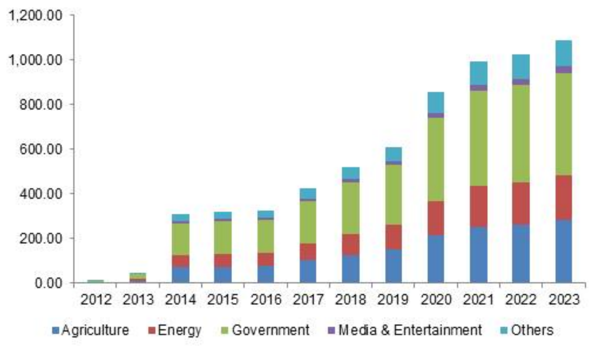


Figure 1.4: US commercial UAV market size, by application, 2012-2023 (USD Million). Published Date: March 2016 [2]

Figure 1.5: UAV civil applications: Agriculture [4]

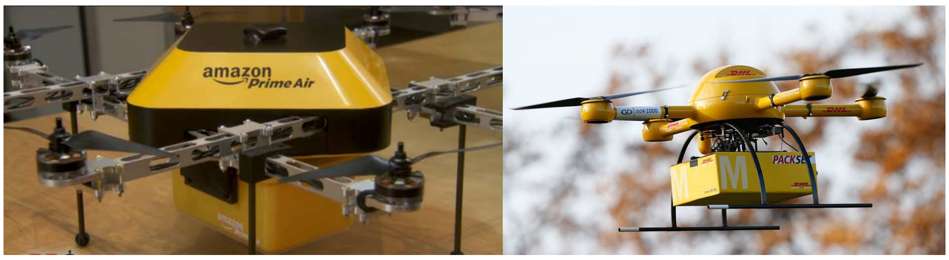


Figure 1.6: UAV civil applications: Cargo transport and delivery [5, 6]

Figure 1.7: UAV civil applications: Free Internet Facebook Tailless aircraft [7]

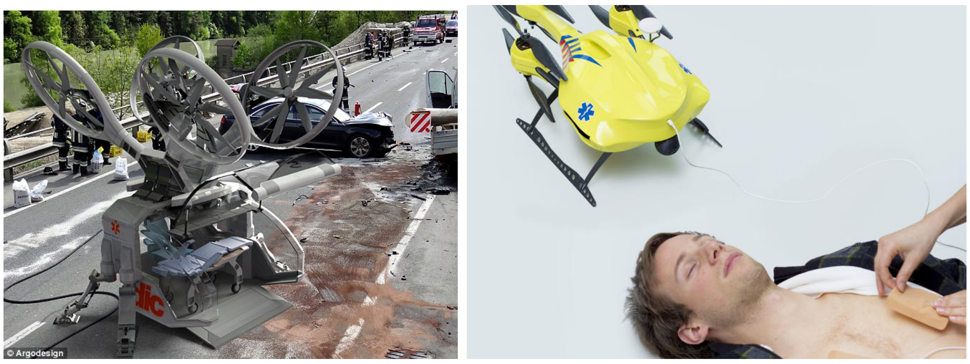


Figure 1.8: UAV civil applications: Air Ambulance [8, 9]

Figure 1.9: UAV civil applications: Blood and medical supplies delivery [10, 11]

Figure 1.10: UAV civil applications: Self-flying taxi [12]

Figure 1.11: UAV civil applications: Search and Rescue [13]

Figure 1.12: Police UAV footage shows the extent devastating fire that ripped through the UK 'oldest' hotel in Exeter [14]

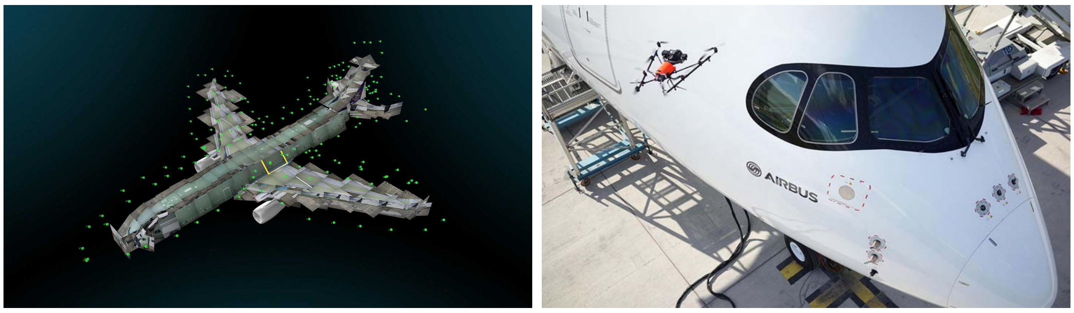


Figure 1.13: UAV civil applications: Airplane Visual Inspection [15]

Figure 1.14: UAV civil applications: Construction Site Survey [16]

Figure 1.15: UAV civil applications: Oil and Gas production [17]

Figure 1.16: UAV civil applications: Real Estate [18]

the recent exciting applications of UAVs is for autonomous self-flying passenger transport.

In a recent unveiling in Dubai (see Figure 1.10), a giant autonomous quadrotor with four arms and eight propellers for transporting passenger was proposed [12]. Airbus also envisaged a future of passenger transporting UAVs (VAHANA) to reflect the importance of the idea [43, 44]. Volocopter [45, 46] is a German company that recently did a two minute flight using an 18 propellers multirotor flying taxi in Singapore in collaboration with Intel. In New Zealand, a company called CORA [47, 44] introduced a 12 propeller self flying taxi. Surefly company based in Cincinnati introduced an 8 propellers multirotor aircraft [48, 49]. Vertical Aerospace based in Bristol, UK also introduced a quadrotor prototype named POC and then Seraph, a 12 propellers flying taxi [50]. In Japan, NEC also announced a successful 1 minute flight of its own flying car [51]. Uber also intended to enter this market collaborating with NASA [52]. Recently, Ucan Araba an octorotor single passenger aircraft is introduced by Cezeri-Baykar company in Turkey [53]. Other prominent companies also have joined the fray, highlighting the importance of the passenger based ‘flying taxi/transport’. See for example Aston Martin with its Volante Vision Concept aircraft [54] , Rolls-Royce [55], Embraer [56] and Opener single-seat BlackFly VTOL [57]. Recently, Airbus has announced a successful untethered test of its CityAirbus flying taxi [58].

With the increase of the applications of UAVs especially in civil and commercial applications, there is also pressure to ensure that the UAVs are *safe to operate, especially in the absence of pilots*. In the case of autonomous self-flying passenger transporting UAV taxi drone, the safety of the passengers is paramount. Despite low manufacturing and operating cost for most UAVs applications (e.g. for filming and photography), where UAVs can be easily replaced, It’s the safety of people on the ground and properties is important. Unfortunately, with the increase in UAV applications in civil and commercial use, there has been an increase in reported near-collision and injuries to the general public. For example, there have been various incidents reported in [19, 59, 60] where UAVs hit an aircraft, cars, bridge and building. Meanwhile as reported in [22], an athlete had a near miss and unfortunately in [21] where an athlete was injured after being struck by a crash landing UAVs.

The more serious case is the recently reported case in [20] when a toddler was hit by a drone and his right eye was sliced in half by the propeller after the operator lost control of the RC drone (see Figure 1.17). There has also been a reported case of death caused by UAV in [61, 62] (although indirectly caused by the UAVs, the fact that UAV has been used in the car chase and lead to the fatal accident). The serious implication to public safety is also highlighted by many cases of death directly related to RC helicopters. An

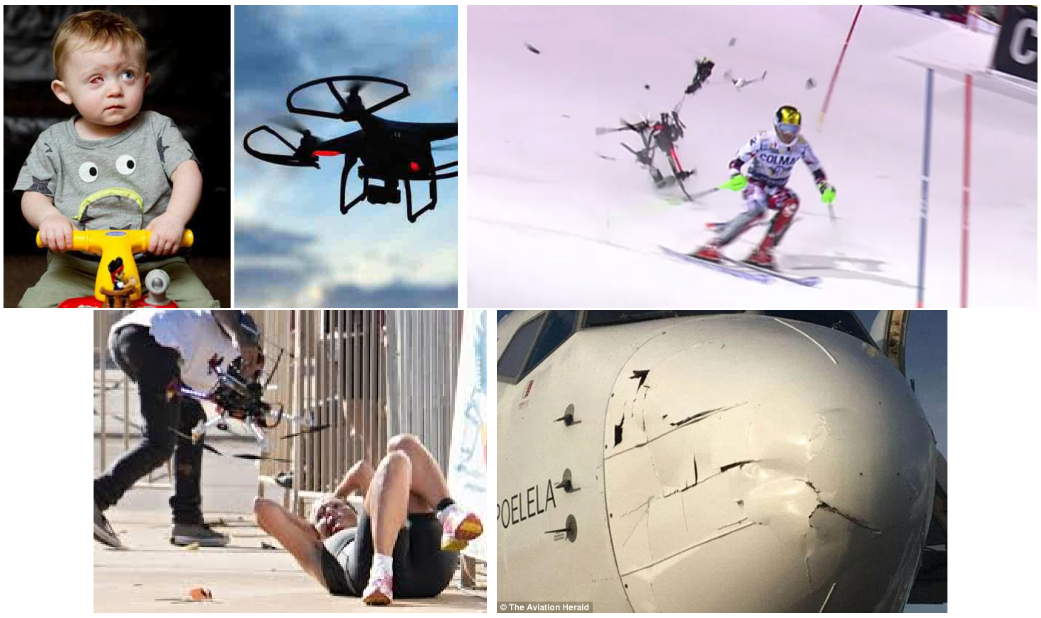


Figure 1.17: Some accidents related to UAV [19, 20, 21, 22]

Figure 1.18: Google Titan Solar UAV [23]

18 years old boy was killed by a small model helicopter that sliced his head in New York 2013 [63, 64]. A Swiss man in July 2013 was also killed in a similar way by his helicopter RC model. It was reported also in Brazil in 2008, South Korea in 2005 and Houston in 2003 all remote helicopter deaths [63].

In other cases, there have been reported incidents involving UAV with prominent companies such Google solar internet UAV and also the Facebook internet UAV [65], (Figure 1.18), to the extent that Google and Facebook confirmed the end of internet drone projects [66, 67]. Most of the incident involving UAVs is arguably due to the operator's shortcoming. But the serious implications to the general public and properties are undeniable.

1.3 Challenges of FTC for UAVs

Most common feedback control strategies do not have the ability to handle system faults/failures or any abnormal system conditions [68] *where the fault term refers to a partial loss in the effectiveness of an actuator and the failure term refers to the complete loss.* Scenarios such as sensor, actuator or component failures could lead to a dramatic degradation in system performance which could result in a catastrophic system collapse [69, 70]. Hence, control schemes that are tolerant to faults/failures without the need for complex and redundant hardware are beneficial.

In the case of large manned passenger aircraft, in the event of emergency situations, there have been many reported cases in the past where the action of the pilots have managed to land the aircraft safely [24, 71, 25] (see for example Figure 1.19 where pilots managed to land the aircraft safely on the Hudson river, while in Figure 1.20, the pilots managed to glide the safely to an emergency landing after ran out of fuel). However, in the absence of pilots in UAVs, there is a need for an automated safety feature that allows the UAVs to land safely. In the manned aircraft, the topic of fault tolerant control – control schemes that are tolerant to faults - has been investigated in the last few decades. However, its application on real aircraft has been scarce. In fact FTC for UAVs, especially multirotor UAVs is an emerging topic. *It is envisioned that fault tolerant control will be implemented on UAVs and will be a key feature, even before the manned aircraft counterpart due to pressing need to ensure safety in the absence of pilots in UAVs.* The next chapter will discuss further literatures on FTC for UAVs.

However, despite significant interest and research activities in the area of FTC, there is still a lack of published work describing the implementation of FTC schemes on real hardware. In recent years, a few researchers have started to use quadrotor unmanned aerial vehicles (UAVs) as a testing platform. This is partly due to the affordability of these UAVs, their small size and the relatively safe and controlled environment in which the tests can be conducted.

This is an aspect that the thesis trying to tackle, providing an open-source hardware rig that can be used by other researchers to implement state-of-the-art flight control. In the open literature, most of the work on multirotor UAV FTC is on the standard four-rotor UAVs (the quadrotor), this thesis will also consider FTC for a novel spherical UAVs with flaps which have not been considered in the literature.

It has to be noted however that most of the FTC implementation work on quadrotors, such as the work done in [72] (gain scheduled PID and LQR), [73] (model predictive control combined with horizon estimation and an unscented Kalman filter for parameter

Figure 1.19: Emergency Landing in Hudson River [24]

Figure 1.20: Emergency Landing after Fuel ran out [24]

estimation), [74, 75, 76, 77] (Sliding Mode Control (SMC)), [78] (nonlinear adaptive control), only considered faults and not total actuator failures due to the lack of redundant control surfaces. (The exception is the work in [79] and [80] (Linear Quadratic Regulator Control (LQR)), which considers total rotor failures, which render the quadrotor an under-actuated system). The references [81] (parametric programming control allocation), [82], [83] (SMC with Control Allocation (CA)), [84] (a bank of PID controllers with a nonlinear sliding mode observer Fault Detection and Isolation (FDI)), [85] (a pre-defined bank of control mixing laws with a nonlinear sliding mode observer FDI), represent notable recent work that considers octorotors and hexarotors for testing FTC. From a practical point of view, the availability of redundant actuators gives these UAVs the potential to handle total failures to certain actuators.

As mentioned earlier, a lot of FTC work on UAVs concentrated on quadrotor and therefore only consider fault (not total failure) due to the absence of redundancy where as discussed before the fault term refers to a partial loss in the effectiveness of an actuator and the failure term refers to the complete loss. This will be an area that the thesis will try to tackle, where an under actuated FTC case will be investigated which allow quadrotor to fail one or two rotors, but control is still possible.

1.4 Thesis structure and contributions

Chapter 2 discusses the topic of fault tolerant control (FTC) especially for UAVs. This chapter provides definitions of common terminologies used in the area of FTC, and also discusses the literature review and survey of FTC for aerospace systems (e.g. aircraft) and especially for UAVs.

The thesis concentrate on the development and applications of robust sliding mode control, where the main idea, revision and typical SMC synthesis is presented in Chapter 3.

This is followed by Chapter 4, where the robustness property of sliding mode control to handle a certain class of uncertainty (which include actuator faults) is combined with control allocation in order to provide automatic control distribution in the event of total actuator failure. Two examples are provided in this chapter. The first is based on a fixed wing aircraft and the other is based on a quadrotor UAV. Both examples provide design and simulation in Matlab and Simulink environment.

Chapter 5 presents a sliding mode control scheme which is combined with control allocation implemented on a multirotor UAV (Quadrotor). Hardware implementation results on IRIS+ Quadrotor using Pixhawk microcontroller via Simulink Pixhawk Support Package (PSP) is presented to show the capability for rapid and automatic build-and-deploy control algorithms. The designed controller is tested on the quadrotor for the purpose of tolerating motor faults. The tests conducted inside a pre-built gimbal designed to provide a safe indoor setup for testing quadrotors. A version of this chapter has appeared in [86, 32].

Chapter 6 presents an FTC scheme for an over actuated spherical UAV by combining sliding mode control and online control allocation. The non linear mathematical model of a spherical UAV is presented in this chapter. The simulation results include fault free and over actuated failure cases. The controller development in this chapter appears in [87]

Chapter 7 investigates the control synthesis of an under actuated mini quadrotor (68 grams), where a complete failure of one motor out of four motors are presented. A reduced attitude non linear mathematical system based on the unit vectors of body axes accelerations rather than Euler angles are used. The controller considers an LPV based sliding mode control which is utilised in both fault free and failure cases. The work in this chapter has been presented in [88].

Chapter 8 presents a unique FTC controller for an octorotor (eight rotors) UAV, which can handle not only the typical over actuated fault/failure scenarios, but also under actuated scenarios. The controller is be designed to tolerate faults and failures of the rotors up to the point where only two rotors remaining (six failed motors). The controller include integral action sliding mode controllers for both position (outer loop) and attitude (inner loop) control. Differential flatness properties of the system are exploited to obtain the appropriate state space models for the outer and inner loop systems and the transition between the two of them. Non linear feedback linearisation will be used to match the non linearities in the system. A version of the chapter is being prepared for submission to the ACC 2021.

Finally chapter 9 provides conclusions of the overall work done in the thesis and highlight the key contributions. This chapter also provides discussions of future work and possible area of research.

Chapter 2

Fault tolerant control and its applications in aerospace and multirotor systems

As highlighted in [25], during an emergency or when an aircraft start to behave abnormally, there are two important questions which may help the pilots: "What has gone wrong" and "what can the pilots do" [25]. The first question relates to the detection and identification of the source of the problem, which is an area of research known in the literature as "fault detection and isolation (FDI)" [25, 26]. The second question of what can be done to mitigate the problem relates to the research area of (FTC) [25]. Note that this thesis will concentrate on the topic on (FTC), although some aspect of the FDI might be investigated in the future works. In this chapter, the definitions for fault and failure and different types of actuator and sensor faults/failures will be discussed. This chapter will also provide general discussions on different FTC techniques from the literature, with emphasis on the FTC implementations on aerospace systems (both manned and unmanned aircraft) and especially for multirotor UAV.

2.1 Definition

The following definitions are based on the IFAC SAFEPROCESS technical committee [25] and will be used as the standard definitions used in this thesis:

Faults: an unpermitted deviation of at least one characteristic property or parameter of the system from the acceptable/usual/standard condition.

Failures: a permanent interruption of a system's ability to perform a required function under specified operating condition.

Note that there is a clear distinction between faults and failures. In other words, fault represents a reduction in the capability of an actuator but is still usable. Failure, however, represents a total failure of the actuator which is no longer being able to be used.

2.2 Types of sensor and actuator faults/failures

Figure 2.2 and 2.1 describe the typical faults and failures on the aircraft sensors and actuators. For the case of actuator fault/failure, some common problems could face the aircraft. In the lock-in-place failure, the actuator might be stuck and becomes immovable. This failure could happen due to lack of lubrication. In float failure, the actuator will move freely with no torque. Float failure may occur due to the loss of the hydraulic fluid. In the runaway/hard over failure, the actuator will be stuck at some maximum positions which are considered one of the most catastrophic failures and could happen due to a failure in an electronic component which will cause a large signal sent to the actuator [25].

For sensor fault/failure, Figure 2.2 shows typical bias, drift, freezing and loss of accuracy. Bias is an offset between the actual and measured signal and could happen due to external source affecting the reading or uncalibrated sensor. Drift is a continuous increase in measurements over time and might happen due to the integration of noisy readings or due to loss of sensitivity. Freezing happens when a sensor sends a constant wrong value. Sensors fault/failure were also categorised in terms of time as abrupt (quickly varying) or incipient (slowly varying).

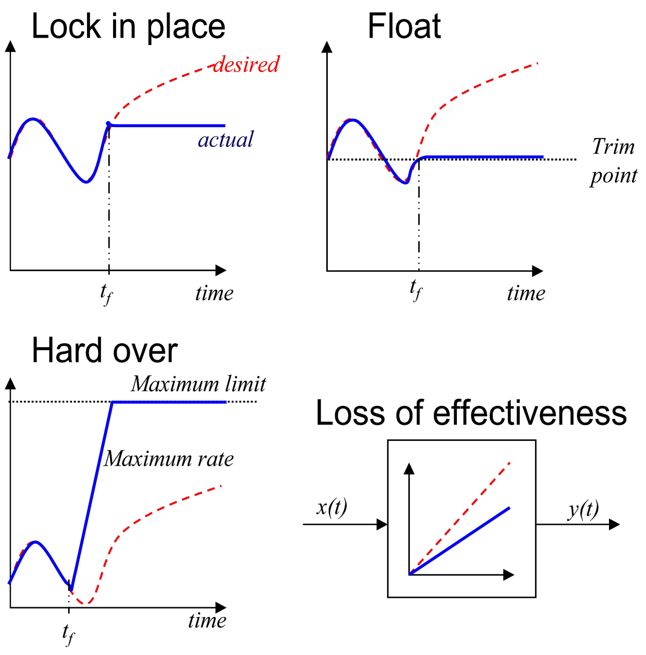


Figure 2.1: Main types of actuator faults and failures [25]

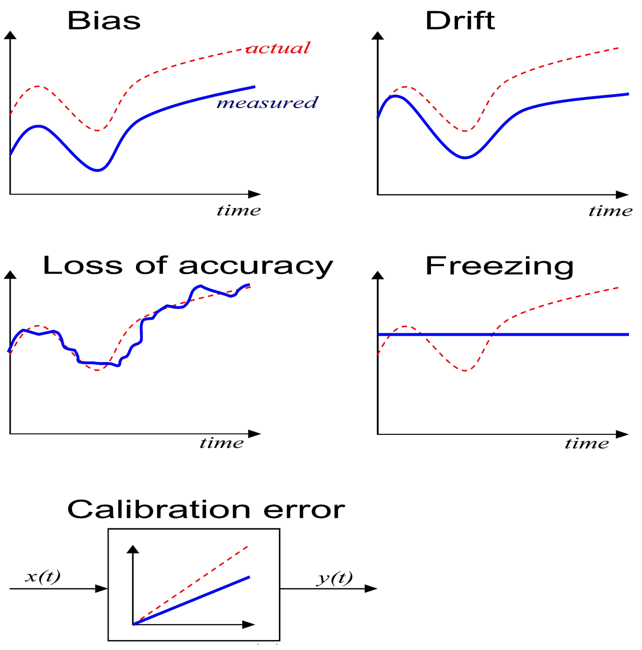


Figure 2.2: Main types of sensor faults and failures [25]

2.3 Fault tolerant control for civil aircraft

Most classical feedback control does not have the ability to handle system faults/failures or any abnormal conditions [68]. Cases such as sensors, actuators or component failures could lead to a dramatic degradation in the system performance which could result in a catastrophic system collapse [69, 70]. Hence, control schemes that are tolerant to faults/failures without the need for more complex and more redundant hardware is crucial. This is the main motivation for fault tolerant control schemes.

During the past few decades, FTC had gained popularity in the field of system control [68] due to its ability to deal with faults and/or failures [89, 69] that may occur in the system. The main objective of FTC is to mitigate the effect of faults and failures in the system before they turn into a seriously unstable condition [90].

In civil aircraft, fault tolerance could be achieved by using multiple redundant hardware [91]. For example, civil aircraft has triple redundancy i.e. they use three hydraulic actuators, three hydraulic lines and three hydraulic pumps for each control surface [25]. Sensors in large civil aircraft also typically rely on triple redundancy to measure the same state and a consolidated measurement will then be used in the flight control computer [25].

A good example of systems with redundancy is the large transport aircraft such as B747 [25] as seen in Figure 2.3, where roll angle could be controlled by ailerons, spoilers or engines. Pitch could be controlled by elevator, horizontal stabiliser or even flaps and engines. Typically stabilisers are used for trimming, but in critical situations, stabilisers could also be used for angles control. Finally, the rudder is typically used for yaw control, but in emergency situations, differential engine thrust can also be used.

Piaggio P 180 Avanti (see Figure 2.4) is another good example of redundancy that available in civil aircraft. Piaggio P 180 Avanti is a small business aircraft that has a roll, pitch and yaw redundancy in the presence of redundant hardware (elevator and canard for pitch control; aileron and spoilers for roll control and rudder and differential thrust for yaw control).

There are obvious benefits for considering redundant actuators for control, especially in an emergency situation. However, it has to be noted that redundancy in aircraft design will also mean an increase in weight, size and complexity and therefore more expensive to build and operate.

Figure 2.3: Boeing 747

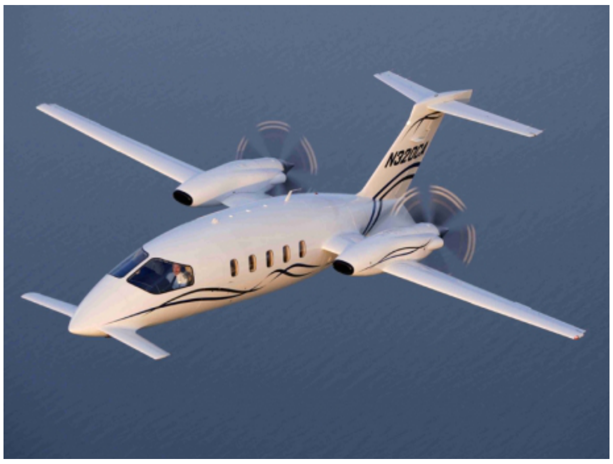


Figure 2.4: Piaggio P.180 Avanti aircraft

2.4 Fault tolerant control for UAV

The issues of multiple redundant hardware which affect large transport aircraft do not affect UAV significantly. This is due to the fact that UAVs has a low operating cost, lightweight and typically less complex and therefore has low manufacturing cost and easy to build. However, the need to use UAV as a reliable civil airspace service, as described in the previous chapter, require an assurance of safety and reliability and subsequently gaining the airworthiness-like approval criteria (although this seems is one of the aspects that the regulators are still working on).

Unfortunately, there have been cases where the general public safety has been affected by the use of UAVs. Figure 1.17 shown some cases where injury and damage that UAVs have caused to the general public without proper care and use. As discussed in the previous chapter, there is a serious implication on the reliability and safety of UAVs to the general public and properties.

The applications of the autonomous UAV for passenger taxi (see for example Dubai Air taxi and Figure 1.10) also represent a great importance for UAV to be designed to be tolerant to faults/failures, to ensure the safety of the passengers. For the case of package delivery, where UAV will have to fly in heavily populated areas, incidents/crash need to be avoided should faults/failures develop onboard during flight, especially in the absence of onboard pilots to take evasive action.

2.5 Types of fault tolerant control

As seen in Figure 2.5, FTC could be classified into two main categories: active and passive FTC [68, 26]. For Active Fault Tolerant Control (AFTC), faults/failures are primary detected and located using fault detection and isolation (FDI) unit. Subsequently, a control law is modified or adapted in order to handle the faults/failures. Therefore AFTC relies on accurate detection and isolation of faults/failures that present in the system. For Passive Fault Tolerant Control (PFTC), the controller is designed a priori to be robust against faults/failures (and also disturbances and uncertainties) without relying on any FDI unit [68, 25].

As described in [25, 26] and Figure 2.5, AFTC can be divided into two subgroups: projection and online controller redesign. For projection AFTC, the controller is pre-designed for all possible faults/failures that might occur in the system. The projected controller will only be activated when a specific predicted fault/failure occurs. For the online controller redesign, fault tolerance can be achieved through controller reconfiguration, either through control adaptation or control allocation. For adaptive control, the control parameters will be adjusted to get the desired performance and this may be done with or without estimating new parameters for the system. For control allocation technique, the control signal will be redistributed to the remaining healthy actuators.

Passive fault tolerant control schemes, on the other hand, are typically designed based on robust control strategy and designed to be tolerant against faults/failures. This is achieved without requiring FDI or reconfiguration, thus the term 'passive'. The controller is designed to be robust to faults/failures and therefore will try to maintain the same level of desired performance, as in the fault-free case, despite the presence of disparities and uncertainties that may be present due to faults/failures. In the last few decades, Sliding Mode Control has gained significant attention in the field of fault tolerant control. This is due to the inherent robustness properties of sliding modes to a certain class of uncertainty (classified as 'matched' uncertainty [25]). These robustness properties include its ability to directly handle actuator faults without requiring the fault to be detected and without requiring controller reconfiguration [25]. Due to this capability, sliding mode control is typically classified as PFTC. However, it has to be noted, sliding mode control can also utilise information provided by the FDI and classified as AFTC. Information provided by the FDI will allow for a less strict requirement for the control law and therefore (theoretically speaking) should be able to achieve better performance and more aggressive as compared to the PFTC counterpart.

Details of sliding mode design and analysis of its robustness properties will be discussed in detail in the next chapter.

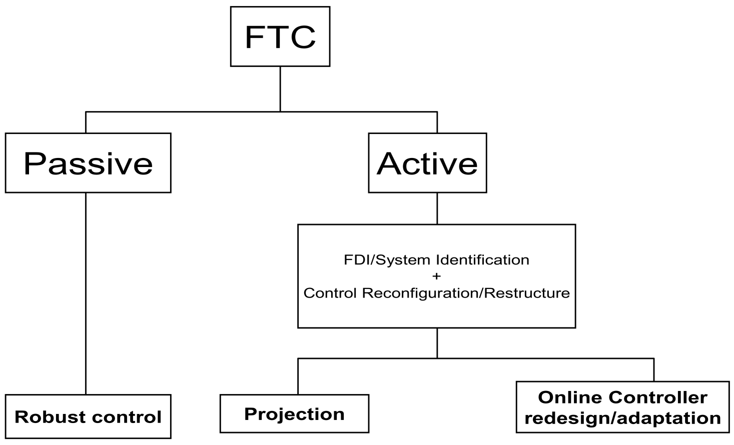

FTC
Passive
Active
FDI/System Identification
+
Control Reconfiguration/Restructure
Robust control
Projection
Online Controller redesign/adaptation

Figure 2.5: Classification of FTC [25, 26]

2.6 Fault tolerant control literature review

In this section, a brief introduction to some of the prominent work that has been done in the area of FTC in literature, for both large manned aircraft and small UAV is presented. This section will also cover FTC literature that considers hardware implementation. This is an area which is currently beginning to receive a lot of attention, partly due to the affordability of building and operating UAV and relative safety of flight testing in a controlled environment.

2.6.1 FTC hardware implementation: civil and fighter aircraft

Propulsion controlled aircraft (PCA) [27] was a project at NASA Dryden flight research centre where a control system was designed to use only engine thrust to control the aircraft. The concept was to control the pitch angle by using collective thrust (increasing the thrust to climb and decrease thrust to descend), while for heading and roll angle control by using left and right engine differentially. The PCA was implemented and flight-tested on a McDonnell-Douglas MD-11 transport aircraft (Figure 2.6) and F-15 fighter aircraft.

Another project at NASA Dryden flight research centre called the Intelligent Flight Control System (IFCS) [28] was designed to provide an online aerodynamic and stability parameters identification of aircraft during a fault or failure conditions. These identified parameters are then used to reconfigure the control to help the pilot safely land the aircraft. The parameter identification was done based on an artificial neural network. The flight test was done on a highly-modified McDonnell Douglas F-15B Eagle aircraft on Dec 6, 2002 (Figure 2.7).

In the integrated resilient aircraft control project (IRAC) [29], a modified F-18 fighter aircraft (see Figure 2.8), was used for testing FTC schemes based on adaptive control to enable safe flight in the presence of structural damage, control surface failure or changes in aerodynamic characteristics.

In Europe, the work done in GERTEUR FM-AG16 project represents one of the earliest collective works on the topic of FTC. The project consists of a consortium of partners from industry, research institutes and universities in Europe. The project aimed to investigate state-of-the-art fault tolerant control strategy for aerospace applications. One of the failure scenarios considered in the project is based on an actual Boeing 747 EL-AL 1862 incident that happens the 1990s. The benchmark was developed as a Matlab/Simulink platform to be used for real-time evaluation of state-of-the-art fault tolerant control techniques. The high fidelity benchmark model represents a full nonlinear simulation of the Boeing 747 aircraft, which include a realistic actuator and engine models and failure modes. The outcome of the project is well documented in [92]. The book showcased different techniques used for FDI and FTC such as Model Predictive Control (MPC), Nonlinear Dynamic Inversion (NDI) and sliding mode control.

The project called ADDSAFE (Advanced Fault Diagnosis for Sustainable Flight Guidance and Control) is another EU funded project which focussed on fault detection and diagnosis (FDD). The project's objective was to investigate and improve FDD design and analysis for aircraft guidance and control that could be used for optimising hardware redundancy and improve aircraft development and maintenance cost [31]. The three-year project started

Figure 2.6: McDonnell Douglas MD-11: Landing under engine power only [27]

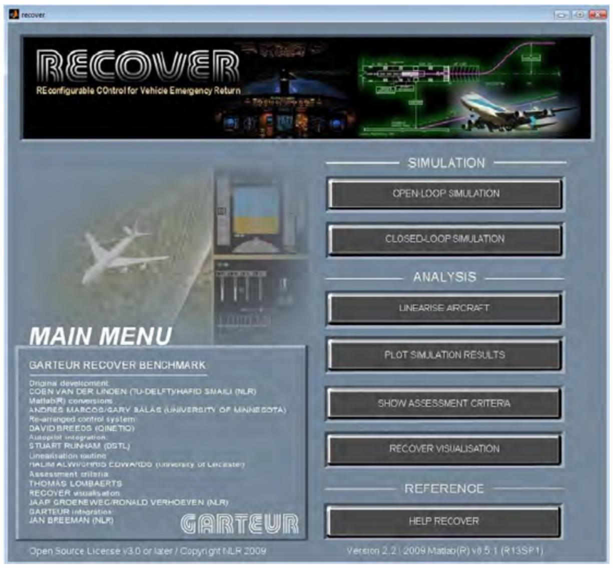
in 2009 and was co-funded by the European Community's Program for International Cooperation under the 7th Framework Program. The project consists of a consortium of academic and industry which involved partners from eight partners from six European countries (Figure 2.10). The output from ADDSAFE includes various publications in the area of sliding mode schemes, polytopic LPV approach and optimisation based techniques (see for examples [93, 94, 95, 96, 97]).

Complementing and continuing the work from ADDSAFE project, the RECONFIGURE (Reconfiguration of Control in Flight for Integral Global Upset Recovery) is another EU funded project in the area of fault tolerant control which runs from 2013-2016. As described in [98], the project was focused on investigation and development of aircraft guidance and control techniques to handle abnormal situations. The used control aimed to reconfigure the aircraft to its optimal flight condition while maintaining the aircraft safety levels. This project considers both FTC and FDI schemes rigorously evaluated on an industrial Functional Engineering Simulator [99, 98], to access the performance of proposed schemes when tested on different types of sensor and actuator failures. Various FTC and FDI schemes have been considered, which includes LPV based approaches, sliding mode schemes, model predictive control.

A recent (2016-present) project called Validation of integrated safety-enhanced intelligent fight control project (VISION) [100] is a Europe-Japan collaborative research project. One of the objectives of the project is to implement FDD and FTC schemes on Japan's JAXA MuPAL–alpha, a manned fly-by-wire experimental aircraft (Figure 2.11) and USOL K-50 UAV in Europe (Figure 2.12). The project also deals with both sensor and actuators failures.

Figure 2.7: NASA Dryden's highly modified F-15B: IFCS project [28]

Figure 2.8: Modified F/A-18A fighter aircraft: Integrated Resilient Aircraft Control project [29]



GARTEUR RECOVER BENCHMARK

Original development:
COEN VAN DER LINDEN (TU-DELFT)/HAFID SMAILI (NLR)
Matlab(R) conversions:
ANDRES MARCOS/GARY BALAS (UNIVERSITY OF MINNESOTA)
Re-arranged control system:
DAVID BREEDS (QINETIQ)
Autopilot integration:
STUART RUNHAM (DSTL)
Linearisation routine:
HALIM ALWI/CHRIS EDWARDS (University of Leicester)
Assessment criteria:
THOMAS LOMBAERTS
RECOVER visualisation:
JAAP GROENEWEG/RONALD VERHOEVEN (NLR)
GARTEUR integration:
JAN BREEMAN (NLR)

Open Source License v3.0 or later / Copyright NLR 2009

Version 2.2 | 2009 Matlab(R) v8.5.1 (R13SP1)

Figure 2.9: GARTEUR RECOVER Benchmark graphical user interface [30]

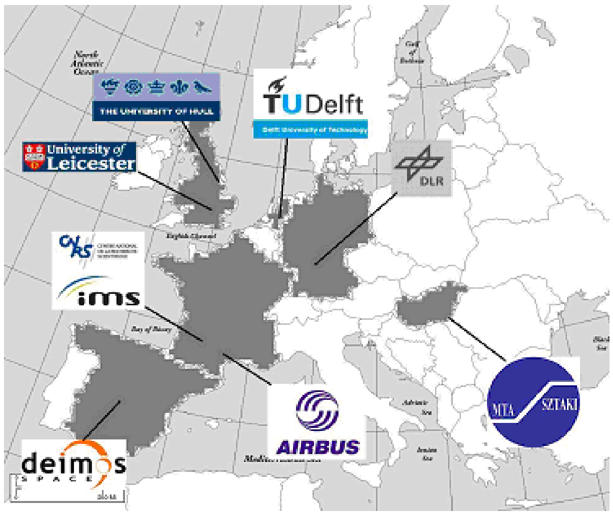


Figure 2.10: ADDSAFE: geographical distribution of partners [31]

Figure 2.11: JAXA MuPAL-alpha experimental airplane [32]

Figure 2.12: USOL K-50 unmanned airplane [33]

2.6.2 FTC hardware implementation: Multirotor UAV

2.6.2.1 FTC on quadrotor

Despite various works on FTC for large civil aircraft, there is a limited number of work that went on to be implemented on a real aircraft (a notable exception is the NASA propulsion controlled aircraft described earlier). This is partly due to cost and safety issues related to testing FTC schemes on the large passenger aircraft. On the other hand, recently, there has been an increase in the number of work that implements FTC schemes on UAV, especially the multirotor systems. This is mainly due to the affordability to manufacture and operate UAV, as well as its small size allowing it to be tested in relative safety in a controlled laboratory environment.

In [72], gain scheduled PID was designed to match each fault situation and the appropriate gains were selected depending on the tracking error or actuator status. For implementation, a UAV called Qball-X4 developed by Quanser Inc. was used after being modified from quadrotor (4 motors) to hexacopter Qball-X6 (6 motors). The Qball UAV has an outer body protection carbon fibre spherical cage to avoid accidental damage to the propeller and safety to the operator. The controller design was built using Matlab/Simulink and uploaded onboard to Gumstix embedded computer running at 200 Hz which is accepted since it is much more than the dynamics of the aircraft. For an indoor evaluation, OptiTrack camera system from NaturalPoint was used for position tracking to replace the GPS. The work in [72] also made a comparison between the LQR and the conventional model reference adaptive control for a propeller damage of up to 16% loss of rotor effectiveness.

In [77], PFTC and AFTC were achieved experimentally based on sliding mode with a comparison between the two methods. In [73], an AFTC scheme for the nonlinear model of the quadrotor was considered where model predictive control was combined with horizon estimation and unscented Kalman filter for parameter estimation. The work also considers actuator saturation although it requires an accurate model for the free fault system to calculate the trim control signals. In [72], model reference adaptive control was tested as a PFTC without explicit FDI i.e. no information of the fault was required. The scheme was tested in the presence of partial damage to one propeller and loss of effectiveness in the total thrust.

AFTC was used in [74] and [75] while in [76] SMC was used for PFTC. A comparison between the two methods in [76], showed that a similar level of performance can be achieved by carefully tuning the passive control or using cascade SMC.

It has to be noted that most of the FTC implementation work on quadrotor described above only considers faults and not total actuator failures. The work in [79] and [80], however, represents a limited number of works that considers total failures on a rotor for a quadrotor, which constitute an under-actuated system. A typical control strategy cannot deal with total failure in the event of total failure of one of the rotors. In that paper, the UAV was allowed to have a free rotation around the yaw (vertical) axis while the remaining healthy rotors are used to maintain control roll and pitch angles. The controller considered was based on a linear quadratic regulator (LQR) and the implementation results on a quadrotor showed a successful safe landing when a failure occurs on one, two or even three rotors.

2.6.2.2 Multirotor with redundancy (i.e. octocopter or hexarotor) for FTC

The work described in this section considers FTC which considers highly redundant multirotor UAV. From a practical point of view, the availability of redundant actuators allows the UAVs the potential to handle total failures to certain actuators.

Despite many works on FTC for the multirotor system, most of the work in the literature is conducted on a quadrotor and therefore only deal with actuator fault. The work by [83, 85, 82, 74], are the notable recent works that consider octocopter for testing FTC.

Although not implemented or tested on any hardware, the work in [83] represents one of the earliest known works that considered octocopter for testing sliding mode FTC. The scheme considers the combinations of sliding mode and control allocations that have previously been simulated and tested on large civil aircraft. In order to deal with large variations of flight conditions, the proposed design considers an LPV based schemes. The results show various cases of simultaneous rotor failure cases with no visible degradation of tracking performance.

The work in [84] and [85] considers an FTC on an octocopter UAV in a co-axial configuration. The FDI is based on a nonlinear sliding mode observer to provide the information to a PID controller. A bank of PID controllers was designed offline, for each fault case and the gains are set up as a look-up table. The implementation was done on an octocopter with BLCTRLV2 controllers (Mikrokopter). In the event when a rotor failed, the observer identified the failed rotor and the controller is reconfigured to stop sending control signals to the failed rotor and it's dual. The control signals are then reallocated for the remaining six motors.

In [78], considers nonlinear adaptive estimators to detect and isolate faults in IMU. The schemes also have the capability to estimate the (bias) fault. A PFTC is designed using nonlinear adaptive fault tolerance controller, where an accepted performance was achieved in the presence of faults.

In [81], a hexacopter was considered for an experiment with a Vicon visual motion tracking system. The FTC depended on parametric programming control allocator to automatically redistribute control signals in the event of faults/failures. When faults/failures occur, the FTC controller has the capability to take about one second for detection, isolation and controller reconfiguration, without affecting the stability of the system.

2.7 Sliding mode FTC

The (SMC) has gained significant attention in the field of FTC. This is due to the inherent robustness properties of sliding modes to a certain class of uncertainty (classified as 'matched' uncertainty [25]). These robustness properties include an ability to directly handle actuator faults without requiring the fault to be detected and without requiring controller reconfiguration [25].

Due to this capability, sliding mode control is typically classified as a form of passive FTC. However, it has to be noted, sliding mode control can also utilise information provided by the FDI unit which would be classified as active fault tolerant control (AFTC). Information provided by the FDI allows for a less strict requirement in terms of the control law and therefore (theoretically speaking) should be able to achieve better performance and a more aggressive level of performance as compared to its passive fault tolerant control (PFTC) counterpart.

Actuator faults, such as 'loss of effectiveness' can be categorised as matched uncertainty and 'classical' sliding mode schemes can deal with it directly without any control design modification providing sliding can be maintained. However, in the event of total actuator failure (and provided redundancy is available), classical sliding mode techniques cannot deal with the failure and hence other fault tolerant techniques have to be considered. In fact, other 'typical' control schemes will also not be able to deal with total actuator failure. However, the combination of sliding modes with a technique called 'control allocation' (CA), as proposed in [25], can deal with this problem without the need to reconfigure the structure of the controller. This will be discussed in the subsequent chapters.

2.8 Summary

This chapter begins with basic terminology and definitions on faults and failures and described typical actuator and sensor faults/failures. This chapter then described the rapid growth of UAV and its civil and commercial applications in the last decades. Being a relatively young industry with rapid growth, there also has been an increase in cases of incident and accidents that involved UAVs. Therefore, there is a great need for technology to ensure safe operations of UAVs. The ideas for fault tolerant control that have been studied in the same period for large transport aircraft can also be applied for UAV, and this has been discussed in this chapter.

This chapter also contains discussions of previous works on FTC for civil aircraft as well as small UAVs, especially the work that was done with actual implementation. It has to be noted that despite work on FTC, there is a scarce of literature on implementation of FTC especially on an actual aircraft (notable exceptions are the work described in Section 2.6.1). These are mainly due to safety and cost factors. It is envisaged that the FTC will be implemented and widely used in UAV application much earlier than in larger civil aircraft. The use of multirotor UAV alleviate the cost issues (as it is affordable and available off the shelf components) and safety restrictions (the test can be done in an enclosed area under a controlled environment) as compared to the manned aircraft. In the event when faults/failures occur, the absence of pilots on board the UAVs and in an autonomous application make the requirement for an FTC even more crucial in order to allow for a safe landing.

Sliding mode scheme will be considered due to the inherent robustness properties to a certain class of uncertainties. When combined with control allocation, the sliding mode schemes provide an excellent FTC strategy that can deal with actuator faults as well as total failures, without loss of performance. These will be one of the facets of the work in this project: to implement state-of-the-art sliding mode control on multirotor UAV and to show the applicability of sliding mode in real hardware. In the next chapter, the concept of sliding mode and its properties (especially the robustness to the so-called 'matched' uncertainty) will be discussed as a precursor to the sliding mode FTC based techniques in the later chapters.

Chapter 3

Sliding mode control

In this chapter, the concept of sliding mode control (SMC) and its properties will be introduced. First, the state space regular form that is compatible with SMC will be discussed, followed by the description of reduced order sliding motion. Then the design of 'sliding surface' and how it influences the closed loop performance will be described. Finally the design of the control law that ensures that the trajectory of the system reach the sliding surface and remain on it will be described. The properties of the sliding mode control especially in terms of robustness to the 'matched uncertainties' will also be discussed in the chapter. Finally, tracking using integral action approach will be discussed with a numerical example of an attitude tracking of longitudinal dynamics of a small (remotely controlled) fixed wing aircraft.

3.1 Regular form

Consider a linear continuous state space model

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{3.1}$$

where $A \in \mathbb{R}^{n \times n}, B \in \mathbb{R}^{n \times m}$ with $1 \leq m \leq n$ and assume that the pair (A, B) is controllable [101]. The system in (3.1) can be transformed into a control canonical form using $T_r \in \mathbb{R}^{n \times n}$ such that

$$T_r B = \begin{bmatrix} 0 \\ B_2 \end{bmatrix} \tag{3.2}$$

where $B_2 \in \mathbb{R}^{m \times m}$ and is non singular, and no need for A matrix to be in the control canonical form. After the coordinate transformation $x \to T_r x$, the states can be partitioned as

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{3.3}$$

where $x_1 \in R^{n-m}$ and $x_2 \in R^m$, so that (in the new coordinates), we will have [101]

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ B_2 \end{bmatrix} u \tag{3.4}$$

The benefit of this transformation is that we could now build a separate two system of equations one of them will be dominant in describing the system before reaching the sliding surface and the other one will describe the system more after reaching the sliding surface [102]. The above system could be operated to these two equations:

$$\dot{x}_1 = \begin{bmatrix} A_{11} & A_{12} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{3.5}$$

$$\dot{x}_2 = \begin{bmatrix} A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + [B_2]\, u \tag{3.6}$$

Assuming now a general linear combination of states to be

$$s(t) = Sx(t) \tag{3.7}$$

where $S \in \mathbb{R}^{m \times n}$ is full rank and S is a matrix could be partitioned into

$$S = \begin{bmatrix} S_1 & S_2 \end{bmatrix} \tag{3.8}$$

where $S_1 \in \mathbb{R}^{m \times (n-m)}$ and $S_2 \in \mathbb{R}^{m \times m}$. During sliding, for all $t > t_s$ where t_s is the time when sliding occurs,

$$s(t) = 0 \tag{3.9}$$

From (3.7),

$$s(t) = Sx(t) = S_1 x_1(t) + S_2 x_2(t) = 0 \tag{3.10}$$

and therefore

$$x_2(t) = -S_2^{-1} S_1 x_1(t) \tag{3.11}$$

Let $M = S_2^{-1} S_1$, yields

$$x_2(t) = -M x_1(t) \tag{3.12}$$

Matrix S can be obtained as [101]

$$S = \begin{bmatrix} S_2M & S_2 \end{bmatrix} \tag{3.13}$$

Notice S_2 has no direct effect on the dynamics of the sliding motion. The matrix S_2 acts only as a scaling factor for the switching function and could be chosen arbitrarily as $S_2 = I_m$. Substituting $x_2(t)$ from (3.12) into (3.5), it yields the reduced order system [101]

$$\dot{x}_1(t) = (A_{11} - A_{12}M)x_1(t) \tag{3.14}$$

Equation (3.14) represents a special case of the original system in equation (3.1). This special case exists only when the system is on the sliding surface. Choosing the sliding surface is only done by choosing the value of M. The matrix $(A_{11} - A_{12}M)$ must be designed to meet stability criteria by any means of control techniques by choosing the appropriate M matrix and hence the sliding surface.

3.2 Design of the sliding surface

In this section, the pole placement method and LQR will be used to design the M matrix and therefore the sliding surface matrix S as discussed in [101].

3.2.1 Pole placement

From equation (3.14),

$$(sI_{l\times l} - A_{11} + A_{12}M)x_1 = 0 \tag{3.15}$$

where s here is the Laplace symbol and $l = n - m$. For the reduced order model to be stable, the poles needed to be stable hence consider

$$sI_{l\times l} - \lambda = 0 \tag{3.16}$$

where $\lambda \in \mathbb{R}^{l\times l}$ is a symmetric positive matrix containing the required stable poles. Hence by comparing the coefficients of both equations (3.15) and (3.16), the matrix M could be obtained .

3.2.2 Linear quadratic regulator

Consider the problem of minimizing the cost function based on LQR is

$$J = \frac{1}{2}\int_{t_s}^{\infty} (x(t)^T Q x(t))dt \tag{3.17}$$

where Q is a symmetric positive definite matrix and t_s is the starting time of sliding. In regular form, the matrix Q could be written as

$$T_r Q T_r^T = \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{bmatrix} \tag{3.18}$$

where $Q_{12} = Q_{21}^T$. Hence

$$J = \frac{1}{2}\int_{t_s}^{\infty} (x_1^T Q_{11} x_1 + 2x_1^T Q_{21}^T x_2 + x_2^T Q_{22} x_2)dt \tag{3.19}$$

Using Utkin and Young method [103] to factorize the last two terms of equation (3.19) and using the fact that $Q_{22} \in \mathbb{R}^{m\times m}$ and is a symmetric matrix yields

$$\begin{aligned} 2x_1^T Q_{21}^T x_2 + x_2^T Q_{22} x_2 &= (x_2 + Q_{22}^{-1} Q_{21} x_1)^T Q_{22} (x_2 + Q_{22}^{-1} Q_{21} x_1) + \\ &\quad - x_1^T (Q_{21}^T Q_{22}^{-1} Q_{21}) x_1 \end{aligned} \tag{3.20}$$

Define

$$\nu = x_2 + Q_{22}^{-1} Q_{21} x_1 \tag{3.21}$$

and

$$\hat{Q} = Q_{11} - Q_{21}^T Q_{22}^{-1} Q_{21} \tag{3.22}$$

After some algebraic manipulations, equation (3.19) becomes

$$J = \frac{1}{2}\int_{t_s}^{\infty} (x_1^T \hat{Q} x_1 + \nu^T Q_{22} \nu)dt \tag{3.23}$$

And from the reduced order equation in (3.5)

$$\dot{x}_1 = A_{11} x_1 + A_{12} x_2 \tag{3.24}$$

and by substituting equation (3.21) into equation (3.24) then

$$\dot{x}_1 = \hat{A}_{11} x_1 + A_{12} \nu \tag{3.25}$$

where $\hat{A}_{11} = (A_{11} - A_{12}Q_{22}^{-1}Q_{21})$. Now optimal control law could be applied and

$$\nu(t) = -(Q_{22}^{-1}A_{12}^{T}P_1)x_1(t) \tag{3.26}$$

where from algebraic Riccati equation, P_1 satisfies

$$\hat{A}_{11}^{T}P_1 + P_1\hat{A}_{11} - P_1A_{12}Q_{22}^{-1}A_{12}^{T}P_1 + \hat{Q} = 0 \tag{3.27}$$

Then from equations (3.21) and (3.26)

$$x_2 = -Q_{22}^{-1}(A_{12}^{T}P_1 + Q_{21})x_1(t) \tag{3.28}$$

and from the sliding surface equation in (3.12)

$$x_2(t) = -Mx_1(t) \tag{3.29}$$

then

$$M = Q_{22}^{-1}(A_{12}^{T}P_1 + Q_{21}) \tag{3.30}$$

and by then S matrix and the sliding surface could be obtained.

3.3 Control law and reachability condition

3.3.1 Control law

It is clear from the previous section that the performance of the closed loop sliding motion system is influenced by the choice of the sliding surface S. The control law is chosen to ensure that the trajectory of the system states reach the sliding surface and remain on it [101]. This condition is called the reachability condition. The reachability condition specifies that the trajectory of the system states must always point towards the sliding surface. For scalar case, $s(t) = 0$ on the sliding surface and from phase plane analysis, $s(t) > 0$ above the sliding surface and to return to the sliding surface, $s(t)$ must decrease and hence it must have $\dot{s} < 0$. Below the sliding surface $s(t) < 0$ and hence it should have $\dot{s} > 0$. This can be expressed from [101, 92, 25] as

$$\lim_{s\to 0^+} \dot{s} \quad < \quad 0 \tag{3.31}$$

$$\lim_{s\to 0^-} \dot{s} \quad > \quad 0 \tag{3.32}$$

3.3.1.1 Reachability condition

For scalar case, equations (3.31) and (3.32) could be combined in one equation as discussed in [101, 92, 25] as

$$\dot{s} = -sgn(s) = -\frac{s}{|s|} \tag{3.33}$$

Near sliding surface $s(t) = Sx(t) = 0$, the above equation can be written as

$$\dot{s}s < 0 \tag{3.34}$$

Equation (3.34) is referred to in the literature review as the 'η-reachability condition' [102, 101]. A more strict reachability condition which ensure that the control law $u(t)$ is designed so that the sliding surface is reached in finite time and despite the presence of uncertainty and this is given by

$$s\dot{s} \leq -\eta|s| \tag{3.35}$$

where η is a positive design scalar. Equation (3.35) is called the 'η-reachability condition'. For multi variable system, the 'η-reachability condition' is given by

$$s^T\dot{s} \leq -\eta \left\|s\right\| \tag{3.36}$$

where $\|s\|$ is the 2-norm of the switching function $s(t)$. From (3.7),

$$\dot{s} = S\dot{x}(t) = S(Ax + Bu) \tag{3.37}$$

a control low could be designed as discussed in [101] where

$$u(t) = -(SB)^{-1}(SAx(t) - \eta sgn(s)) \tag{3.38}$$

This control law in (3.38) will provide a control action to get the trajectory of the states to the sliding surface but will require a high rate change in the control action due to the discontinuous sigmoid function and will cause chattering. This is not suitable for most systems due to wear and tear therefore a more practical control law [101, 92, 25] is given by

$$u(t) = -(SB)^{-1}(SAx(t)) - \rho(SB)^{-1}(\frac{s}{|s| + \delta}) \tag{3.39}$$

where δ is a small positive scalar and ρ is a positive design scalar depends on the magnitude of the uncertainty. A more practical control law is given by [101, 92, 25]

$$u(t) = -(SB)^{-1}(SA - \Phi S)x(t) - \rho(SB)^{-1}(\frac{s}{|s| + \delta}) \tag{3.40}$$

where Φ is a negative scalar. The new Φ term provides an extra design freedom, by moderating how quickly sliding surface is reached. An appropriate choice of ρ also need to be made to ensure that the sliding is reached and subsequently maintained. See [101, 103] for further details of the proof os stability for the choice of the control law in (3.40). As a summary, the following is the practical steps to design a sliding mode controller [101, 92, 25]:

1. Transform the linear system to the regular form in (3.4).
2. Design M to assure stability of reduced order system in (3.14).
3. The switching function is given by (3.7), where $S = \left[\begin{array}{cc} M & I_m \end{array} \right]$.
4. Choose Φ, ρ, δ.
5. The final control law is given by (3.40).

3.4 Properties of sliding mode control

In this section a summary of the properties of using sliding mode and its benefits [101, 92, 25].

- The dynamics of the closed-loop system is just determined by n-m states.
- By changing the sliding surface, the sliding motion change.
- The reduced order motion of the system does not depend on the control signal u(t). This guarantee some level of stability when a problem exists in the control signals like a fault in actuators or uncertainty. Some uncertainties in the system could be completely matched as will discussed next.

Consider the uncertain Linear system

$$\dot{x}(t) = Ax(t) + B(u(t) + \xi(t, x)) \tag{3.41}$$

where the function $\xi \in \mathbb{R}^{m\times 1}$ is unknown and represent uncertainty. During sliding motion,

$$s(t) = \dot{s}(t) = 0 \quad \text{for all}\, t \geq t_s \tag{3.42}$$

hence,

$$\dot{s} = S\dot{x}(t) = S(Ax(t) + Bu(t) + B\xi(t,x)) = 0 \tag{3.43}$$

then

$$u(t) = -(SB)^{-1}(SAx(t) + SB\xi(t,x)) \quad for\, all\, t \geq t_s \tag{3.44}$$

substituting equation (3.44) into equation (3.41) yields

$$\begin{aligned}\dot{x}(t) &= Ax(t) - B(SB)^{-1}(SAx(t) + SB\xi(t,x)) + B\xi(t,x) \\ &= (I_n - B(SB)^{-1}S)Ax(t) + (I_n - B(SB)^{-1}S)B\xi(t,x) \\ &= (I_n - B(SB)^{-1}S)Ax(t) \quad for\, all\, t \geq t_s \end{aligned} \tag{3.45}$$

Form equation (3.45), it could be shown that during ideal sliding motion the dynamics of the system is not affected by the uncertainty $\xi(t,x)$ i.e $\xi(t,x)$ is a matched uncertainty [101, 92, 25]. For a general case of the matched uncertainty [25] the term $B\xi(t,x)$ in equation (3.41) could be replaced with $D\xi(t,x)$, where $D \in \mathbb{R}^{n\times 1}$ and $D = BR$ for some scaling $R \in \mathbb{R}^{m\times 1}$. So D could be equal to B or any multiplication of B but with keeping the shape of B matrix as in equation (3.4). So to sum up, any uncertainty that could be expressed in

$$\dot{x}(t) = Ax(t) + Bu(t) + D\xi(t,x) \tag{3.46}$$

where the range space of D is contained within the range space of B, could be treated as a matched uncertainty. Any other form of uncertainty is described as unmatched uncertainty.

3.5 Unit vector approach

In section (3.4), matched uncertainty was discussed. Here unmatched uncertainties will be discussed [101, 92, 25]. Consider some uncertainty was added to the system in (3.4) where

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ B_2 \end{bmatrix} u + \begin{bmatrix} f_1(t,x) \\ f_2(t,x,u) \end{bmatrix} \tag{3.47}$$

where the functions $f_1 : \mathbb{R} \times \mathbb{R}^n \to \mathcal{R}(B)^\perp$ and $f_2 : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m \to \mathcal{R}(B)$ are unkown but bounded and

$$\|f_1(t,x)\| \leq k_1 \|x(t)\| + k_2 \tag{3.48}$$

and

$$\|f_2(t,x,u)\| \leq k_3\,\|u(t)\| + \alpha(t,x) \tag{3.49}$$

where $k_1, k_2, k_3 \geq 0$ and $\alpha(.)$ are known. Consider the mapping,

$$T_s = \begin{bmatrix} I & 0 \\ S_1 & S_2 \end{bmatrix} \tag{3.50}$$

where

$$\begin{bmatrix} x_1 \\ s \end{bmatrix} = T_s \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{3.51}$$

this implies

$$\begin{bmatrix} \dot{x}_1 \\ \dot{s} \end{bmatrix} = \tilde{A} \begin{bmatrix} x_1 \\ s \end{bmatrix} + T_s \begin{bmatrix} 0 \\ B_2 \end{bmatrix} u + T_s \begin{bmatrix} f_1(t,x) \\ f_2(t,x,u) \end{bmatrix} \tag{3.52}$$

where

$$\tilde{A} = T_s A T_s^{-1} \tag{3.53}$$

and

$$\tilde{A} = \begin{bmatrix} \tilde{A}_{11} & \tilde{A}_{12} \\ \tilde{A}_{21} & \tilde{A}_{22} \end{bmatrix} \tag{3.54}$$

hence

$$\begin{bmatrix} \dot{x}_1 \\ \dot{s} \end{bmatrix} = \begin{bmatrix} \tilde{A}_{11} & \tilde{A}_{12} \\ \tilde{A}_{21} & \tilde{A}_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ s \end{bmatrix} + T_s \begin{bmatrix} 0 \\ B_2 \end{bmatrix} u + \begin{bmatrix} f_1(t,x) \\ S_1 f_1(t,x) + S_2 f_2(t,x,u) \end{bmatrix} \tag{3.55}$$

This mapping is considered to get a formula for $\dot{s}$ where

$$\dot{s}(t) = \tilde{A}_{21} x_1(t) + \tilde{A}_{22} s(t) + S_2 B_2 u(t) + S_1 f_1(t,x) + S_2 f_2(t,x,u) \tag{3.56}$$

For the uncertainty case $\dot{s}(t)$ could be considered to be $\dot{s}(t) < -\rho sgn(s)$ if ρ was greater than or equal to the uncertainty maximum magnitude and hence ρ satisfies the following inequality,

$$\begin{aligned} \rho &\geq \|S_1 f_1(t,x)\| + \|S_2 f_2(t,x,u)\| \\ &\geq \|S_1\|\,k_1\,\|x(t)\| + \|S_1\|\,k_2 + \|S_2\|\,k_3\,\|u(t)\| + \|S_2\|\,\alpha(t,x) \end{aligned} \tag{3.57}$$

and as $u(t) = u_l + u_n$ and getting $u(t)$ from equation (3.56) then the linear term will be

$$u_l = -(S_2B_2)^{-1}(\tilde{A}_{21}x_1(t) + \tilde{A}_{22}s(t) + S_1f_1(t,x) + S_2f_2(t,x,u)) \tag{3.58}$$

and because the function $f_1(t,x)$ and $f_2(t,x,u)$ are not exactly defined, then the linear term could be

$$u_l = -(S_2B_2)^{-1}(\tilde{A}_{21}x_1(t) + \tilde{A}_{22}s(t) - \Phi s(t)) \tag{3.59}$$

where $\Phi \in \mathbb{R}^{m\times m}$ is any stable design matrix. The nonlinear term will be

$$u_n(t) = -\rho(t,x)(S_2B_2)^{-1}\frac{P_2s(t)}{\|P_2s(t)\|} \tag{3.60}$$

where $P_2 \in \mathbb{R}^{m\times m}$ is a symmetric positive definite matrix and both Φ and P_2 are satisfying the Lyapunov equation

$$P_2\Phi + \Phi^TP_2 = -I_m \tag{3.61}$$

For the nonlinear control term, $\|u_n(t)\| = \rho$ and equation (3.57) will be

$$\begin{aligned}\rho(t,x) &\geq \|S_1\|k_1\|x(t)\| + \|S_1\|k_2 + \|S_2\|k_3\|u(t)\| + \|S_2\|\alpha(t,x)\\ &\geq \|S_1\|k_1\|x(t)\| + \|S_1\|k_2 + \|S_2\|k_3\|u_l(t)\| + \|S_2\|k_3\rho + \|S_2\|\alpha(t,x)\\ &\geq \frac{\|S_1\|k_1\|x(t)\| + \|S_1\|k_2 + \|S_2\|k_3\|u_l(t)\| + \|S_2\|\alpha(t,x)}{(1-\|S_2\|k_3)}\end{aligned} \tag{3.62}$$

So by introducing $\rho(t,x)$ satisfying equation (3.62) which depending on the magnitude of the uncertainty term, the sliding mode control could have the ability to overcome this type of uncertainty. But using these values of ρ and Φ need first to be tested for stability as in the following section.

3.6 Lyapunov stability analysis for unit vector approach

In the previous section, some sort of unmatched uncertainty was discussed. It was claimed that using the inequality in equation (3.62) could treat with the uncertainty terms in (3.47). In this section, a stability analysis for the inequality in (3.62) will be discussed based on Lyapunov stability approach [101, 92, 25]. By substituting equations (3.60) and (3.59) into equation (3.56),

$$\dot{s} = \Phi s(t) - \rho(t,x)\frac{P_2s(t)}{\|P_2s(t)\|} + S_1f_1(t,x) + S_2f_2(t,x,u) \tag{3.63}$$

Consider Lyapunov function $V(s) = s^T P_2 s$ where $V(0) = 0$ and $V(s) \to \infty$ when $s \to \infty$. By differentiating $V(s)$ yields

$$\begin{aligned}
\dot{V} &= \dot{s}^T P_2 s + s^T P_2 \dot{s} \\
&= (\Phi s - \rho \frac{P_2 s}{\|P_2 s\|} + S_1 f_1 + S_2 f_2)^T P_2 s + s^T P_2 (\Phi s - \rho \frac{P_2 s}{\|P_2 s\|} + S_1 f_1 + S_2 f_2) \\
&= s^T (\Phi^T P_2 + P_2 \Phi) s - 2\rho \frac{1}{\|P_2 s\|} s^T P_2 P_2 s + 2 s^T P_2 (S_1 f_1 + S_2 f_2) \\
&= -s^T s - 2\rho \|P_2 s\| + 2 s^T P_2 (S_1 f_1 + S_2 f_2) \qquad (3.64)
\end{aligned}$$

And considering

$$s^T P_2 (S_1 f_1 + S_2 f_2) \leq \|P_2 s\| (\|S_1\| \|f_1\| + \|S_2\| \|f_2\|) \qquad (3.65)$$

then

$$\dot{V} \leq -\|s\|^2 - 2\|P_2 s\| (\rho - (\|S_1\| \|f_1\| + \|S_2\| \|f_2\|)) \qquad (3.66)$$

from equation (3.57), $\rho \geq (\|S_1\| \|f_1\| + \|S_2\| \|f_2\|)$ and hence $\dot{V}$ will always be a negative number. Hence from Lyapunov, the system will be stable and the sliding will take place in a finite time calculated in [25] and [101].

3.7 Integral action

The control law presented earlier has so far only considered state regulation i.e. the control law has been designed so that states return to the equilibrium point once perturbed i.e. the required states values are equal to zero. Here, a tracking requirement will be discussed, i.e. the required state values will not be equal zero. Discussed in detail, tracking using integral action [101, 92, 25].

3.7.1 Integral action approach

Consider a nominal linear system that is in regular form given by:

$$\dot{x}(t) = Ax(t) + Bu(t) \qquad (3.67)$$

$$y(t) = Cx(t) \qquad (3.68)$$

where $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$ and $y(t) \in \mathbb{R}^m$. Assume R is the required vector. For smoothing the required value some delay could be added to R by a low pass filter. If R is a constant demand which is not differentiable at certain time instants the differentiation will go to infinity and hence the low pass filter is required to remove 'derivative kick'. Assume R is the required value for a single state hence

$$\dot{r} = \gamma(R - r) \tag{3.69}$$

where γ is a positive value responsible for the settling time i.e. signal smoothing and r is the filtered required value. For general required vector

$$\dot{r}(t) = \Gamma(r(t) - R(t)) \tag{3.70}$$

where $\Gamma \in \mathbb{R}^{m\times m}$ is a stable design matrix. Consider additional states $x_r(t) \in \mathbb{R}^m$ defined as:

$$\dot{x}_r(t) = r(t) - Cx(t) \tag{3.71}$$

By adding the new states to the system

$$\tilde{x} = \begin{bmatrix} x_r \\ x \end{bmatrix} \tag{3.72}$$

hence,

$$\begin{bmatrix} \dot{x}_r \\ \dot{x} \end{bmatrix} = \begin{bmatrix} 0 & -C \\ 0 & A \end{bmatrix} \begin{bmatrix} x_r \\ x \end{bmatrix} + \begin{bmatrix} 0 \\ B \end{bmatrix} u + \begin{bmatrix} I_p \\ 0 \end{bmatrix} r \tag{3.73}$$

If the original pair (A, B) was in regular form so the new pairs also are in regular form and the states in $\tilde{x}$ could be partitioned to

$$\tilde{x} = \begin{bmatrix} \tilde{x}_1 \\ \tilde{x}_2 \end{bmatrix} \tag{3.74}$$

where $\tilde{x}_1 \in \mathbb{R}^n$ and $\tilde{x}_2 \in \mathbb{R}^m$ and hence

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} \tilde{A}_{11} & \tilde{A}_{12} \\ \tilde{A}_{21} & \tilde{A}_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ B_2 \end{bmatrix} u + \begin{bmatrix} B_r \\ 0 \end{bmatrix} r \tag{3.75}$$

where the new $\tilde{A}$ matrix is given by

$$\tilde{A} = \begin{bmatrix} \tilde{A}_{11} & \tilde{A}_{12} \\ \tilde{A}_{21} & \tilde{A}_{22} \end{bmatrix} = \left[\begin{array}{cc|c} 0 & -C_1 & -C_2 \\ 0 & A_{11} & A_{12} \\ \hline 0 & A_{21} & A_{22} \end{array}\right] \tag{3.76}$$

and

$$B_r = \begin{bmatrix} I_m \\ 0 \end{bmatrix} \tag{3.77}$$

Note that the pair $(\tilde{A}_{11}, \tilde{A}_{11})$ is controllable as proved in [101, 92, 25]. To design the sliding surface, it is required to choose some variables that are required to reach zero in some finite time and in the same time achieve the tracking. The variables that could be used for this purpose in integral action approach are $\dot{x}_r$, x_r and x_2.

The system is ready now to design the sliding surface and the control low. For the sliding surface

$$S = \tilde{x} \in \mathbb{R}^{n+m} : S\tilde{x} = S_r r \tag{3.78}$$

where $S \in \mathbb{R}^{m\times(n+m)}$ and $S_r \in \mathbb{R}^{m\times m}$ are the sliding surface parameters and should be designed to meet the stability of the reduced order system. The matrix S can be partitioned as

$$S = \begin{bmatrix} S_1 & S_2 \end{bmatrix} \tag{3.79}$$

During an ideal sliding motion

$$S_1 x_1 + S_2 x_2 = S_r r \tag{3.80}$$

and therefore

$$x_2 = S_2^{-1} S_r r - M x_1 \tag{3.81}$$

From equation (3.75)

$$\dot{x}_1 = \tilde{A}_{11} x_1 + \tilde{A}_{12} x_2 + B_r r \tag{3.82}$$

then

$$\dot{x}_1(t) = (\tilde{A}_{11} - \tilde{A}_{12} M) x_1(t) + (\tilde{A}_{12} S_2^{-1} S_r + B_r) r(t) \tag{3.83}$$

From here M should be designed to assure stability of the sliding surface regardless of the required signal $r(t)$. Now, the control law needs to be designed. First $\dot{s}$ must be obtained. $\dot{s}(t)$ could be obtained easily by adding $s(t)$ as a state to the system. Specifically define

$$T_s = \begin{bmatrix} I_n & 0 \\ S_1 & S_2 \end{bmatrix} \tag{3.84}$$

let

$$\begin{bmatrix} x_1 \\ s \end{bmatrix} = T_s \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{3.85}$$

By applying this mapping to the system in equation (3.75),

$$\begin{aligned} \begin{bmatrix} \dot{x}_1 \\ \dot{s} \end{bmatrix} &= T_s \begin{bmatrix} \tilde{A}_{11} & \tilde{A}_{12} \\ \tilde{A}_{21} & \tilde{A}_{22} \end{bmatrix} T_s^{-1} \begin{bmatrix} x_1 \\ s \end{bmatrix} + T_s \begin{bmatrix} 0 \\ B_2 \end{bmatrix} u + T_s \begin{bmatrix} B_r \\ 0 \end{bmatrix} r \\ &= \begin{bmatrix} \overline{A}_{11} & \overline{A}_{12} \\ S_2\overline{A}_{21} & S_2\overline{A}_{22}S_2^{-1} \end{bmatrix} \begin{bmatrix} x_1 \\ s \end{bmatrix} + \begin{bmatrix} 0 \\ \Lambda \end{bmatrix} u + \begin{bmatrix} B_r \\ S_1 B_r \end{bmatrix} r \end{aligned} \tag{3.86}$$

where $\overline{A}_{11} = \tilde{A}_{11} - \tilde{A}_{12}M$, $\bar{A}_{21} = M\bar{A}_{11} + \tilde{A}_{21} - A_{22}M$, $\bar{A}_{22} = M\tilde{A}_{12} + A_{22}$, $\overline{A}_{12} = \tilde{A}_{12}S_2^{-1}$, and $\Lambda = S_2B_2$. Hence

$$\dot{x}_1(t) = \overline{A}_{11}x_1(t) + \overline{A}_{12}s(t) + B_r r(t) \tag{3.87}$$

$$\dot{s}(t) = S_2\overline{A}_{21}x_1(t) + S_2\overline{A}_{22}S_2^{-1}s(t) + \Lambda u(t) + S_1B_r r(t) \tag{3.88}$$

By obtaining a formula for $\dot{s}(t)$ as in equation (3.88), now the control law could be obtained. The control action will have a linear term and a nonlinear term.

$$u = u_l + u_n \tag{3.89}$$

where

$$u_l(x_1, s, r) = \Lambda^{-1}(-S_2\overline{A}_{21}x_1 + (\Phi - S_2\overline{A}_{22}S_2^{-1})s - (\Phi S_r + S_1B_r)r) \tag{3.90}$$

and

$$u_n = \begin{cases} -\rho\Lambda^{-1}\frac{\bar{P}_2(s-S_r r)}{||\bar{P}_2(s-S_r r)||} & if\ s \neq S_r r \\ 0 & otherwise \end{cases} \tag{3.91}$$

where $\Phi \in \mathbb{R}^{m\times m}$ is any stable design matrix and $\bar{P}_2$ is a symmetric positive definite matrix that satisfies,

$$\bar{P}_2\Phi + \Phi^T\bar{P}_2 = -I \tag{3.92}$$

Returning to the original coordinates, equation (3.90) could be written as

$$u_l = L\tilde{x} + L_r r \tag{3.93}$$

where

$$L = -\Lambda^{-1}(S\tilde{A} - \Phi S) \tag{3.94}$$

and

$$L_r = -\Lambda^{-1}(\Phi S_r + S_1 B_r) \tag{3.95}$$

where the matrix $\tilde{A}$ is obtained from equation (3.76).

3.7.2 Example: tracking requirement

Consider the longitudinal dynamics calculated for the low super trainer RC model aircraft [104] where the states u, w, q and θ are velocity in x body axis, velocity in z body axis, pitch rate and pitch angle respectively. The control input considered was only the elevator δ_e for trim condition and neglecting the change in thrust input δ_{th} but assuming throttle to maintain the trim condition of the aircraft then

Figure 3.1: Low Super Trainer RC model aircraft

$$A = \begin{bmatrix} -0.0887 & 0.0011 & 0 & -9.8 \\ -0.0025 & -15.15 & 1 & 0 \\ 0.15 & -167.47 & -17.63 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{3.96}$$

and

$$B = \begin{bmatrix} 0 \\ 0 \\ 216.4486 \\ 0 \end{bmatrix} \tag{3.97}$$

The B matrix is required to be in the regular form as in equation (3.4). Hence a change in the order of states could reach the regular form as a special case with this example. So by permutating the states to be θ, u, w, q yields the A, B matrices in regular form given by

$$\bar{A} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ -9.8 & -0.0887 & 0.0011 & 0 \\ 0 & -0.0025 & -15.15 & 1 \\ 0 & 0.15 & -167.47 & -17.63 \end{bmatrix} \tag{3.98}$$

$$\bar{B} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 216.4486 \end{bmatrix} \tag{3.99}$$

and this satisfies the regular form. Consider the pitch angle θ to be the controlled state and hence

$$C = [\ 1 \quad 0 \quad 0 \quad 0\ ] \tag{3.100}$$

Consider designing the sliding surface based on LQR design with

$$Q = \begin{bmatrix} 4000 & 0 & 0 & 0 & 0 \\ 0 & 1.0 & 0 & 0 & 0 \\ 0 & 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0 & 1.0 \end{bmatrix} \tag{3.101}$$

It gives the sliding surface S as

$$S = \begin{bmatrix} -0.2922 & 0.0522 & 0 & 0 & 0.0046 \end{bmatrix} \tag{3.102}$$

and consider $\Phi = -1$, $\rho = 1$ and $\delta = 0.01$ then for the control input

$$L = \begin{bmatrix} 0.2922 & -0.3444 & -0.0007 & 0.7738 & 0.0246 \end{bmatrix} \tag{3.103}$$

and

$$L_r = 0.3444 \tag{3.104}$$

and

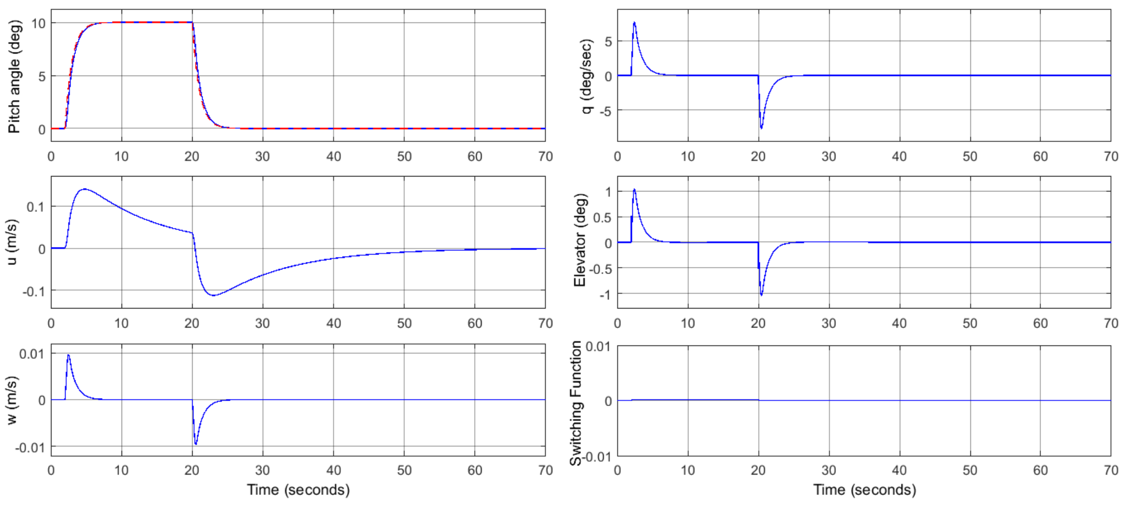


Figure 3.2: Simulation Results: Desired value in red and measured values in blue.

$$S_r = 0.0522 \tag{3.105}$$

In the simulation that follows, Figure 3.2 show the simulation results of the aircraft longitudinal tracking performance where it shows good pitch angle tracking performance with the sliding is maintained close to zero. The simulation was conducted using SIMULINK with a fixed time solver $ode3$ with a time step of $0.001s$.As discussed earlier, the input is the elevator and the controlled output is the pitch angle. It also shows a reliable control surface deflections (δ_e). This example shows the ability of sliding mode control to maintain the tracking performance when the desired state values are not equal to zero using the integral action approach.

3.8 Summary

In this chapter, the main concept, design process and properties of sliding mode control have been discussed. The (state space) regular form has been presented and the robustness property of sliding mode to the matched uncertainties was discussed. An integral action approach was also discussed in this chapter with an illustrating example of attitude tracking using small remote control model aircraft.

Chapter 4

Sliding mode with control allocation

In the last chapter, the ability of sliding mode to directly deal with 'matched' uncertainty was discussed. Actuator fault, such as the 'loss of effectiveness' as described in Chapter 2 can be categorised as matched uncertainty and the 'classical' sliding mode scheme described in the last chapter will be able to deal with it directly without any modification. However, in the event of total actuator failure (and provided redundancy is available), the sliding mode technique described in the previous chapter will not be able to deal with the failure and another fault tolerant technique will have to be considered. (In fact other typical control schemes will not be able to deal with total actuator failure). In this chapter, the combination of sliding mode with a technique called 'control allocation' as proposed in [25] will be discussed. The combination of sliding mode control (with its robustness property to matched uncertainty) with the control allocation scheme (which will allow automated control signal redistribution) results in a simple, yet effective FTC schemes, that have the capability to deal with both actuator faults and total actuator failures. This is achieved without the need to reconfigure the structure of the controller. The synthesis procedure and analysis originally described in [25] will be briefly discussed in this thesis. Finally, at the end of this chapter, two numerical examples will be used to showcase the capability of the scheme to deal with actuator faults and total failures.

4.1 Problem formulation and stability analysis

Consider the nth order linear time invariant system with m inputs. The system is considered to be an over actuated system i.e $m < n$. With the existence of faults/failures, the system can be represented by the following expression [25]:

$$\dot{x}(t) = Ax(t) + BWu(t) \tag{4.1}$$

where $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$. The matrix $W \in \mathbb{R}^{m\times m}$ is a diagonal element matrix with a diagonal element $0 \leq w_i \leq 1$. If $w_i = 1$, it means that the corresponding control element u_i has no problems and working efficiently. For $w_i = 0$ it means that u_i has a complete failure and the corresponding actuator is completely out of service. For $0 < w_i < 1$, it is the faulty situation for u_i but the actuator can still be used. Noting that failure is a 100% fault.

In aircraft equation of motion, forces and moments have their direct effect mainly on the linear and angular accelerations of the aircraft i.e. the velocity states. But Euler angles and displacement states are not directly affected by forces and moments. So for most systems, B matrix can be factorized in two sub matrices

$$B = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} \tag{4.2}$$

where $B_1 \in \mathbb{R}^{(n-l)\times m}$ and $B_2 \in \mathbb{R}^{l\times m}$ and $l < m$. The matrix $B_1$is related to Euler angles and displacement states and $B_2$is related to linear and angular velocity states. Hence,$\|B_1\|$ will be significantly less than $\|B_2\|$. Hence it is acceptable to assume $\|B_1\|$ approximately equal to zero for most cases. Hence the control design could neglect B_1as long as it does not affect the closed loop stability. Hence the strategy here is to find the limits where the control design can go for fault/ failure cases while neglecting B_1 without affecting the closed loop system stability. Thus consider

$$\dot{x}(t) = Ax(t) + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} Wu(t) \tag{4.3}$$

Assume

$$v(t) = B_2 u(t) \tag{4.4}$$

where $v(t) \in \mathbb{R}^l$ is a virtual control input to the system. Hence

$$u(t) = B_2^{+} v(t) \tag{4.5}$$

where $B_2^{+}$is the pseudo inverse of B_2 matrix where

$$B_2^{+} = WB_2^T (B_2 W B_2^T)^{-1} \tag{4.6}$$

Then

$$\dot{x}(t) = Ax(t) + \begin{bmatrix} B_1WB_2^+ \\ B_2WB_2^+ \end{bmatrix} v(t) \tag{4.7}$$

Equation (4.7) could be factorized as

$$\dot{x}(t) = Ax(t) + \begin{bmatrix} 0 \\ I \end{bmatrix} v(t) + \begin{bmatrix} B_1WB_2^+ \\ B_2WB_2^+ - I \end{bmatrix} v(t) \tag{4.8}$$

Equation (4.8) could be separated in two parts, a regular form part and an extra part. The strategy is to design the sliding surface for the regular part and get the limits of the last part that guarantee stability. Note that the second part will be zero in the fault free case with $B_1 = 0$. Hence for design S consider,

$$\dot{x}_{\text{reg}}(t) = Ax(t) + \begin{bmatrix} 0 \\ I \end{bmatrix} v(t) \tag{4.9}$$

where

$$\dot{x}(t) = \dot{x}_{\text{reg}}(t) + \begin{bmatrix} B_1WB_2^+ \\ B_2WB_2^+ - I \end{bmatrix} v(t) \tag{4.10}$$

and design the sliding surface for the system in (4.9) as before where the sliding surface S is supposed to be

$$S = \begin{bmatrix} M & I_l \end{bmatrix} \tag{4.11}$$

where $M \in \mathbb{R}^{l\times(n-l)}$.To get $\dot{s}$ for the full system in (4.7), use the transformation

$$T_s = \begin{bmatrix} I & 0 \\ M & I \end{bmatrix} \tag{4.12}$$

hence

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{s}(t) \end{bmatrix} = \begin{bmatrix} \tilde{A}_{11} & \tilde{A}_{12} \\ \tilde{A}_{21} & \tilde{A}_{22} \end{bmatrix} \begin{bmatrix} x_1(t) \\ s(t) \end{bmatrix} + \begin{bmatrix} B_1WB_2^+ \\ (MB_1 + B_2)WB_2^+ \end{bmatrix} v(t) \tag{4.13}$$

where $\widetilde{A} = T_sAT_s^{-1}$. During sliding, $s(t) = \dot{s}(t) = 0$ then from the upper and lower equations in (4.13)

$$\dot{x}_1(t) = \tilde{A}_{11}x_1(t) + B_1WB_2^{+}\upsilon_{eq}(t) \tag{4.14}$$

and

$$\tilde{A}_{21}x_1(t) + (MB_1 + B_2)WB_2^{+}\upsilon_{eq}(t) = 0 \tag{4.15}$$

then

$$\dot{x}_1(t) = \tilde{A}_{11}x_1(t) - B_1WB_2^{+}((MB_1 + B_2)WB_2^{+})^{-1}\tilde{A}_{21}x_1(t) \tag{4.16}$$

The second term in equation (4.16) must be checked for stability so consider rewriting the equation (4.16) as follows

$$\dot{x}_1(t) \quad = \quad \tilde{A}_{11}x_1(t) + B_1\tilde{u}(t) \tag{4.17}$$

and

$$\tilde{y}(t) \quad = \quad \tilde{A}_{21}x_1(t) \tag{4.18}$$

then an open loop dynamical system could be considered as

$$\tilde{y}(s) = G(s)\tilde{u}(s) \tag{4.19}$$

where $G(s)$ is stable and

$$G(s) = \tilde{A}_{21}(sI - A_{11})^{-1}B_1 \tag{4.20}$$

and the closed loop control law as

$$\tilde{u}(t) = -WB_2^{+}((MB_1 + B_2)WB_2^{+})^{-1}\tilde{y}(t) \tag{4.21}$$

From the small gain theorem [105], if

$$\|G(s)\|_{\infty} \left\|WB_2^{+}((MB_1 + B_2)WB_2^{+})^{-1}\right\| < 1 \tag{4.22}$$

then (4.16) will be stable.

4.2 Control law

The control law will be designed based on the system in (4.9). By applying the transformation in (4.12) to the system in (4.9). Then the control law will be

$$v(t) \;=\; \dot{s}(t) - \tilde{A}_{21}x_1(t) - \tilde{A}_{22}s(t) \tag{4.23}$$

even the actual system will have the sliding surface performance as in equation (4.13) where

$$\dot{s}(t) \;=\; \tilde{A}_{21}x_1(t) + \tilde{A}_{22}s(t) + (MB_1 + B_2)WB_2^{+}v(t) \tag{4.24}$$

The control law comprises linear and nonlinear components given by

$$v(t) = v_l + v_n \tag{4.25}$$

hence

$$\begin{aligned}
\dot{s}(t) &= -v_l(t) + (MB_1 + B_2)WB_2^{+}(v_l(t) + v_n(t)) \\
&= -v_l(t) + (MB_1 + B_2)WB_2^{+}(v_l(t) - \rho(t,x)\frac{s(t)}{\|s(t)\|}) \\
&= ((MB_1 + B_2)WB_2^{+} - I)v_l(t) - (MB_1 + B_2)WB_2^{+}\rho(t,x)\frac{s(t)}{\|s(t)\|}
\end{aligned} \tag{4.26}$$

and so

$$\begin{aligned}
s^T\dot{s}(t) &= s^T((MB_1 + B_2)WB_2^{+} - I)v_l(t) - s^T(MB_1 + B_2)WB_2^{+}\rho(t,x)\frac{s(t)}{\|s(t)\|} \\
&\leq \|s(t)\| \,(\left\|(MB_1 + B_2)WB_2^{+} - I\right\| \|v_l(t)\| \\
&\quad - \left\|(MB_1 + B_2)WB_2^{+} - I\right\| \rho(t,x) - \rho(t,x))
\end{aligned} \tag{4.27}$$

From [25], it is shown that, in order to ensure the reachability condition to exist, $\rho(t,x)$ must be chosen to satisfy

$$s(t)^T\dot{s}(t) \;\leq\; -\eta\,\|s(t)\| \tag{4.28}$$

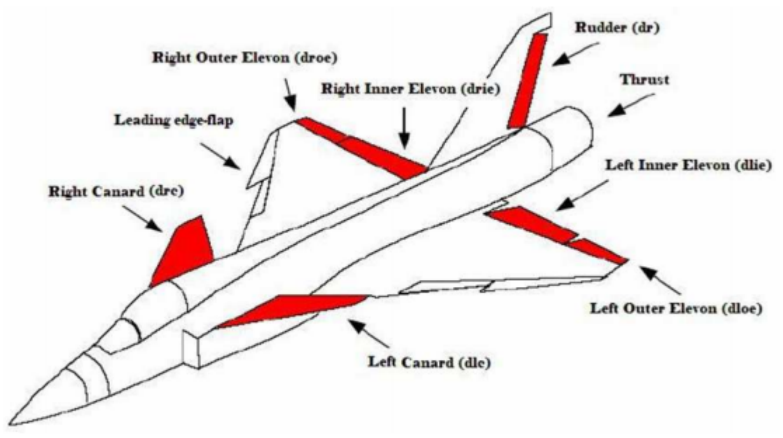


Figure 4.1: ADMIRE Aircraft [34]

hence from equation (4.27)

$$\rho(t,x) \quad := \quad \frac{\eta + \gamma \left\| \upsilon_l(t) \right\|}{\gamma + 1} \tag{4.29}$$

where

$$\gamma = \left\| (MB_1 + B_2) W B_2^+ - I \right\| \tag{4.30}$$

and the proof for stability and that the sliding surface is reached in a finite time could be found in [101].

4.3 Examples

4.3.1 ADMIRE fault tolerance example

The linear model of the ADMIRE model aircraft [25] is as follow

$$A = \begin{bmatrix} -0.5432 & 0.0137 & 0 & 0.9778 & 0 \\ 0 & -0.1179 & 0.2215 & 0 & -0.9661 \\ 0 & -10.5128 & -0.9967 & 0 & 0.6176 \\ 2.6221 & -0.0030 & 0 & -0.5057 & 0 \\ 0 & 0.7075 & -0.0939 & 0 & -0.2127 \end{bmatrix} \tag{4.31}$$

and

$$B = \left[\begin{array}{cccc} 0.0069 & -0.0866 & -0.0866 & 0.0004 \\ 0 & 0.0119 & -0.0119 & 0.0287 \\ \hline 0 & -4.2423 & 4.2423 & 1.4871 \\ 1.6532 & -1.2735 & -1.2735 & 0.0024 \\ 0 & -0.2805 & 0.2805 & -0.8823 \end{array}\right] \begin{array}{l} \left.\vphantom{\begin{array}{c}0\\0\end{array}}\right\} B_1 \\ \left.\vphantom{\begin{array}{c}0\\0\\0\end{array}}\right\} B_2 \end{array} \tag{4.32}$$

where the states are α_a, β, p, q, r which represent the angle of attack, side slip angle, roll rate, pitch rate and yaw rate respectively. The used actuators are canard, right elevon, left elevon and rudder $\delta_c, \delta_{re}, \delta_{le}, \delta_r$ respectively. The model is not in the regular form but as discussed, B matrix could be separated into two matrices $B_1$and B_2 as shown in equation (4.32) where $\|B_1\| \ll \|B_2\|$. For designing the sliding surface use the system A, B_s and for stability analysis and control design use the system A, B where

$$B_s = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} \approx \begin{bmatrix} 0 \\ B_2 \end{bmatrix} \tag{4.33}$$

Consider $\rho = 10$, $\delta = 0.05$. For controlling the states α, β, p the A matrix for closed loop system could be obtained using equation (3.76) and using the LQR with Q is a diagonal matrix with diagonal element [2000,2000, 2000, 1, 1, 1, 1, 1] then the sliding surface S is given by

$$S = \begin{bmatrix} 0.0024 & 1.8980 & 44.6811 & -0.0004 & -0.2239 & -1 & 0 & 0 \\ 44.7213 & -0.0323 & -0.0011 & -9.0768 & -0.0066 & 0 & -1 & 0 \\ -0.0322 & -44.6811 & 1.8980 & 0.0066 & 9.5456 & 0 & 0 & -1 \end{bmatrix} \tag{4.34}$$

and for the control law parameters

$$\begin{aligned} L &= \begin{bmatrix} 0.0024 & 1.8980 & 44.6811 & -0.0026 \\ 44.7213 & -0.0323 & -0.0011 & -51.4897 \\ -0.0322 & -44.6811 & 1.8980 & 0.0352 \end{matrix} \\ &\quad \begin{matrix} 8.4173 & -44.7340 & -0.0004 & -0.4013 \\ -0.0949 & -0.0004 & -9.3696 & 0.0064 \\ 52.3938 & 0.3102 & 0.0064 & -10.0093 \end{bmatrix} \\ L_r &= \begin{bmatrix} 0.0029 & 2.1219 & 45.6811 \\ 54.3537 & -0.0397 & -0.0011 \\ -0.0388 & -54.3487 & 2.1273 \end{bmatrix} \end{aligned}$$

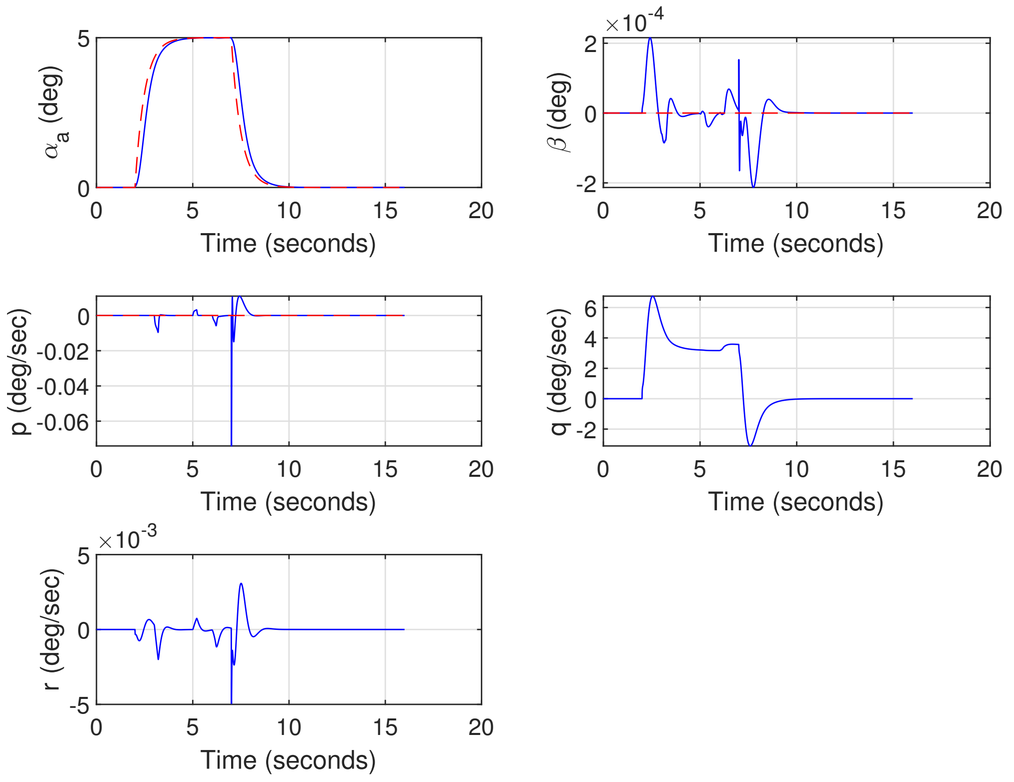


Figure 4.2: ADMIRE States Response: Required signals in red and output signals in blue

$$S_r = \begin{bmatrix} -0.0004 & -0.2239 & -1 \\ -9.6323 & 0.0074 & 0 \\ 0.0066 & 9.6676 & -0.2293 \end{bmatrix} \tag{4.35}$$

In the simulation that follows, Figures 4.2,4.3,4.4 show the simulation results associated with the faults/failures profile in Figure 4.3. The simulation was conducted using SIMULINK with a fixed time solver $ode3$ with a time step of $0.001s$. Figure 4.2 shows good angle of attack (AoA), side-slip and roll rate tracking performance, despite the presence of faults. Figure 4.3 shows that sliding is maintained close to zero, despite the presence of faults and failures. Finally Figure 4.4 shows the effect of faults/failures to the control surface deflections. In particular, it can be seen that the canard is ineffective due to total failure after $t = 6sec$, but the control signals have been redistributed to the remaining control surfaces, in order to maintain the performance of the controller. This example shows the robustness of sliding mode control to work passively as the active control allocation technique.

4.3.2 IRIS+ 3DR quadcopter example

For the linear model of IRIS+ 3DR Quad copter, the states are z, ϕ, θ, ψ, $\dot{z}, p$, q, r which represent altitude, Euler angles (roll, pitch and yaw), vertical velocity and angular rates (roll, pitch and yaw rates) respectively, and using four motors as actuators u.

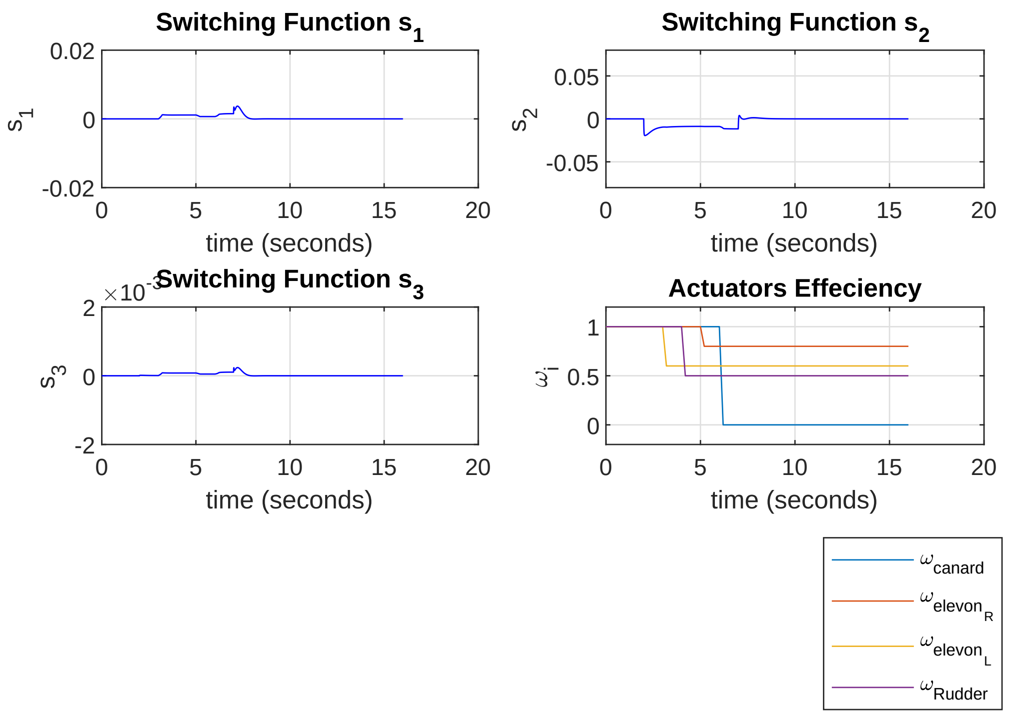


Figure 4.3: ADMIRE Switching function and Actuator efficiencies

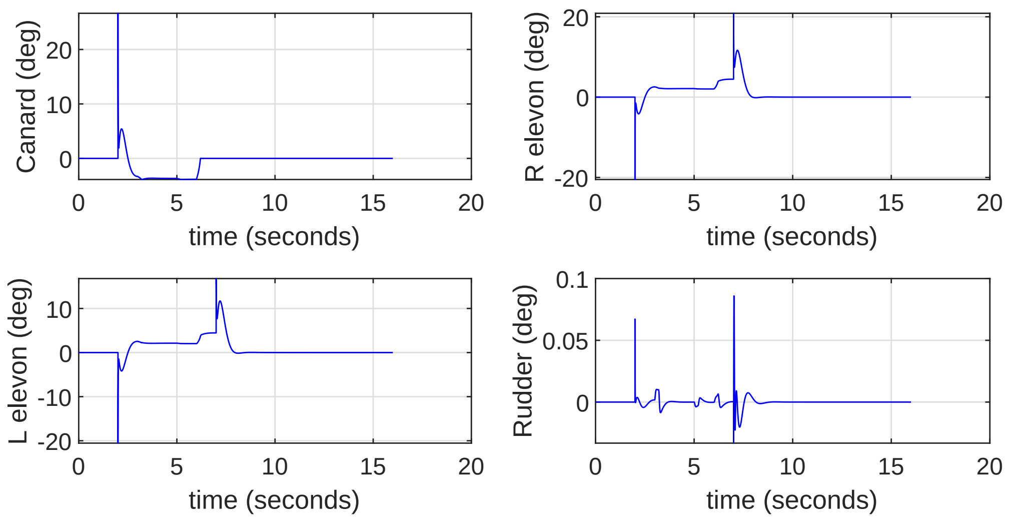


Figure 4.4: ADMIRE Control Signals

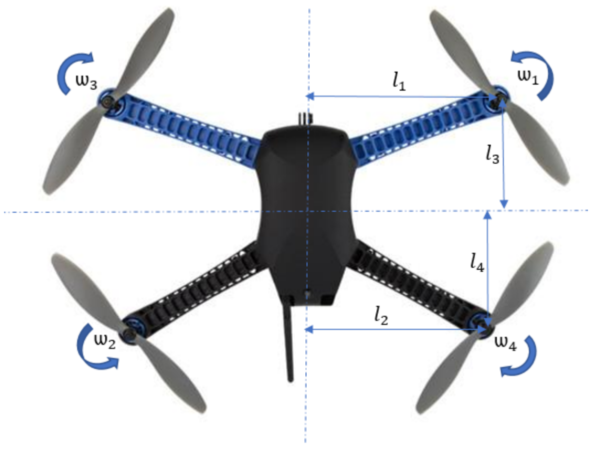


Figure 4.5: IRIS+ 3DR Quad Copter [35]

Remark: In this thesis, especially for the multirotor UAV, the term actuator will also used to refer to motor, motor speed control and the propeller assembly unit. This general lumping of the components as actuator indicate that any faults in the components will influence the ability for the 'actuator' to produce the required thrust and therefore reduction in effectiveness of the actuator.

The Linear model will be considered as in [83]. A matrix it is assumed to be

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.36}$$

The forces and torques vector τ considered in this model are force in z body axis direction, torques about x,y,z body axes $F_z, \mathcal{L}, \mathcal{M}, \mathcal{N}$ respectively. Forces and torques are affecting the linear and angular acceleration of the system. Forces and torques are generating from the motors which can be represented by square of the motors rotational velocities Ω_i^2 where most brush-less motors forces have a linear relation with the square of the rotational speeds of the corresponding motors. Hence

$$B = B_\tau B_\Omega \tag{4.37}$$

where

$$B_\tau = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{m_{kg}} & 0 & 0 & 0 \\ 0 & \frac{1}{I_{xx}} & 0 & 0 \\ 0 & 0 & \frac{1}{I_{yy}} & 0 \\ 0 & 0 & 0 & \frac{1}{I_{zz}} \end{bmatrix} \tag{4.38}$$

where $m_{kg}, I_{xx}, I_{yy}, I_{zz}$ are the total mass of the quad rotor, Inertia about x,y,z body axes respectively. And the relation between torques and motors forces will be

$$B_\Omega = \begin{bmatrix} b & b & b & b \\ -b\ell_1 & b\ell_2 & b\ell_1 & -b\ell_2 \\ b\ell_3 & -b\ell_4 & b\ell_3 & -b\ell_4 \\ d & d & -d & -d \end{bmatrix} \tag{4.39}$$

where $b, d, \ell_1, \ell_2, \ell_3, \ell_4$ are thrust and drag factors and moment arm lengths respectively. Motors numbering and moment arms as in Figure 4.5. Hence

$$B = \begin{bmatrix} 0 \\ B_2 \end{bmatrix} \tag{4.40}$$

Assume $c_1 = \frac{1}{m_{kg}}, c_2 = \frac{1}{I_{xx}}, c_3 = \frac{1}{I_{yy}}, c_4 = \frac{1}{I_{zz}}$ then

$$B_2 = \begin{bmatrix} c_1 b & c_1 b & c_1 b & c_1 b \\ -c_2 b\ell_1 & c_2 b\ell_2 & c_2 b\ell_1 & -c_2 b\ell_2 \\ c_3 b\ell_3 & -c_3 b\ell_4 & c_3 b\ell_3 & -c_3 b\ell_4 \\ c_4 d & c_4 d & -c_4 d & -c_4 d \end{bmatrix} \tag{4.41}$$

and hence

$$\dot{x}(t) = Ax(t) + \begin{bmatrix} 0 \\ B_2 \end{bmatrix} u(t) \tag{4.42}$$

where

$$u(t) = \begin{bmatrix} \Omega_1^2 & \Omega_2^2 & \Omega_3^2 & \Omega_4^2 \end{bmatrix}^T \tag{4.43}$$

and assuming $v(t) = B_2 u(t)$, then

$$\dot{x}(t) = Ax(t) + \begin{bmatrix} 0 \\ I \end{bmatrix} v(t) \tag{4.44}$$

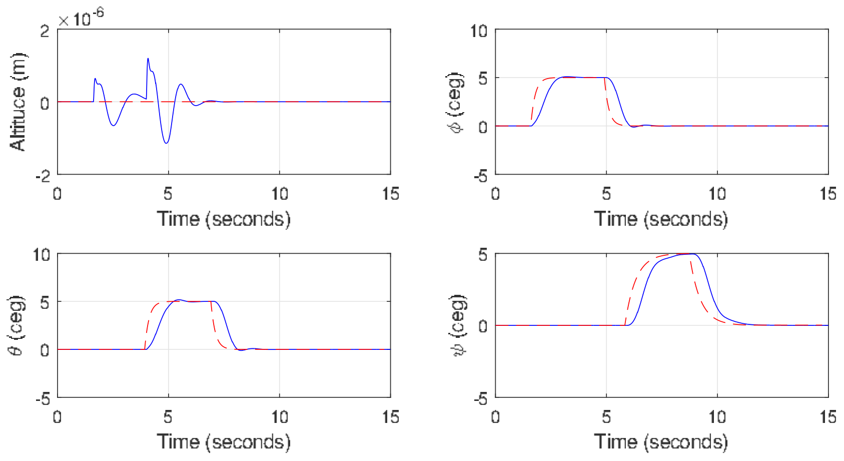


Figure 4.6: 3DR States Response: Required signals in red and output signals in blue

Consider $\rho = 6$, $\delta = 0.05$. For controlling the states z, ϕ, θ, ψ then

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.45}$$

The A matrix for closed loop system could be obtained using equation (3.76) and using the LQR with Q is a diagonal matrix with diagonal elements Q_{diag} equal

$$Q_{diag} = [100, 1000, 100, 100, 1, 1, 1, 1, 0.5, 0.5, 0.5, 0.5] \tag{4.46}$$

and to be taken into consideration that torques of the motors would not exceed the limits. The sliding surface S is given by

$$S = \begin{bmatrix} 14.1421 & 0 & 0 & 0 & -5.5031 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 44.7214 & 0 & 0 & 0 & -9.5626 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 14.1421 & 0 & 0 & 0 & -5.5031 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 14.1421 & 0 & 0 & 0 & -5.5031 & 0 & 0 & 0 & -1 \end{bmatrix} \tag{4.47}$$

and for the control law parameters

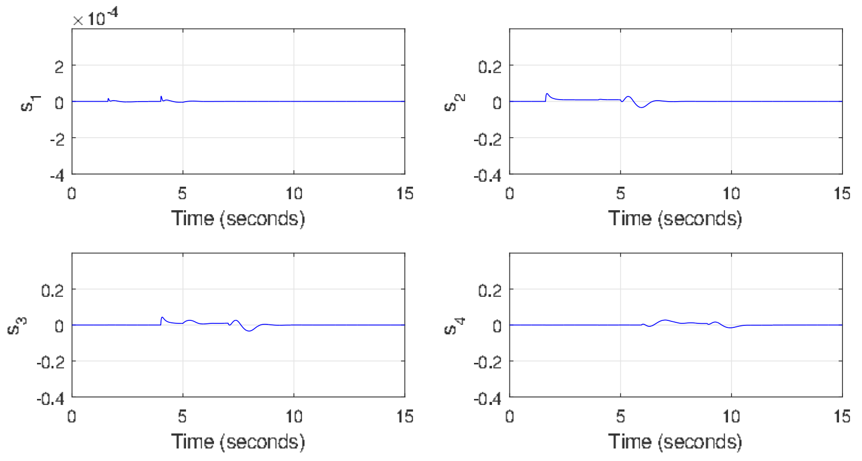


Figure 4.7: 3DR Switching function

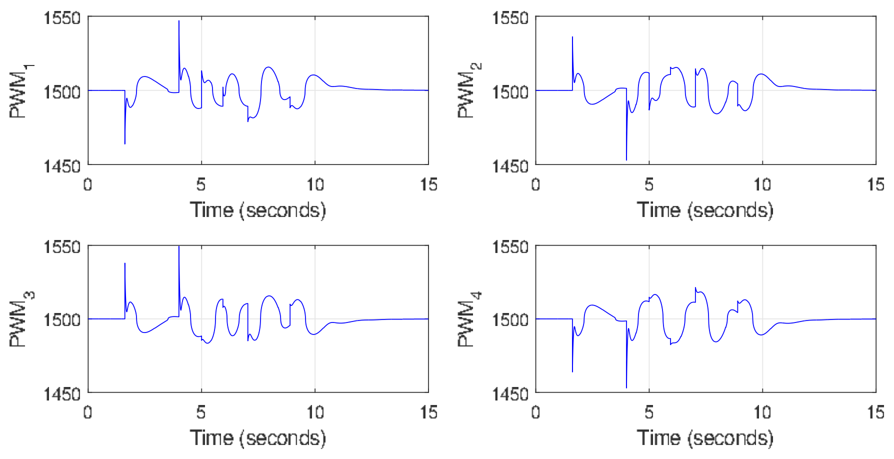


Figure 4.8: 3DR PWM signals

$$
\begin{aligned}
L &= \begin{bmatrix} 14.1421 & 0 & 0 & 0 & -19.6453 & 0 \\ 0 & 44.7214 & 0 & 0 & 0 & -54.2839 \\ 0 & 0 & 14.1421 & 0 & 0 & 0 \\ 0 & 0 & 0 & 14.1421 & 0 & 0 \end{bmatrix. \\
&\qquad \left. \begin{matrix} 0 & 0 & -6.5031 & 0 & 0 & 0 \\ 0 & 0 & 0 & -10.5626 & 0 & 0 \\ -19.6453 & 0 & 0 & 0 & -6.5031 & 0 \\ 0 & -19.6453 & 0 & 0 & 0 & -6.5031 \end{matrix} \right] \\
L_r &= \begin{bmatrix} 19.6453 & 0 & 0 & 0 \\ 0 & 54.2839 & 0 & 0 \\ 0 & 0 & 19.6453 & 0 \\ 0 & 0 & 0 & 19.6453 \end{bmatrix} \\
S_r &= \begin{bmatrix} -5.5031 & 0 & 0 & 0 \\ 0 & -9.5626 & 0 & 0 \\ 0 & 0 & -5.5031 & 0 \\ 0 & 0 & 0 & -5.5031 \end{bmatrix}
\end{aligned} \tag{4.48}
$$

Consider all motors were only 15% effective at time $t = 5sec$. In the simulation that follows, Figures 4.6,4.7,4.8 show the simulation results associated with that fault. The simulation was conducted using SIMULINK with a fixed time solver $ode3$ with a time step of $0.001s$. Figure 4.6 shows good Altitude, roll, pitch and yaw angles tracking performance, despite the presence of faults. Figure 4.7 shows that sliding is maintained close to zero, despite the presence of faults. Finally Figure 4.8 shows the effect of faults/failures to the motors PWM. The PWM limits from 1000 to 2000 and as shown, the control signals were kept in the limits. In particular, the desired response signals were considered to be smooth to be reliable for motors signals. This example shows the robustness of sliding mode control in the existence of faults but not failures in the quadrotor motors.

4.4 Summary

This chapter described a fault tolerant control scheme originally developed in [25] where sliding mode has been combined with online control allocation. This results in a simple yet effective FTC scheme to deal with both actuator faults and failures without the need to reconfigure the baseline controller. The synthesis exploits the separation between the design of a baseline controller which produce the ‘virtual’ control signals and control allocation to redistribute the virtual control signals to all available (redundant) actuators. When actuator faults/failure occurs, the control allocation scheme will automatically redistribute the virtual control signals to the remaining healthy actuators, without any changes to the baseline controller. To illustrate the capability of the proposed scheme, two numerical examples from the literature have been considered. Simulation results from both the ADMIRE fixed wing aircraft model and 3DR IRIS+ Quadrotor shows good tracking performance despite the presence of faults and failures.

Chapter 5

Implementation of sliding mode fault tolerant control on the IRIS+ quadrotor

The last chapter has introduced the key concepts pertaining to sliding mode with control allocation. This chapter investigates the implementation of the idea presented in the last chapter to actual hardware. This chapter implements a sliding mode control scheme combined with control allocation on a multirotor UAV. The controllers were designed to be applicable to both fault-free and faulty conditions. The implementation was done using the PSP for rapid and automatic build-and-deploy control algorithms. The designed controller was tested on a quadrotor for the purpose of tolerating motor faults. The results show the robustness of the proposed scheme.

5.1 Introduction

This chapter will consider a quadrotor as an initial test bed (and therefore the chapter only deals with faults and not total failures). The 3DR IRIS+ [36] has been considered here as it was set up to exploit the Pixhawk [106] as the flight control computer. The main contribution of this chapter concerns the implementation of a sliding mode fault tolerant control allocation scheme, using Simulink supported tools. This allows rapid prototyping and testing of advanced flight controllers without the need for manual coding (which takes time and can be laborious if the structure of the controller has to be changed). The techniques and skills developed in this chapter should be viewed as a stepping stone to implementing an FTC scheme for multirotor UAV with more redundancy (a hex or octorotor).

This chapter concentrates on attitude control for indoor flight. The idea is to control the three Euler angles (roll, pitch and yaw angle) using the available motors in fault-free as well as faulty conditions. In the following control design, a sliding mode controller combined with CA will be considered for fault/failure tolerant control. However, since in this chapter a quadrotor will be used for hardware implementation, no total failure conditions will be considered. (That said, the proposed control could be used later for failure conditions for over actuated systems such as hexarotors and octorotors.)

5.2 Preliminaries

5.2.1 Integral action approach with control allocation

Consider a general nth order linear time invariant system with m inputs given by

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) && (5.1) \\ y_c(t) &= C_c x(t) && (5.2) \end{aligned}$$

where $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C_c(t)\in\mathbb{R}^{l\times n}$ and $u(t) \in \mathbb{R}^m$. Note that $y_c(t)$ represents the quantities to be controlled. For tracking purposes, consider $R\in\mathbb{R}^l$ as the demand vector where $l \leq m$ i.e. the system is considered to be over actuated.

Remark: Note that the assumption that $l \leq m$ can be considered restrictive. But this assumption is true for most multirotor systems and will be exploited later in this chapter.

Here a low pass filter is used for smoothing the required demand signals to remove any 'derivative kick' caused by an abrupt change in the required signal. Specifically

$$\dot{r}(t) = \Gamma(r(t) - R(t)) \tag{5.3}$$

where $\Gamma \in \mathbb{R}^{l\times l}$ is a Hurwitz design matrix. Consider additional states $x_r(t) \in \mathbb{R}^l$ defined as:

$$\dot{x}_r(t) = r(t) - C_c x(t) \tag{5.4}$$

By augmenting the new states to the system states, a new state space model can be obtained based on

$$\tilde{x}(t) = \begin{bmatrix} x_r(t) \\ x(t) \end{bmatrix} \tag{5.5}$$

where
$$\begin{bmatrix} \dot{x}_r(t) \\ \dot{x}(t) \end{bmatrix} = \begin{bmatrix} 0 & -C_c \\ 0 & A \end{bmatrix} \begin{bmatrix} x_r(t) \\ x(t) \end{bmatrix} + \begin{bmatrix} 0 \\ B \end{bmatrix} u(t) + \begin{bmatrix} I_l \\ 0 \end{bmatrix} r(t) \tag{5.6}$$

5.2.2 Faults/failures mitigation

Assume that the input distribution matrix in (5.6) can be perfectly factorised as

$$B = B_\upsilon B_2 \tag{5.7}$$

where $B_\upsilon \in \mathbb{R}^{n\times l}$ and $B_2 \in \mathbb{R}^{l\times m}$ and both are rank l. By a suitable change of coordinate, without loss of generality

$$B_\upsilon = \begin{bmatrix} 0 \\ I_l \end{bmatrix} \tag{5.8}$$

If the augmented states (in the new coordinate system) are partitioned as $(\tilde{x}_1, \tilde{x}_2)$ where $x_2 \in \mathbb{R}^l$, then in the presence of faults/failures, equation (5.6) can be represented as:

$$\begin{bmatrix} \dot{\tilde{x}}_1(t) \\ \dot{\tilde{x}}_2(t) \end{bmatrix} = \begin{bmatrix} \tilde{A}_{11} & \tilde{A}_{12} \\ \tilde{A}_{21} & \tilde{A}_{22} \end{bmatrix} \begin{bmatrix} \tilde{x}_1(t) \\ \tilde{x}_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ B_2 \end{bmatrix} Wu(t) + \begin{bmatrix} B_r \\ 0 \end{bmatrix} r(t) \tag{5.9}$$

The matrix $W \in \mathbb{R}^{m\times m}$ is a diagonal matrix with the diagonal element $0 \leq w_i \leq 1$. If $w_i = 1$, the corresponding actuator u_i is fault free and is working efficiently. When $w_i = 0$, the corresponding actuator u_i has completely failed (i.e. out of service). If $0 < w_i < 1$, the corresponding actuator u_i contains faults but can still be used albeit with degraded performance. Define a 'virtual' control

$$\upsilon(t) = B_2 u(t) \tag{5.10}$$

where $\upsilon(t) \in \mathbb{R}^l$. The actual control signals sent to the actuator is given by

$$u(t) = B_2^+ \upsilon(t) \tag{5.11}$$

where $B_2^+$is a pseudo inverse of B_2 matrix where

$$B_2^+ = \Xi B_2^T (B_2 \Xi B_2^T)^{-1} \tag{5.12}$$

where $\Xi \in \mathbb{R}^{m\times m}$ is a symmetric positive definite diagonal weighting matrix. In this chapter, Ξ will be chosen as $\Xi = W$, i.e. the control allocation will depend on the effectiveness level of the actuator (i.e. 'online control allocation' [25]). As such W must be computed from information supplied by an FDI unit. Define

$$\hat{\upsilon}(t) = B_2 W B_2^+ \upsilon(t) \tag{5.13}$$

Using (5.12) and (5.13), equation (5.9) can be written as

$$\begin{bmatrix} \dot{\tilde{x}}_1(t) \\ \dot{\tilde{x}}_2(t) \end{bmatrix} = \begin{bmatrix} \tilde{A}_{11} & \tilde{A}_{12} \\ \tilde{A}_{21} & \tilde{A}_{22} \end{bmatrix} \begin{bmatrix} \tilde{x}_1(t) \\ \tilde{x}_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ I_l \end{bmatrix} \hat{\upsilon}(t) + \begin{bmatrix} B_r \\ 0 \end{bmatrix} r(t) \tag{5.14}$$

5.2.3 Controller synthesis

Define a switching function $\sigma(t) \in \mathbb{R}^l$ to be

$$\sigma(t) = S\tilde{x}(t) - S_r r(t) \tag{5.15}$$

where $S \in \mathbb{R}^{l\times(n+l)}$ and $S_r \in \mathbb{R}^{l\times l}$. A suitable sliding surface is

$$\mathcal{S} = \{\tilde{x} \in \mathbb{R}^{n+l} : S\tilde{x} = S_r r\} \tag{5.16}$$

The matrix S can be parametrized as

$$S = \begin{bmatrix} M & I_l \end{bmatrix} \tag{5.17}$$

where $M \in \mathbb{R}^{l\times n}$. During an ideal sliding motion $\sigma(t) = \dot{\sigma}(t) = 0$ [101] and therefore from (5.15) and (5.17) during sliding

$$\tilde{x}_2 = S_r r - M\tilde{x}_1 \tag{5.18}$$

Substituting (5.18) into the top equation of (5.14), the sliding motion can be written in the form

$$\dot{x}_1(t) = (\tilde{A}_{11} - \tilde{A}_{12}M)x_1(t) + (\tilde{A}_{12}S_r + B_r)r(t) \tag{5.19}$$

The matrix M must be designed to ensure the reduced order sliding motion $(\tilde{A}_{11} - \tilde{A}_{12}M)$ is stable. Later in this chapter, an LQR-like technique will be used [101] to synthesise M. In order to obtain the control law, as in [101] consider a state transformation for the system in (5.14)

$$\begin{bmatrix} \tilde{x}_1 \\ s \end{bmatrix} = T_s \begin{bmatrix} \tilde{x}_1 \\ \tilde{x}_2 \end{bmatrix} \tag{5.20}$$

where

$$T_s = \begin{bmatrix} I_n & 0 \\ M & I_l \end{bmatrix} \tag{5.21}$$

In the new coordinate, system (5.14) becomes

$$\begin{bmatrix} \dot{\tilde{x}}_1 \\ \dot{s} \end{bmatrix} = \begin{bmatrix} \overline{A}_{11} & \overline{A}_{12} \\ \overline{A}_{21} & \overline{A}_{22} \end{bmatrix} \begin{bmatrix} \tilde{x}_1 \\ s \end{bmatrix} + \begin{bmatrix} 0 \\ I_l \end{bmatrix} \hat{\upsilon} + \begin{bmatrix} B_r \\ MB_r \end{bmatrix} r \tag{5.22}$$

where $\overline{A}_{11} = \tilde{A}_{11} - \tilde{A}_{12}M$, $\overline{A}_{21} = M\overline{A}_{11} + \tilde{A}_{21} - A_{22}M$, $\overline{A}_{22} = M\tilde{A}_{12} + A_{22}$, and $\overline{A}_{12} = \tilde{A}_{12}$. The virtual controller law is chosen to have a linear term and a nonlinear term and is given by

$$\hat{\upsilon} = \hat{\upsilon}_L(\tilde{x}_1, s, r) + \hat{\upsilon}_N(s, r) \tag{5.23}$$

Hence

$$\hat{\upsilon}_L(\tilde{x}_1, s, r) = \big(-\overline{A}_{21}\tilde{x}_1 + (\Phi - \overline{A}_{22})s - (\Phi S_r + MB_r)r + S_r\dot{r}\big) \tag{5.24}$$

and

$$\hat{\upsilon}_N(s, r) = \begin{cases} -\rho\frac{\overline{P}_2\sigma}{||\overline{P}_2\sigma)||} & \text{if}\, \sigma = 0 \\ 0 & \text{otherwise} \end{cases} \tag{5.25}$$

where $\Phi \in \mathbb{R}^{l\times l}$ is a designed stable design matrix and $\bar{P}_2$ is a symmetric positive definite matrix that satisfies

$$\bar{P}_2\Phi + \Phi^T\bar{P}_2 = -I \tag{5.26}$$

In the original coordinates, equation (5.24) can be written as

$$\upsilon_L(\tilde{x}, r) = L\tilde{x} + L_r r + L_{\dot{r}}\dot{r} \tag{5.27}$$

where

$$L = -(S\tilde{A}{-}\Phi S) \tag{5.28}$$

$$L_r = -(\Phi S_r + S_1 B_r) \tag{5.29}$$

$$L_{\dot{r}} = S_r \tag{5.30}$$

5.2.4 Control signals sent to the actuators

The actual control signal $u(t)$ sent to the actuator can be obtained from equation (5.11) and (5.13) and is given by

$$u(t) = WB_2^T(B_2W^2B_2^T)^{-1}\hat{\nu}(t) \tag{5.31}$$

where $\hat{\nu}(t)$ is given in (5.23)-(5.30).

Remark: The online control allocation in (5.31) depends on information about the health of the actuators. In this chapter, it will be assumed that this information is available (this information can be obtained from a fault detection unit or from a direct comparison between measurements of the actuator position and the control signals $u(t)$).

5.2.5 Stability analysis

Proposition 1. *Suppose the hyperplane matrix M from (5.17) has been chosen so that $\tilde{A}_{11} - \tilde{A}_{12}M$ is stable, then choosing*

$$\rho > 0 \tag{5.32}$$

ensures a sliding motion takes place on $\mathcal{S}$ in finite time.

Proof: Substituting equations (5.24) and (5.25) into equation (5.22) yields

$$\dot{\tilde{x}}_1(t) = \overline{A}_{11}\tilde{x}_1(t) + \overline{A}_{12}s(t) + B_r r(t) \tag{5.33}$$

and

$$\dot{s}(t) = \Phi s - \Phi S_r r + S_r \dot{r} - \rho \frac{\overline{P}_2 \sigma}{||\overline{P}_2 \sigma||} \tag{5.34}$$

By choice of S_r and $S_r\dot{r}$, (5.34) can be written as

$$\dot{\sigma}(t) = \Phi\sigma - \rho \frac{\overline{P}_2 \sigma}{||\overline{P}_2 \sigma||} \tag{5.35}$$

Consider $V(\sigma) = \sigma^T \overline{P}_2 \sigma$ as a Lyapunov function for (5.35). Differentiating the Lyapunov function and using (5.26)

$$\dot{V}(\sigma) = \sigma^T \underbrace{\left(\Phi^T \overline{P}_2 + \overline{P}_2 \Phi\right)}_{-I} \sigma - 2\rho\|\overline{P}_2\sigma\| \tag{5.36}$$

Therefore using (5.32), the Lyapunov function satisfies

$$\dot{V}(\sigma) \leq -\|\overline{P}_2\sigma\|^2 - 2\rho\|\overline{P}_2\sigma\| \tag{5.37}$$

Inequality (5.37) will now be used to show that sliding occurs on $\mathcal{S}$ in finite time. Using the Rayleigh principle

$$\begin{aligned} \|\overline{P}_2\sigma\|^2 &= \left(\|\overline{P}_2^{1/2}\sigma\right)^T \overline{P}_2 \left(\overline{P}_2^{1/2}\sigma\right) \\ &\geq \lambda_{min}(\overline{P}_2)\|\overline{P}_2^{1/2}\sigma\|^2 \end{aligned}$$

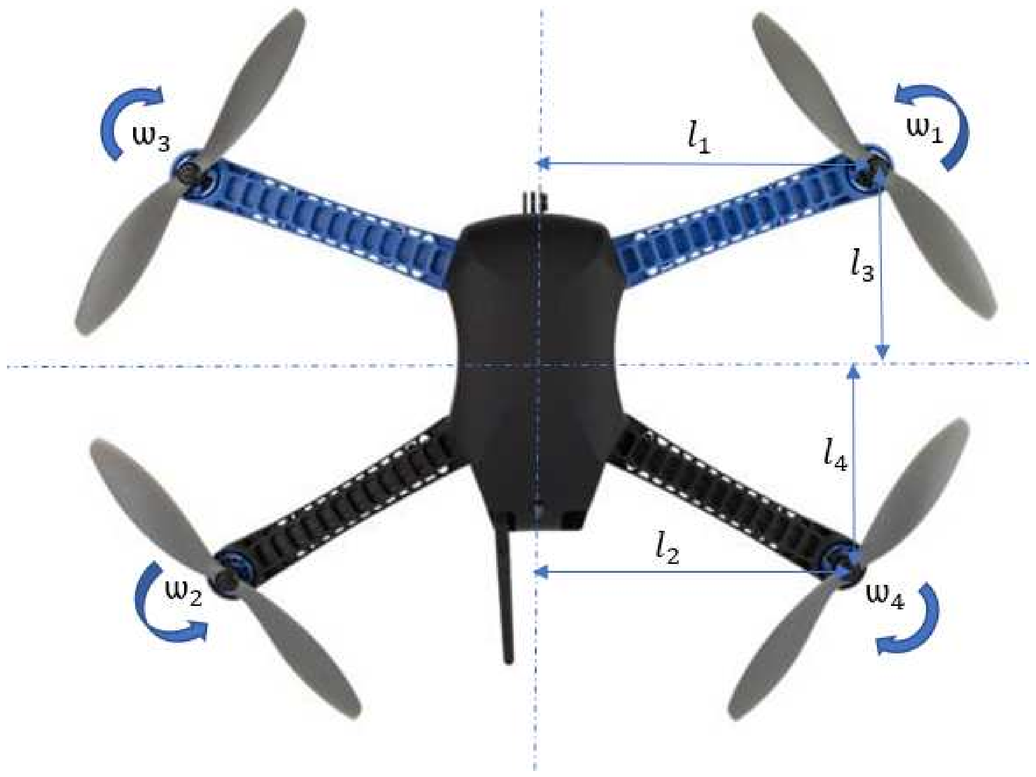


Figure 5.1: 3DR IRIS+ quadrotor [36]

$$= \lambda_{min}(\overline{P}_2)V(\sigma) \tag{5.38}$$

Equation (5.37) can then be written as

$$\dot{V}(\sigma) \leq -2\rho\sqrt{\lambda_{min}(\overline{P}_2)}\sqrt{V} \tag{5.39}$$

Integrating (5.39) yields

$$t_s \leq \rho^{-1}\sqrt{V(\sigma_0)/\lambda_{min}(\overline{P}_2)} \tag{5.40}$$

which represents a bound on the time taken to reach the sliding surface (where σ_0 is the initial value of $\sigma(t)$ at $t = 0$). ■

5.3 IRIS+ 3DR quadrotor equation of motion

The nonlinear equation of motion of the 3DR IRIS+ quadrotor [36] is given by

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ \dot{u} \\ \dot{v} \\ \dot{w} \\ \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} p + qsin(\phi)tan(\theta) + rcos(\phi)tan(\theta) \\ qcos(\phi) - rsin(\phi) \\ qsin(\phi)sec(\theta) + rcos(\phi)sec(\theta) \\ vr - qw - gsin(\theta) \\ pw - ur + gcos(\theta)sin(\phi) \\ uq - pv + gcos(\theta)cos(\phi) \\ qr(I_{yy} - I_{zz})/I_{xx} \\ pr(I_{zz} - I_{xx})/I_{yy} \\ qr(I_{xx} - I_{yy})/I_{zz} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \frac{F_z}{m_{kg}} \\ \frac{\mathcal{L}}{I_{xx}} \\ \frac{\mathcal{M}}{I_{yy}} \\ \frac{\mathcal{N}}{I_{zz}} \end{bmatrix} \tag{5.41}$$

and

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} cos(\theta)cos(\psi) & sin(\phi)sin(\theta)cos(\psi) - cos(\phi)sin(\psi) & cos(\phi)sin(\theta)cos(\psi) + sin(\phi)sin(\psi) \\ cos(\theta)cos(\psi) & sin(\phi)sin(\theta)sin(\psi) + cos(\phi)cos(\psi) & cos(\phi)sin(\theta)sin(\psi) - sin(\phi)cos(\psi) \\ -sin(\theta) & sin(\theta)cos(\theta) & cos(\phi)cos(\theta) \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{5.42}$$

The states are roll angle, pitch angle, yaw angle, velocities in the x, y, z axes, roll rate, pitch rate, yaw rate while (x, y, z) represent position in the x, y, z axes. The variables $F_z, \mathcal{L}, \mathcal{M}$ and $\mathcal{N}$ represent the total thrust, roll torque, pitch torque and yaw torque, while c, s represents cos and sin respectively. The system parameters are mass m_{kg} and I_{xx}, I_{yy}, I_{zz} which represent inertia about the x,y,z body axes respectively.

For the implementation work considered in this chapter, only the roll, pitch and yaw rates will be controlled using sliding mode control scheme developed earlier (altitude will be controlled using a separate outer loop control). By linearizing the nonlinear equation of motion in (5.41)) about hover and considering the attitude states only i.e.

$$X = \begin{bmatrix} \phi & \theta & \psi & p & q & r \end{bmatrix}^T$$

a linear attitude state space model can be written as

$$\dot{X}(t) = AX(t) + Bu(t) \tag{5.43}$$

where

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{5.44}$$

while

$$B = \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \frac{1}{I_{xx}} & 0 & 0 \\ 0 & \frac{1}{I_{yy}} & 0 \\ 0 & 0 & \frac{1}{I_{zz}} \end{bmatrix}}_{B_\tau} \underbrace{\begin{bmatrix} -b\ell_1 & b\ell_2 & b\ell_1 & -b\ell_2 \\ b\ell_3 & -b\ell_4 & b\ell_3 & -b\ell_4 \\ d & d & -d & -d \end{bmatrix}}_{B_\Omega} \tag{5.45}$$

The input

$$u(t) = \begin{bmatrix} \Omega_1^2 & \Omega_2^2 & \Omega_3^2 & \Omega_4^2 \end{bmatrix}^T \tag{5.46}$$

where Ω_i^2 represents the square of the individual motor rotational velocities. The parameters $b, d, \ell_1, \ell_2, \ell_3, \ell_4$ are thrust and drag factors, and moment arm lengths respectively. Note the rotor number and moment arms are labelled as in Fig. 5.1.

Using the convention employed in (5.7), equation (5.45) can be written as

$$B = \underbrace{\begin{bmatrix} 0 \\ I_3 \end{bmatrix}}_{B_v} B_2 \tag{5.47}$$

where

$$B_2 = \begin{bmatrix} -c_2b\ell_1 & c_2b\ell_2 & c_2b\ell_1 & -c_2b\ell_2 \\ c_3b\ell_3 & -c_3b\ell_4 & c_3b\ell_3 & -c_3b\ell_4 \\ c_4d & c_4d & -c_4d & -c_4d \end{bmatrix} \tag{5.48}$$

and $c_2 = \frac{1}{I_{xx}}, c_3 = \frac{1}{I_{yy}}, c_4 = \frac{1}{I_{zz}}$.

Since the states to be controlled are ϕ, θ, ψ , the controlled output distribution matrix is given by

$$C_c = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{5.49}$$

This linear model will be used in the design of the controller which will be described in the next section.

5.4 Implementation

The UAV used in this work is the IRIS+ multirotor UAV (see Fig. 5.1) developed by 3DR [36] which comes ready to fly. More importantly, the IRIS+ multirotor UAV is controlled using the Pixhawk flight controller [36] as shown in Fig. 5.2. This flight controller is open source hardware and the firmware (which is coded in C and contains a prototypical (PID) controller) is also available as open source software [107]. The open nature of the firmware and hardware of the Pixhawk makes it popular not only among hobbyist, but also researchers. This has motivated many researchers such as Polak [108] and Heartley [109], to create support packages for Matlab and Simulink through the ArduPilotMega (APM). Recently, Mathworks provided official support through their PSP [37]. This allows any control scheme to be rapidly implemented and tested directly from Simulink block diagrams without the need to manually programme the controller in C. This facilitates rapid prototyping and testing.

The IRIS+ multirotor UAV physical properties have been identified in [36]. The work in [36] also describes in detail procedures to implement a PID controller using the Simulink support software and complements the information contained in the document provided by Simulink.

5.4.1 Pixhawk support package

Pixhawk [106] originated from the Pixhawk project at ETH Zurich (Computer Vision and Geometry Lab). The objective of the project was to provide low-cost high availability open hardware and software for academics, hobbyists and industry. Pixhawk comprises a main processor (168 MHz Cortex M4F CPU, 256 kb Ram and 2MB Flash), sensors (3D accelerometer, gyroscope, magnetometer and barometers) an interface (e.g. microSD slot, 14 PWM/servo outputs, 5 UARTs, CAN, I2C, SPI, ADC) in a compact package ($50 \times 15.5 \times 81.5mm$). Details of the hardware can be found in [106].

5.4.2 Gimbal setup

Due to health and safety and regulation from the Civil Aviation Authority (CAA), most flight tests in this project will be done indoor. However, in the absence of a suitable large flying arena, the tests that have been considered so far mainly focused on controlling the Euler angles and not position tracking.

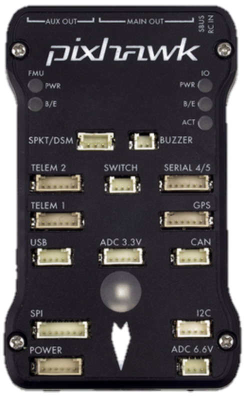


Figure 5.2: Pixhawk Autopilot [36]

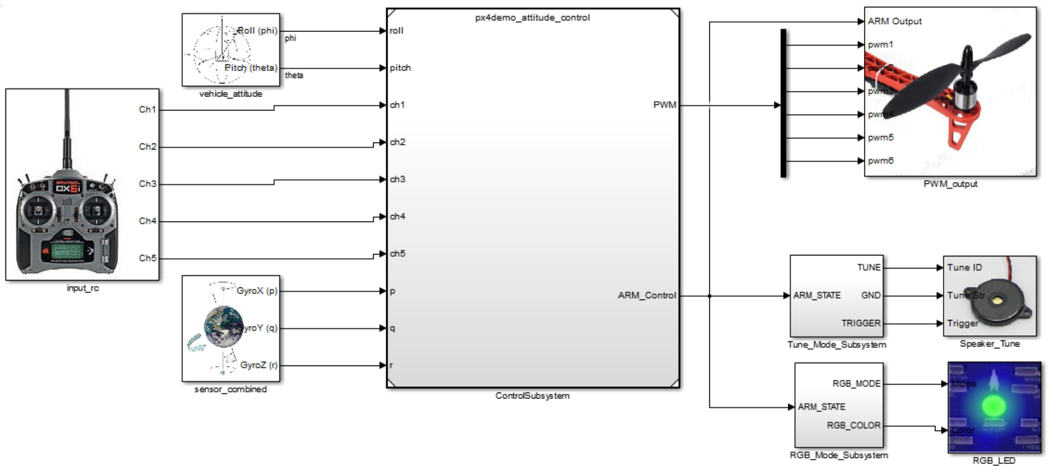


Figure 5.3: PX4 Attitude System Control [37]

Therefore, a test rig is mandatory that enables the aircraft to rotate smoothly at different Euler angles while fixing the position in space. Lots of suggested setups were introduced to handle this. Initially, a pivot test (ball joint stand) was tested with IRIS+ quadrotor using a tripod stand with ball head as in Figure 5.4. The test could be used for small roll and pitch angles for about ± 5 degrees. For higher angles, the quadrotor seems to be struggling to produce good results. One possible reason is due to the centre of gravity for the quadrotor and the ball are not identical and it is required a horizontal body axes force to push the quad back to its stable point. One way to solve this issue is to use a pendulum setup (for example by hanging the quadrotor to the roof). The 'string test' [110] could help in tuning the gains for each axis separately as in Figure 5.5. This test was proved to provide reliable results for tuning Euler angles separately especially when tuning PID controller. However, it is not possible to tune all Euler angles control at the same time to test the coupling effect between different axes. Gimbal (gyroscopic) test rig, as in Figure 5.6, could provide a means to test all the Euler angles simultaneously.

Special attention was given to the material used for fabricating the gimbal so that it will not have a significant effect on the mass or inertia of the quadrotor. Lightweight carbon fibre tubes were used in the fabrication of the gimbal (see Fig. 5.7). As shown in Figure 5.8, the setup consists of an outer aluminium strut cubic frame, see Figure 5.9. The aluminium strut dimension was $40 \times 40mm$ with 8mm groove and 2000mm length. The gimbal was built from a carbon fiber 8mm diameter and 1000mm length rods. The rods were connected together by using prebuilt plastic connections using a 3D printing machine as shown in figure 5.10. As shown in Figure 5.8, the aircraft will rotate around rod E when for roll motion and the square D and rod E will rotate for pitch motion. Rod E, squares D and B will rotate around the tripod C for yaw motion. All the changes in inertias were calculated and added in table 5.1.

5.4.3 Design

The physical properties of the IRIS+ quadrotor with the gimbal are given in Table 5.1. By applying the FTC approach described in the previous section, an LQR-like optimal design methodology [25, 101] was used to select the hyperplane matrix where the design matrix Q was chosen to be $Q = \text{diag}(13, 13, 13, 2, 2, 2, 0.02, 0.02, 0.17)$. The design matrices Γ and Φ have been chosen as $\Gamma = -I_3$ and $\Phi = \text{diag}(-3, -1, -1)$ respectively.

To aid in tuning the controller, the discontinuity in the signum term in (5.25) has been split channel-wise (by exploiting the diagonal structure of $s(t)$ as a result of the design above) and then 'smoothed' to create a sigmoidal function. The 'smoothing factors' δ_i have been chosen as $0.12, 0.17, 0.1$ respectively while the individual modulation gains ρ_i

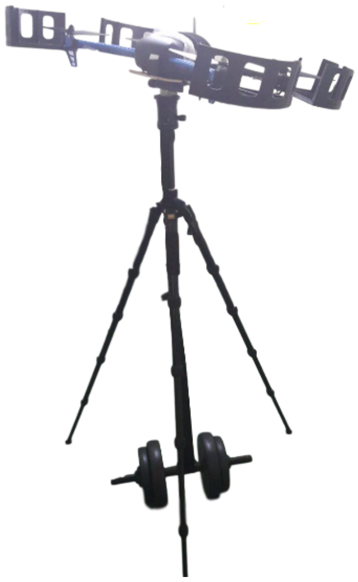


Figure 5.4: Tripod Test setup with IRIS+ Quadrotor

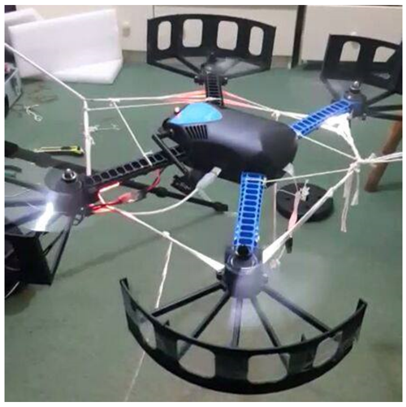


Figure 5.5: String Test

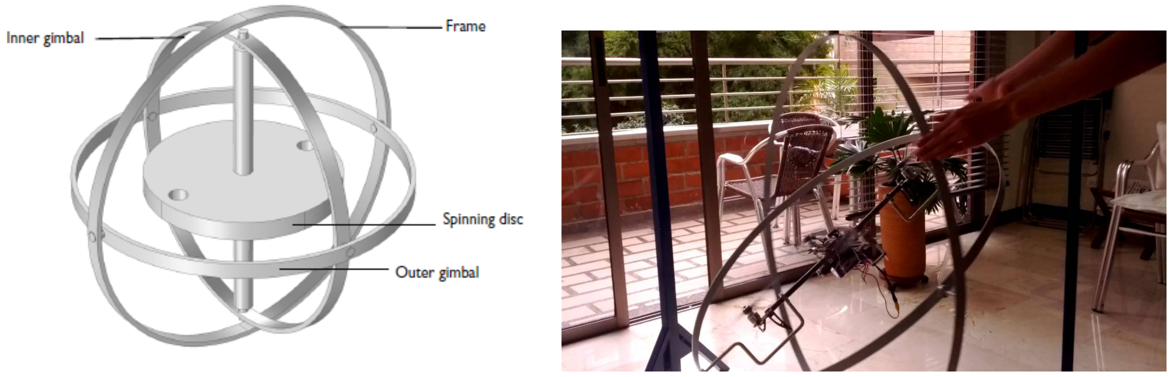


Figure 5.6: Gimbal Test (Gyroscope) [38, 39]

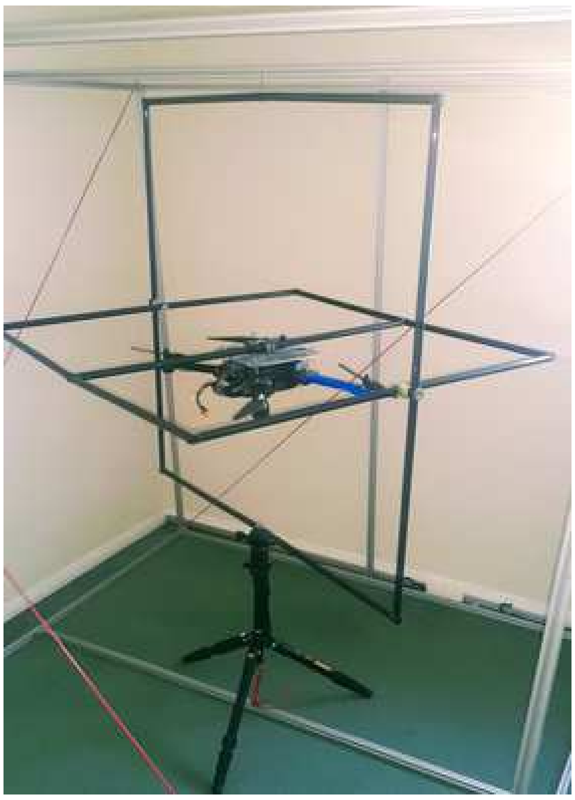


Figure 5.7: 3DR IRIS+ Quadrotor in gimbal

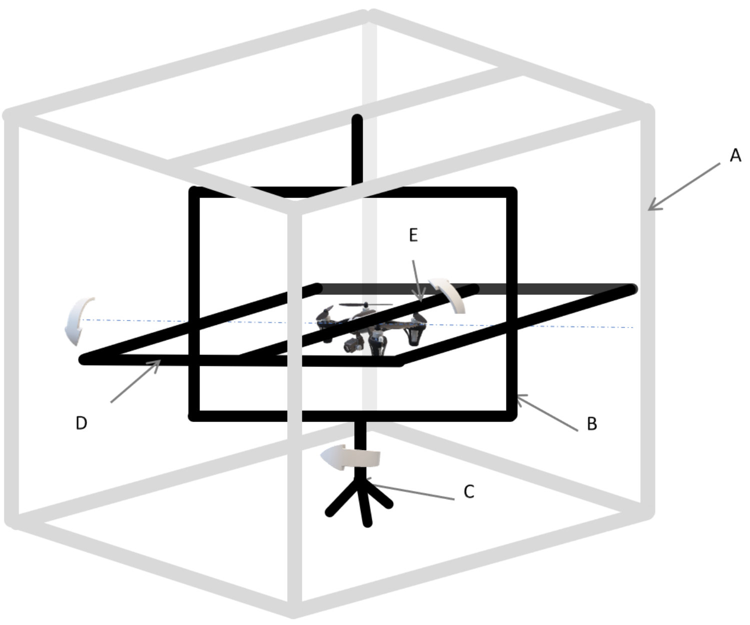


Figure 5.8: Gimbal Design

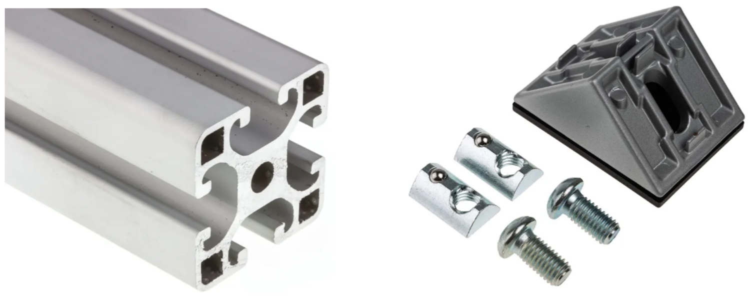


Figure 5.9: aluminium Alloy Strut and Connector Bracket [40]

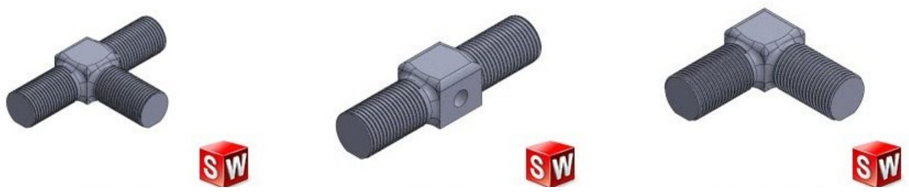


Figure 5.10: 3D Printed Plastic Connectors

Table 5.1: 3DR IRIS+ Physical parameters (including test rig)

Parameter	Value	Unit
m_{kg}	1.698	kg
I_{xx}	0.0296	kgm^2
I_{yy}	0.0638	kgm^2
I_{zz}	0.0686	kgm^2
b	7.1127×10^{-6}	kgm
d	1.6473×10^{-7}	kgm^2
ℓ_1	0.23	m
ℓ_2	0.21	m
ℓ_3	0.13	m
ℓ_4	0.13	m

have been chosen as $5, 1, 1$ respectively. Hence the resulted sliding surface S is given by

$$S = \begin{bmatrix} -0.2550 & 0 & 0 & 0.1229 & 0 & 0 & 0.0100 & 0 & 0 \\ 0 & -0.2550 & 0 & 0 & 0.1229 & 0 & 0 & 0.0100 & 0 \\ 0 & 0 & -0.0874 & 0 & 0 & 0.0541 & 0 & 0 & 0.0100 \end{bmatrix} \tag{5.50}$$

and for the control law parameters

$$\begin{aligned} L &= \begin{bmatrix} 0.7649 & 0 & 0 & -0.2745 & 0 & 0 & -0.1529 & 0 & 0 \\ 0 & 0.2550 & 0 & 0 & -0.2159 & 0 & 0 & -0.1329 & 0 \\ 0 & 0 & 0.0874 & 0 & 0 & -0.1415 & 0 & 0 & -0.0641 \end{bmatrix} \\ L_r &= \begin{bmatrix} 0.6236 & 0 & 0 \\ 0 & 0.3778 & 0 \\ 0 & 0 & 0.1415 \end{bmatrix} \\ S_r &= \begin{bmatrix} 0.1229 & 0 & 0 \\ 0 & 0.1229 & 0 \\ 0 & 0 & 0.0541 \end{bmatrix} \end{aligned} \tag{5.51}$$

5.4.4 Implementation and test results

Note that the outputs of the control law $u(t)$ in (5.46) are the squares of the individual motor rotational speeds (Ω_i^2). However, for implementation, the brushless motor Electric Speed Controller (ESC) only accepts PWM signals. The conversion between the control law $u(t)$ in $(rpm)^2$ to PWM for each motor is given by:

$$PWM_i = (2000 - trim) \times \left(\frac{u_i}{u_{i,max}}\right) + trim \tag{5.52}$$

Here the $u_{i,max}$ represent the square of the maximum rotational speed (i.e. $u_{i,max} = (10,000rpm)^2$). Note that that the PWM signals supplied to the ESC range between

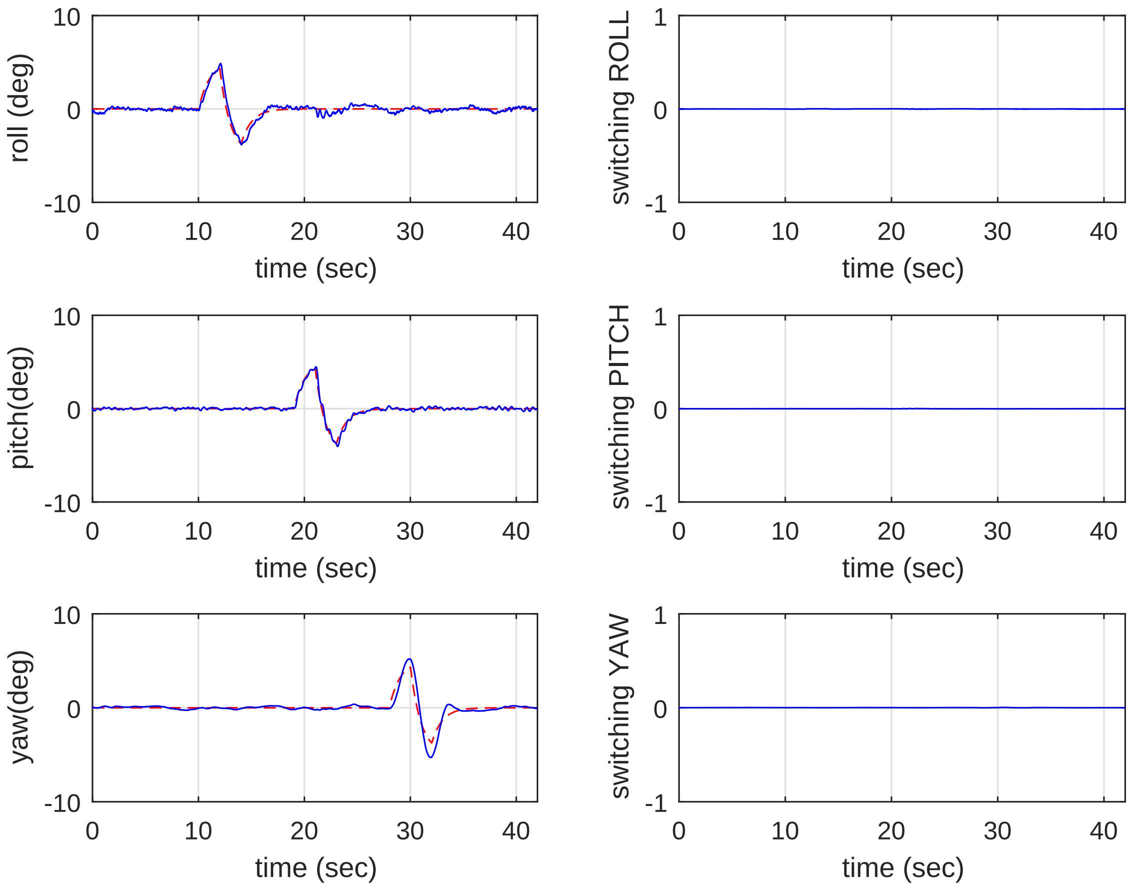


Figure 5.11: Controlled states and switching functions (Fault-Free Case), Desired Values in Red and measured values in blue

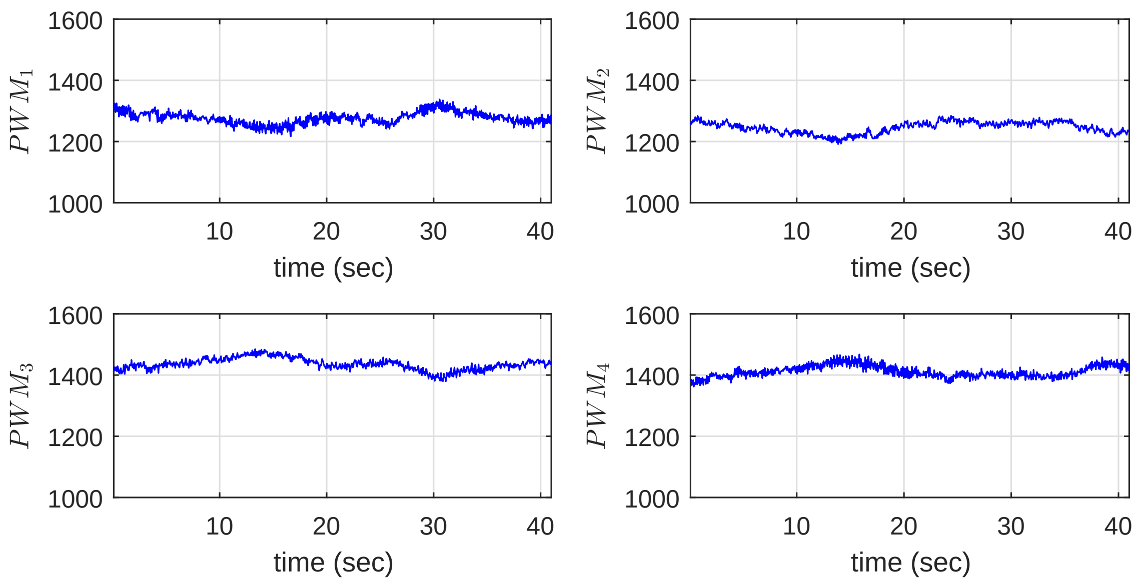


Figure 5.12: PWM signals (Fault-Free Case)

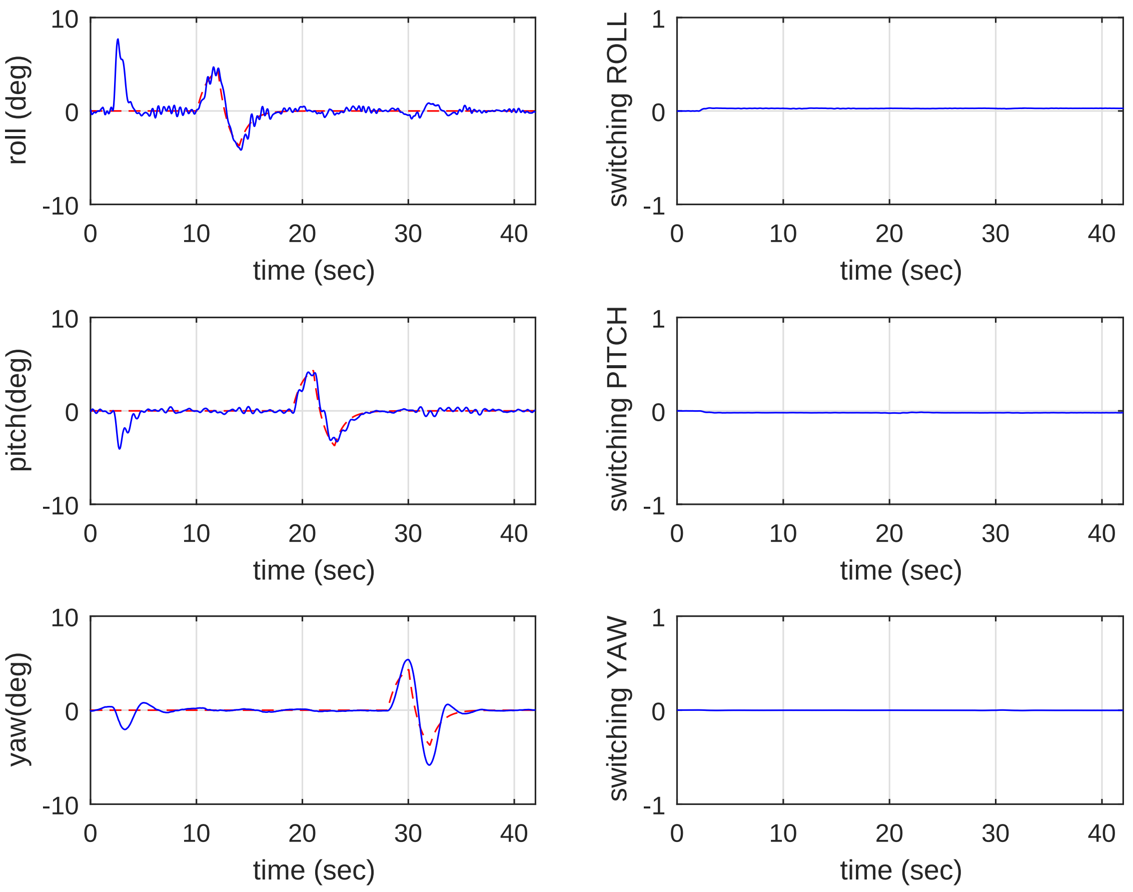


Figure 5.13: Controlled states and switching functions (Faulty case: $w_1 = 0.75 @ t = 2sec$), , Desired values in red and measured values in blue

$trim = 1000$ (no rotation) to 2000 (maximum rotation).

The results from two implementations will be shown in this chapter. The first scenario will consider the fault free case. While the second scenario considers a fault on the first motor (specifically a 25% reduction in the effectiveness of motor 1) i.e. $w_1 = 0.75$ at $t = 2sec$. (In this chapter, the drop in the effectiveness levels is introduced at a software level in order to preserve the capability of the IRIS+ for future tests).

The implementation of the proposed controller was done on Pixhawk using PSP [37].

Figs. 5.11, 5.12, 5.13, 5.14 show the implementation results (Fig. 5.11, 5.12 for the fault free case and Fig. 5.13, 5.14 for the faulty cases). Fig. 5.11 shows the Euler angle tracking performance in the fault free case. It can be seen that sliding is maintained. The motors' PWMs are shown in Fig. 5.12.

Fig. 5.13 shows good Euler angle tracking performance despite the presence of faults. The sliding surfaces show very small deviation from zero even during the fault. Finally Fig. 5.14 shows the effect of the fault on the PWM. The PWM signals are limited from 1000 to 2000 as shown. This implementation shows the robustness of sliding mode control in the presence of faults in the quadrotor motors.

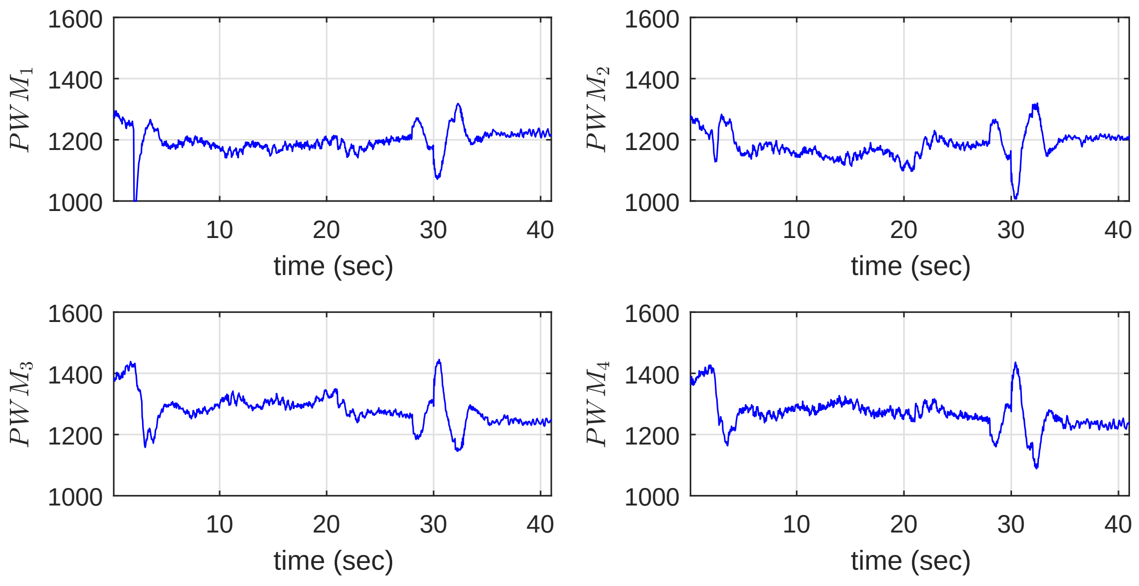


Figure 5.14: PWM signals (Faulty case: $w_1 = 0.75 \,@\, t = 2sec$)

5.5 Summary

In this chapter, an implementation of a sliding mode fault tolerant control allocation scheme using Pixhawk has been conducted. The use of the Pixhawk (though the Simulink PSP) and the gimbal test rig, provides a good test platform for rapid prototyping and testing of the advanced controller. This will be a good stepping stone before further flight tests can be conducted (either in a flying arena or outdoors). The rapid implementation using PSP could save effort and time. The results from the implementation of the sliding mode fault tolerant control allocation scheme showed good results in both fault free and faulty conditions.

Chapter 6

Spherical UAV fault tolerant control

The last chapter presented an implementation work of sliding mode in an FTC context to a quadrotor UAV. It has to be noted that typical multirotor UAV configuration such as quadrotor has become quite common in the control literature. Therefore this chapter investigates a novel UAV configuration and its FTC capability. This chapter proposes FTC schemes for a spherical UAV for the purpose of increasing reliability. The spherical UAV prototype consists of two counter-rotating propellers and eight control vanes. The proposed FTC scheme is based on combining sliding mode control with online control allocation to exploit the available actuators redundancy. The online control allocation exploits knowledge of the actuator effectiveness levels to redistribute the control signals to the healthy actuators. The controller design is based on a linear state space model at a hovering trim condition, but the simulations were done on the full nonlinear spherical UAV model in the presence of faults/failures. The simulation results show good tracking performance for various fault/failure scenarios.

6.1 Introduction

Compared to typical multirotor UAVs, spherical UAVs (e.g. Fig. 6.1 or [111]) have a lower centre of gravity (and thus are more stable) and have a spherical exterior frame which allows the UAV to land at any attitude angle, and to move on the ground. Furthermore, it is more tolerant to collisions in its environments (e.g. walls) compared to normal multirotor UAVs. At the same time, the exterior spherical frame also gives better protection while flying – especially in the event of faults/failures. The manufacturing costs for a spherical UAV is much lower as compared to a typical multirotor UAV (a quadcopter requires 4 motors and 4 electronic controllers (ESCs) as compared to one or two motors, two ESCs and some servos). The uniqueness and advantages of the spherical UAV, has caught the attention of researchers and has received an increased amount of attention in the last few years. The work in [111, 112] considers the modelling, analysis and control design (PID) of a spherical UAV with a single propeller and four flaps. The work in [113] considers a PD and $\mu-$ synthesised robust controller for a special type of spherical UAV (two propellers and four flaps). The work in [114] also considered (in simulation) a nonlinear controller exploiting disturbance observer based on adaptive neural networks for the same UAV. However, none of the work described above exploited the capability of the spherical UAVs in terms of FTC, which will be the focus of this chapter.

Whilst actuator faults, such as 'loss of effectiveness' can be categorised as matched uncertainty and 'classical' sliding mode schemes can deal with such a situation directly without any control design modification, classical sliding mode techniques cannot deal with total failures. However, the combination of sliding modes with 'control allocation' (CA) as proposed in [25], can deal with this problem without the need to reconfigure the structure of the controller. In this chapter, a sliding mode control allocation scheme (similar to the one in [25]) will be considered to achieve FTC for the spherical UAV. One of the main contributions of this chapter is the detailed description of the nonlinear mathematical model of the spherical UAV. Although loosely based on the model in [115], the description here is more detailed and includes multiple redundancies (especially the counter-rotating propeller) which makes it suitable as a development and simulation platform for any FTC scheme. The other main contribution of this chapter is that for the first time (as far as the authors are concerned), an FTC scheme have been considered for an over-actuated spherical UAV. This chapter also utilises genetic algorithm numerical search, which provide rapid solutions as compared to the 'brute force' methods considered in [25].

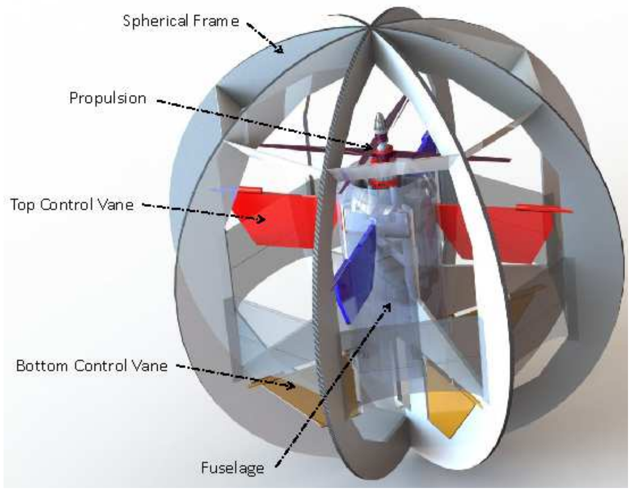
6.2 Spherical UAV equation of motion

6.2.1 Nonlinear model

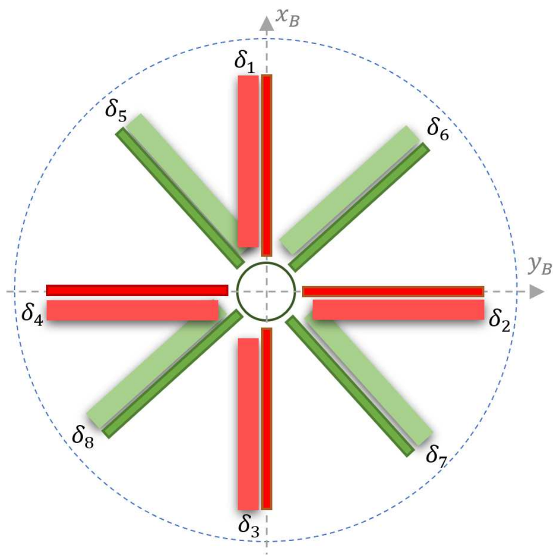
The configuration of the spherical UAV considered in this chapter is loosely based on the UAV from [115], with two counter-rotating rotors at the top and 8 flaps at the bottom and a spherical outer protective cover as an outer shell (see Figure 6.1). The additional rotor provides more capability to carry the payload and will be exploited as extra redundancy.

The states of the UAV can be represented by

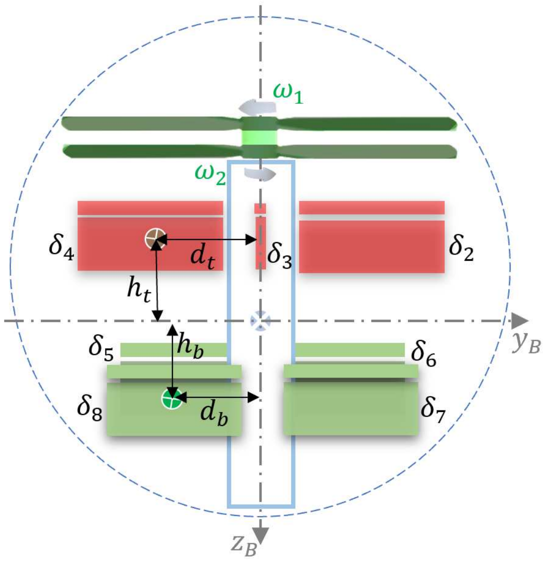
$$x_p(t) = \begin{bmatrix} \phi & \theta & \psi & u & v & w & p & q & r \end{bmatrix}^T \tag{6.1}$$



(a) overall view [115]



(b) top view



(c) side view

Figure 6.1: Spherical UAV

which represents roll angle (rad), pitch angle (rad), yaw angle (rad), velocities in the body axes u, v, w (m/s); and roll rate (rad/s), pitch rate (rad/s) and yaw rate (rad/s) respectively.

Based on [115] and the equation of motion of a rigid body aircraft in [116], with the assumptions of constant mass and rigid body motion and considering that the x and y axes are symmetric i.e. the off-diagonal moment of inertia components $I_{xy} = I_{xz} = I_{yz} = 0$, then the nonlinear equation of motions can be written as

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ \dot{u} \\ \dot{v} \\ \dot{w} \\ \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} p + qsin(\phi)tan(\theta) + rcos(\phi)tan(\theta) \\ qcos(\phi) - rsin(\phi) \\ qsin(\phi)sec(\theta) + rcos(\phi)sec(\theta) \\ vr - qw - gsin(\theta) \\ pw - ur + gcos(\theta)sin(\phi) \\ uq - pv + gcos(\theta)cos(\phi) \\ qr(I_{yy} - I_{zz})/I_{xx} \\ pr(I_{zz} - I_{xx})/I_{yy} \\ qr(I_{xx} - I_{yy})/I_{zz} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ F_x/m_{kg} \\ F_y/m_{kg} \\ F_z/m_{kg} \\ \mathcal{L}/I_{xx} \\ \mathcal{M}/I_{yy} \\ \mathcal{N}/I_{zz} \end{bmatrix} \tag{6.2}$$

where g is gravity. The forces and moments from the actuators in the x, y and z axes in (6.2) can be represented by

$$\begin{bmatrix} F_x, & F_y, & F_z, & \mathcal{L}, & \mathcal{M}, & \mathcal{N} \end{bmatrix}^T = \bar{B}u(t) \tag{6.3}$$

where $u(t)$ is the inputs given by

$$u(t) = \begin{bmatrix} \delta_1 & \delta_2 & \delta_3 & \delta_4 & \delta_5 & \delta_6 & \delta_7 & \delta_8 & \Omega_1^2 & \Omega_2^2 \end{bmatrix}^T \tag{6.4}$$

which represent four upper flaps $\delta_1, \delta_2, \delta_3, \delta_4$, four lower flaps $\delta_5, \delta_6, \delta_7, \delta_8$ and thrusts from two counter-rotating rotors Ω_1 and Ω_2. The orientations of the flaps and rotor are similar to the one in [115] as shown in Fig. 6.1.

$$\bar{B} = \left[\begin{array}{cccc|c} 0 & \bar{a} & 0 & \bar{a} & -\bar{b} \\ \bar{a} & 0 & \bar{a} & 0 & -\bar{b} \\ \bar{c}\,\mathrm{sgn}(\delta_1) & \bar{c}\,\mathrm{sgn}(\delta_2) & \bar{c}\,\mathrm{sgn}(\delta_3) & \bar{c}\,\mathrm{sgn}(\delta_4) & \bar{d}\,\mathrm{sgn}(\delta_5) \\ \bar{a}h_t & \bar{e}\,\mathrm{sgn}(\delta_2) & \bar{a}h_t & -\bar{e}\,\mathrm{sgn}(\delta_4) & \bar{b}h_b - \bar{f}\,\mathrm{sgn}(\delta_5) \\ -\bar{e}\,\mathrm{sgn}(\delta_1) & \bar{a}h_t & \bar{e}\,\mathrm{sgn}(\delta_3) & \bar{a}h_t & \bar{b}h_b - \bar{f}\,\mathrm{sgn}(\delta_5) \\ \bar{a}d_t & -\bar{a}d_t & -\bar{a}d_t & \bar{a}d_t & -\bar{b}d_b\sqrt{2} \end{array}\right.$$

$$\left.\begin{array}{ccc|cc} -\bar{b} & -\bar{b} & -\bar{b} & 0 & 0 \\ \bar{b} & -\bar{b} & \bar{b} & 0 & 0 \\ \bar{d}\,\mathrm{sgn}(\delta_6) & \bar{d}\,\mathrm{sgn}(\delta_7) & \bar{d}\,\mathrm{sgn}(\delta_8) & -k_t & -k_t \\ -\bar{b}h_b + \bar{f}\,\mathrm{sgn}(\delta_6) & \bar{b}h_b + \bar{f}\,\mathrm{sgn}(\delta_7) & -\bar{b}h_b - \bar{f}\,\mathrm{sgn}(\delta_8) & 0 & 0 \\ \bar{b}h_b - \bar{f}\,\mathrm{sgn}(\delta_6) & \bar{b}h_b + \bar{f}\,\mathrm{sgn}(\delta_7) & \bar{b}h_b + \bar{f}\,\mathrm{sgn}(\delta_8) & 0 & 0 \\ \bar{b}d_b\sqrt{2} & \bar{b}d_b\sqrt{2} & -\bar{b}d_b\sqrt{2} & -k_m & k_m \end{array}\right] \tag{6.5}$$

The term $\bar{B}$ in (6.3) is defined in (6.5) where the constants are defined as

$$\begin{aligned} \bar{a} &= \bar{Q}S_tC_{L_\alpha}, & \bar{b} &= \bar{Q}S_bC_{L_\alpha}cos(\pi/4) \\ \bar{c} &= QS_tC_{D_\alpha}, & \bar{d} &= \bar{Q}S_bC_{D_\alpha} \\ \bar{e} &= \bar{Q}S_tC_{D_\alpha}d_t, & \bar{f} &= \bar{Q}S_bC_{D_\alpha}d_bcos(\pi/4) \end{aligned} \tag{6.6}$$

where $\bar{Q} = \frac{1}{2}\rho_{air}V^2$ represents the dynamic pressure, ρ_{air} is the air density, the velocity V is given by $V = \sqrt{u^2 + v^2 + w^2}$ and S_t, S_b, $C_{L\alpha}$ and $C_{D\alpha}$ are the upper and lower fin surface areas and aerodynamic lift and drag curve slope coefficients respectively. In (6.5), h_t and h_b represent the vertical distances between the vane's aerodynamic centre and the UAV centre of gravity for the upper and lower vanes respectively (see Fig. 6.1). The terms d_t and d_b represent the horizontal distances between the vanes aerodynamic centre and the UAV centre of gravity for the upper and lower vanes respectively. Finally k_t and k_m represent the motor thrust and torque coefficients respectively.

Using the analysis in [116], the velocity (and therefore position) of the UAV in the earth axes can be obtained using the following conversion from the body axes speed u, v, w

$$\begin{bmatrix} \dot{x}_e \\ \dot{y}_e \\ \dot{z}_e \end{bmatrix} = \begin{bmatrix} cos(\theta)cos(\psi) & sin(\phi)sin(\theta)cos(\psi) - cos(\phi)sin(\psi) & cos(\phi)sin(\theta)cos(\psi) + sin(\phi)sin(\psi) \\ cos(\theta)sin(\psi) & sin(\phi)sin(\theta)sin(\psi) + cos(\phi)cos(\psi) & cos(\phi)sin(\theta)sin(\psi) - sin(\phi)cos(\psi) \\ -sin(\theta) & sin(\theta)cos(\theta) & cos(\phi)cos(\theta) \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{6.7}$$

6.2.2 Linearisation

As in [115], since the drag from the fins is small relative to the lift coefficient, the scalars $\bar{c}, d, e, f$ are neglected and therefore (6.5) can be written as

$$\bar{B} = \left[\begin{array}{cccc|cccc|cc} 0 & \bar{a} & 0 & \bar{a} & -\bar{b} & -\bar{b} & -\bar{b} & -\bar{b} & 0 & 0 \\ \bar{a} & 0 & \bar{a} & 0 & -\bar{b} & \bar{b} & -\bar{b} & \bar{b} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -k_t & -k_t \\ \bar{a}h_t & 0 & \bar{a}h_t & 0 & \bar{b}h_b & -\bar{b}h_b & \bar{b}h_b & -\bar{b}h_b & 0 & 0 \\ 0 & \bar{a}h_t & 0 & \bar{a}h_t & \bar{b}h_b & \bar{b}h_b & \bar{b}h_b & \bar{b}h_b & 0 & 0 \\ \bar{a}d_t & -\bar{a}d_t & -\bar{a}d_t & \bar{a}d_t & -\bar{b}d_b\sqrt{2} & \bar{b}d_b\sqrt{2} & \bar{b}d_b\sqrt{2} & -\bar{b}d_b\sqrt{2} & -k_m & k_m \end{array}\right] \tag{6.8}$$

The trim value for the two rotor speeds during a stable hover trim condition can be obtained using the $\dot{w}$ equation in (6.2). During hover $\dot{w} = 0$, then assuming that both rotors rotate at the same speed, the trim value is given by $\Omega_1 = \Omega_2 = \sqrt{m_{kg} g/(2k_t)}$. Since the counter-rotating rotors cancelled the yaw motion, all the other 8 flaps trim values are zero. The other 9 states trim values are also zero. Therefore linearizing the nonlinear equations of motion in (6.2) about a stable hover condition yield

$$\dot{x}_p(t) = A_p x_p(t) + B_p u(t) \tag{6.9}$$

where the system pair (A_p, B_p) is given by

$$A_p = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & -g & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ g & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{6.10}$$

$$B_p = \left[\begin{array}{cccc|c} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & \bar{a}/m_{kg} & 0 & \bar{a}/m_{kg} & -\bar{b}/m_{kg} \\ \bar{a}/m_{kg} & 0 & \bar{a}/m_{kg} & 0 & -\bar{b}/m_{kg} \\ 0 & 0 & 0 & 0 & 0 \\ \bar{a}h_t/I_{xx} & 0 & \bar{a}h_t/I_{xx} & 0 & \bar{b}h_b/I_{xx} \\ 0 & \bar{a}h_t/I_{yy} & 0 & \bar{a}h_t/I_{yy} & \bar{b}h_b/I_{yy} \\ \bar{a}d_t/I_{zz} & -\bar{a}d_t/I_{zz} & -\bar{a}d_t/I_{zz} & \bar{a}d_t/I_{zz} & -\bar{b}d_b\sqrt{2}/I_{zz} \end{array}\right.$$

$$\left.\begin{array}{ccc|cc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -\bar{b}/m_{kg} & -\bar{b}/m_{kg} & -\bar{b}/m_{kg} & 0 & 0 \\ \bar{b}/m_{kg} & -\bar{b}/m_{kg} & \bar{b}/m_{kg} & 0 & 0 \\ 0 & 0 & 0 & -k_t/m_{kg} & k_t/m_{kg} \\ -\bar{b}h_b/I_{xx} & \bar{b}h_b/I_{xx} & -\bar{b}h_b/I_{xx} & 0 & 0 \\ \bar{b}h_b/I_{yy} & \bar{b}h_b/I_{yy} & \bar{b}h_b/I_{yy} & 0 & 0 \\ \bar{b}d_b\sqrt{2}/I_{zz} & \bar{b}d_b\sqrt{2}/I_{zz} & -\bar{b}d_b\sqrt{2}/I_{zz} & -k_m/I_{zz} & k_m/I_{zz} \end{array}\right] \tag{6.11}$$

6.3 Controller synthesis

6.3.1 System with faults/failures

Consider a generic n-th order linear time invariant system with m inputs.

The system is considered to be an over actuated system in the presence of faults/failures, the system can be represented by the following expression [25]:

$$\dot{x}(t) = Ax(t) + BWu(t) \tag{6.12}$$

where $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$. The matrix $W \in \mathbb{R}^{m\times m}$ is diagonal with the diagonal element $0 \leq w_i \leq 1$. If $w_i = 1$, it means that the corresponding control element u_i has no problem and is working efficiently. If $w_i = 0$ it means that u_i has a complete failure and the corresponding actuator is completely out of service. If $0 < w_i < 1$, then u_i is faulty but the actuator can still be used.

In the equations of motion of aircraft, forces and moments have a direct effect mainly on the linear and angular accelerations of the aircraft i.e. the velocity states. The Euler angles and displacement states are not directly affected by forces and moments. Consequently, for most aircraft systems, the B matrix can be factorized into two submatrices

$$B = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} \tag{6.13}$$

where $B_1 \in \mathbb{R}^{(n-l)\times m}$ and $B_2 \in \mathbb{R}^{l\times m}$ and $l < m$. The sub-partition B_2 is associated with the ability of the inputs (2 rotors and 8 flap surfaces) to produce total vertical thrust as well as roll, pitch and yaw moments. Meanwhile, the sub-partition B_1 is associated with the effect of the inputs on the forces of the UAV. It is a well-known characteristic of aircraft that input surfaces will be much more effective in generating total thrust and moments than in causing changes in forces, and therefore, B_2 has a more dominant contribution than B_1 i.e. $\|B_2\| \gg \|B_1\|$. To aid in the controller synthesis, since $\text{rank}(B_2) = l$, it will be assumed without loss of generality that the states of the system in (6.12) have been transformed so that $B_2B_2^T = I_l$ and therefore $\|B_2\| = 1$.

6.3.2 Controller synthesis

Here it is assumed that the states of the system in (6.12) have been pre-transformed so that $B_2B_2^T = I_l$ (this will be exploited later in the proofs). Hence in this section, consider a faulty Linear Time Invariant (LTI) system

$$\dot{x}(t) = Ax(t) + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} Wu(t) \tag{6.14}$$

Assume

$$\nu(t) = B_2 u(t) \tag{6.15}$$

where $\nu(t) \in \mathbb{R}^l$ is a virtual control input to the system. Hence

$$u(t) = B_2^{\dagger} \nu(t) \tag{6.16}$$

where $B_2^{\dagger}$ is the pseudo inverse of B_2 matrix given by

$$B_2^{\dagger} = WB_2^T (B_2 W B_2^T)^{-1} \tag{6.17}$$

satisfies (6.15). Substituting (6.16) and (6.17) into (6.14) yields

$$\dot{x}(t) = Ax(t) + \begin{bmatrix} B_1 W B_2^{\dagger} \\ B_2 W B_2^{\dagger} \end{bmatrix} \nu(t) \tag{6.18}$$

Define

$$\bar{\nu}(t) = (B_2 W B_2^T)^{-1} \nu(t) \tag{6.19}$$

then equation (6.18) can be written as

$$\dot{x}(t) = Ax(t) + \begin{bmatrix} B_1 B_2^T \\ I \end{bmatrix} \bar{\nu}(t) - \begin{bmatrix} B_1 (I - W^2) B_2^T \\ B_2 (I - W^2) B_2^T \end{bmatrix} \bar{\nu}(t) \tag{6.20}$$

Notice that in the fault-free case where $W = I$ the last term in equation (6.20) will be zero. To get the sliding mode control regular form for the fault-free case as in [101], a state transformation $x \mapsto T_r x(t) = \hat{x}(t)$ will be introduced where

$$T_r = \begin{bmatrix} I & -B_1 B_2^T \\ 0 & I \end{bmatrix} \tag{6.21}$$

then (6.20) will become

$$\dot{\hat{x}}(t) = \hat{A}\hat{x}(t) + \begin{bmatrix} 0 \\ I \end{bmatrix} \bar{\nu}(t) - \begin{bmatrix} B_1 B_2^N (I - W^2) B_2^T \\ B_2 (I - W^2) B_2^T \end{bmatrix} \bar{\nu}(t) \tag{6.22}$$

where $\hat{A} = T_r A T_r^{-1}$ and

$$B_2^N = (I - B_2^T B_2) \tag{6.23}$$

From the assumption that $B_2 B_2^T = I_l$, it follows directly that $B_2^N B_2^T = B_2^T - B_2^T B_2 B_2^T = 0$, hence

$$\dot{\hat{x}}(t) = \hat{A}\hat{x}(t) + \begin{bmatrix} 0 \\ B_2 W^2 B_2^T \end{bmatrix} \bar{\nu}(t) + \begin{bmatrix} B_1 B_2^N W^2 B_2^T \\ 0 \end{bmatrix} \bar{\nu}(t) \tag{6.24}$$

Consider

$$\hat{\nu}(t) = B_2 W^2 B_2^T \bar{\nu}(t) \tag{6.25}$$

then

$$\dot{\hat{x}}(t) = \hat{A}\hat{x}(t) + \underbrace{\begin{bmatrix} 0 \\ I \end{bmatrix}}_{B_\nu} \hat{\nu}(t) + \begin{bmatrix} B_1 B_2^N B_2^+ \\ 0 \end{bmatrix} \hat{\nu}(t) \tag{6.26}$$

where

$$B_2^+ = W^2 B_2^T (B_2 W^2 B_2^T)^{-1} \tag{6.27}$$

It can be shown in the fault-free case, $B_1 B_2^N B_2^+ = 0$. The sliding mode control will be designed based on the nominal fault-free system in its regular form exploiting this property.

In this chapter, sliding mode technique will be considered for the synthesis of the 'virtual' control law $\hat{\nu}$ in (6.26). Define the switching function $s(t) : \mathbb{R}^n \to \mathbb{R}^l$ as

$$s(t) = S\hat{x}(t) \tag{6.28}$$

where $S \in \mathbb{R}^{l \times n}$ and $SB_\nu = I_l$. The objective is to design the 'virtual' control to force the closed loop trajectory of the system on to the surface $\mathcal{S} = \{\hat{x} \in \mathbb{R}^n : Sx(t) = 0\}$ in finite time and make it remain there [25, 101].

In the regular form coordinate, one suitable choice of S is

$$S = \begin{bmatrix} M & I_l \end{bmatrix} \tag{6.29}$$

where $M \in \mathbb{R}^{l \times (n-l)}$. The state transformation

$$T_s = \begin{bmatrix} I & 0 \\ M & I \end{bmatrix} \tag{6.30}$$

results in

$$\begin{bmatrix} \dot{\hat{x}}_1(t) \\ \dot{s}(t) \end{bmatrix} = \begin{bmatrix} \tilde{A}_{11} & \tilde{A}_{12} \\ \tilde{A}_{21} & \tilde{A}_{22} \end{bmatrix} \begin{bmatrix} \hat{x}_1(t) \\ s(t) \end{bmatrix} + \begin{bmatrix} B_1 B_2^N B_2^+ \\ I + M B_1 B_2^N B_2^+ \end{bmatrix} \hat{\nu}(t) \tag{6.31}$$

where $\widetilde{A}_{11} = \hat{A}_{11} - \hat{A}_{12}M$, $\tilde{A}_{21} = M\tilde{A}_{11} + \hat{A}_{21} - \hat{A}_{22}M$ and $\tilde{A}_{22} = M\hat{A}_{12} + \hat{A}_{22}$. During sliding, $s(t) = \dot{s}(t) = 0$ then from the upper and lower equations in (6.31)

$$\dot{\hat{x}}_1(t) = \tilde{A}_{11}\hat{x}_1(t) + B_1 B_2^N B_2^+ \hat{\nu}_{eq}(t) \tag{6.32}$$

and

$$\tilde{A}_{21}\hat{x}_1(t) + (I + MB_1B_2^N B_2^+)\hat{\nu}_{eq}(t) = 0 \tag{6.33}$$

Rearranging (6.33) to obtain an expression for $\hat{\nu}_{eq}(t)$ and substituting ((6.32)) yields

$$\dot{\hat{x}}_1(t) = \tilde{A}_{11}\hat{x}_1(t) - B_1B_2^N B_2^+(I + MB_1B_2^N B_2^+)^{-1}\tilde{A}_{21}\hat{x}_1(t) \tag{6.34}$$

6.3.3 Stability analysis

Equation (6.34) which governs the sliding motion must be checked for stability as the second term was not taken into consideration while designing the control law. Consider rewriting (6.34) as follows

$$\dot{x}_1(t) \quad = \quad \tilde{A}_{11}x_1(t) + B_1B_2^N\tilde{u}(t) \tag{6.35}$$

and

$$\tilde{y}(t) \quad = \quad \tilde{A}_{21}x_1(t) \tag{6.36}$$

then if

$$\tilde{y}(s) = G(s)\tilde{u}(s) \tag{6.37}$$

where

$$G(s) = \tilde{A}_{21}(sI - A_{11})^{-1}B_1B_2^N \tag{6.38}$$

Equation (6.34) can then be considered as the closed-loop system obtained for using the 'control law'

$$\tilde{u}(t) = -B_2^+(I + MB_1B_2^N B_2^+)^{-1}\tilde{y}(t) \tag{6.39}$$

From the small gain theorem [105], if

$$\|G(s)\|_\infty \left\|B_2^+(I + MB_1B_2^N B_2^+)^{-1}\right\| < 1 \tag{6.40}$$

then (6.34) will be stable. From [25], the closed-loop system will be stable if the following condition is satisfied

$$0 \leq \frac{\gamma_2\gamma_0}{1-\gamma_1\gamma_0} < 1 \tag{6.41}$$

where the scalars

$$\gamma_0 = \|B_2^+\| \tag{6.42}$$

$$\gamma_1 = \|MB_1B_2^N\| \tag{6.43}$$

$$\gamma_2 = \|G(s)\|_\infty \tag{6.44}$$

Note that γ_1 and γ_2 depend on the choice of sliding surface.

6.3.4 Control law

The virtual sliding mode controller $\hat{\nu}$ is designed based on the system in (6.31) in a fault-free condition and will be guaranteed to be stable for all combinations of faults/failures if condition (6.41) is satisfied. Here, the virtual controller $\hat{\nu}$ is a combination of a linear and nonlinear structure given by

$$\hat{\nu}(t) = \hat{\nu}_l(t) + \hat{\nu}_n(t) \tag{6.45}$$

where

$$\hat{\nu}_l(t) = \tilde{A}_{21}\hat{x}_1(t) - \tilde{A}_{22}s(t) \tag{6.46}$$

while

$$\hat{\nu}_n(t) = -\rho(t,x)\frac{s(t)}{\|s(t)\|} \quad \text{if } s(t) \neq 0 \tag{6.47}$$

while the switching function $s(t)$ is defined in (6.28). It is shown in [25] that if the following conditions are satisfied, then a sliding motion will take place on $\mathcal{S}$ in finite time:

- matrix M from (6.29) must be chosen to ensure that $\tilde{A}_{11} = \hat{A}_{11} - \hat{A}_{12}M$ from (6.31) stable,
- condition in (6.41) holds,
- the modulation gain $\rho(t,x)$ from (6.47) is chosen to satisfies

$$\rho(t,x) = \frac{\gamma_1\gamma_0\|\hat{\nu}_l(t)\| + \eta}{1-\gamma_1\gamma_0} \tag{6.48}$$

  where η is a small positive scalar.

Note that the sliding mode has been designed using the virtual control $\nu(t)$ in (6.45) based on the virtual system (6.31). The actual control $u(t)$ sent to all 10 actuators (2 motors and 8 flaps) is given by

$$u(t) = WB_2^T(B_2W^2B_2^T)^{-1}\hat{\nu}(t) \tag{6.49}$$

which is determined from (6.16), (6.19), (6.25) and (6.45).

6.4 Design and simulations

Here, the physical UAV parameters (i.e. mass and inertia and the motor thrust and torque coefficients) are similar to the one from [114] as summarised in Table 6.1. The stable hover trim is $\Omega = 439.2rad/s$ for both rotors. Substituting the values from Table 6.1 into (6.10)-(6.11), yields the (A, B) matrix pair used for controller design.

Table 6.1: UAV Physical Parameters

Parameter	$Value$	Unit
m_{kg}	1.18	kg
I_{xx}	3393×10^{-6}	kgm^2
I_{yy}	3918×10^{-6}	kgm^2
I_{zz}	2745×10^{-6}	kgm^2
k_t	3×10^{-5}	kgm
k_m	7.5×10^{-7}	kgm^2
d_b, d_t	$0.2, 0.2$	m
h_b, h_t	$0.2, 0.2$	m

6.4.1 States tracking, integral action and augmented system

The objective is to maintain tracking performance for yaw ψ as well as the velocities in the body axes u, v, w. To provide tracking performance, integral action states [25, 101] are augmented into the plant states. The integral action states are given by

$$\dot{x}_c(t) = y_c(t) - C_c x \tag{6.50}$$

where

$$C_c = \begin{bmatrix} 0_{4\times2} & I_4 & 0_{4\times3} \end{bmatrix} \tag{6.51}$$

is the distribution matrix associated with the tracked outputs ψ, u, v, w. The reference signal $y_c(t)$ is differentiable and is chosen to satisfy

$$\dot{y}_c(t) = \Gamma(y_c - Y_c) \tag{6.52}$$

where Y_c is the demand vector while $\Gamma \in \mathbb{R}^{4\times4}$ is chosen to be stable. Augmenting the integral states from (6.50) with the system states in (6.12) yields

$$\underbrace{\begin{bmatrix} \dot{x}_c(t) \\ \dot{x}(t) \end{bmatrix}}_{\dot{x}_a} = \underbrace{\begin{bmatrix} 0 & C_c \\ 0 & A \end{bmatrix}}_{A_a} \underbrace{\begin{bmatrix} x_r(t) \\ x \end{bmatrix}}_{x_a(t)} + \underbrace{\begin{bmatrix} 0 \\ B \end{bmatrix}}_{B_a} u(t) + \underbrace{\begin{bmatrix} I_4 \\ 0 \end{bmatrix}}_{B_c} y_c(t) \tag{6.53}$$

It is shown in [101] that the pair (A_a, B_a) is controllable if the pair (A, B) is controllable and (A, B, C_c) does not have any zeros at the origin. The switching function for the augmented systems $s_a \in \mathbb{R}^4$ becomes

$$s_a = S_a \hat{x}_a \tag{6.54}$$

where S_a is in the coordinate system of (6.28), The control law in (6.46)-(6.47) becomes

$$\hat{\nu}_l(t) = \tilde{A}_{a,21}\hat{x}_1(t) - \tilde{A}_{a,22}s(t) + B_{c,1}y_c(t) \tag{6.55}$$

while

$$\hat{\nu}_n(t) = -\rho(t,x)\frac{s_a(t)}{\|s_a(t)\|} \quad \text{if } s_a(t) \neq 0 \tag{6.56}$$

Note that the last term in (6.55) contains the feed-forward states command term. The final control law sent to the actuator has the same form as in (8.67) with the virtual control law from (6.55) and (6.56) with appropriate B_2 for the augmented system.

6.4.2 Design and stability analysis

In order to provide tracking for ψ, u, v, w, when faults/failures occur, it is assumed that the system has at least four functional actuators. Specifically, at least one motor (to control w) and three flaps are available (to control ψ, u, v). Based on these assumptions, a numerical search using Genetic Algorithm (GA) [117] has been conducted to find the worst possible γ_0 from (6.42) for all possible combinations of faults/failures. Here, the GA objective is to maximise the fitness function (equation (6.42)) where the 'decision variables' are $w_1, \ldots, w_{10}$ (the effectiveness levels of flaps 1-8 and rotors 1-2 as described in (6.4)). An initial random population size of 20 (for each decision variables), a mutation size of 0.01, 'best' individual for parent selection, 'maximum' fitness for new generation and a maximum of 1000 iterations are used for the numerical search [117], which is repeated ten times with random initial population to ensure that the solution obtained is the global maxima. The results from the GA search yield $\gamma_0 = 3.3417$. Note that using GA is an improvement to the 'brute force' methods that have been used previously used in [25].

Here, a solution can be found much faster despite the ten repetitions of the algorithm.

A quadratic optimal design has been used to design the sliding hyperplane S_a in (6.54), which depends on the matrix M associated with equation (6.29) [25, 101]. The sliding hyperplane was synthesised so that the conditions in (6.41) are satisfied. The symmetric positive definite state weighting matrix has been chosen as $Q = \text{diag}(100, 10I_3, I_9)$ which results in the poles for the reduced order sliding motion being given by $\{-2.2913 \pm 2.1794i, -2.0924 \pm 2.7141i, -2.0924 \pm 2.7141i, -2.6414, -3.1623, -2.6414\}$. Based on this designed M, simple calculations using (6.43) and (6.44) yield $\gamma_1 = 0.0772$ and $\gamma_2 = \|G(s)\|_\infty = 0.1064$ respectively. Therefore, $\gamma_1\gamma_0 = 0.2579 < 1$ and

$$0 \leq \frac{\gamma_2\gamma_0}{1-\gamma_1\gamma_0} = 0.4793 < 1$$

which satisfies the conditions in (6.41). The pre-filter Γ in (6.52) has been chosen as $\Gamma = -10I_4$. The nonlinear modulation gain in (6.56) has been chosen as $\rho = diag(2, I_3)$. To reduce 'chattering', the discontinuous signum term in (6.56) has been replaced with a 'smooth' sigmoidal approximation $\frac{s_a(t)}{\|s_a(t)\|+\delta}$ where the smoothing term has been chosen as $\delta = 0.01$.

The final control law in (8.67) sent to the actuators is dependent on the effectiveness levels of the actuators through the matrix W. In this chapter, it will be assumed that this information is available either from fault detection and isolation (FDI) units (see for example fault reconstructions schemes in [25]) or direct measurements of the two rotors speeds and deflections of the flaps.

In order to provide the outer loop position (x_e, y_e, z_e) control, proportional controllers are used to provide the demand signals for the inner loop u, v, w. The proportional gains are set as $K_{x_e} = 0.5$, $K_{y_e} = 0.5$ and $K_{z_e} = 0.2918$ for x_e, y_e, z_e position controls respectively. The yaw angle is controlled using the inner-loop controller directly.

6.5 Results

The results presented here are from the nonlinear model described in equations (6.1)-(6.7). The simulation was conducted using SIMULINK with a fixed time solver *ode*3 with a time step of $0.001s$. Three different sets of tests have been considered; one fault-free and two with failure conditions. The configuration and locations of the actuator failures are described in Table 6.2. For failure Case 1, one of the rotors as well as the lower flaps 5-8 have failed. The failure Case 2 considers a more challenging scenario in which only the two rotors and flaps 7 and 8 are available. In all test cases, the same manoeuvre has been considered for comparison purposes (as shown in Figures 6.2 and 6.3). Figure 6.2

Table 6.2: Test configurations

config.	w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9	w_{10}
fault-free	1	1	1	1	1	1	1	1	1	1
failure 1	1	1	1	1	0	0	0	0	1	0
failure 2	0	0	0	0	0	0	1	1	1	1

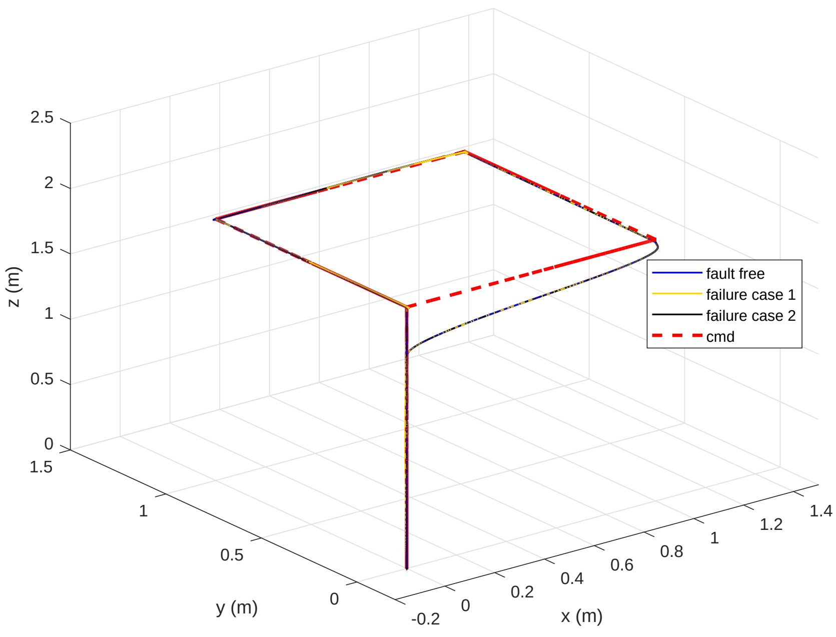


Figure 6.2: Trajectory of the UAV (fault-free, failure case 1 and 2)

shows the trajectory of the UAV for the three cases (vertical takeoff, followed by a square change of position and finally land at the same spot as takeoff). Figure 6.3 shows tracking performance for position control (which provides the ψ, u, v, w commands signals for the inner loop sliding mode scheme). Figure 6.2 shows a small position deviation at the start of the simulation for the failure challenging Case 2 – although Figure 6.3 shows similar tracking performance between the failure cases and the fault-free case.

6.5.1 Fault-free

Figure 6.4 shows the responses for the fault-free case. Figure 6.4a shows good ψ, u, v, w tracking performances where the red dashed lines are the command signals from the outer loop position control, while the solid blue line is the actual state response (note that the yaw (ψ) command is set to zero). Figure 6.4a also shows the roll, roll rate, pitch and pitch rate responses.

Figure 6.4b shows no visible deviation of the switching functions $s(t)$ from zero, indicating a good sliding motion during fault-free condition. Figure 6.4c shows the eight flaps deflections and the two motors speeds (left y-axis – blue lines) and their respective

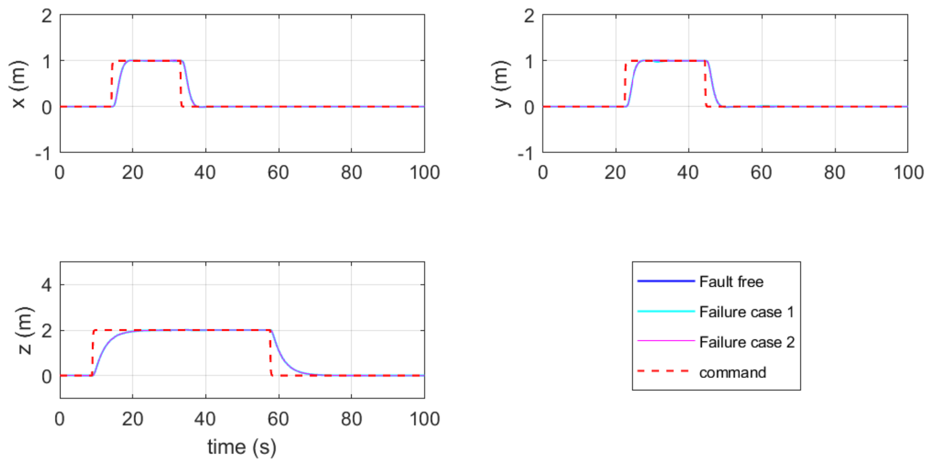


Figure 6.3: x, y, z position (fault-free, failure case 1 and 2)

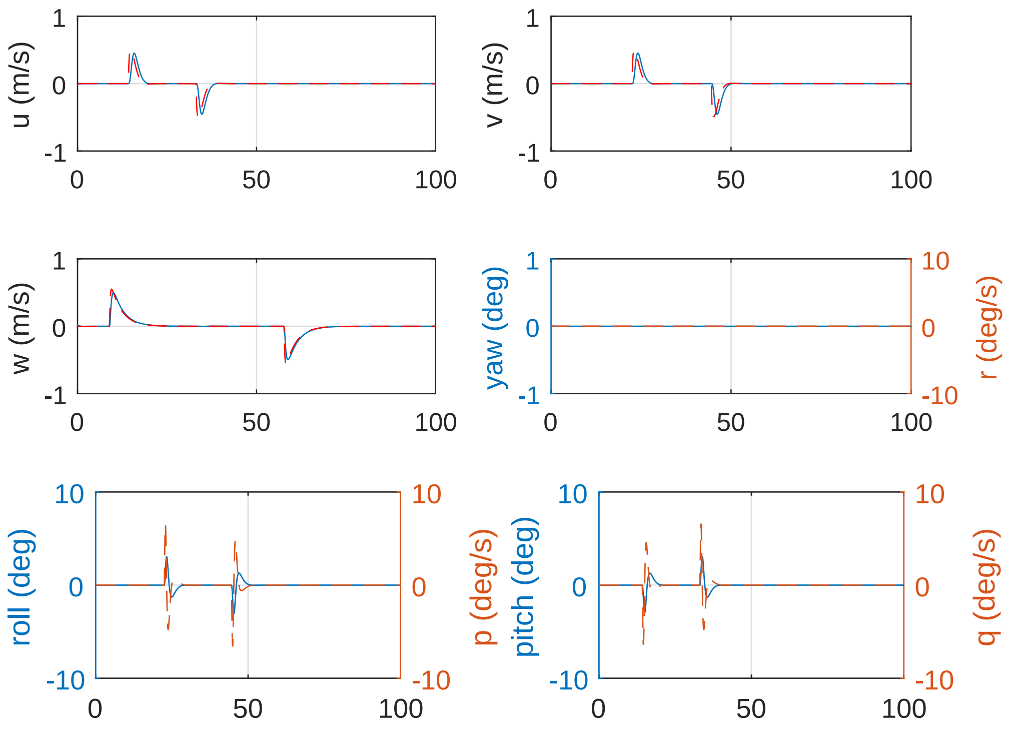
effectiveness levels where $w_1, \ldots, w_{10} = 1$ for the fault-free case (right y-axis – red lines). The flaps deviate around the 0 deg trim position during manoeuvres, while the two motors vary around the hover trim rotor speed.

6.5.2 Failure case 1

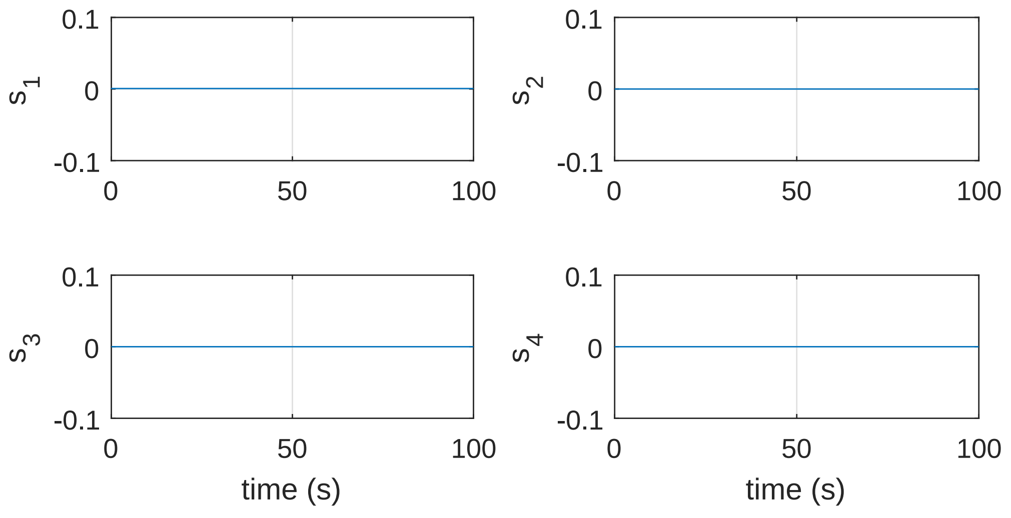
Figure 6.5 shows the results for failure Case 1 as specified in Table 6.2, where one of the rotors, as well as the lower flaps 5-8 fail at 30 sec (total loss of effectiveness). Despite the presence of actuator failures, the tracking performance is similar to the fault-free case (Fig. 6.5a). There is a small nonzero roll angle due to the asymmetry condition (from the loss of one of the motors, which is compensated by the flaps), however, the states tracking remain unaffected. Figure 6.5b shows sliding are being maintained as switching functions remain close to zero despite the presence of failures. Figure 6.5c shows the effect when flaps 5-8 and rotor 2 failed at 30 sec. Consequently, after the failure, flaps 1-4 and rotor 1 become more active to compensate for the failed actuators.

6.5.3 Failure case 2

Figure 6.6 shows the results for the challenging failure Case 2 as specified in Table 6.2. Here flaps 1-6 have fail at 30 sec (blue lines) and its corresponding effectiveness levels drop to zero (red lines) (see Figure 6.6c). Subsequently, flap 7-8, as well as rotor 1 and 2 are used to compensate for the failed actuators. The same level of tracking performance is maintained as seen in Figure 6.6a. Again, as in the previous cases, sliding is maintained and the switching functions remain close to zero despite the presence of failures (see Figure 6.6b).

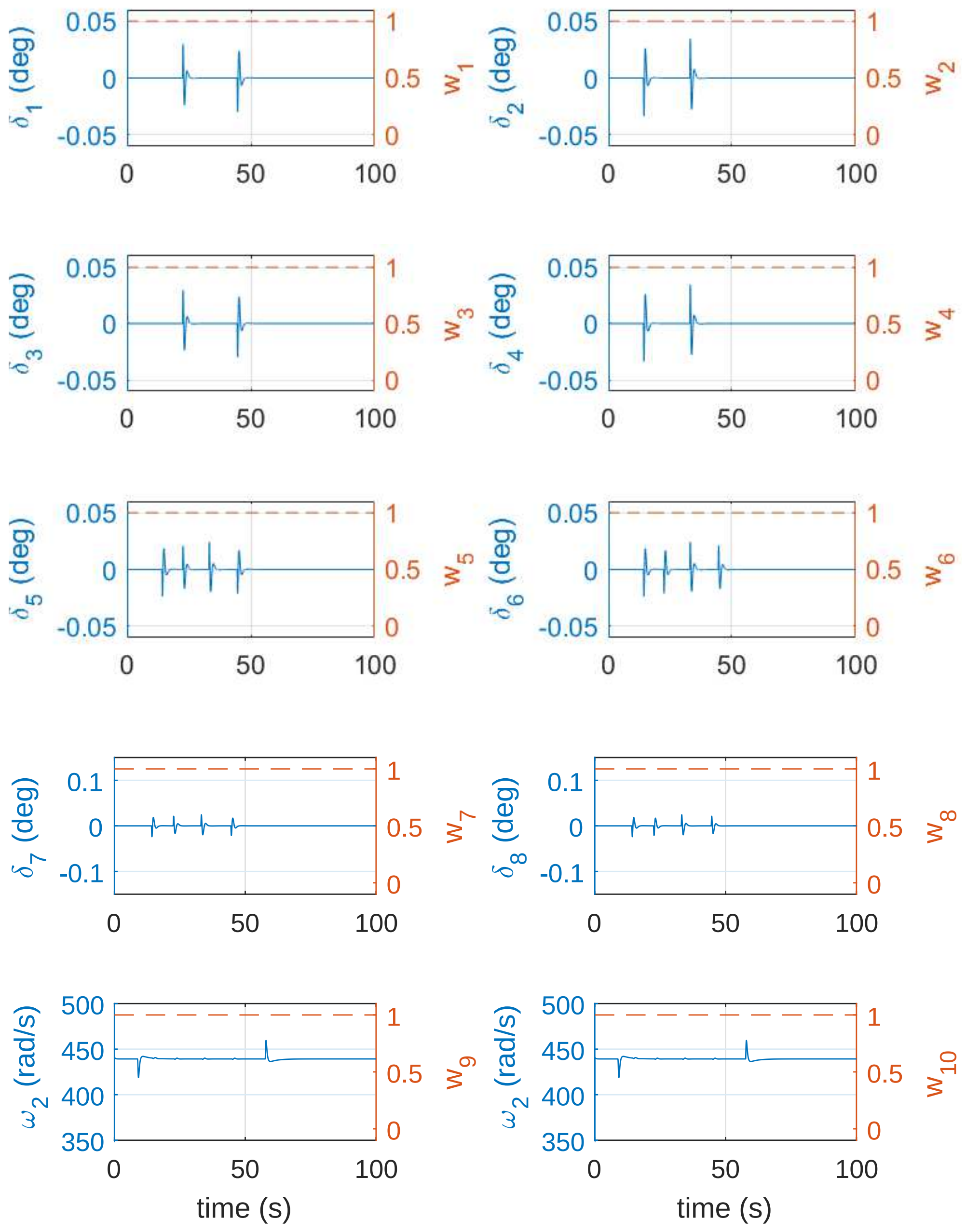


(a) States



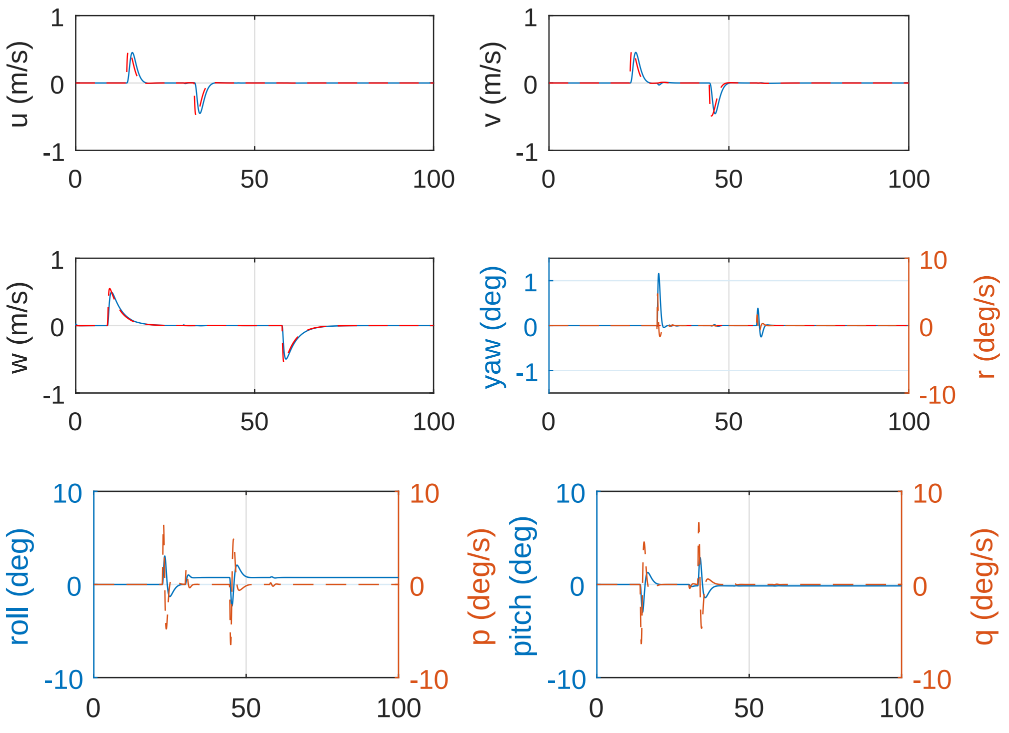
(b) Switching Functions

Figure 6.4: Fault-Free

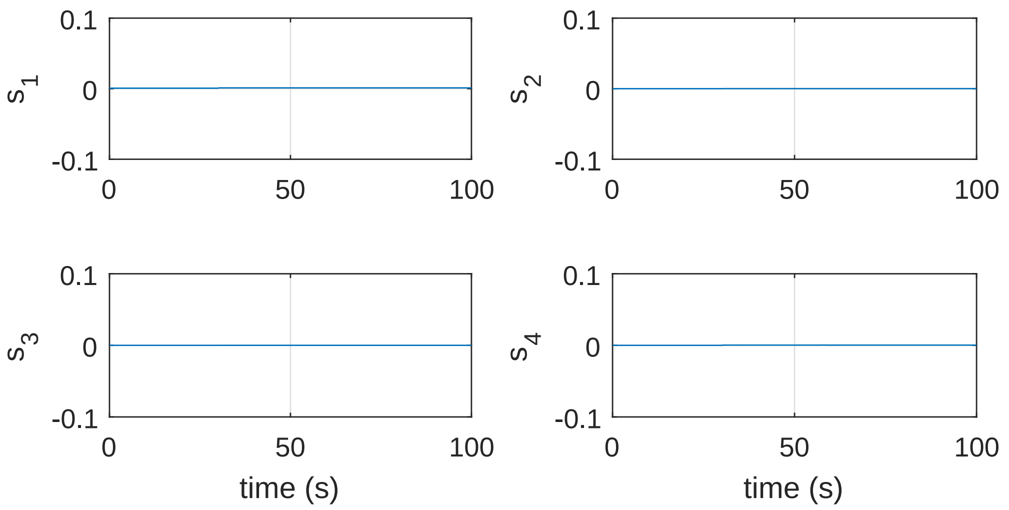


(c) Control Surface Deflections and Effectiveness

Figure 6.4: Fault-Free

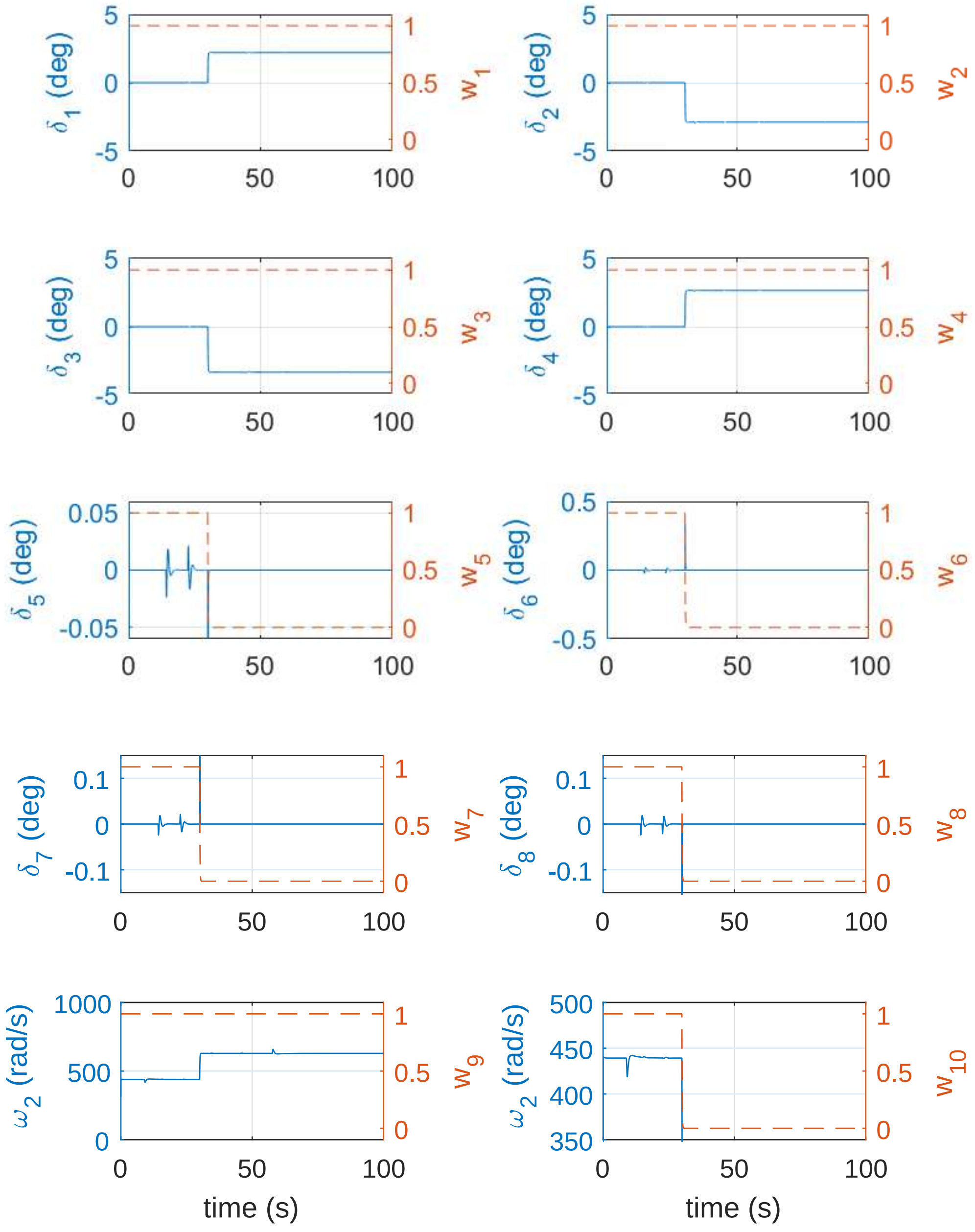


(a) states



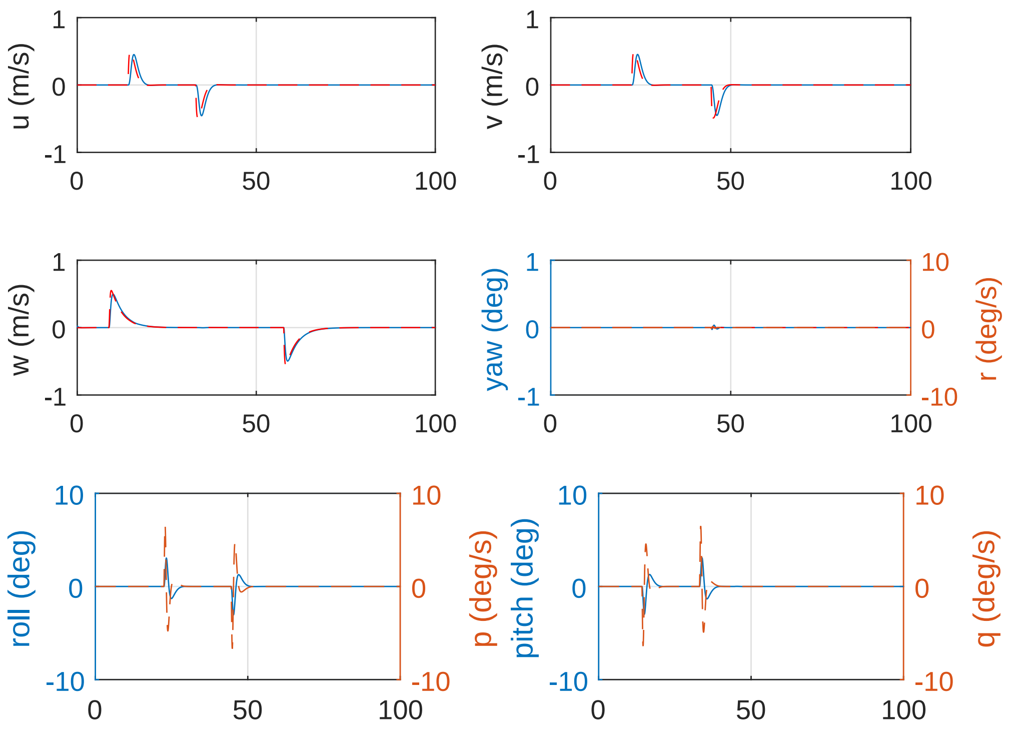
(b) switching functions

Figure 6.5: Failure case 1

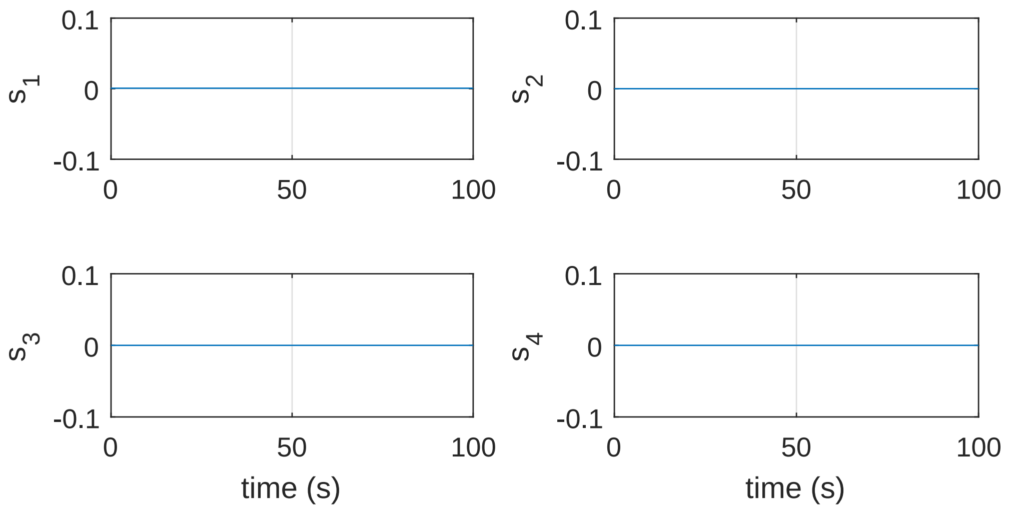


(c) control surface deflections and effectiveness

Figure 6.5: Failure case 1

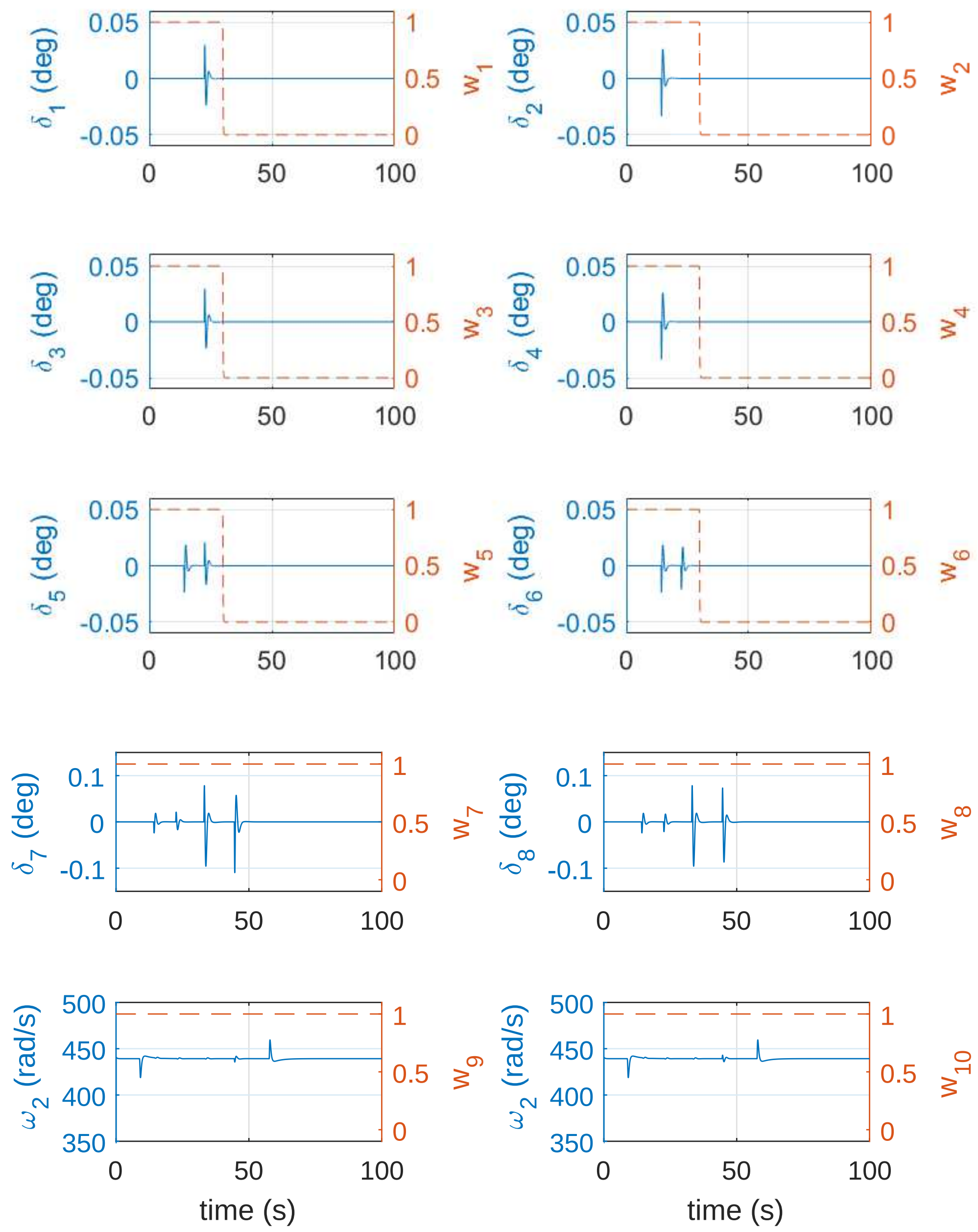


(a) states



(b) switching functions

Figure 6.6: Failure case 2



(c) control surface deflections and effectiveness

Figure 6.6: Failure case 2

6.6 Summary

This chapter has presented an FTC scheme for a spherical UAV. A detailed nonlinear mathematical model development of the spherical UAV with redundant flaps and two counter-rotating rotors has been discussed in the chapter. Unlike most of the work on similar spherical UAV in the literature, the scheme proposed in this chapter exploits the available redundant flaps and rotors to achieve a resilient UAV that is tolerant to faults and failures. The proposed FTC scheme is based on the combination of sliding mode ideas and online control allocation. The idea is to use the effectiveness levels of the actuators in the event of actuator faults/failures to redistribute the control signals to the remaining healthy actuators. A controller is synthesised based on a linear model around a hover condition, but all the simulation were conducted using the nonlinear model. Three sets of tests have been conducted where a fault-free and two failures cases have been considered. The results show good tracking performance even in the presence of failures to six actuators.

Chapter 7

Mitigating total rotor failure in quadrotor using LPV based sliding mode control scheme

The last chapter presents a novel spherical UAV for FTC application by exploiting the available redundant control inputs through sliding mode and control allocation. This chapter revert back to the traditional quadrotor configuration. However, this chapter will consider the case where redundancy is not available.

This chapter presents a control scheme to handle total rotor failure in quadrotor. The controller utilises an LPV based scheme to control the reduced attitude dynamics system rather than the typical Euler angle control. The idea is to sacrifice the yaw control (and to let the quadrotor to spin around an axis) and manipulate the body-axis angular velocities so that a primary axis n (which is fixed with respect to the body axis) is tilted to align with the desired unit vector that points in the direction of the desired position. The simulation results conducted on a nonlinear model show good performance of the proposed scheme for both fault-free and failure conditions.

7.1 Introduction

As mentioned in Chapter 1 and 2, Unmanned aerial vehicles (UAVs) have now a high demand in remote aerial services such as photography, monitoring and search and rescue, construction and recently seen as good potential for autonomous 'taxi drone' [118]. However, one area of major concern is the safe operation of UAVs which have attracted many researchers in the area of FTC for UAVs in the last few years (see for example [119, 77, 120]). However, many of these works only deal with faults and not total failures on quadrotor due to lack of redundancy.

There has been a recent emergence of a handful of work that considered a total actuator failure for quadrotor (see for example [121, 79, 122]. Many of these works suggested sacrificing yaw control and utilise only three states to control the position of the aircraft. This was carried out by controlling the altitude, pitch angle and roll angle as in [121, 123]. Other works such as [79, 80, 124, 125] suggested controlling altitude and acceleration-based load factors in x and y body axes. Most of these work considers linear based design around a level trim condition and that the quadrotor rotates horizontally around a vertical axis parallel to the total thrust axis. However, in the event of one motor fails, this cannot be guaranteed because of the asymmetry in thrust generated by the three remaining motors.

In order to guarantee the level trim condition, the adjacent motor to the failed one should not be used. However, this strategy decreases the number of available rotors resulted in less manoeuvrability and increases the trim thrust values of the two remaining active motors, making these motors prone to saturation limits. Using the altitude and load factor technique proposed in [79, 80] can mitigate this issue by considering the asymmetric thrust trim condition. This slightly tilts the quadrotor away from level trim, but with the three motors still operational rather than two. This increases the total thrust produced and ensure all the motors operate away from its saturation limits.

In [80], a linear quadratic regulator (LQR) based controller was considered and assumed that motor failure can be detected and isolated. In [125], a geometric controller was considered and detection and isolation of the failed motor is also assumed to be available. In [124], fault detection and isolation approach were used with nonlinear dynamic inversion controller.

Recently, the work in [122] considers an LPV based control to ensure full control from the initial transition when the quadrotor started to spin (loss of yaw control), to the steady state when constant/maximum yaw rotation is achieved. This is different in comparison to [79, 80] which only consider the linear case when constant yaw rate has been achieved.

This chapter proposes a sliding mode FTC to maintain roll and pitch control, while separate altitude and yaw control also designed to provide full control during fault-free conditions. This is one of the main differences to the work in [79, 80, 122] where these works only consider failure situation. Similar to [79, 80] in the event of one rotor failure, the quadrotor is allowed to rotate freely around a primary axis which is fixed with respect to the body axis while maintaining hover. In order to provide position tracking, the primary axis is tilted to the desired position (through angular translation) to allow motion/acceleration in the desired direction. In comparison with [122], the LPV formulation considered in this chapter is different as it uses an extra LPV parameter n_z

(which is not assumed to be 1).

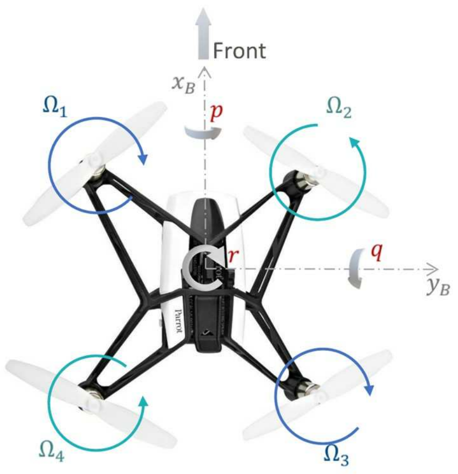


Figure 7.1: Motor numbering and direction of rotation

7.2 Equation of motion

7.2.1 The nonlinear equations of motion

A typical quadrotor UAV with a mass m_{kg} considered in this chapter is given in Fig. 7.1. The speed of the rotors $\Omega_1 \dots \Omega_4$ can be controlled independently and the direction of rotation and body axis are shown in Fig. 7.1.

7.2.2 Control strategy

This chapter concentrates on the event of total failure to one of the motors (although the same idea can be extended to two-motor failure case). In this case, most FTC schemes will not be able to control the quadrotor due to the lack of available redundancy. Instead, the idea is to sacrifice the yaw control (similar to [121, 79, 122]) and to let the quadrotor to spin on the body z axis while maintaining the roll, pitch and altitude control and therefore still retaining x, y, z position control.

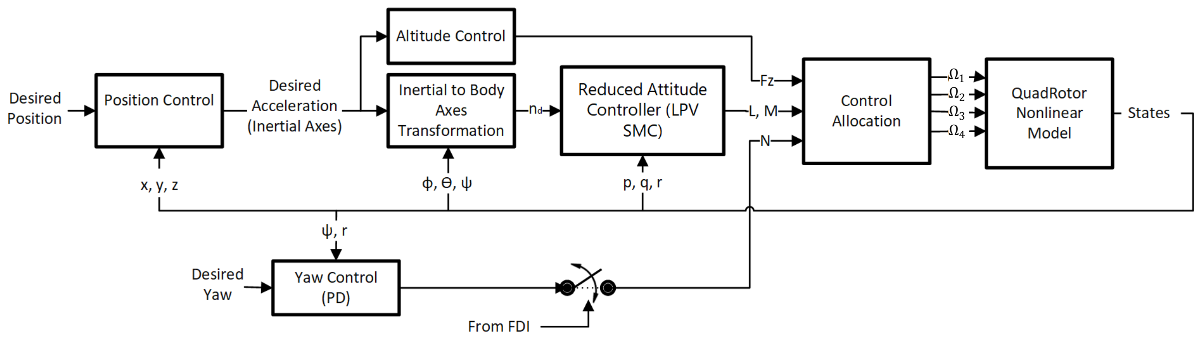


Figure 7.2: Overall structure of the proposed design scheme

The overall control strategy is illustrated in Fig. 7.2. This figure shows the cascaded controller structure where the outer-loop position control provides the desired unit vector command for the inner-loop reduced attitude controller and to a separately designed altitude control. A PID based yaw control is used for a fault-free condition and in the event of motor failure, the FDI unit disabled the yaw control.

The analysis which follows, the description for the reduced attitude dynamics which is used to design the inner-loop controller. The reduced order attitude is based on the roll and pitch rate dynamics together with some unit vector dynamics (rather than typical Euler angles) [121, 79, 122].

Consider a unit vector n, defined as

$$n = \begin{bmatrix} n_x & n_y & n_z \end{bmatrix}^T \tag{7.1}$$

which is fixed in the body axis and in the same direction as the total thrust vector F_z (see Fig. 7.3). Also, consider the unit vector n_d defined as

$$n_d = \begin{bmatrix} n_{d_x} & n_{d_y} & n_{d_z} \end{bmatrix}^T \tag{7.2}$$

which is the desired unit vector pointing to the direction of desired position (see Fig. 7.5). (Note that the unit vector n and n_d represents unit acceleration vectors in the fixed body axis.)

Similar to [121, 79], the main idea is to align the body axis unit vector n to the desired unit vector n_d (i.e. $n = n_d$ as seen in Fig. 7.3 and 7.4). This is achieved by manipulating the reduced attitude kinematic [121, 79, 122] to control the quadrotor.

7.2.3 Reduced attitude kinematics

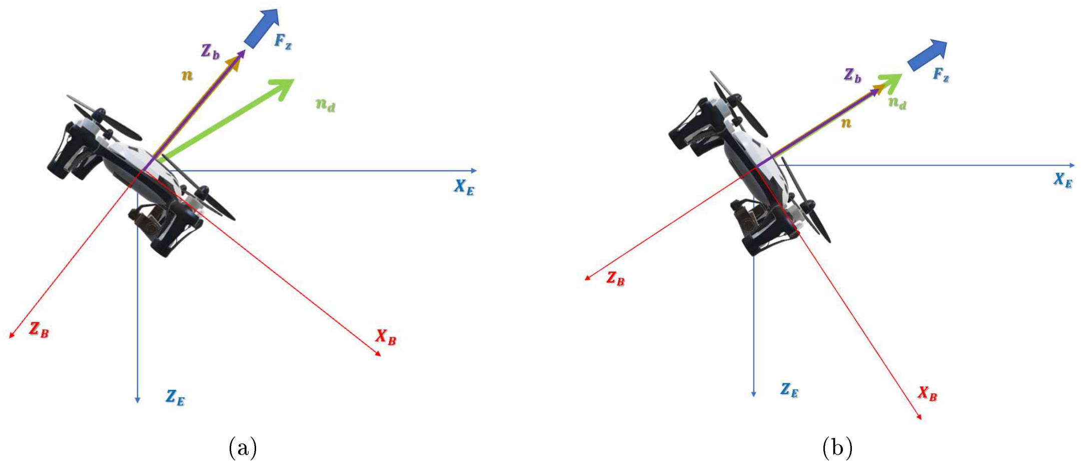
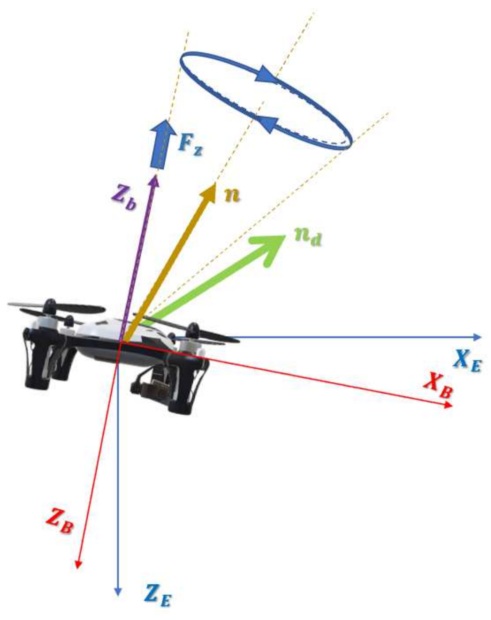
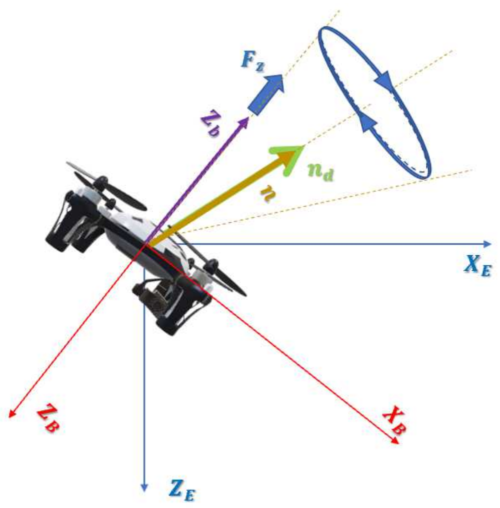


Figure 7.3: Control strategy during fault-free: Z_b and n are in the same direction.



(a)



(b)

Figure 7.4: Control strategy during one motor failure: The idea is to align the vector n with n_d while the aircraft is rotating around n.

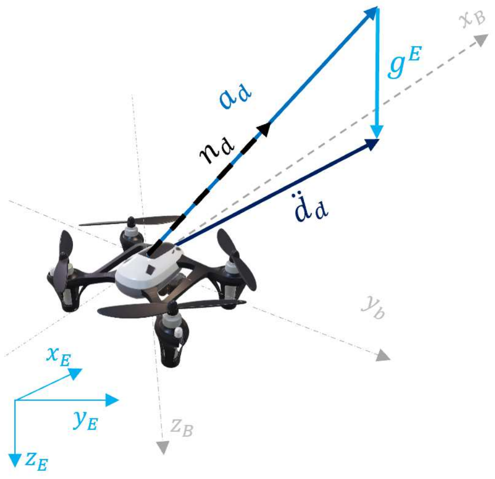


Figure 7.5: The relationship between acceleration vectors

Consider the vector

$$d^E = [\ x \quad y \quad z\ ]^T \tag{7.3}$$

which represents the position of the quadrotor in inertial axes, d_d^E as the desired position and $\ddot{d}_d^E$ as the desired acceleration of the quadrotor. Then the desired acceleration (in inertial axes) a_d^E can be obtained as shown in Fig. 7.5 as

$$a_d^E = \ddot{d}_d^E - g^E \tag{7.4}$$

where $g^E = [\ 0 \quad 0 \quad 9.81\ ]^T$ is the gravitational acceleration vector in inertial axes. The unit vector $n_d^E = [\ n_{d_x}^E \quad n_{d_y}^E \quad n_{d_z}^E\ ]^T$ can be used to describe the direction a_d^E where

$$n_d^E = \frac{a_d^E}{\|a_d^E\|} \tag{7.5}$$

The vector $n_d^E = [\ n_{d_x} \quad n_{d_y} \quad n_{d_z}\ ]^T$ (which is in inertial axis) can be represented in body axis as

$$n_d = R_b^i \times n_d^E \tag{7.6}$$

where R_b^i is the direction cosine matrix defined in Eq.(A.4).

The evolution of the desired unit vector n_d is given by the following differential equation (with cross product)

$$\begin{aligned} \dot{n}_d &= -\omega \times n_d \\ &= \begin{bmatrix} -n_{d_z} q + n_{d_y} r \\ n_{d_z} p - n_{d_x} r \\ -n_{d_y} p + n_{d_x} q \end{bmatrix} \end{aligned} \tag{7.7}$$

where

$$\omega = [\ p \quad q \quad r\ ]^T \tag{7.8}$$

is the angular velocity (in body axis). Equation (7.7) is referred to as the reduced attitude kinematics which is utilised later to control the quadrotor (instead of typical roll, pitch and yaw angle).

7.2.4 Equilibrium condition

In a fault-free hover case, from (7.7), there exists a static equilibrium for the unit vector n which is stationary in the body axis in all its axes (i.e. stationary equilibrium in n_x, n_y, n_z). When one (or two) rotor failed, there is no static equilibrium, since the quadrotor spins in the vertical axis due to the loss of yaw control. However, it is shown in [79], that there exists a constant angular velocity Ω and therefore exist a (dynamic) equilibrium condition for unit vector n. This equilibrium condition is used in [79] to specify the linear model of the reduced attitude kinematics in ((7.7)) and design a linear controller.

In a (static/dynamic) equilibrium hover condition, the unit vector n in (7.1) can be used to describe the quadrotor primary axis about which the vehicle rotates [79]. In fault-free hover condition, $n = \begin{bmatrix} 0 & 0 & -1 \end{bmatrix}^T$ which is in the same direction as the motor total thrust axis Z_b i.e. $n = Z_b$ (see Fig. 7.3). In the event of the failure of two adjacent rotors, then the quadcopter typically lose yaw control but spin around the same primary axis $n = \begin{bmatrix} 0 & 0 & -1 \end{bmatrix}^T$ due to the symmetry of thrust from remaining rotors. In the event of one motor failed, the quadcopter also loses yaw control and since there is a non-symmetry of thrust from 3 rotors, the primary axis $n \neq \begin{bmatrix} 0 & 0 & -1 \end{bmatrix}^T$, but instead slightly tilted at some angle relative to the body axis Z_b. In this case $n \neq Z_b$ and the vertical body axis Z_b rotates around the primary axis n (see Fig. 7.4). See [79] for detailed discussions and analysis for the equilibrium conditions and the primary axis for different rotor failures.

7.2.5 Reduced attitude control

As shown in Fig. 7.2, an outer loop position control provides acceleration a_d^E with a resultant unit vector n_d^E which is transformed to the body axis as n_d. The idea here is to manipulate the body-axis angular velocities p, q so that the primary axis n (which is fixed with respect to body axis) align with the desired unit vector n_d [79, 80]. For control synthesis, the dynamics of the desired reduced attitude kinematics n_{d_x}, n_{d_y} from (7.7) and angular velocities in the body axis p and q in Eq. (A.1) are considered to control the x and y (inertial axes) positions where

$$\begin{bmatrix} \dot{n}_{d_x} \\ \dot{n}_{d_y} \\ \dot{p} \\ \dot{q} \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & r & 0 & -n_{d_z} \\ -r & 0 & n_{d_z} & 0 \\ 0 & 0 & 0 & rc_1 \\ 0 & 0 & rc_2 & 0 \end{bmatrix}}_{A_p(r, n_{d_z})} \underbrace{\begin{bmatrix} n_{d_x} \\ n_{d_y} \\ p \\ q \end{bmatrix}}_{x(t)} + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ c_4 & 0 \\ 0 & c_5 \end{bmatrix}}_{B_p} \underbrace{\begin{bmatrix} \mathcal{L} \\ \mathcal{M} \end{bmatrix}}_{v(t)} \tag{7.9}$$

Note that in the reduced attitude system representation described above, the desired unit vector n_d dynamics (which is in body axis) is considered since the primary axis unit vector n is stationary in the body axis. Note that c_1, c_2, c_3, c_4 and c_5 are systems constant which is given in detail in Appendix A.

7.3 Sliding mode control synthesis

The reduced attitude system in (7.9) can be written as an affine LPV form as

$$\dot{x}(t) = A_p(\varrho)x(t) + B_p v(t) \tag{7.10}$$

where

$$\varrho(t) = [\ \varrho_1(t) \quad \varrho_2(t)\ ] = [\ r(t) \quad \hat{n}_z(t)\ ] \tag{7.11}$$

and $\hat{n}_z(t) = n_{d_z}(t) + 1$. The scheduling gain ϱ is assumed to be measured and lie in an envelope defined by a compact set, $\Theta \in \mathbb{R}^2$. The matrix $A_p(\varrho)$ in (7.10) is defined as

$$A_p(\varrho) = \begin{bmatrix} 0 & r & 0 & 1-\hat{n}_z \\ -r & 0 & \hat{n}_z - 1 & 0 \\ 0 & 0 & 0 & rc_1 \\ 0 & 0 & rc_2 & 0 \end{bmatrix} \tag{7.12}$$

which can be represented in LPV form as

$$A_p(\varrho) = A_0 + A_1 r + A_2 \hat{n}_z \tag{7.13}$$

where

$$A_0 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, A_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & c_1 \\ 0 & 0 & c_2 & 0 \end{bmatrix}, A_2 = \begin{bmatrix} 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{7.14}$$

7.3.1 Integral action approach

The LPV system (7.10)-(7.11) is used for design which is discussed in the subsequent section. Note that $\hat{n}_z$ have been chosen for the LPV parameters to allow a fixed matrix A_0 to have the structure defined in (7.14). This convenient structure is exploited for designing the sliding surface which will be discussed later.

Note that the use of the LPV system for design is different to the one in [79, 80] which only considers control design using LTI model when the system has reached the maximum yaw rate (equilibrium condition). The LPV based method proposed in this chapter also covers the transient period (from the moment a motor failed until it reached the maximum yaw rate i.e. equilibrium). This strategy is also being considered in recent work by [122]. The proposed method considers two LPV parameters - r (yaw rate) and $\hat{n}_z$.

The controlled states are n_{d_x}, n_{d_y}. Define $x_r(t) \in \mathbb{R}^2$ as:

$$\dot{x}_r(t) = r(t) - Cx(t) \tag{7.15}$$

where $r(t) \in \mathbb{R}^2$ is the commanded signals and

$$C = \begin{bmatrix} I_2 & 0_{2\times 2} \end{bmatrix} \tag{7.16}$$

By augmenting the states $x_r(t)$ to the system in (7.10) yield

$$\tilde{x}(t) = \begin{bmatrix} x_r(t) \\ x(t) \end{bmatrix} \tag{7.17}$$

and

$$\dot{\tilde{x}}(t) = \tilde{A}(\varrho)\tilde{x}(t) + \tilde{B}v(t) + \tilde{B}_r r(t) \tag{7.18}$$

where

$$\tilde{A}(\varrho) = \begin{bmatrix} 0 & -C \\ 0 & A(\varrho) \end{bmatrix}, \tilde{B} = \begin{bmatrix} 0 \\ B \end{bmatrix}, \tilde{B_r} = \begin{bmatrix} I_2 \\ 0_{4\times 2} \end{bmatrix} \tag{7.19}$$

The states in $\tilde{x}$ can be partitioned as

$$\tilde{x} = \begin{bmatrix} \tilde{x}_1 \\ \tilde{x}_2 \end{bmatrix} \tag{7.20}$$

where $\tilde{x}_1 \in \mathbb{R}^4$ and $\tilde{x}_2 \in \mathbb{R}^2$ and therefore

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} \tilde{A}_{11}(\varrho) & \tilde{A}_{12}(\varrho) \\ \tilde{A}_{21}(\varrho) & \tilde{A}_{22}(\varrho) \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0_{4\times 2} \\ B \end{bmatrix} v + \begin{bmatrix} B_r \\ 0_{2\times 2} \end{bmatrix} r \tag{7.21}$$

where

$$\begin{aligned} \tilde{A}(\varrho) &= \begin{bmatrix} \tilde{A}_{11}(\varrho) & \tilde{A}_{12}(\varrho) \\ \tilde{A}_{21}(\varrho) & \tilde{A}_{22}(\varrho) \end{bmatrix} \\ &= \left[\begin{array}{cc|c} 0 & -C_1 & -C_2 \\ 0 & A_{11}(\varrho) & A_{12}(\varrho) \\ \hline 0 & A_{21}(\varrho) & A_{22}(\varrho) \end{array}\right] \end{aligned} \tag{7.22}$$

and

$$B_r = \begin{bmatrix} I_2 \\ 0_{2\times 2} \end{bmatrix} \tag{7.23}$$

Define the sliding surface as

$$\mathcal{S} = \tilde{x} \in \mathbb{R}^6 : S\tilde{x}(t) = S_r r(t) \tag{7.24}$$

where $S \in \mathbb{R}^{2\times 6}$ and $S_r \in \mathbb{R}^{2\times 2}$ are the sliding surface which are designed to ensure the stability of the reduced order system. The matrix S can be partitioned as

$$S = \left[\begin{array}{cc} S_1 & S_2 \end{array} \right] \tag{7.25}$$

where $S_1 \in \mathbb{R}^{2\times 4}$ and $S_2 \in \mathbb{R}^{2\times 2}$. By defining

$$s(t) = S\tilde{x}(t) - S_r r(t) \tag{7.26}$$

then

$$\dot{s}(t) = S\dot{\tilde{x}}(t) - S_r \dot{r}(t) \tag{7.27}$$

then from equation (7.18)

$$\dot{s}(t) = S(\tilde{A}(\varrho)\tilde{x}(t) + \tilde{B}\nu(t) + \tilde{B}_r r(t)) - S_r \dot{r}(t) \tag{7.28}$$

Note that $S\tilde{B} = S_2 B_2$ and $S\tilde{B}_r = S_1 B_r$ then the control law $\nu(t)$ could be obtained as

$$\nu(t) = \Lambda^{-1}(\dot{s}(t) - (S\tilde{A}(\varrho))\tilde{x}(t) - (S_1 B_r) r(t)) + S_r \dot{r}(t)) \tag{7.29}$$

where $\Lambda = S_2 B_2$. The control law could be factorised into two terms, a linear term $\nu_l(t)$ and a nonlinear term $\nu_n(t)$ as

$$v(t) = v_l(t) + v_n(t) \tag{7.30}$$

where

$$v_l(t) \quad = \quad L\tilde{x}(t) + L_r r(t) + L_{\dot{r}} \dot{r}(t) \tag{7.31}$$

where

$$\begin{aligned} L &= -\Lambda^{-1}(S\tilde{A}(\varrho) - \Phi S) \\ L_r &= -\Lambda^{-1}(S_1 B_r + \Phi S_r) \\ L_{\dot{r}} &= \Lambda^{-1} S_r \end{aligned} \tag{7.32}$$

The nonlinear term

$$v_n = \begin{cases} -\rho\Lambda^{-1}\frac{P_2 s(t)}{||P_2 s(t)||} & \text{if } s(t) \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{7.33}$$

where ρ is a scalar that depends on the uncertainties in the system [25], $\Phi \in \mathbb{R}^{2\times 2}$ is a stable design matrix and $P_2 \in \mathbb{R}^{2\times 2}$ is a symmetric positive definite matrix satisfying the following equation

$$P_2\Phi + \Phi^T P_2 = -I \tag{7.34}$$

7.3.2 Design of the sliding surface

From equation (7.21), the reduced order sliding motion [25] is given by

$$\dot{x}_1(t) = \tilde{A}_{11}(\varrho)x_1(t) + \tilde{A}_{12}(\varrho)x_2(t) + B_r r(t) \tag{7.35}$$

From equation (7.26), during sliding motion

$$s(t) = S_1 x_1(t) + S_2 x_2(t) - S_r r(t) = 0 \tag{7.36}$$

Substituting x_2 from (7.36) into (7.35) yield

$$\dot{x}_1(t) = \bar{A}(\varrho)x_1(t) + (\tilde{A}_{12}(\varrho)S_2^{-1}S_r + B_r)r(t) \tag{7.37}$$

where $\bar{A}(\varrho) = (\tilde{A}_{11}(\varrho) - \tilde{A}_{12}(\varrho)M)$ and $M = S_2^{-1}S_1$. The design matrix M is chosen to ensure $\bar{A}$ is stable. (Note that the last term in (7.37) depends only on the bounded command signal $r(t)$ and not considered during the design of M). A linear matrix inequality (LMI) based approach based on a single quadratic Lyapunov function $V = \bar{A}(\varrho)^T P_1 \bar{A}(\varrho)$ where

$$\bar{A}(\varrho)^T P_1 + P_1\bar{A}(\varrho) \;<\; 0 \tag{7.38}$$

$$P_1 \;>\; 0 \tag{7.39}$$

can be considered for the design of M. The existing MATLAB LMI multi-model state feedback synthesis code $msfsyn$ [126] can be used to solve the LMI in (7.38) and to obtain the solution for P_1 and M.

Table 7.1: Quadrotor physical parameters

Parameter	Value	Unit
m_{kg} (mass)	0.068	kg
I_{xx} (moment of inertia)	0.0686×10^{-3}	kgm^2
I_{yy} (moment of inertia)	0.092×10^{-3}	kgm^2
I_{zz} (moment of inertia)	0.1366×10^{-3}	kgm^2
b (thrust factor)	6.5328×10^{-4}	kgm
d (drag factor)	1.6×10^{-6}	kgm^2
ℓ (moment arm)	0.044	m

7.4 Design and simulation

7.4.1 Parrot quadrotor

The quadrotor model used for simulation in this chapter is based on the Parrot rolling spider [127] quadrotor. The quadrotor physical parameters are summarized in Table 7.1. The LPV variables range are $\varrho_1 = [-40 : 40]$ and $\varrho_2 = [0 : 0.2]$. The solution for the Lyapunov matrix P_1 and therefore the sliding matrix M (as discussed in Section 7.3.2) was obtained using the standard MATLAB LMI function "$msfsyn$" [126]. Here, a pole placement method has been utilised to ensure the closed loop poles of the reduced order sliding motion are within a specified region of the complex plane. In this design, the selected LMI region is set between two vertical lines through -1 and -6.

The motor limits are $[0 \;\; 700]$ $(rad/sec)^2$. For practical implementation, the discontinuous (signum) term in (7.33) have been replaced with a smooth sigmoidal approximation [25] given by $\frac{P_2 s(t)}{\|P_2 s(t)\|+\delta}$ where δ is a small positive scalar. For this design, the $\delta = 0.001$ has been chosen. The nonlinear modulation gain ρ in (39) was chosen as 0.2. The design matrix Φ in (7.32) was chosen as $\Phi = diag(-1, -1)$.

As seen in Fig. 7.2, an outer loop x and y position control as well as the separate control for altitude and yaw (for the fault-free case) will be discussed in the following subsection.

7.4.2 Yaw control

A PD controller was designed for yaw control to calculate the required yaw torque $\mathcal{N}$ to be sent to motors. This controller is only active in a fault-free case where the four motors are working properly. The PD gains were set at $K_P = 0.004$ and $K_D = 0.0012$. Once one of the motors have failed, this controller is switched off using FDI (which is assumed to be available).

7.4.3 Position control

The idea here is for the position controller to provide the desired accelerations (and therefore desired unit vector) for the reduced attitude controller (see Fig. 7.2). As in [79], the controller is designed such that the deviation between the desired and actual position of the quadrotor behaves like a second order system (with damping coefficient ζ and natural frequency ω_n), by introducing the desired acceleration $\ddot{d}_d^E$, such that

$$\ddot{d}_d^E + 2\zeta\omega_n \dot{d}^E + \omega_n^2(d^E) = 0 \tag{7.40}$$

The parameters are chosen as $\zeta = [0.7, 0.7, 1]^T$, and $\omega_n = [1.5, 1.5, 4]^T$. Using $\ddot{d}_d^E$ from above, unit vector n_d^E is obtained from (7.5) and converted into body axis n_d using ((7.6)). This desired unit vector is used as the command for the inner-loop control as shown in Fig. 7.2.

7.4.4 Altitude control

Similar to [79, 80], the total force from all rotors F_z (in Z_b direction) is obtained from the desired acceleration produced by the outer-loop position control and given by

$$F_z = \frac{m_{kg} \left\| a_d^E - g^E \right\|}{Z_b.n} \tag{7.41}$$

7.4.5 Control allocation

The control law Eq.(7.30) is designed to manipulate the system input ν which represents the total roll and pitch moment $\mathcal{L}$ and $\mathcal{M}$ as defined in Eq.(7.9). These need to be converted into control signals to be sent into the four individual rotors trough control allocation/ control mixing [83].

In a fault-free case, the signals sent to the four motors is defined as

$$u = \begin{bmatrix} \Omega_1^2 & \Omega_2^2 & \Omega_3^2 & \Omega_4^2 \end{bmatrix}^T \tag{7.42}$$

where Ω_i is the i_{th} motor rotational speed in rad/s and can be calculated as

$$u = B_{\tau_1}^+ F_z + B_{\tau_2}^+ v + B_{\tau_3}^+ \mathcal{N} \tag{7.43}$$

where F_z is the total thrust from Eq.(7.41), $\mathcal{N}$ is the yaw controller and ν is the roll and pitch moment from the reduced attitude controller in Eqs.(7.30)-(7.33). The input matrix is given by

$$B_{\tau_1} = \begin{bmatrix} -b & -b & -b & -b \end{bmatrix} \tag{7.44}$$

$$B_{\tau_2} = \begin{bmatrix} bl & -bl & -bl & bl \\ bl & bl & -bl & -bl \end{bmatrix} \tag{7.45}$$

$$B_{\tau_3} = \begin{bmatrix} -d & d & -d & d \end{bmatrix} \tag{7.46}$$

where b,d are the rotor thrust and torque coefficient respectively and ℓ is the moment arm (distance from the motor to the aircraft centre of gravity) [83]. The parameter B_i^+ is the pseudo-inverse of B_i matrix as is defined as

$$B_i^+ = WB_i^T(B_iWB^T)^{-1} \tag{7.47}$$

where the matrix $W \in \mathbb{R}^{4\times4}$ is a diagonal matrix defining the effectiveness of the motors with diagonal elements $w_i = 1$ for fault-free motors and $w_i = 0$ for failure motors.

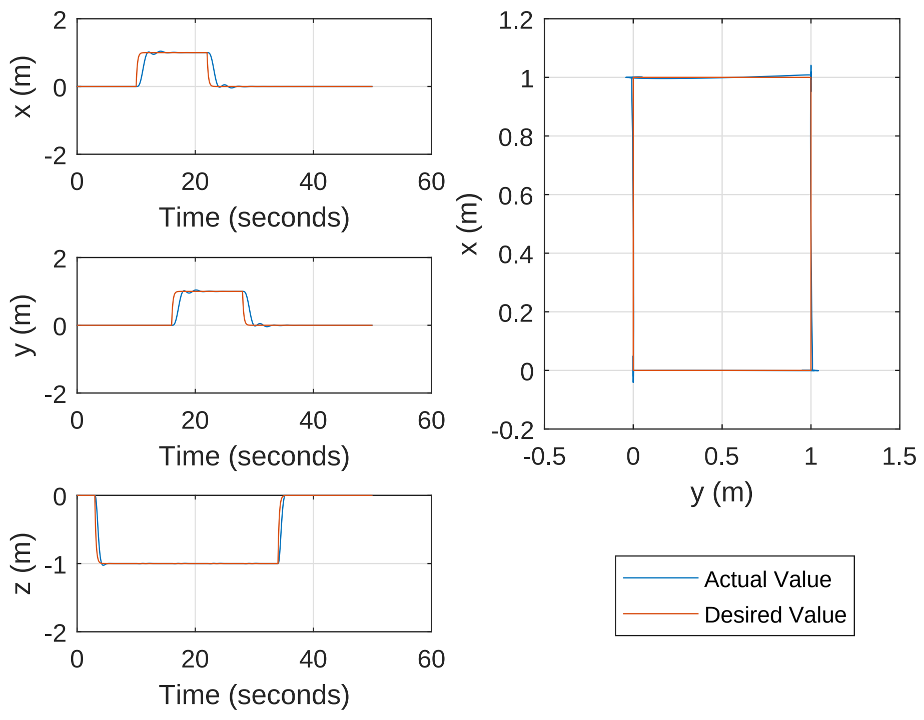
Note that the control law (7.43), information of the effectiveness level W of each rotor is assumed to be available from a fault detection and isolation (FDI) units. In the simulation that follows, it is assumed that motor 4 fail. When this occurs, the FDI detected and isolated the failure, and $w_4 = 0$ and therefore the 4th element of the control law in (7.42) is zero, and no control signal sent to the failed motor.

7.4.6 Simulation results

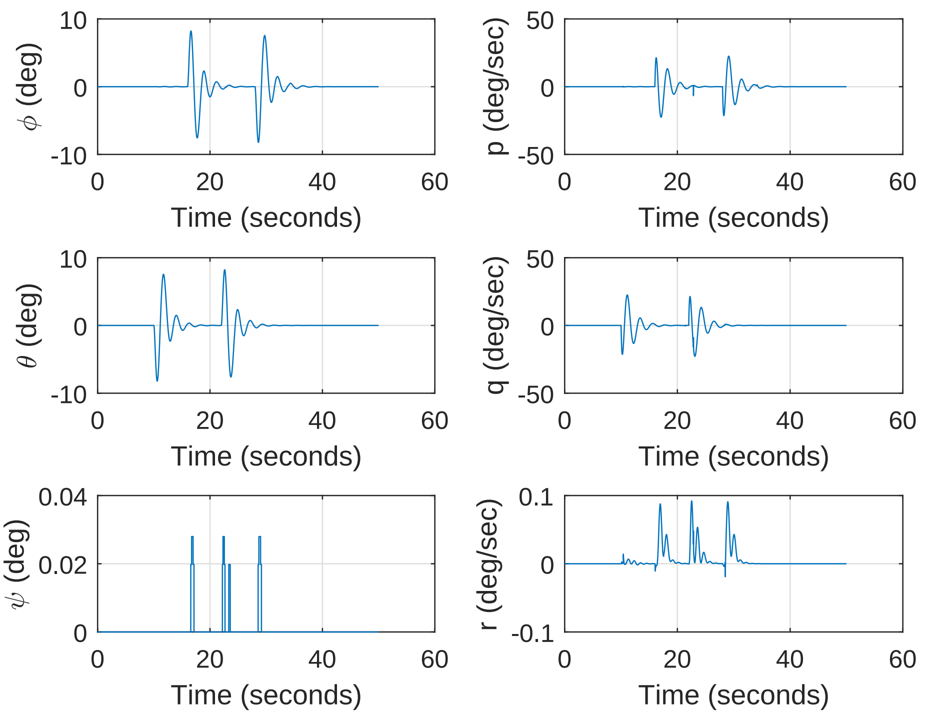
The results shown in this section are obtained using the nonlinear model simulation representing the Parrot Quadcopter. The simulation was conducted using SIMULINK with a fixed time solver $ode3$ with a time step of $0.005s$.

7.4.6.1 Fault-free case

During fault-free simulation, the controller showed a good performance in position control as shown in Fig. 7.6a. Fig. 7.6b shows a reasonable performance for the Euler angles and the vehicle's angular velocities. The tracking of the desired acceleration unit vector showed acceptance response and the switching functions were maintained very close to zero as shown in Fig. 7.6c. The motors signals were kept in the limits with reasonable performance as shown in Fig. 7.6d. Note that during fault-free condition, the primary unit vector n (body axis) has an equilibrium value $n = \begin{bmatrix} 0 & 0 & -1 \end{bmatrix}^T$

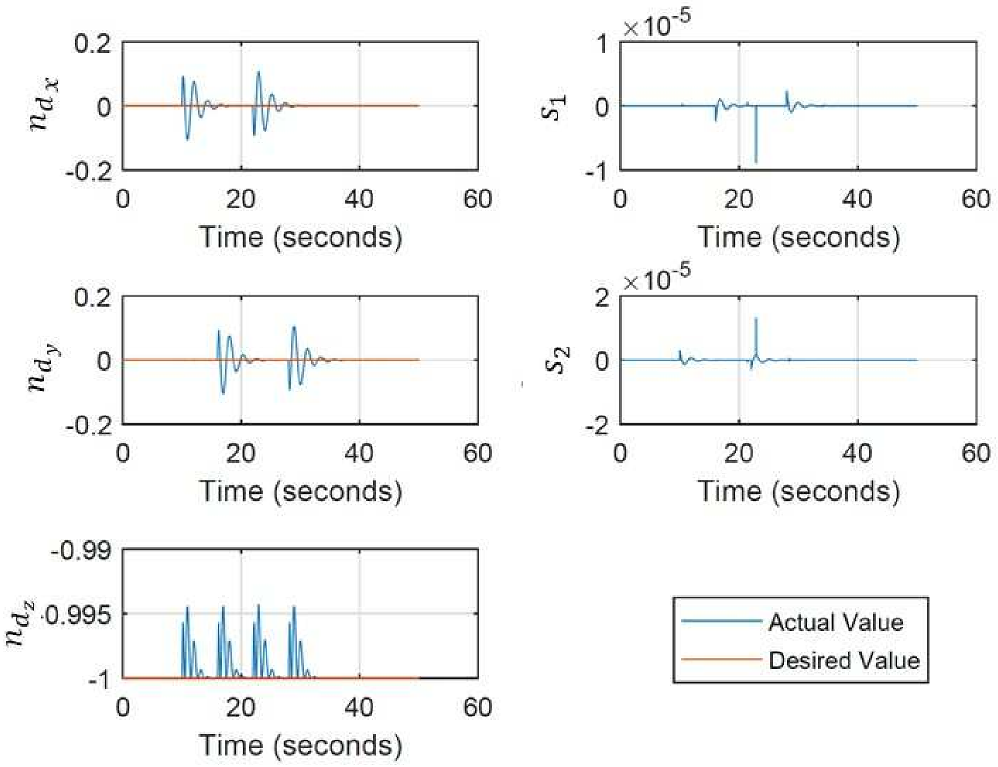


(a) Position Control

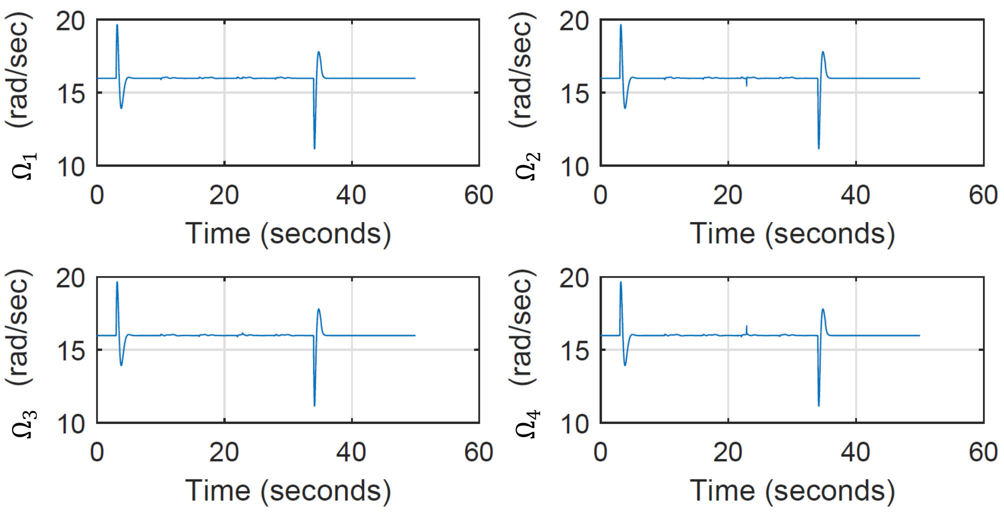


(b) Euler Angles and rates

Figure 7.6: Fault-Free Case



(c) Acceleration Unit Vector in Body Axes and Switching Functions



(d) Motors Speed

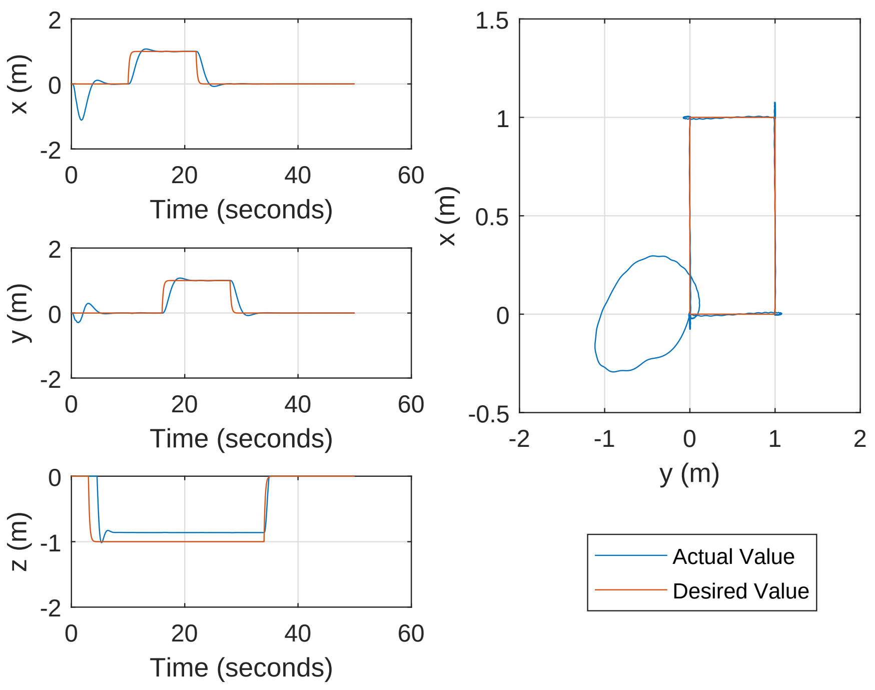
Figure 7.6: Fault-Free Case

7.4.6.2 One motor failure case

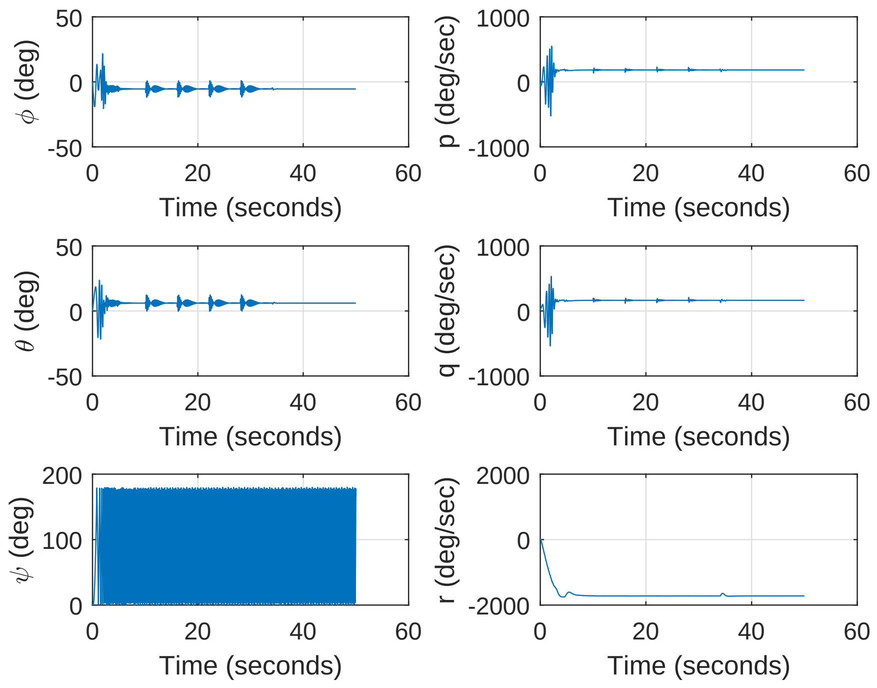
For the case of one motor failure, as shown in Fig. 7.7d, motor 4 was set to fail from the start of the simulation and the controller stopped sending control signals to motor 4. At the beginning of the simulation while the yaw rate approaching its equilibrium value, only motor 1 and motor 3 were active (Fig. 7.7d). Once the yaw rate reached an equilibrium rotation at $r \approx -1800 deg/sec$, motor 2 becomes active (see Fig. 7.7d) to extra additional lift. In this failure condition, once the yaw rate has reached its equilibrium condition (i.e. when $r \approx -1800 deg/s$) the primary axis n has an equilibrium $n = \begin{bmatrix} -0.1 & -0.1 & -0.9899 \end{bmatrix}^T$ which is slightly tilted from the vertical body axis Z_b. Due to the tilt of n, three motors are utilised (rather than two motors) and hence motor 2 was activated and started to work with motors 1 and 3 as shown in Figs. 7.7c and 7.7d. Activating motor 2 decrease the control load on the motors 1 and 3 and hence reduce the control signal away from the motor saturation limit of motors as shown in Fig. 7.7a. Fig. 7.7b shows the Euler angles and the vehicle's angular velocities where at equilibrium p and q are approximately 180 deg/s and yaw rate reached ≈ -1800 deg/s Fig. 7.7c showed the desired acceleration unit vector response and the sliding is maintained where switching functions which were maintained close to zero.

7.5 Summary

This chapter studied the under actuated control of quadrotor in the presence of one motor failure. An LPV based sliding mode control has been utilised to control a reduced attitude system rather than the typical Euler angle control. The proposed scheme can operate in faulty-free as well as in the event of one motor failure cases. The control strategy is to allow the aircraft to rotate around a primary axis (fixed in body axes) when failure occurred to one of the motors. The remaining three motors allow the vehicle to translate in space by utilising an outer-loop control which provides the desired acceleration and therefore desired unit vector in the direction of travel. This desired unit vector is then used by the inner-loop LPV based SMC controller. Simulation results showed good responses during fault-free condition and in the event when one of the motors failed.

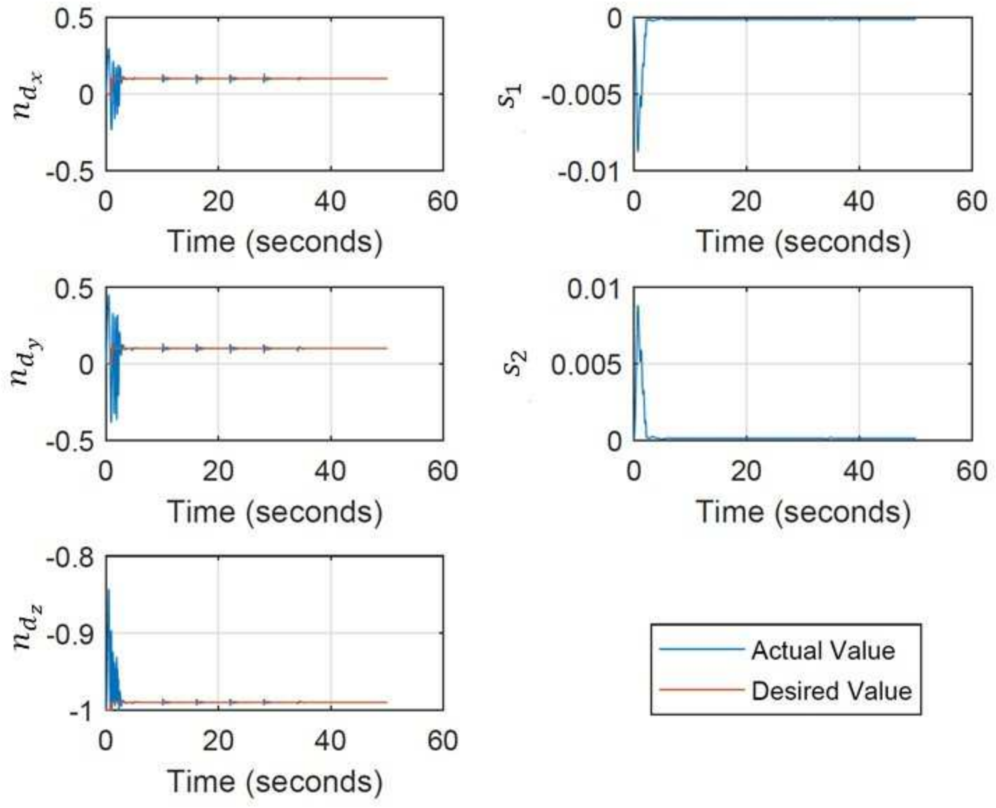


(a) Position Control

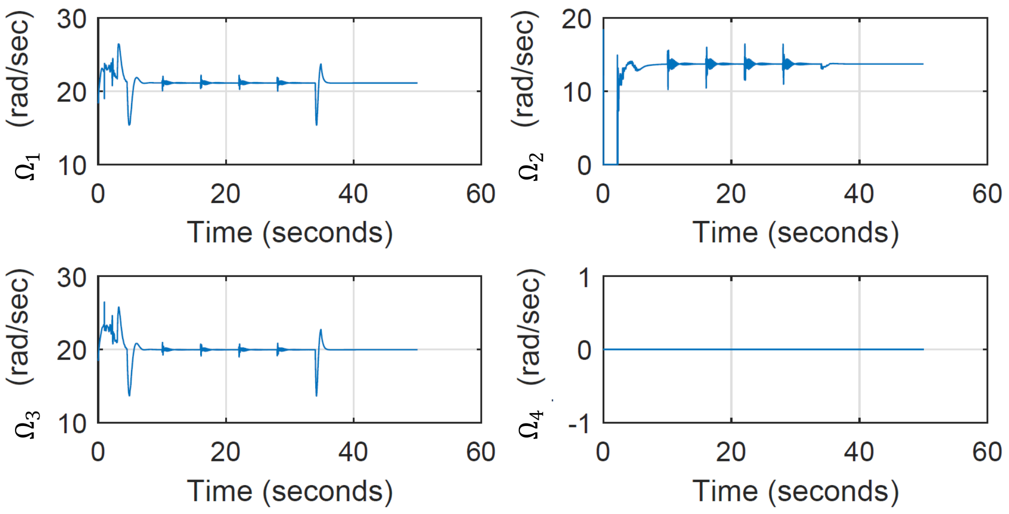


(b) Euler Angles and rates

Figure 7.7: One Motor Failure



(c) Acceleration Unit Vector in Body Axes and Switching Functions



(d) Motors Speed

Figure 7.7: One Motor Failure

Chapter 8

Control of octocopter with up to six motors failure

This chapter proposes an FTC scheme for trajectory control of octorotors. The stated FTC scheme has the ability to control the octorotor (eight rotors) during over actuation, sufficient actuation and under actuation scenarios (with up to six motors failing). Both the outer loop position control and the inner loop attitude control are based on sliding mode control methods. The inner loop control combines SMC with online control allocation to exploit the available actuator redundancy. The differential flatness property of the system is exploited in the synthesis of the outer loop position control. Meanwhile, feedback linearisation is used to eliminate non-linearities in the inner loop system. The simulation results conducted on the nonlinear model show a good tracking performance under various (over actuation, sufficient actuation and under actuation) scenarios.

8.1 Introduction

In this chapter, the stability and control of an octorotor will be studied. The octorotor is an eight propeller UAV. The conventional configuration of octorotors could have four arms and four pair of coaxial rotors, or they could have eight arms and eight separate rotors. The coaxial configuration has some inherent nonlinearities from the interaction between the upper and lower rotors [84]. The conventional eight arm form has less interaction among propellers and will be considered under this study.

Octorotors like quadrotors require at least four separate inputs to fully control the UAV position and orientation. Octorotors are considered over actuated with four redundant actuators. Various studies considered the fault tolerant control of octorotors keeping it as an over actuated system, for example [83] and [85]. However, this chapter will consider fault-tolerant control of the octorotor starting from a fault-free over actuated system, a

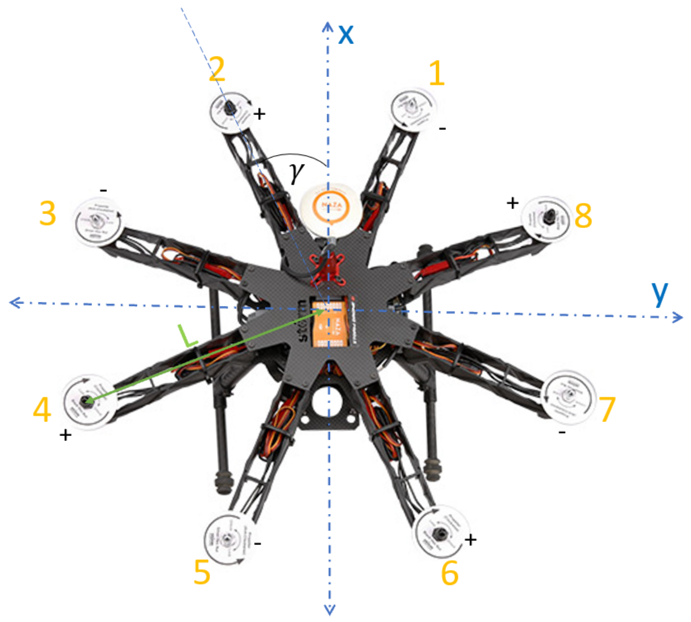


Figure 8.1: Storm8 octorotor Motor Configuration [41]

faulty over actuated system with up to four motors remaining, and the under actuated cases. Two cases will be considered in the under actuated scenario. The first involved a three motors scenario (where yaw angle will be sacrificed) and second, where only two motors remain (which will sacrifices both yaw and pitch angle in order to control the translational position of the aircraft). In [80], the work considers the under actuated cases of quadrotors considering a bespoke system and controller for each case. In this chapter, a simpler control scheme that covers various cases will be proposed. The control design contains an outer loop position control and an inner loop attitude control. In [79, 80] and in chapter 7, the body acceleration unit vector representation was used as the inner loop system. However the typical Euler angles representation will be used in this chapter, which makes the analysis easier to understand and more natural to design.

In [128], an LQR controller was used for outer loop position control and PID controllers were considered for inner loop attitude control. However in this chapter, the controller will be synthesised using integral action sliding mode control for both outer and inner control loops, combined with feedback linearization to eliminate non linearities in the system.

The differential flatness properties (as described in [129] and [128]), will be utilised in this chapter to design appropriate state-space based controllers for the inner and outer loops.

8.2 Equation of motion

Common to any rigid body dynamical model, the octorotor equation of motion in body axes is given by:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \hline \dot{\psi} \\ \dot{u} \\ \dot{v} \\ \dot{w} \\ \hline \dot{p} \\ \dot{q} \\ \hline \dot{r} \end{bmatrix} = \begin{bmatrix} p + qsin(\phi)tan(\theta) + rcos(\phi)tan(\theta) \\ qcos(\phi) - rsin(\phi) \\ \hline qsin(\phi)sec(\theta) + rcos(\phi)sec(\theta) \\ vr - qw - gsin(\theta) \\ pw - ur + gcos(\theta)sin(\phi) \\ uq - pv + gcos(\theta)cos(\phi) \\ \hline qr(I_{yy} - I_{zz})/I_{xx} + J_r q\Omega_r/I_{xx} \\ pr(I_{zz} - I_{xx})/I_{yy} - J_r p\Omega_r/I_{yy} \\ \hline qr(I_{xx} - I_{yy})/I_{zz} \end{bmatrix} + \underbrace{\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{m_{kg}} & 0 & 0 & 0 \\ \hline 0 & \frac{1}{I_{xx}} & 0 & 0 \\ 0 & 0 & \frac{1}{I_{yy}} & 0 \\ \hline 0 & 0 & 0 & \frac{1}{I_{zz}} \end{bmatrix}}_{B_\tau} \underbrace{\begin{bmatrix} F_z \\ \hline \mathcal{L} \\ \hline \mathcal{M} \\ \hline \mathcal{N} \end{bmatrix}}_{\tau(t)} \quad (8.1)$$

where g is gravity. The states ϕ, θ, ψ are the Euler angles (roll, pitch and yaw), p, q, r are the angular velocities in the body axes (roll, pitch and yaw rates), u, v, w are the linear velocities in body axes and I_{xx}, I_{yy}, I_{zz} are the inertias about x, y, z body axes respectively. The parameter m_{kg} is the mass of the octorotor, J_r is the rotor inertia and Ω_r is the overall residual propeller speed from unbalanced rotor rotation and is defined as

$$\Omega_r = \Omega_1 - \Omega_2 + \Omega_3 - \Omega_4 + \Omega_5 - \Omega_6 + \Omega_7 - \Omega_8 \quad (8.2)$$

where $\Omega_1, \dots \Omega_8$ are the individual propeller angular speeds. The velocity of the octorotor in the earth axes is given by

$$\begin{bmatrix} \dot{x}_e \\ \dot{y}_e \\ \dot{z}_e \end{bmatrix} = (R_b^i(\phi, \theta, \psi))^T \begin{bmatrix} u \\ v \\ w \end{bmatrix} \quad (8.3)$$

where $(.)^T$ is the transpose of a matrix and R_b^i is the direction cosine matrix defined in Eq.(A.4). The position of the octorotor can be obtained by integrating the velocity Equation (8.3).

The inputs of the system (8.1), are the total force in the z body axis (F_z), and the moments and torques around the x, y, z body axes - ($\mathcal{L}, \mathcal{M}, \mathcal{N}$) respectively. These total force and moments are mapped from all the eight individual rotors. Based on the 'Storm8' octorotor [41] motor configuration shown in Figure (8.1), the mapping is given by

$$\left[\begin{array}{c} F_z \\ \hline \mathcal{L} \\ \mathcal{M} \\ \hline \mathcal{N} \end{array}\right] = \left[\begin{array}{cccccccc} -b & -b & -b & -b & -b & -b & -b & -b \\ \hline -b\ell_1 & b\ell_1 & b\ell_2 & b\ell_2 & b\ell_1 & -b\ell_1 & -b\ell_2 & -b\ell_2 \\ b\ell_2 & b\ell_2 & b\ell_1 & -b\ell_1 & -b\ell_2 & -b\ell_2 & -b\ell_1 & b\ell_1 \\ \hline d & -d & d & -d & d & -d & d & -d \end{array}\right] \underbrace{\left[\begin{array}{c} \Omega_1^2(t) \\ . \\ . \\ . \\ \Omega_8^2(t) \end{array}\right]}_{u_\Omega(t)} \tag{8.4}$$

where b, d are the constant thrust and torque coefficients of the motors. Consider ℓ is the motor arm of the 'Storm8' octorotor, hence $\ell_1 = \ell cos(\gamma)$ and $\ell_2 = \ell sin(\gamma)$ and $\gamma = 22.5^o$.

8.3 Overall control strategy

The overall control system strategy proposed in this chapter is shown in Figure 8.2. There are two main control loop in cascade. The inner-loop SMC attitude control which provides the 'virtual' control laws (which generate the desired roll moment $\mathcal{L}$ and desired roll moment $\mathcal{M}$). A separate yaw control (based on PID), to produce the desired yaw moment $\mathcal{N}$ is also available during fault free and sufficient control conditions. The differential flatness properties ([129] [128]) are exploited directly to convert outer-loop altitude position control to the desired vertical thrust (force) F_z. The control allocation unit converts these virtual control laws $(F_z, \mathcal{L}, \mathcal{M}, \mathcal{N})$ into individual rotor speeds. An outer-loop position control based on SMC exploits the differential flatness properties to generate the desired roll and pitch angle for the inner-loop control, as well as producing the total thrust F_z. Note that Figure 8.2 also contains two switches activated by an FDI unit (which is assumed to be available). The first is used to disable yaw control in the case when only three rotors remaining (under actuated case). When two rotors remain operational (the worst under actuated case considered in this chapter and thesis), the second switch (activated by an FDI) is used to disable pitch control. In this case, only roll and altitude control remain operational and are sufficient to provide some level of control of the octorotor. The detailed descriptions of each of the blocks in Figure 8.2 will be described in detail in the subsequent sections.

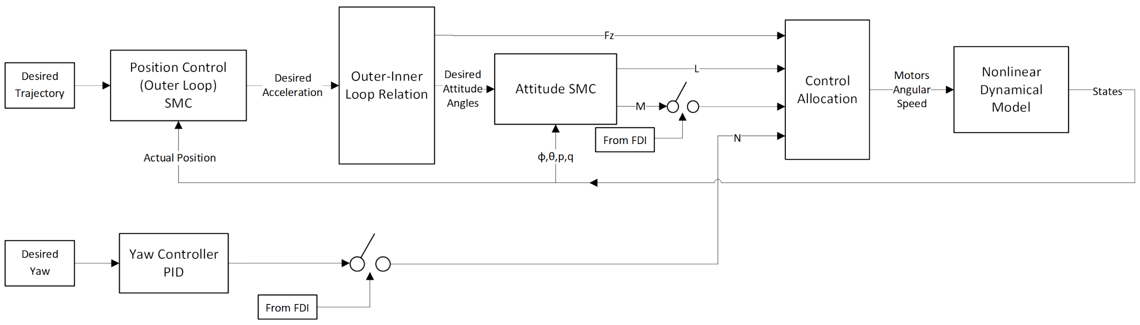


Figure 8.2: Storm8 octorotor Control Block Diagram

8.4 Inner loop control

8.4.1 Roll and pitch moment ($\mathcal{L}, \mathcal{M}$) control

This section provides the analysis for the inner loop roll and pitch moment ($\mathcal{L}, \mathcal{M}$) control. Exploiting the decoupling of the input force and moments from (8.1), only the following reduced order dynamics:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{p} \\ \dot{q} \end{bmatrix} = \begin{bmatrix} p + qsin(\phi)tan(\theta) + rcos(\phi)tan(\theta) \\ qcos(\phi) - rsin(\phi) \\ c_1qr + c_3q\Omega_r \\ c_2pr - c_4p\Omega_r \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \frac{\mathcal{L}}{I_{xx}} \\ \frac{\mathcal{M}}{I_{yy}} \end{bmatrix} \tag{8.5}$$

where $c_1 = \frac{I_{yy}-I_{zz}}{I_{xx}}$, $c_2 = \frac{I_{zz}-I_{xx}}{I_{yy}}$, $c_3 = \frac{J_r}{I_{xx}}$ and $c_4 = \frac{J_r}{I_{yy}}$ is considered. Note that the reduced order dynamics in (8.5) are nonlinear and are part of the overall nonlinear equations of motion from (8.1). For design purposes, the system in (8.5) can be written as

$$\underbrace{\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{p} \\ \dot{q} \end{bmatrix}}_{\dot{x}_p(t)} = \underbrace{\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}}_{A_p} \underbrace{\begin{bmatrix} \phi \\ \theta \\ p \\ q \end{bmatrix}}_{x_p(t)} + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{1}{Ixx} & 0 \\ 0 & \frac{1}{Iyy} \end{bmatrix}}_{B_{p\tau}} \underbrace{\begin{bmatrix} \mathcal{L} \\ \mathcal{M} \end{bmatrix}}_{\tau(t)}$$

$$+ \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ c_3 & 0 \\ 0 & c_4 \end{bmatrix}}_{D_p} \underbrace{\begin{bmatrix} q\Omega_r(t) \\ p\Omega_r(t) \end{bmatrix}}_{\varsigma(t)} + \underbrace{\begin{bmatrix} sin(\phi)tan(\theta)q + cos(\phi)tan(\theta)r \\ (cos(\phi) - 1)q - sin(\phi)r \\ c_1qr \\ c_2pr \end{bmatrix}}_{\zeta(t)} \tag{8.6}$$

Using (8.4), Eq. (8.6) can be written as

$$\dot{x}_p = A_p x_p + \underbrace{B_{p\mathcal{T}} B_\Omega}_{B_p} u_p(t) + D_p \varsigma(t) + \zeta(t) \tag{8.7}$$

where B_Ω is given by

$$B_\Omega = \begin{bmatrix} -b\ell_1 & b\ell_1 & b\ell_2 & b\ell_2 & b\ell_1 & -b\ell_1 & -b\ell_2 & -b\ell_2 \\ b\ell_2 & b\ell_2 & b\ell_1 & -b\ell_1 & -b\ell_2 & -b\ell_2 & -b\ell_1 & b\ell_1 \end{bmatrix} \tag{8.8}$$

Note that in (8.6), the term Ω_r defined in (8.2) is assumed to be uncertain but bounded. Due to the structure of D_p (8.6), this uncertainty is considered 'matched' which sliding mode control is robust against. This will be exploited later during the sliding mode controller synthesis. Meanwhile, the nonlinear term $\zeta(t)$ in (8.6) is known as it depends on known constants c_1, c_2 and states $\phi, \theta, \psi, p, q, r$ which is assumed to be measured. In (8.7), the matrix B_p can be written in detail as

$$B_p = B_{p\mathcal{T}} B_\Omega = \begin{bmatrix} 0_{2\times 8} \\ B_2 \end{bmatrix} \tag{8.9}$$

where

$$B_2 = \begin{bmatrix} -\frac{1}{I_{xx}}b\ell_1 & \frac{1}{I_{xx}}b\ell_1 & \frac{1}{I_{xx}}b\ell_2 & \frac{1}{I_{xx}}b\ell_2 & \frac{1}{I_{xx}}b\ell_1 & -\frac{1}{I_{xx}}b\ell_1 & -\frac{1}{I_{xx}}b\ell_2 & -\frac{1}{I_{xx}}b\ell_2 \\ \frac{1}{I_{yy}}b\ell_2 & \frac{1}{I_{yy}}b\ell_2 & \frac{1}{I_{yy}}b\ell_1 & -\frac{1}{I_{yy}}b\ell_1 & -\frac{1}{I_{yy}}b\ell_2 & -\frac{1}{I_{yy}}b\ell_2 & -\frac{1}{I_{yy}}b\ell_1 & \frac{1}{I_{yy}}b\ell_1 \end{bmatrix} \tag{8.10}$$

8.4.2 States tracking - Sliding mode integral action approach

For the inner-loop controller, the controlled states are ϕ, θ. Define $x_r(t) \in \mathbb{R}^2$ as:

$$\dot{x}_r(t) = r(t) - C x_p(t) \tag{8.11}$$

where $r(t) \in \mathbb{R}^2$ is the commanded signals and

$$C = \begin{bmatrix} I_2 & 0_{2\times 2} \end{bmatrix} \tag{8.12}$$

By augmenting the states $x_r(t)$ to the system in (8.6) yields the augmented states

$$x(t) = \begin{bmatrix} x_r(t) \\ x_p(t) \end{bmatrix} \tag{8.13}$$

Therefore the resulting augmented system is given by

$$\dot{x}(t) = Ax(t) + Bu_p(t) + B_r r(t) + D\varsigma(t) + E\zeta(t) \tag{8.14}$$

where

$$\begin{aligned} A &= \begin{bmatrix} 0 & -C \\ 0 & A_p \end{bmatrix}, B = \begin{bmatrix} 0 \\ B_p \end{bmatrix}, \\ B_r &= \begin{bmatrix} I_2 \\ 0 \end{bmatrix}, D = \begin{bmatrix} 0 \\ D_p \end{bmatrix}, E = \begin{bmatrix} 0 \\ I_4 \end{bmatrix} \end{aligned} \tag{8.15}$$

The controller synthesis will be based on the augmented system in (8.14)-(8.15). The augmented system can be partitioned as

$$\begin{aligned} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ B_2 \end{bmatrix} u_p(t) \\ + \begin{bmatrix} B_r \\ 0 \end{bmatrix} r(t) + \begin{bmatrix} 0 \\ D_2 \end{bmatrix} \varsigma(t) + \begin{bmatrix} E_1 \\ E_2 \end{bmatrix} \zeta(t) \end{aligned} \tag{8.16}$$

where $x_1 \in \mathbb{R}^4$, $x_2 \in \mathbb{R}^2$ and

$$\begin{aligned} A &= \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = \left[\begin{array}{cc|c} 0 & -C_1 & -C_2 \\ 0 & A_{11} & A_{12} \\ \hline 0 & A_{21} & A_{22} \end{array}\right] \\ D_2 &= \begin{bmatrix} c_3 & 0 \\ 0 & c_4 \end{bmatrix}, E_1 = \begin{bmatrix} 0_{2\times 2} & 0_{2\times 2} \\ I_2 & 0_{2\times 2} \end{bmatrix}, E_2 = \begin{bmatrix} 0_{2\times 2} & I_2 \end{bmatrix} \end{aligned} \tag{8.17}$$

8.4.3 Feedback linearisation

In the event of rotors fault/failure, the augmented system (8.14)-(8.17) can be written as

$$\dot{x}(t) = Ax(t) + Bu_p(t) - BKu_p(t) + B_r r(t) + D\varsigma(t) + E\zeta(t) \tag{8.18}$$

where $K = diag(k_1 \ldots k_8)$ represents the fault/failure in each rotors. The scalars k_i model each individual rotor reduction in effectiveness level and satisfy $0 \leq k_i \leq 1$. In the fault free case $k_i = 0$, in the faulty case $0 < k_i < 1$, and when $k_i = 1$ the rotor has failed totally.

The control law in (8.18), is comprised of two parts:

$$u_p(t) = u(t) + \underbrace{(-B^{\dagger}E\zeta(t))}_{u_F(t)} \tag{8.19}$$

where $u(t)$ is the stabilizing sliding mode controller (to be discuss in the sequel) and $B^{\dagger}$ is the pseudo-inverse of B given by

$$B^{\dagger} = B^T(BB^T)^{-1} \tag{8.20}$$

The control signal $u_F(t)$ is the feedback linearisation term which is used to cancel the nonlinear term $E\zeta(t)$ in (8.18), thus linearising the model, which then will be used for the sliding mode design. This is possible since it is assumed that ζ, which is defined in (8.6), is known (i.e. all the states and the system parameters/constants are known), thus the cancellation of $\zeta(t)$ is perfect. Therefore, substituting (8.19) into (8.18) to yield the linear system

$$\dot{x}(t) = Ax(t) + Bu(t) - BK(\underbrace{u(t) + u_F(t)}_{u_p(t)}) + B_r r(t) + D\varsigma(t) \tag{8.21}$$

Note that the system (8.21) will be used for the subsequent SMC synthesis.

8.4.4 Control allocation

Define $W = I - K = diag(w_1 \dots w_8)$, where W represents the effectiveness of each of the rotors. As in previous chapters, in the fault free case $w_i = 1$, in the faulty case $w_i < 1$, and when $w_i = 0$ the rotor has failed totally. Applying the control law in (8.19), the linearised augmented system (8.21) can be written as

$$\dot{x}(t) \quad = \quad Ax(t) + BWu(t) + B_r r(t) + \underbrace{\begin{bmatrix} D & -B \end{bmatrix}}_{D_a} \underbrace{\begin{bmatrix} \varsigma(t) \\ Ku_F(t) \end{bmatrix}}_{\xi(t)} \tag{8.22}$$

where $\xi(t)$ is considered as an accumulated uncertainty and is assumed to be bounded and satisfies

$$\|\xi(t)\| \leq \gamma\|\hat{\nu}(t)\| + \alpha(t, x) \tag{8.23}$$

where $\alpha(\cdot)$ is a known function while the gain γ is a known constant. Note that $\xi(t)$ is 'matched' uncertainty due to the structure of B and D in (8.16).

Due to the structure of the matrix B_p in (8.7)-(8.10), the augmented matrix B in (8.15) and (8.18) can be factorised as

$$B = \underbrace{\begin{bmatrix} 0 \\ I_2 \end{bmatrix}}_{B_\nu} B_2 \tag{8.24}$$

where B_2 is defined in (8.10). To assist in the controller synthesis, a virtual control $v(t)$ is defined as

$$v(t) = B_2 u(t) \tag{8.25}$$

Using (8.25) the control signal $u(t)$ is given by

$$u(t) = B_2^\dagger v(t) \tag{8.26}$$

As in the previous chapters

$$B_2^\dagger = W B_2^T (B_2 W B_2^T)^{-1} \tag{8.27}$$

is the weighted pseudo-inverse of B_2. Note that in (8.27), the pseudo-inverse is weighted using the effectiveness level of each rotors through W.
Using (8.24) and (8.25)-(8.27), the augmented system in (8.22) can be written as

$$\begin{aligned} \dot{x}(t) &= Ax(t) + B_v B_2 W B_2^\dagger v(t) + B_r r(t) + D_a \xi(t) \\ &= Ax(t) + B_v \hat{v}(t) + B_r r(t) + D_a \xi(t) \end{aligned} \tag{8.28}$$

where

$$\hat{v}(t) := B_2 W^2 B_2^T (B_2 W B_2^T)^{-1} v(t) \tag{8.29}$$

8.4.5 Sliding mode control

In the subsequent analysis, a sliding mode controller is designed based on the augmented system in (8.28). Since the reference signal $r(t)$ in (8.28) is known and bounded, the signal does not influence stability and therefore will be ignored in the subsequent analysis. Also note that $\xi(t)$ in (8.28) is 'matched uncertainty' [101, 25] (due to the structure of D and B in (8.16) and (8.24) respectively), which sliding modes are inherently robust against. In the event when a total failure occurs, and provided enough redundancy still exists in the system (the condition $\det(B_2 W B_2^T) \neq 0$ i.e. at least two adjacent rotors remaining), control allocation will be used to redistribute the control signals to the remaining healthy rotors to achieve fault tolerance.

Ignoring the $r(t)$ term, Equation (8.28) can be written in detail as

$$\underbrace{\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix}}_{\dot{x}(t)} = \underbrace{\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}}_{x(t)} + \underbrace{\begin{bmatrix} 0 \\ I_4 \end{bmatrix}}_{B_\nu} \hat{\nu}(t) + \underbrace{\begin{bmatrix} 0 \\ D_{a2} \end{bmatrix}}_{D_a} \xi(t) \tag{8.30}$$

where D_{a2} is the lower half of the matrix D_a. Note that due to the structure of B_ν, Eq. (8.30) is in 'regular form' [101, 25]. For the synthesis of the 'virtual' control $\hat{\nu}(t)$, first define

$$s(t) = Sx(t) \tag{8.31}$$

which represents the sliding mode switching function, and $S \in \mathbb{R}^{2\times 6}$ is the sliding hyperplane matrix. For convenience in the subsequent analysis, it will be assumed that the system has been scaled such that $SB_\nu = I_2$. Also define

$$\mathcal{S} = \{x \in \mathbb{R}^6 \ : \ Sx = 0\}$$

as the sliding hyperplane which define the system's closed-loop performance. The control law is chosen to ensure that the closed-loop system trajectories are forced onto the hyperplane $\mathcal{S}$ in finite time and constrains the states to remain there. Since the system (8.30) in regular form, the choice

$$S = \begin{bmatrix} M & I_2 \end{bmatrix} \tag{8.32}$$

is a suitable design for the sliding hyperplane matrix and $M \in \mathbb{R}^{2\times 4}$ is the design freedom. Introduce a transformation so that $(x_1, x_2) \mapsto T_s x = (x_1, s)$ where

$$T_s = \begin{bmatrix} I_4 & 0 \\ M & I_2 \end{bmatrix} \tag{8.33}$$

is the nonsingular transformation matrix. In the new coordinates, Equation (8.30) becomes

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{s}(t) \end{bmatrix} = \begin{bmatrix} \hat{A}_{11} & \hat{A}_{12} \\ \hat{A}_{21} & \hat{A}_{22} \end{bmatrix} \begin{bmatrix} x_1(t) \\ s(t) \end{bmatrix} + \begin{bmatrix} 0 \\ I_4 \end{bmatrix} \hat{\nu}(t) + \begin{bmatrix} 0 \\ D_{a2} \end{bmatrix} \xi(t) \tag{8.34}$$

where

$$\begin{aligned} \hat{A}_{11} &:= A_{11} - A_{12}M, \\ \hat{A}_{21} &:= M\hat{A}_{11} + A_{21} - MA_{22}, \quad \hat{A}_{22} := MA_{12} + A_{22} \end{aligned} \tag{8.35}$$

If a control law can be designed so that closed-loop state trajectories reach the sliding hyperplane and remain there, then during ideal sliding $\dot{s}(t) = s(t) = 0$. Therefore from the top partition of (8.34), the reduced order sliding motion is given by:

$$\dot{x}(t) = \hat{A}_{11}\hat{x}_1(t) \tag{8.36}$$

By choice of M in (8.32), the matrix $\hat{A}_{11}$ (as defined in (8.35)) can be made stable.
The virtual control law is given by

$$\hat{\nu}(t) = \hat{\nu}_l(t) + \hat{\nu}_n(t) \tag{8.37}$$

where the linear component is defined as

$$\hat{\nu}_l(t) = -\hat{A}_{21}\hat{x}_1(t) - (\hat{A}_{22} - \Phi)s(t) \tag{8.38}$$

and $\Phi \in \mathbb{R}^{2\times 2}$ is a stable design matrix. The nonlinear component is defined as

$$\hat{\nu}_n(t) = -\rho(t,x)\frac{P_2 s(t)}{\|P_2 s(t)\|} \qquad \text{if } s(t) \neq 0 \tag{8.39}$$

where $P_2 \in \mathbb{R}^{2\times 2}$ is symmetric positive definite matrix which satisfies

$$P_2\Phi + \Phi^T P_2 = -I_2 \tag{8.40}$$

Due to the matched uncertainty structure in (8.34), the influence of uncertainty does not affect the reduced order sliding motion in (8.36), which is the well known property of sliding mode [101, 25]. Therefore, the stability analysis of the closed-loop system with matched uncertainty becomes the problem of ensuring that sliding can be maintained despite the presence of uncertainty or faults.
If the matrix M has been chosen so that $\hat{A}_{11}$ in (8.35) stable and $\rho(t,x)$ in (8.39) is chosen as

$$\rho(t,x) \geq \frac{\|D_{a2}\|\left(\gamma\|\hat{\nu}_l\| + \alpha(t,x)\right) + \eta}{(1 - \gamma\|D_{a2}\|)} \tag{8.41}$$

then the controller in (8.37)-(8.40) ensures a sliding motion takes place on $\mathcal{S}$ despite in the presence of (matched) uncertainty. The proof is similar to the one in Chapter 4 and in [101, 25]. Note that the bound of the uncertainty in (8.23) has been utilise to obtain (8.41).

8.4.6 Final control law

From (8.19), the overall control law is given by

$$u_p(t) = \underbrace{WB_2(B_2W^2B_2^T)^{-1}\hat{\nu}(t)}_{u(t)} + \underbrace{(-B^\dagger E\zeta(t))}_{u_F(t)} \tag{8.42}$$

where $u_F(t)$ is the feedback linearisation term describe in Section 8.4.3. The control law $u(t)$ is the sliding mode control term obtained using (8.26)-(8.27), (8.29) and (8.37)-(8.40). Note that the sliding mode control law $u(t)$ sent to the actuators is dependent on the effectiveness gains w_i (from the diagonal weighting matrix W). Here, it will be assumed that this information is available from a fault detection and isolation (FDI) scheme (see for example [25]).

8.5 Outer loop control

In most sliding mode FTC literature (see for example [25]), the focus has been in the inner loop control and the outer loop position tracking is typically provided by a simple PID control. Although it is simple and effective, this strategy is not without limitations (e.g. limited robustness guarantees of position control). In this chapter, a sliding mode control will be considered instead. This is achieved by exploiting the differential flatness properties of the multirotor UAV (see for example [128]). The idea is for the outer loop SMC control to produce the desired/commanded reference command signal for the inner loop SMC controller. The reference signals for the outer loop control are the desired position/trajectory of the UAV, supplied by the UAV operator. To aid in the analysis that follows, it will be assumed that the desired trajectory provided by the operator will be smooth to the fourth order.

8.5.1 Position and altitude control

The equation that describe the translational motion of the octorotor in the inertial axis is given by

$$\begin{aligned}
\begin{bmatrix} \ddot{x}_e(t) \\ \ddot{y}_e(t) \\ \ddot{z}_e(t) \end{bmatrix} &= R_b^i(\phi,\theta,\psi)^T \begin{bmatrix} 0 \\ 0 \\ -T_h \end{bmatrix} \left(\frac{1}{m_{kg}}\right) + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \\
&= \begin{bmatrix} c_\phi s_\theta c_\psi + s_\phi s_\psi \\ c_\phi s_\theta s_\psi - s_\phi c_\psi \\ c_\phi c_\theta \end{bmatrix} \left(\frac{-T_h}{m_{kg}}\right) + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \qquad (8.43)
\end{aligned}$$

where T_h is the total thrust acting on the octorotor (upwards), and $R_b^i(\cdot)$ is the direction cosine matrix defined in (??). For outer loop position control purposes, define outer loop states as

$$x_o(t) = \begin{bmatrix} x_e(t) & y_e(t) & z_e(t) & \dot{x}_e(t) & \dot{y}_e(t) & \dot{z}_e(t) \end{bmatrix}^T \tag{8.44}$$

The nonlinear dynamics of the translational motion can be written in a state space form as

$$\dot{x}_o(t) = A_o x_o(t) + B_o u_o(t) + b_o g \tag{8.45}$$

where g is gravity and

$$A_o = \begin{bmatrix} 0_{3\times3} & I_{3\times3} \\ 0_{3\times3} & 0_{3\times3} \end{bmatrix}, \quad B_o = \begin{bmatrix} 0_{3\times3} \\ I_{3\times3} \end{bmatrix}^T, \quad b_o = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}^T \tag{8.46}$$

Similar to [128], it will be assumed that the inner loop controller will be able to achieve the desired input attitude (i.e. $\phi = \phi_d$, $\theta = \theta_d$ and the desired total thrust $T_h = T_d$). In (8.45), the input $u_o(t)$ represents a non linear 'mapping' given by

$$u_o(t) = \begin{bmatrix} u_{o1}(t) \\ u_{o2}(t) \\ u_{o3}(t) \end{bmatrix} = (R_b^i(\phi_d, \theta_d, \psi))^T \begin{bmatrix} 0 \\ 0 \\ -T_h \end{bmatrix} \left(\frac{1}{m_{kg}}\right) \tag{8.47}$$

Writing $u_o(t)$ as a 'mapping' in (8.47), allows the nonlinear model (8.43) to be written as the linear state space form (8.45).

8.5.2 Outer-Inner sliding mode control

The control law $u_o(t)$ will be designed using sliding mode methods. Here, since the gravitational term g is known, it will be ignored in this section and will be 'cancelled' in the subsequent section.

As in Section 8.4.2, integral action will be considered to provide tracking capability for position $x_e(t), y_e(t), z_e(t)$. Here, the integral action state is given by

$$\dot{x}_c(t) = y_c(t) - C_c x(t) \tag{8.48}$$

where $y_c(t)$ is the smooth and differentiable reference signal which represents the command signals for the desired $x_e(t), y_e(t), z_e(t)$ position, and $C_c = \begin{bmatrix} I_3 & 0_{3\times3} \end{bmatrix}$ is the controlled output distribution matrix. Augmenting the states from (8.45) with the integral action states in (8.48) to yield

$$\dot{x}_a(t) = A_a x_a(t) + B_a u_o(t) + B_c y_c(t) \tag{8.49}$$

where the augmented states is defined as $x_a(t) := \text{col}(x_c(t), x(t))$ and

$$A_a = \begin{bmatrix} 0 & -C_c \\ 0 & A_o \end{bmatrix}, \quad B_a = \begin{bmatrix} 0 \\ B_o \end{bmatrix}, \quad B_c = \begin{bmatrix} I_3 \\ 0 \end{bmatrix} \tag{8.50}$$

The switching function for the outer loop controller is given by

$$s_a(t) = S_a x_a(t) = \begin{bmatrix} M_a & I_3 \end{bmatrix} \tag{8.51}$$

where $M_a \in \mathbb{R}^{3\times 6}$. As in section (8.4.5), the control law comprises linear and nonlinear components given by

$$u_o(t) = u_{o_l}(t) + u_{o_n}(t) \tag{8.52}$$

The linear component contains a feed-forward reference term due to the reference signal $y_c(t)$, given by

$$u_{o_l}(t) = L_a x_a(t) + L_c y_c(t) \tag{8.53}$$

where

$$L_a = -(S_a A_a - \Phi_a S_a) \text{ and } L_c = -M_a B_c \tag{8.54}$$

The matrices A_a, B_a and S_a are from (8.50) and (8.51) which are already in regular form while Φ_a is the design freedom (analogous to Φ in (8.40)). The nonlinear component is defined as

$$u_{o_n}(t) = -\rho_o(t, x_a)\frac{s_a(t)}{\|s_a(t)\|} \quad \text{for } s_a(t) \neq 0 \tag{8.55}$$

where $\rho_o(t, x_a)$ is the modulation gain for the outer loop SMC, which is chosen to satisfy

$$\rho_o(t, x_a) > \eta_o \tag{8.56}$$

where η_o is a positive design scalar. As shown in [101, 25], since the system contains no uncertainties, the choice of the modulation gain (8.56) is sufficient to ensure that sliding is attained in and subsequently maintained.

8.5.3 Outer-Inner loop relation

Once the outer-loop sliding mode controller has been designed, the idea is to use the control signal $u_o(t)$ in (8.52) to extract the desired thrust as well as the roll and pitch demand signals, by exploiting the linear 'mapping' in (8.47).

Using (8.47), (8.45) and (8.43), the signal $u_o(t)$ can be viewed as the desired inertial acceleration

$$u_o(t) = \begin{bmatrix} \ddot{x}_d & \ddot{y}_d & \ddot{z}_d \end{bmatrix}^T \tag{8.57}$$

Since the aircraft is under the gravitational acceleration $g = 9.807 m/s^2$, define

$$u_g(t) = u_o(t) - \begin{bmatrix} 0 & 0 & g \end{bmatrix}^T \tag{8.58}$$

Using free body diagram analysis, and calculating the total forces in vertical direction, the total magnitude of thrust required from the motors T_h to overcome the gravity and weight is given by:

$$T_h = m_{kg} \|u_g(t)\| \tag{8.59}$$

Since the $z - axis$ is facing downward then the total required thrust F_{z_d} would be

$$F_{z_d} = -T_h \tag{8.60}$$

The signal F_{z_d} (which does not depends on the inner loop control), will be sent directly to the control allocation unit shown in Figure 8.2 in order to find the required speed of each remaining rotors.

As shown in Figure 8.2, the desired roll and pitch used by the inner-loop controller is provided by the outer-loop position control exploiting the outer-inner loop relation.

Using the same idea as in [128], a further transformation of the inertial axes around z inertial axes by a yaw angle ψ will be considered. As in [128], this transformation produces a 'control axes' system where the desired acceleration from the rotors is given by:

$$u_C(t) = R_b^i(\psi) \times u_g(t) \tag{8.61}$$

where $u_g(t)$ is defined in (8.58) and

$$R_b^i(\psi) = \begin{bmatrix} cos(\psi) & sin(\psi) & 0 \\ -sin(\psi) & cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{8.62}$$

During a steady state condition, the linear acceleration affects on the body frame from the rotors would be only the thrust in z body axes, hence

$$u_C = \begin{bmatrix} u_{c_1} \\ u_{c_2} \\ u_{c_3} \end{bmatrix} = R_b^{i^T}(\theta_d) R_b^{i^T}(\phi_d) \begin{bmatrix} 0 \\ 0 \\ F_{z_D}/m_{kg} \end{bmatrix} \tag{8.63}$$

where

$$R_b^i(\phi_d) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\phi_d) & sin(\phi_d) \\ 0 & -sin(\phi_d) & cos(\phi_d) \end{bmatrix}, \quad R_b^i(\theta_d) = \begin{bmatrix} cos(\theta_d) & 0 & -sin(\theta_d) \\ 0 & 1 & 0 \\ sin(\theta_d) & 0 & cos(\theta_d) \end{bmatrix} \tag{8.64}$$

Hence through the differential flatness property of the translational dynamics [128], it is possible to find a relationship between desired acceleration and desired attitude angles using (8.63). Therefore, once u_{c_1}, u_{c_2} and u_{c_3} is known, the desired roll and pitch angles can be obtained using the following:

$$\phi_d = sin^{-1}(u_{c_2}) \left(\frac{m_{kg}}{T_h} \right) \tag{8.65}$$

and

$$\theta_d = tan^{-1} \left(\frac{u_{c_1}}{u_{c_3}} \right) \tag{8.66}$$

As shown in shown in Figure 8.2, these signals represent the desired roll and pitch angles which used by the inner-loop controller.

8.5.4 Overall forces and moments

Since in the fault-free case, the octorotor is an over actuated system, to find the angular rotational speed of the individual rotors, Figure 8.2 shows a control allocation unit it is required. From (8.4), once the force F_z and moments $\mathcal{L}, \mathcal{M}, \mathcal{N}$ have been obtained from the controllers, the angular velocity of each rotors Ω_i can be obtained through the control allocation unit. Since different controllers produces $F_z, \mathcal{L}, \mathcal{M}, \mathcal{N}$, the final control law (which provide individual rotor speed) is given by:

$$u_\Omega(t) = \begin{bmatrix} \Omega_1^2 & \dots & \Omega_8^2 \end{bmatrix}^T = B^\dagger_{\Omega_{F_z}} F_z + B^\dagger_{\Omega_N} \mathcal{N} + u_p(t) \tag{8.67}$$

where $u_p(t)$ is the control law for the inner loop control in (8.42), F_z is obtained from (8.60) and $\mathcal{N}$ is the desired yaw moment from the yaw (PID) control (which will be discussed in

Table 8.1: Chosen Control input vectors and there representing matrices

	Fault Free Case and remaining rotors ≥ 4	3 rotors remaining	two rotors remaining
$\tau'(t)$	$\begin{bmatrix} F_z & \mathcal{L} & \mathcal{M} & \mathcal{N} \end{bmatrix}^T$	$\begin{bmatrix} F_z & \mathcal{L} & \mathcal{M} \end{bmatrix}^T$	$\begin{bmatrix} F_z & \mathcal{L} \end{bmatrix}^T$

detail later). In (8.67), the various input matrices are given by:

$$B_{\Omega_{F_z}} = \begin{bmatrix} -b & -b & -b & -b & -b & -b & -b & -b \end{bmatrix} \tag{8.68}$$

$$B_{\Omega_N} = \begin{bmatrix} d & -d & d & -d & d & -d & d & -d \end{bmatrix} \tag{8.69}$$

In (8.67), the parameter $B^{\dagger}_{\Omega_i}$ represents the weighted pseudo-inverse of $B^{\dagger}_{\Omega_i}$ defined as

$$B^{\dagger}_{\Omega_i} = WB^T_{\Omega_i}\left(B_{\Omega_i}WB^T_{\Omega_i}\right)^{-1} \tag{8.70}$$

where W is a diagonal matrix whose diagonal elements varies between $[0:1]$ representing the effectiveness level of each rotors. Note that the inner loop control law $u_p(t)$ in (8.67) as defined in (8.42), is used to produced the desired roll and pitch moment $\mathcal{L}, \mathcal{M}$.

As shown in Figure 8.2, in the event of three rotors remaining, the yaw control will be sacrificed and therefore the terms related to the yaw moment $\mathcal{N}$ control in (8.67) will be removed (using an FDI switch described in Section 8.3). In the most extreme case when only two adjacent rotors remain, the pitch control will be sacrificed and the control law $u_p(t)$ in (8.67) and (8.42) is slightly modified. In this extreme case, the second row of B_2 in (8.10) (associated with pitch moment) is removed and the control allocation only depends on the pseudo-inverse of the top row of B_2 given by

$$B_{21} = \begin{bmatrix} -\frac{1}{I_{xx}}b\ell_1 & \frac{1}{I_{xx}}b\ell_1 & \frac{1}{I_{xx}}bvl_2 & \frac{1}{I_{xx}}b\ell_2 & \frac{1}{I_{xx}}b\ell_1 & -\frac{1}{I_{xx}}b\ell_1 & -\frac{1}{I_{xx}}b\ell_2 & -\frac{1}{I_{xx}}b\ell_2 \end{bmatrix} \tag{8.71}$$

The exclusion of the pitch moment control (using an FDI switch) is possible since the system in (8.30) in which the inner loop control was designed is decoupled in the roll and pitch axis due to the structure of the original systems in (8.6). As a summary, depending on the fault/failure cases, the variation of final control law is summarized in Table (8.1).

8.5.5 Summary of control structure

This subsection provides a summary of the control synthesis for the proposed scheme. This complements Figure 8.2 and the high level description of the proposed scheme given in Section 8.3.

From Figure 8.2, the aircraft's 'desired acceleration' signal, described in inertial axes (see Equation (8.57)) on the left side of Figure 8.2 is calculated using the desired position trajectory, the measured (actual) position and the aircraft inertial velocity (via SMC). In the 'outer-inner loop relation' block, the 'desired acceleration' signal is converted to the desired acceleration in the control axes (see Equation (8.61)). Using the three components of this signal, the desired total thrust F_{zd} can be be obtained using Equation (8.60) after the gravitational force has been extracted through Equations (8.58)-(8.59). Also using the desired acceleration in the control axes in (8.57), the desired attitude roll (ϕ) and pitch (θ) angles can be calculated using (8.65) and (8.66). In the 'attitude SMC' block, the desired roll and pitch moments ($\mathcal{L}$ and $\mathcal{M}$) are calculated from the desired and measured roll and pitch angles and the measured angular velocities rates p, q as given in Equation (8.42). The 'yaw controller PID' block generates the desired yaw moment ($\mathcal{N}$) using a typical PID based design.

The 'control allocation' unit converts the 'virtual' control law (the desired force F_z as well as the desired roll, pitch and yaw moments $\mathcal{L}$, $\mathcal{M}$ and $\mathcal{N}$) to individual rotor speed. As in the previous chapter, the control allocation uses the rotor effectiveness levels to redistribute the control signals to the remaining available rotors to achieved the desired forces and moments. In this chapter, it will be assumed that FDI will be available to provide information on each rotor's effectiveness level. This information is used in the control allocation unit to redistribute the control signals to the remaining rotors (when more than four rotors are available) and also to disable the yaw and pitch moment control when there are only three and two motors remaining (under actuated cases).

In the fault free case or in the event of faults/failures where the total number of failed rotors are less than 4, the system still has a sufficient number of rotors and therefore all the 'virtual inputs' $F_z, \mathcal{L}, \mathcal{M}, \mathcal{N}$ will be considered. When only 3 rotors are left, the system becomes under actuated, and the yaw control is sacrificed and removed from the overall control (as shown in the FDI-activated switch in Figure 8.2). In this scenario, only $F_z, \mathcal{L}, \mathcal{M}$ will be utilised. When only 2 opposite motors are left, pitch control is also sacrificed and removed from the overall control and the 'virtual' inputs, leaving only F_z and $\mathcal{L}$ as the remaining control (as shown in the FDI-activated switch in Figure 8.2). This will be sufficient to control the octorotor in both roll and pitch rotation directions, and will still allow the octorotor to achieve some position tracking performance.

Table 8.2: Storm8 octorotor physical parameters

Parameter	Value	Unit
m_{kg}	1.214	kg
I_{xx}	0.009565	kgm^2
I_{yy}	0.013746	kgm^2
I_{zz}	0.017866	kgm^2
b	7.8×10^{-8}	N/rpm^2
d	1.8065×10^{-9}	Nm/rpm^2
ℓ	0.21	m

Table 8.3: Chosen Control input vectors and there representing matrices

System	Q	Φ	ρ	δ
Outer Loop	$diag([0.0001, 0.0001, 0.0001, 1, 1, 1, 1, 1, 1])$	-1	0.1	0.1
Inner Loop	$diag([0.6, 0.6, 40, 40, 1, 1])$	-20	100	0.01

8.6 Design

8.6.1 Storm8 octorotor physical parameters

The Storm8 octorotor [41] parameters are summarized in Table 8.2. Note that the moments of inertia of the octorotor are obtained using bifilar pendulum method [130, 42] as described in the Appendix B.

8.6.2 Sliding mode control parameters

For the inner and outer loop systems in (8.7) and (8.45), integral action is considered to provide tracking capabilities to the sliding mode controller (see Section 8.4.2 and 8.5.2). Both the inner and outer loop controllers are designed based on an LQR-like synthesis for the reduced order system. The design parameters are given in Table (8.3). Note that the first two entry in the diagonal Q design matrix are associated with integral action states and are therefore least weighted. Also note that the inner-loop controller is designed to be more aggressive as compared to the outer loop as the inner loop dynamics are faster and therefore need to be stabilised quicker. This is reflected in Table (8.3) where the values of Φ and ρ are larger (ensuring sliding is attained faster), while δ is much smaller (ensuring robustness is attained while sacrificing the smoothing effect of the pseudo sigmoidal term).

8.6.3 Yaw control

As shown in Figure 8.2, a separate yaw control is designed using a more typical PID controller. The yaw control only active when there are sufficient rotors available (i.e. the number of remaining rotors are ≥ 4). In the event when the remaining rotors drop to 3 or below, the system becomes under actuated and the FDI switch is used to remove the yaw control, thus allowing the octorotor to rotate freely in the yaw axis. As discussed earlier, this is a similar strategy to the one used in the last chapter.

Here, the desired angular acceleration $\dot{r}_d$ is given by

$$\dot{r}_d = K_d(\psi_d - \psi) - K_d r + K_i \int (\psi_d - \psi)dt \tag{8.72}$$

where ψ_d is the desired yaw angle. The PID controller parameters are manually tuned and are given by $K_p = 4$, $K_d = 4$, $K_i = 0.1$. These gains ensure a closed loop natural frequency $\omega_n = 2$ and damping ration $\zeta_d = 1$. The desired yaw moment N_d is given by

$$\mathcal{N}_d = I_{zz}\dot{r}_d \tag{8.73}$$

8.7 Results and summary

During simulation, the desired position trajectory x_{o_d} (in inertial axis) is given by:

$$x_{o_d} = \begin{bmatrix} sin\,(2\omega_d t)\,cos\,(\omega_d t) \\ sin\,(2\omega_d t)\,sin\,(\omega_d t) \\ z_{des} \end{bmatrix} \tag{8.74}$$

where $\omega_d = 0.1 rad/sec$ and $z_{des} = -10m$. Note that this trajectory is smooth to the fourth order (which satisfies the requirement for the differential flatness). This desired position results in a distinctive trajectory as shown in Figure 8.3a. This manoeuvre will be considered for all tests considered in this chapter for consistency and allow for direct comparisons of performance for different fault/failure scenarios.

In this chapter, five different scenarios are considered as summarised in Table 8.4. The simulations start with the fault free case to highlight the nominal performance. The remaining scenarios are set to be of increasing difficulty/challenge (from faults to four, five and six rotors failures as shown in Table 8.4). Note that the effectiveness of each of the rotors is assumed to be known from an FDI (see for example FDI from [25]). The simulation was conducted using SIMULINK with a fixed time solver *ode*3 with a time step of $0.001s$.

8.7.1 Fault free

Figure 8.3 presents the fault-free scenario. The desired and the actual x, y, z position trajectories are shown in Figure 8.3a. The inner loop desired and measured states are shown in Figure 8.3b. Here, it can be seen that the roll, pitch and yaw angles track the

Table 8.4: Test scenarios

Test	No. of failed rotors	W_{diag}
Fault-Free	0	[1 1 1 1 1 1 1 1]
Faulty (Over Actuated)	2, (3 Faulty)	[1 0 0.7 0.5 1 0.3 0 1]
Four Motors (X-Configuration)	4	[1 0 1 0 0 1 0 1]
Three Motors (Under Actuated)	5	[0 0 0 1 0 0 1 1]
Two Motors (Under Actuated)	6	[1 0 0 0 1 0 0 0]

desired angles closely. The switching function for both outer (s_x, s_y, s_z) and inner loops $(sin(\phi), sin(\theta))$ are shown in Figure 8.3c. Here the inner loop switching function are kept very small of order 1×10^{-4} and the outer loop was kept lower than 1×10^{-1}, indicating that sliding is maintained throughout the simulation. The individual rotor speeds $\Omega_1, \ldots, \Omega_8$ are shown in Figure 8.3d, where the nominal (hover) rotor speed is around $4000rpm$.

8.7.2 Faulty (Over actuated)

The second scenario represents a combination of faults and failures but still over actuated. Here two rotors are completely failed, while six other rotors are at different effectiveness levels i.e. $W_{diag} = [\ 1\ \ 0\ \ 0.7\ \ 0.5\ \ 1\ \ 0.3\ \ 0\ \ 1\ ]$. The results are shown in Figure 8.4 which show the same fault-free performance still being maintained. The outer loop position tracking as well as inner loop roll, pitch and yaw angle tracking remains unchanged as compared to the fault free case. The sliding motions are still maintained as the switching functions remain close to zero. The effect of faults and failures to all the rotors can be seen in Figure 8.4. Here, rotor 2 and 7 are completely failed (rotor speed are zero), rotors 1, 5 and 8 are fully effective while rotors 3, 4 and 6 are 70%, 50% and 30% respectively. The rotor speed for rotor 1,5 and 8 are found to increase to the maximum nominal value at about $6000rpm$ to compensate for the faults and failures to the other rotors.

8.7.3 Four motors (X-configuration) remaining

Figure 8.5 shows the results when rotors 2, 4, 5 and 7 fail (rotor speed are zero), while the remaining rotors are fully effective. Note that the remaining rotors have higher nominal speeds to compensate for the failed rotors, and a maximum value of $7000rpm$ has been reached. In this scenario, the octorotor behaves like an equivalent x-shaped quadrotor (although as shown in Figure 8.1, it is not a symmetric quadcopter). As in the previous scenarios, Figure 8.5 shows no degradation in the position and angles tracking performance and the switching functions remain close to zero, similar to the fault free case.

8.7.4 Under actuated scenarios

The remaining two scenarios (Figures 8.6 and 8.7) are significantly more challenging than then the previous scenarios. Here the number of remaining rotors drop below 4 (the minimum number required to fly nominally) and the system becomes under actuated.

8.7.4.1 Three motors remaining

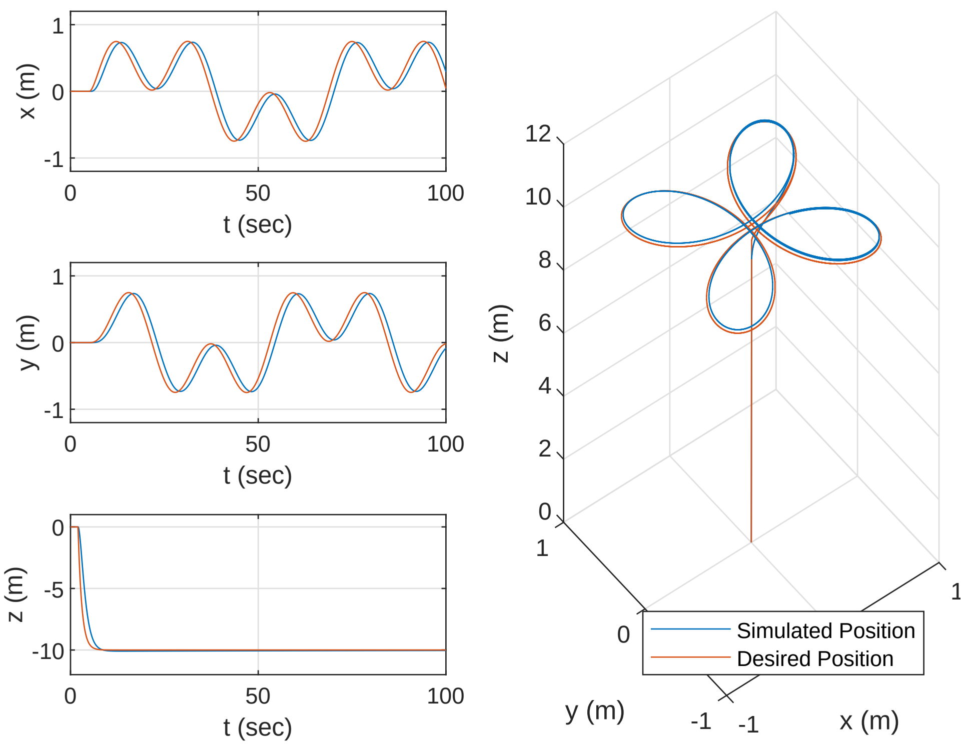
In Figure 8.6, five rotors have failed (rotors 1, 2, 3, 5 and 6) and only three rotors remain. In this case, the system becomes under actuated and as described earlier, the yaw angle ψ control is sacrificed (the octorotor rotates in the yaw axis and the yaw angle fluctuates between ± 180 deg) and the yaw rate r reached about $1100 deg/s$ (about $3 rev/sec$). Despite only three rotors remaining, Figure 8.6b shows a small change to the position tracking performance and the octorotor is still able to complete the manoeuvre. Since the failure occurs from the beginning of the simulation when the octorotor takeoffs from the ground, the switching functions start at some high values as shown in Figure 8.6c. However, after a few seconds sliding is achieved and subsequently maintained close to zero. It has to be noted that in the case when only three rotors remain, it is possible to configure the controller to switch off one of the rotors and only maintain the two opposite rotors, while allowing the roll and pitch angles at zero degree angles. However this will increase the burden (and rotational speed) of the two remaining rotors. In this chapter, by assuming a non zero trim roll and pitch angle is considered (as shown in Figure 8.6b), the three motors will be kept operational, albeit one rotor has a slower speed than the other two. This is shown in Figure 8.6d. Figure 8.6b shows both the roll and pitch angles ϕ_d, θ_d have a desired values of 1 deg. This non zero trim conditions allow the 7^{th} rotor to remain operational at about $2000 rpm$ (see Figure 8.6d).

8.7.4.2 Two motors remaining

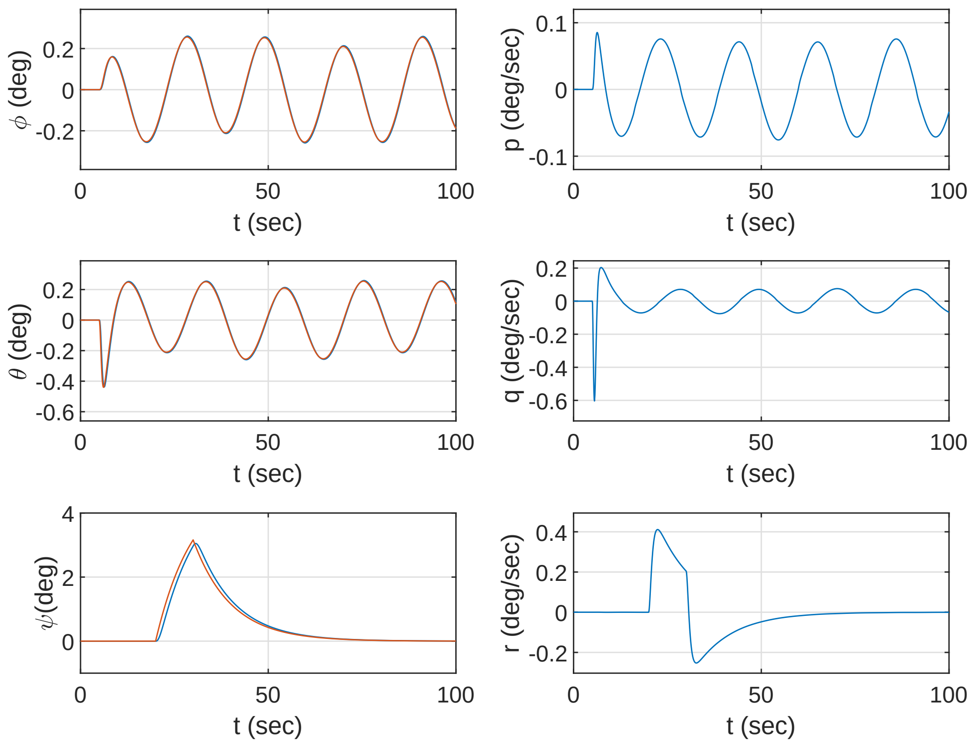
The final scenario is the most challenging case. Here only two rotors remain (six rotors totally failed). In this case, yaw and pitch control will be sacrificed, maintaining only altitude and roll control. Figure 8.7 shows the yaw and the pitch angles ψ, θ are sacrificed and only the roll angle ϕ is used to track x, y position. The yaw rate r reaches a maximum of about $3 rev/sec$ (1100 deg/sec). Since the aircraft is continuously rotating in the yaw axis, roll angle (inner loop) control is sufficient to control both x, y positions albeit with some small degradation as shown in Figure 8.7a. Despite this small degradation, the octorotor still manages to complete the manoeuvre. The switching functions are slightly higher as compared to the previous scenarios (around 0.2) although this is still small. The remaining rotors 1 and 5 have a nominal speed of $8000rpm$ which is much higher than the fault free case (around $4000rpm$). This is due to the fact that the two remaining rotors are compensating for the loss of the other six rotors and requires higher thrust to maintain hover.

8.8 Summary

This chapter has presented an FTC scheme for octorotors. The control strategy is capable of controlling the aircraft during over/full/under actuated scenarios. Unlike most of the work done on similar octorotor in the literature, the scheme proposed in this chapter uses SMC for both outer and inner loop control. This is done utilising the differential flatness properties of the octorotor to build the appropriate state-space models and nonlinear feedback linearisation to eliminate the nonlinearities in the system. For inner loop control, online control allocation was combined with SMC to redistribute the control signals depending on the effectiveness level (health) of the rotors. In total, five scenarios have been considered in this chapter. The first scenario is the fault-free case to highlight the capabilities of the controller in a nominal over actuated condition. Over actuated faults/failures scenarios have also been conducted where the number of failed rotors are two and the remaining six rotors have different effectiveness levels. The third scenario considers a sufficiently actuated case where 4 rotors remain. The UAV behaves like a typical quadrotor albeit a nonsymmetric one. Finally, two extreme scenarios were considered where only three and two remaining rotors are present (under actuated scenarios). In the three rotors scenario, yaw control was sacrificed to maintain position tracking. In the two motors scenario, both yaw and pitch control was sacrificed while maintaining altitude and roll tracking. The continuous rotation in the yaw axis is sufficient to allow roll tracking to control x and y position. The simulation results showed a good tracking performance in all the tested scenarios indicating the efficiency of the proposed scheme.

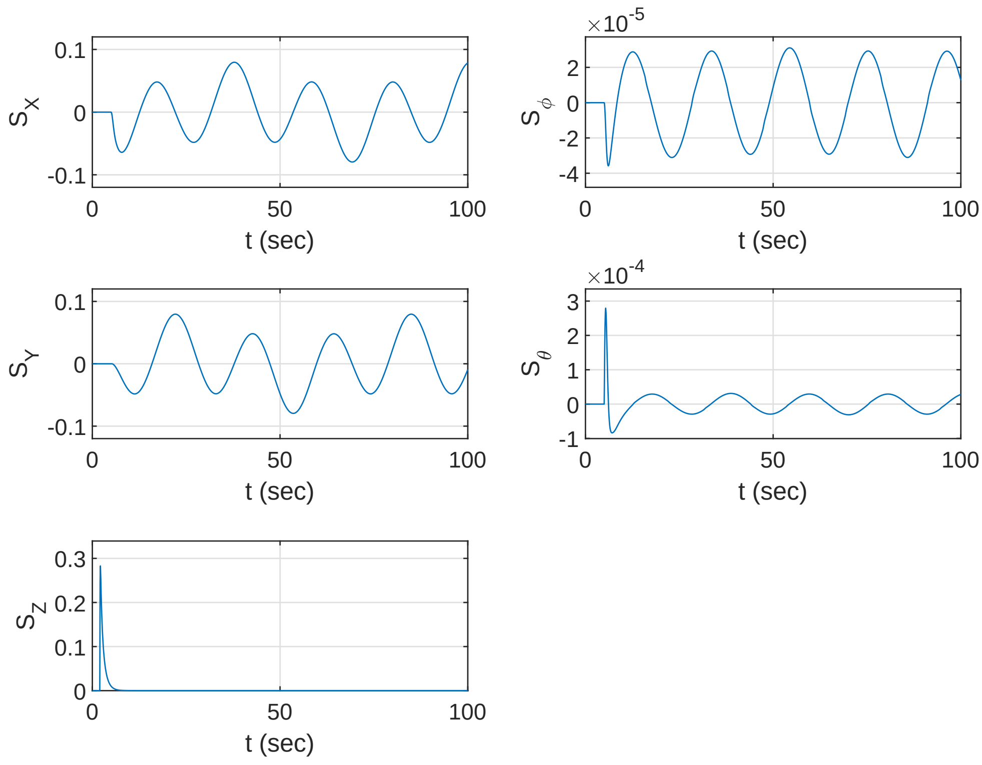


(a) Desired and Simulated Aircraft Position

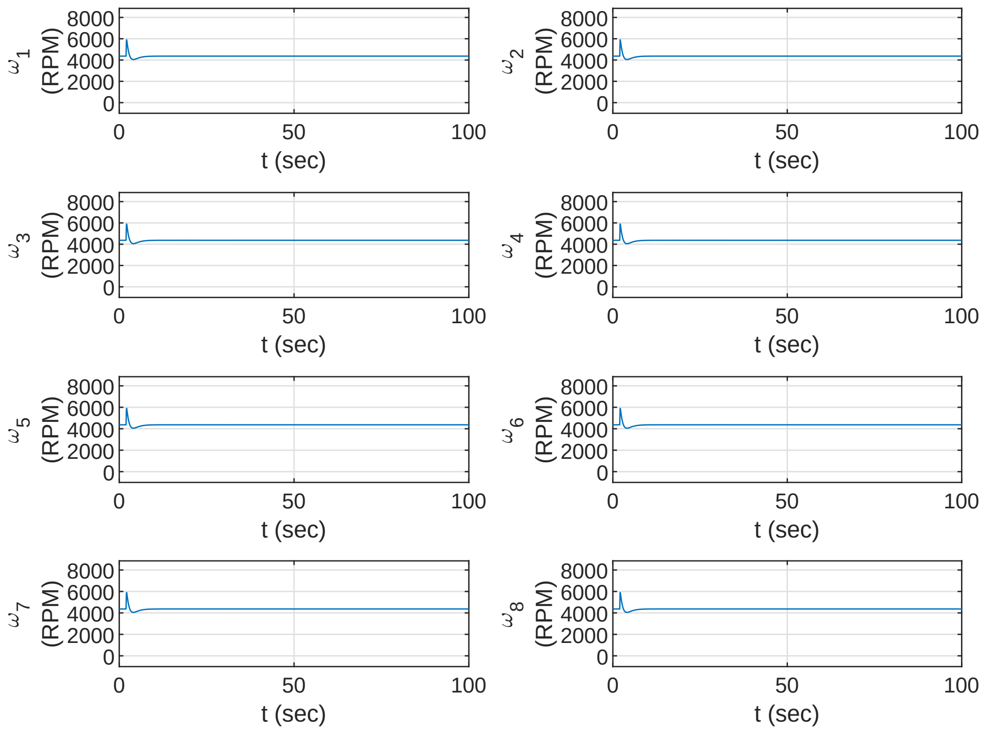


(b) Desired and Simulated Euler angles and Simulated Angular Rates

Figure 8.3: Fault Free

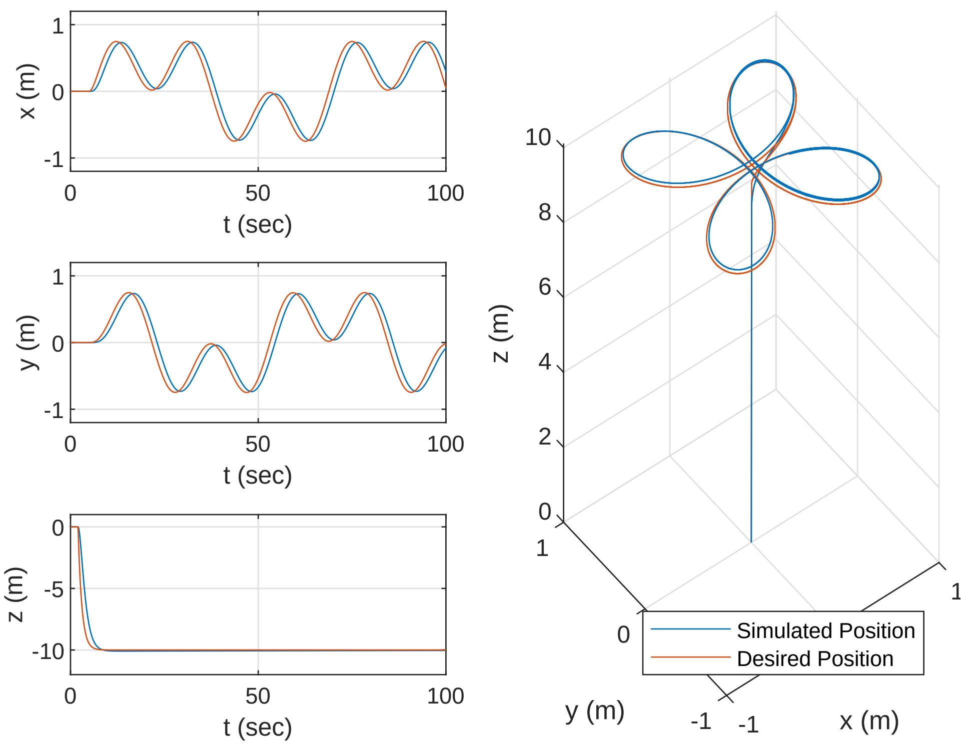


(c) Switching Functions

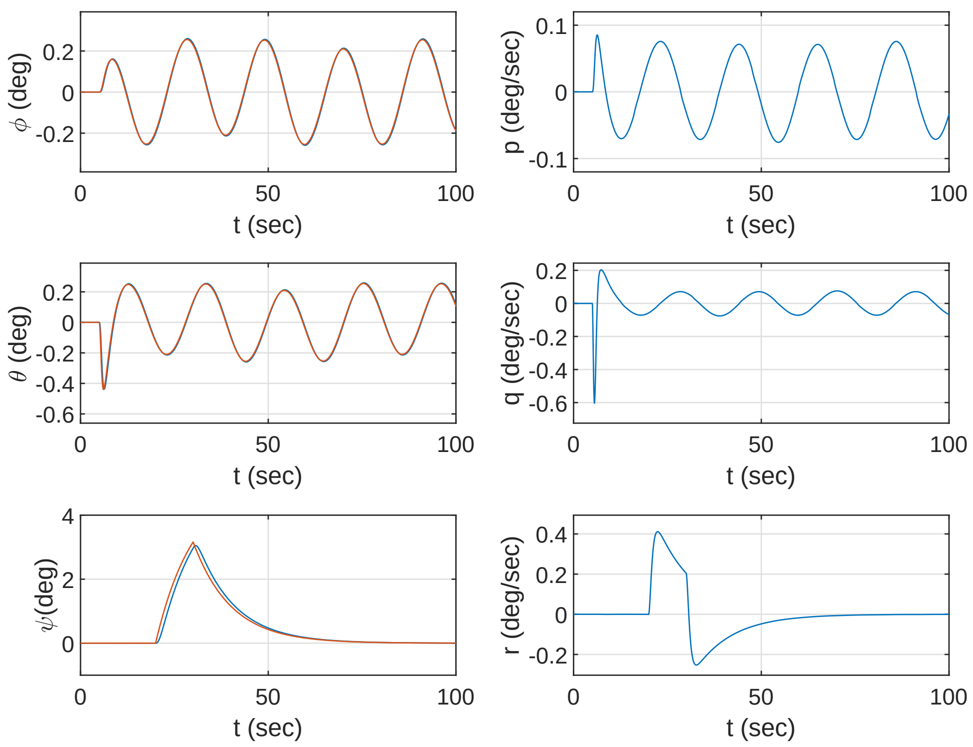


(d) Simulated Motors Angular Velocities

Figure 8.3: Fault Free

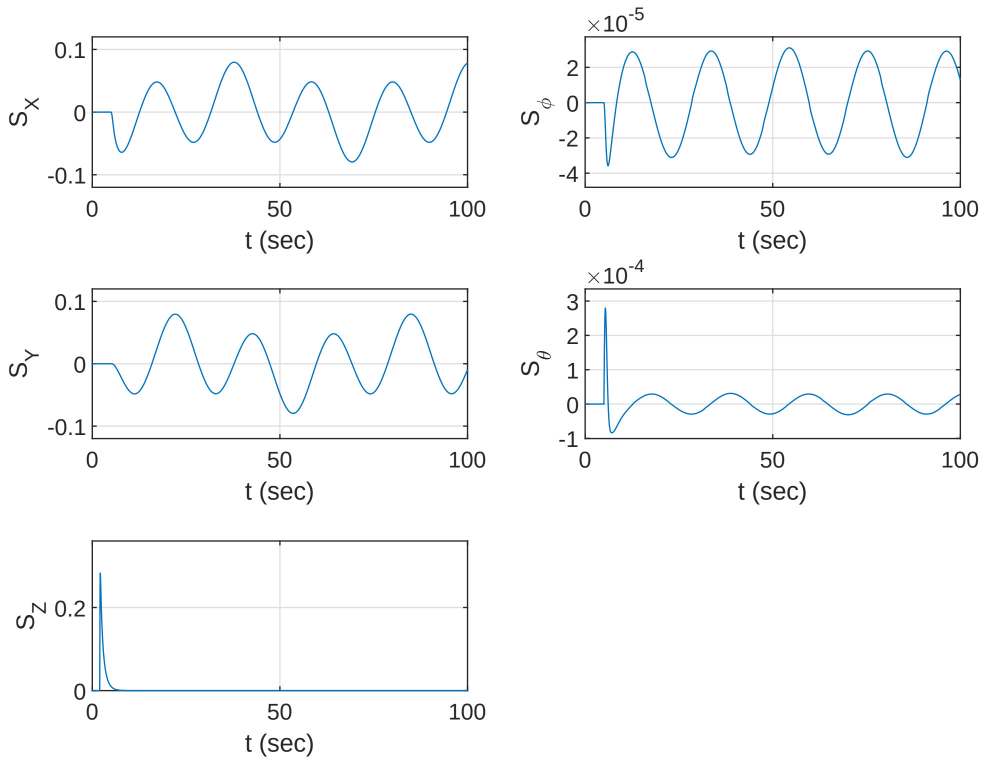


(a) Desired and Simulated Aircraft Position

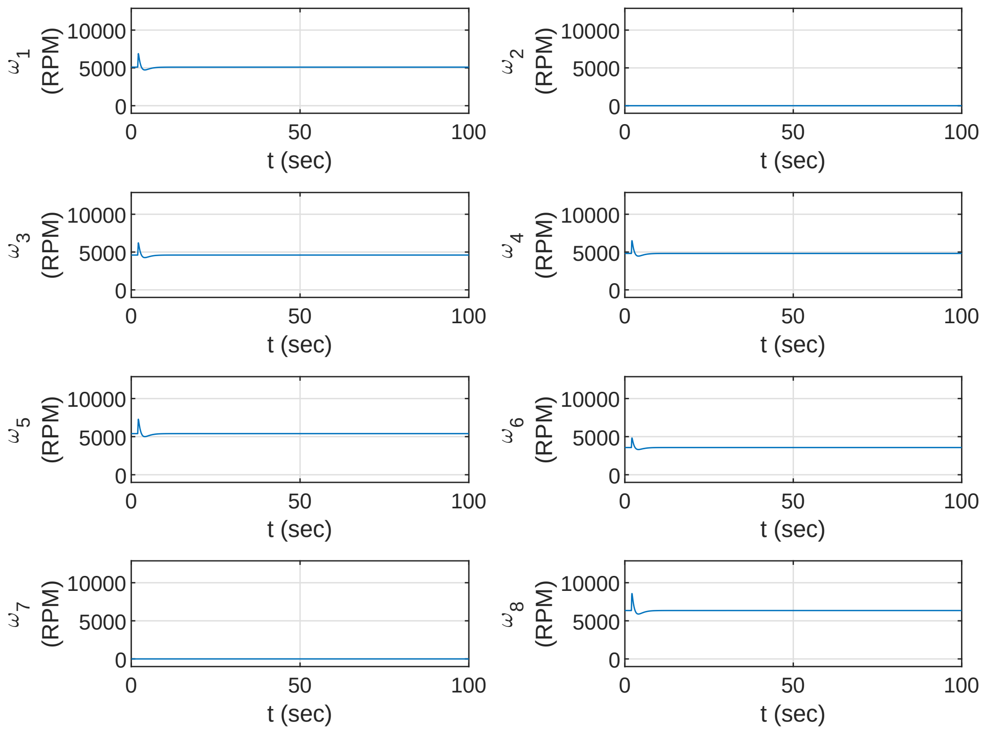


(b) Desired and Simulated Euler angles and Simulated Angular Rates

Figure 8.4: Faulty Scenario, $W_{diag} = [\ 1 \quad 0 \quad 0.7 \quad 0.5 \quad 1 \quad 0.3 \quad 0 \quad 1\ ]$

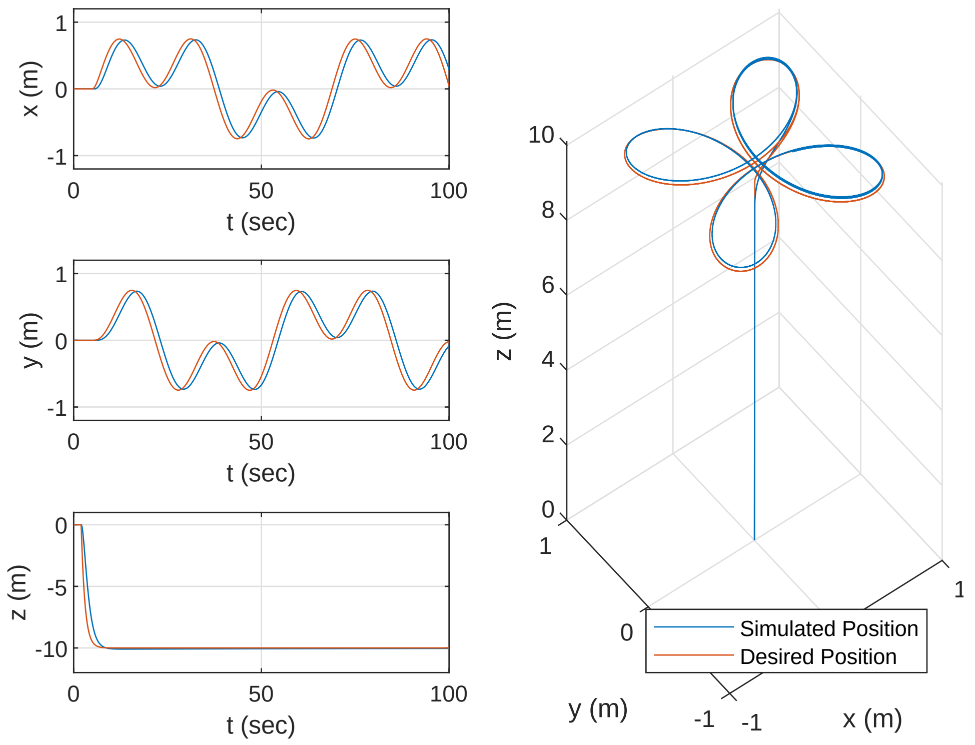


(c) Switching Functions

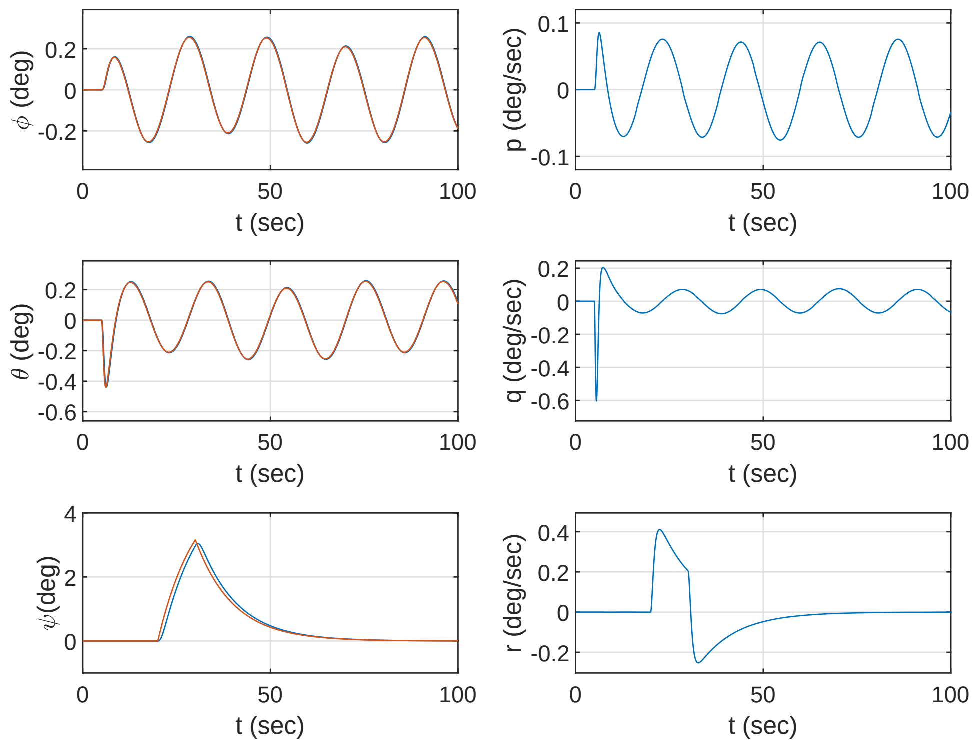


(d) Simulated Motors Angular Velocities

Figure 8.4: Faulty Scenario, $W_{diag} = [\ 1 \quad 0 \quad 0.7 \quad 0.5 \quad 1 \quad 0.3 \quad 0 \quad 1\ ]$

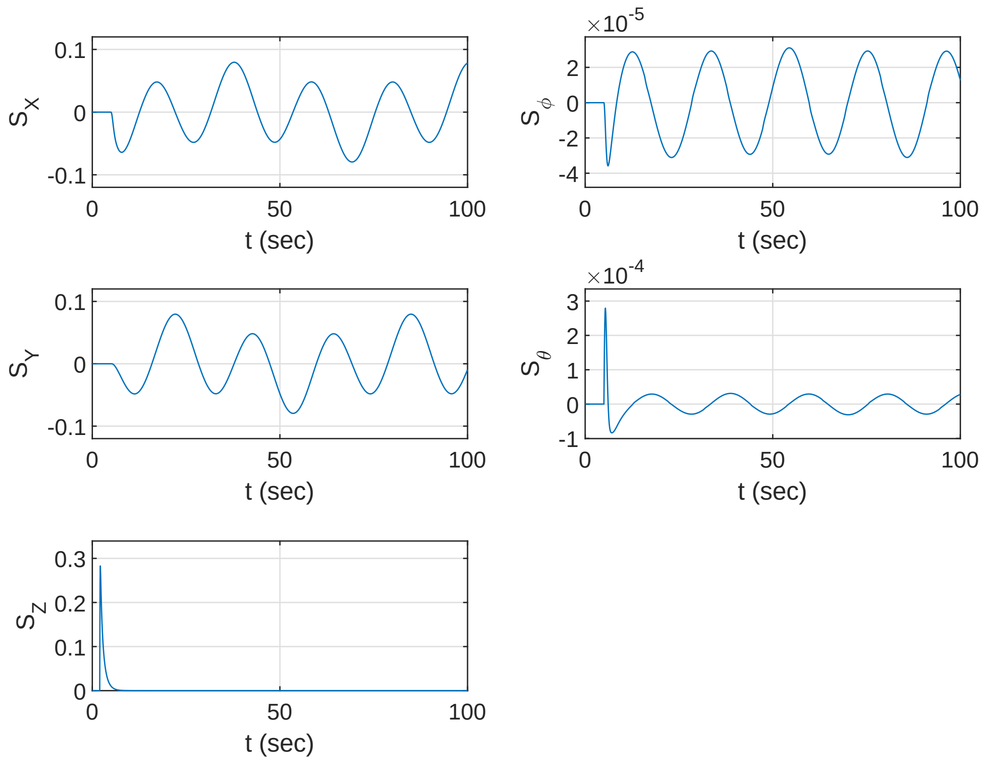


(a) Desired and Simulated Aircraft Position

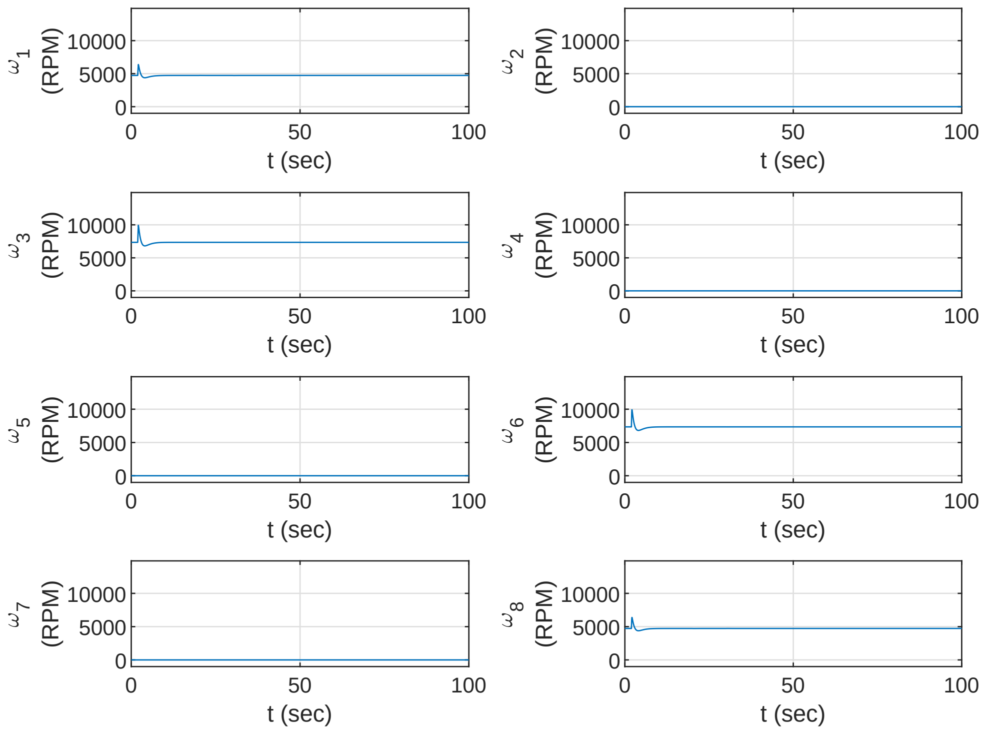


(b) Desired and Simulated Euler angles and Simulated Angular Rates

Figure 8.5: Four Motor (X Configuration Scenario)

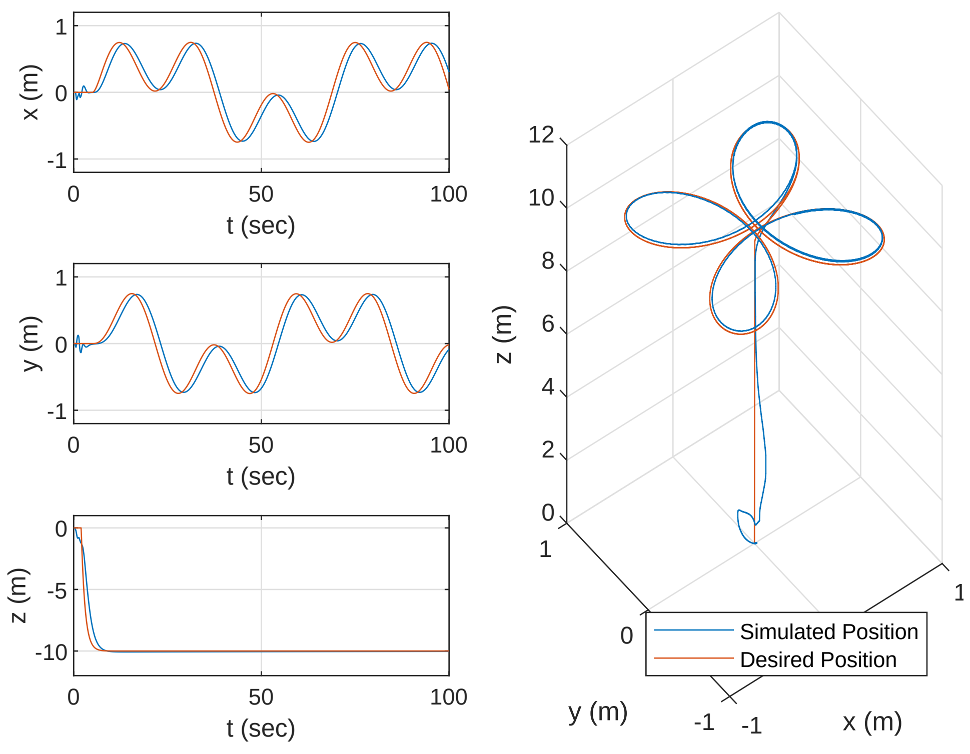


(c) Switching Functions

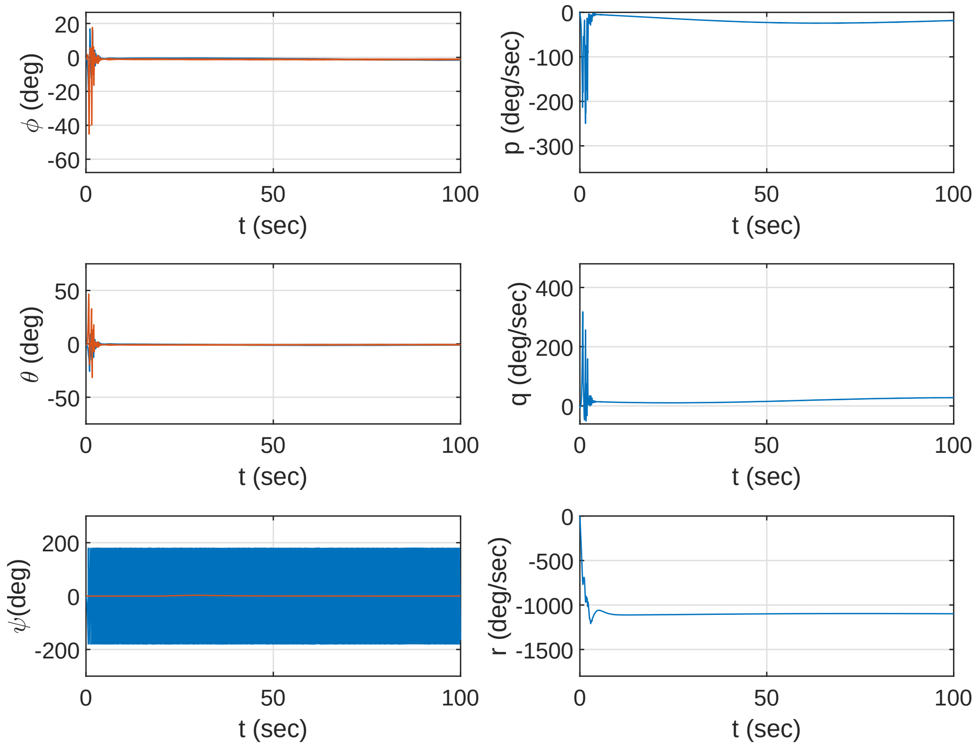


(d) Simulated Motors Angular Velocities

Figure 8.5: Four Motor (X Configuration Scenario)

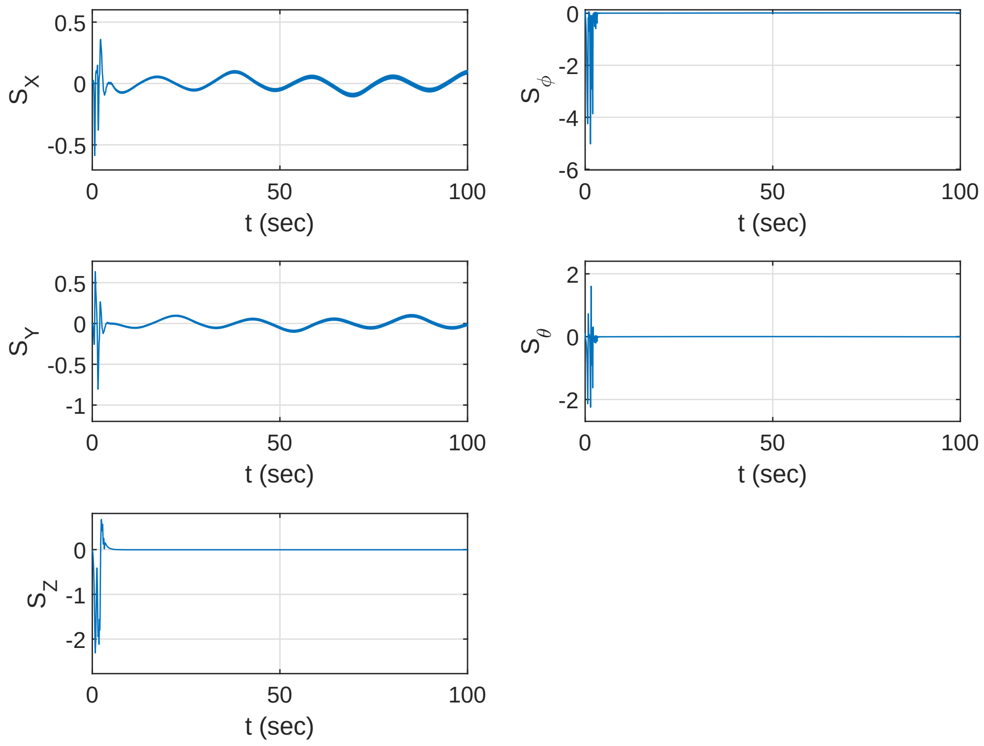


(a) Desired and Simulated Aircraft Position

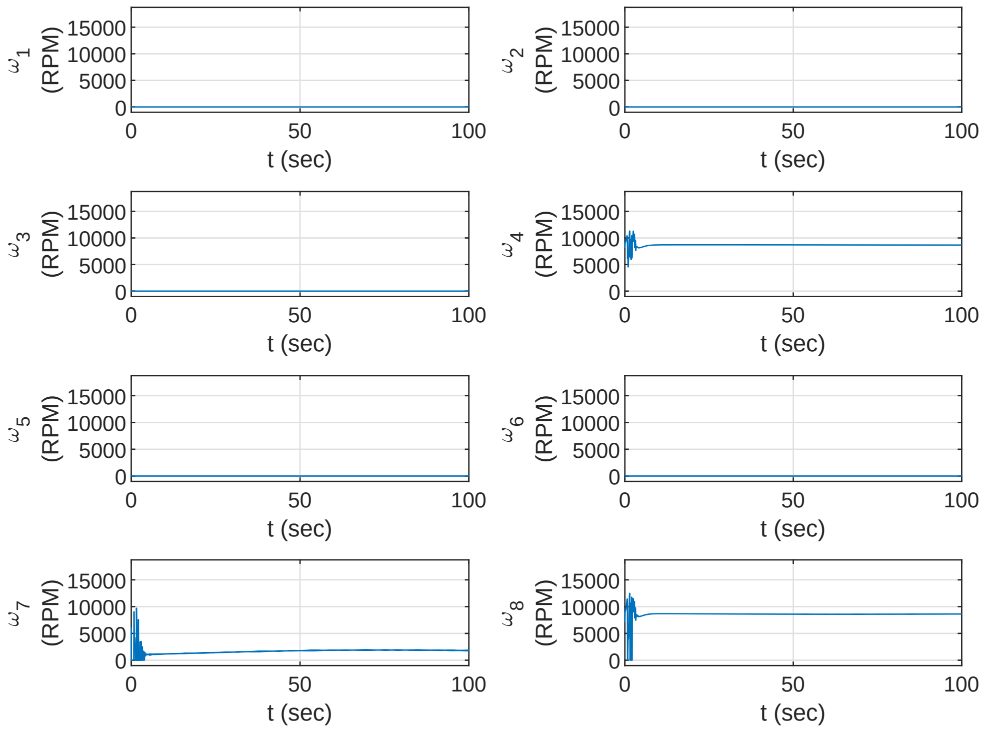


(b) Desired and Simulated Euler angles and Simulated Angular Rates

Figure 8.6: Three motor (Under Actuated) Scenario

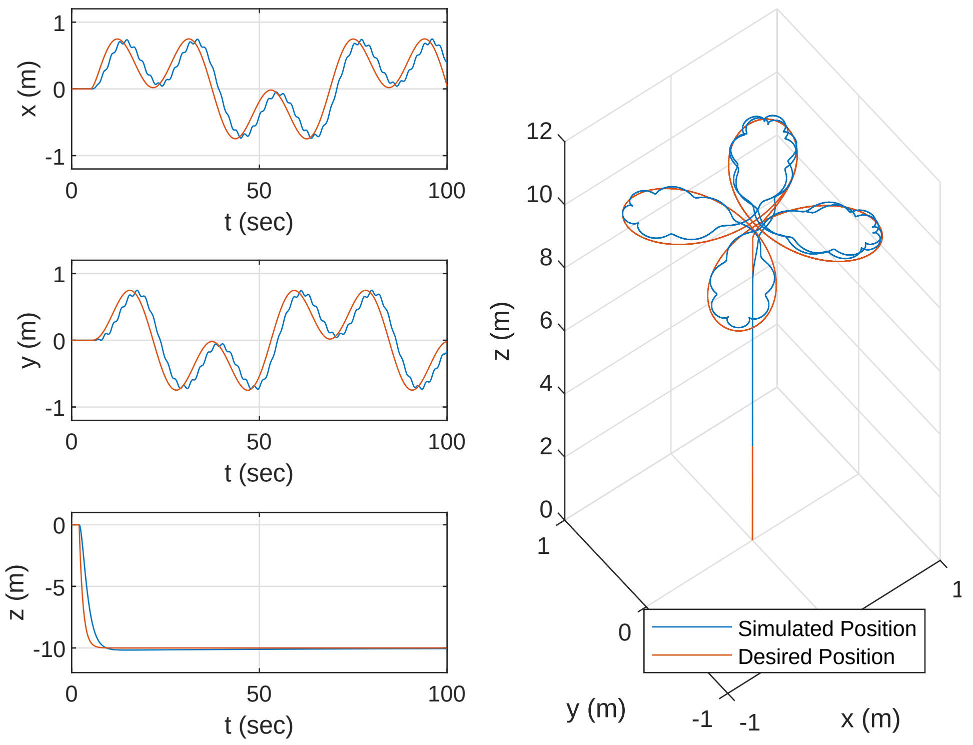


(c) Switching Functions

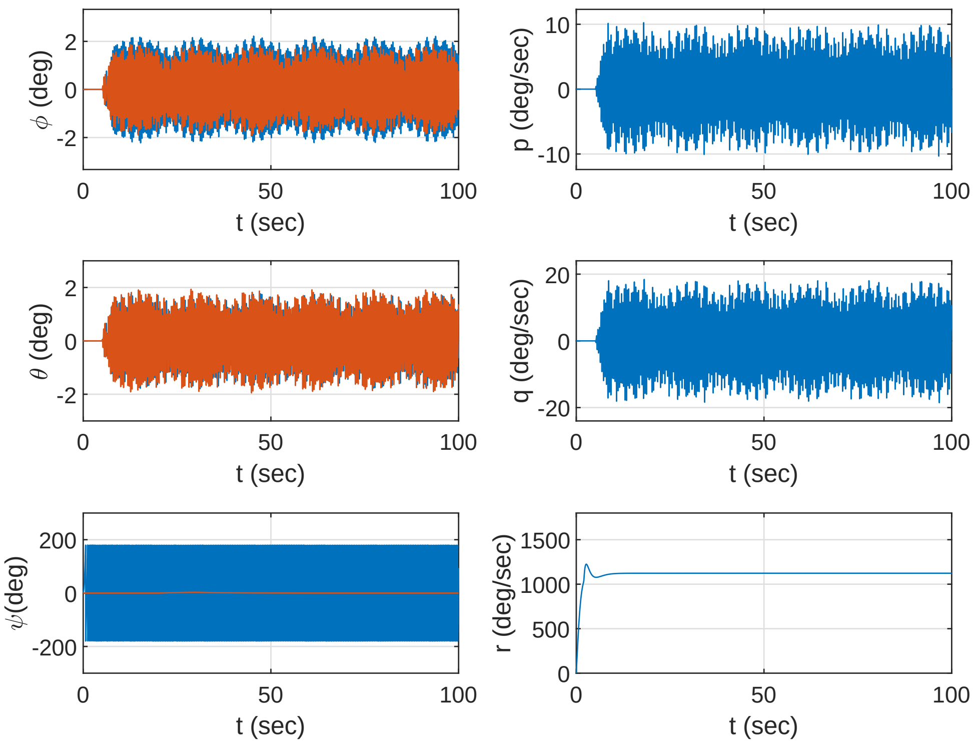


(d) Simulated Motors Angular Velocities

Figure 8.6: Three motor (Under Actuated) Scenario

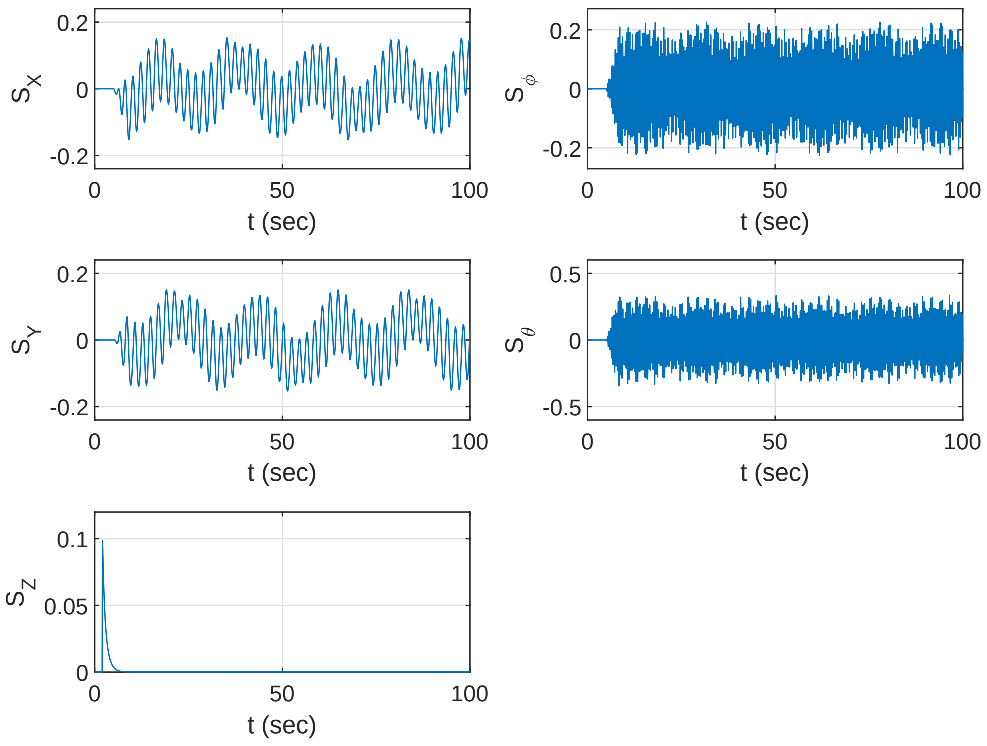


(a) Desired and Simulated Aircraft Position

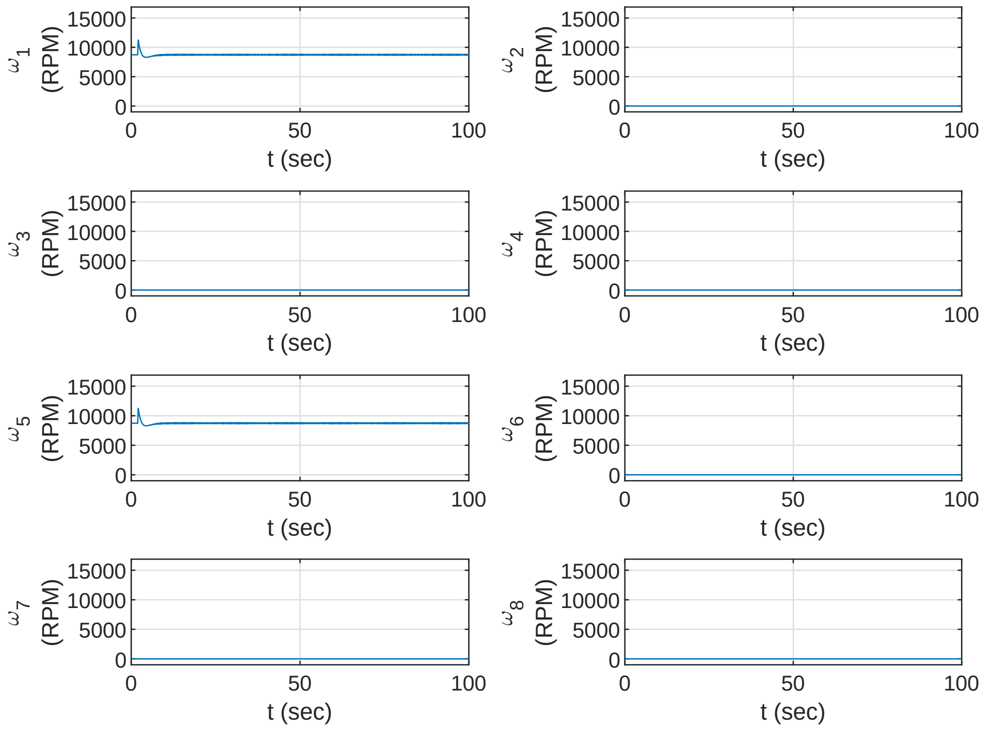


(b) Desired and Simulated Euler angles and Simulated Angular Rates

Figure 8.7: Two motors (under actuated) Scenario



(c) Switching Functions



(d) Simulated Motors Angular Velocities

Figure 8.7: Two motors (under actuated) Scenario

Chapter 9

Conclusions and future work

9.1 Summary

The earlier chapters of this thesis have focussed on the basic terminology and definitions on faults and failures of aircraft and unmanned aerial vehicles systems. Importance of UAVs and its market growth and the subsequent increase of incidents and accidents involving UAVs has been discussed. This motivates the need for the fault-tolerant control systems for UAVs especially in the absence of onboard pilot. The earlier chapters also discussed various FTC techniques used in the literature on both large transport aircraft and UAVs. One of the challenges faced by FTC researcher is the cost and safety issues to implement FTC schemes. The earlier chapters have also argued for the suitability of small unmanned multi-rotor UAVs as a testing platform for testing FTC schemes due to the affordability and relative safe lab environment that the UAVs can be tested in comparison to large transport aircraft.

Chapter 3 has focussed on the main concept of sliding mode control. The (state space) regular form has been presented and the inherent robustness property to a certain class of uncertainties has been discussed. A controller synthesis with tracking capability has also been discussed in this chapter with an illustrating example of a small UAVs.

Chapter 4 described an FTC scheme which can deal with total actuator failure. Here, the combination of sliding mode schemes with control allocation provides an excellent FTC strategy that can deal with actuator faults as well as total failures, without reconfiguring the controller and with the ability to maintain fault-free performance. The synthesis exploits the separation between the design of a baseline controller which produce the 'virtual' control signals and control allocation to redistribute the virtual control signals to all available (redundant) actuators. This results in a simple yet effective FTC scheme to deal with both actuator faults and failures without the need to reconfigure the baseline controller. When actuator faults/failure occurs, the control allocation scheme will automatically redistribute the virtual control signals to the remaining healthy actuators, without any changes to the baseline controller. Two numerical examples from the literature have been considered, to illustrate the capability of the proposed scheme. Simulation results from both a fixed-wing aircraft (ADMIRE) and a Quadrotor (3DR IRIS+) show a good tracking performance despite the presence of various fault and failure scenarios.

Chapter 5 describe one of the major contributions of this thesis. Here, an implementation of a sliding mode fault-tolerant control allocation scheme using Pixhawk (flight controller) has been conducted on the 3DR IRIS+ quadrotor. The use of the Pixhawk through the Simulink PSP and the gimbal test rig provided a good test platform for rapid prototyping and testing of the advanced controller and a good stepping stone in preparation for future flight tests (flying arena or outdoors). The rapid implementation using PSP also save time and efforts in conducting future novel control schemes. This chapter also highlights the low computational load of the proposed scheme when implemented on the real time, limited processing power of the Pixhawk flight controller. The good implementation results further highlight the efficacy of the proposed scheme, in both fault free and faulty conditions.

Chapter 6 described an FTC scheme for a novel multirotor UAVs. In this chapter, a detailed nonlinear mathematical model development of the over actuated spherical UAV with two counter-rotating rotors and eight flaps has been discussed. The proposed scheme exploits the available redundant flaps and rotors to achieve a resilient UAV that is tolerant to faults and failures. This is unlike most of the work on similar spherical UAV in the literature. The proposed FTC scheme is similar to the one proposed in Chapter 5 - the combination of sliding mode ideas and online control allocation. The idea is to exploit the robustness property of sliding mode to design a baseline controller. The control allocation uses the effectiveness levels of the actuators in the event of actuator faults/failures to redistribute the control signals to the remaining healthy actuators. In this chapter, the controller is synthesised based on a linear model (around a hover condition), but all the simulation were conducted using the nonlinear model. The results show good tracking performance in fault free and in the presence of failures to six actuators.

Chapter 7 described one of the major breakthroughs in terms of FTC scheme for a multirotor UAVs without any redundant actuator e.g. quadrotor. Unlike typical schemes in the literature, the scheme proposed here has the capability to handle total rotor failures while able to operate in fault free condition. An LPV based reduced attitude system formulation has been developed rather than the typical Euler angle based system. An LPV based sliding mode control has been proposed to exploit this LPV form. When a failure occurred to one of the motors in a quadrotor, typical control strategy will fail as the system becomes uncontrollable due to lack of actuator. The idea in this chapter is to sacrifice yaw control and allow the aircraft to rotate around a vertical axis (the so-called primary axis). The remaining three motors have been utilised to provides desired unit vectors in the direction of travel, and therefore to allow the vehicle to translate in the 3D space. Good simulation results highlight the efficacy of the proposed scheme.

Chapter 8 provide an alternative FTC framework as compared to Chapter 7. Rather than quadrotor, this chapter considers octocopters and proposed a single resilient FTC controller which able to operate in fault-free, faulty, failure with sufficient redundancy and failure with no redundancy (under actuated). One major breakthrough in this chapter is the synthesis of SMC control for both outer and inner control loops. This is unlike most of the work done on similar octocopter in the literature, where typically only the inner loop is SMC. This was achieved by exploiting the differential flatness property of the system to build the appropriate state-space models for the outer loop navigation/position control. This chapter also considers a nonlinear feedback linearisation to cancel the nonlinearities in the system and allow for a typical linear based design to be utilised. The inner loop control considers the combination of SMC and control allocation to distribute the control efforts to the available actuators. This allows the controller to handle fault free as well as failure cases to up to four motors remaining. When only three motors remain operational (under actuated case), yaw motion was sacrificed to sustain tracking the aircraft translational (3D) position. In the case when only two motors remaining, both yaw and pitch motion was sacrificed where the continuous rotation of the aircraft was sufficient to control the translational position of the aircraft. Simulation results that consider the fault-free, faulty, failure with sufficient redundancy and failure with no redundancy (under actuated) shows the efficacy of the proposed scheme.

9.2 Future work

9.2.1 Implementations on spherical and octorotor UAVs

One of the planned future work is to implement and evaluate different sliding mode control schemes developed in this thesis on the spherical UAVs developed in Chapter 6 and a highly redundant multirotor UAV such as octocopters e.g. the 3DR's X8 [35] (Figure 9.1) or the Helipal's Storm Drone 8 [41] (Figure 9.2). Evaluations on the fixed test rig will provide opportunities to fine tune the controller. Actual flight test can then be considered in an appropriate and approved outdoor test range. These implementation tests (fixed test rig and actual flight tests) will provide actual verification of tracking performance and robustness of various sliding mode control schemes (existing and ones that was developed in this project), rather than just simulations results. The tests will include both nominal

Figure 9.1: 3D Robotics RTF X8+ Multi copter [35]

and fault/failures conditions and will exploit the availability of redundant rotors in the Octorotor.

9.2.2 Catastrophic failure scenario

Apart from the typical faults/failures such as loss of effectiveness (e.g. damage rotor) and total loss of rotor (propeller or motor) considered in this thesis, a more catastrophic scenario such as structural damage or loss of arm can also be explored. This will simulate the change in mass, inertia and centre of gravity, which is believed have not yet been formally analysed in the open FTC literature. The change in mass due to structural damage is also beneficial, as the same controller can also be used to handle varying payload, which is a typical variation in the operation of a UAV.

9.2.3 Novel multirotor designs

In Chapter 6, a novel multirotor UAVs with counter-rotating propellers and eight flaps has been considered. This configuration is different as compared to the typical quadrotor UAVs and much more efficient due to only two motors are required. Other design and configurations of multirotor UAVs can be explored to analyse the FTC capabilities.

9.2.4 Fault detection and isolation (FDI)

The work that has been described in this thesis are have been based on the assumptions that the information about the faults/failures are available or the control is robust enough to handle unknown faults or failures. This information is utilised in the online control allocation to automatically redistribute the control signals to the remaining healthy rotors. This is an avenue of a possible area of research; is to develop, implement and verify sliding mode based FDI schemes in conjunction with the FTC schemes developed in this thesis. Rigorous tests can be conducted to evaluate the combined computational load, especially when implemented on the limited computational power of the Pixhawk flight controller.

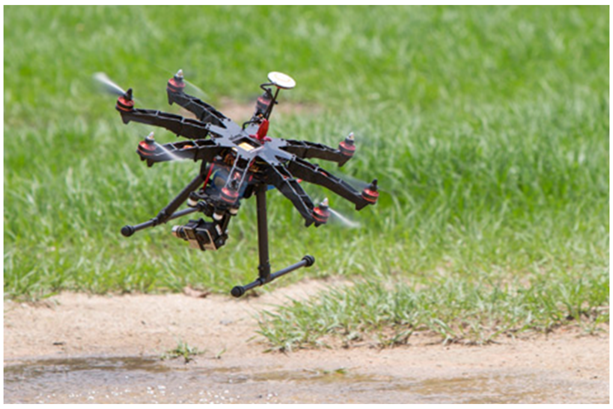


Figure 9.2: Storm8 UAV [41]

Appendix A

Rigid body equation of motion of multirotor UAVs

The nonlinear equation of motion of a general Multirotor UAV (assuming rigid body equation of motion with symmetry around x and y body axes and neglecting propeller inertia) [116] is given by

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ \dot{u} \\ \dot{v} \\ \dot{w} \\ \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} p + qsin(\phi)tan(\theta) + rcos(\phi)tan(\theta) \\ qcos(\phi) - rsin(\phi) \\ qsin(\phi)sec(\theta) + rcos(\phi)sec(\theta) \\ vr - qw - gsin(\theta) \\ pw - ur + gcos(\theta)sin(\phi) \\ uq - pv + gcos(\theta)cos(\phi) \\ c_1qr \\ c_2pr \\ c_3qr \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \frac{F_z}{m_{kg}} \\ c_4\mathcal{L} \\ c_5\mathcal{M} \\ c_6\mathcal{N} \end{bmatrix} \tag{A.1}$$

and

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = R_b^i(\phi, \theta, \psi)^T \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{A.2}$$

where ϕ, θ, ψ are the Euler angles (roll, pitch and yaw angles). The variables u, v, w represent body axis linear velocity and p, q, r are body axis angular velocities while $\dot{x}, \dot{y}, \dot{z}$ are the linear velocities in inertial axes. Total motor thrust forces are represented by F_z while roll, pitch and yaw torques in the body axis are represented by $\mathcal{L}, \mathcal{M}, \mathcal{N}$ respectively. The system parameters are mass m_{kg} and I_{xx}, I_{yy}, I_{zz} which represent inertia about the

x, y, z body axes respectively. The constants $c_1, \ldots, c_6$ are defined as

$$\begin{aligned} c_1 &= (I_{yy} - I_{zz})/I_{xx}, \quad c_2 = (I_{zz} - I_{xx})/I_{yy} \\ c_3 &= (I_{xx} - I_{yy})/I_{zz}, \quad c_4 = 1/Ixx \\ c_5 &= 1/Iyy, \qquad\qquad\quad c_6 = 1/Izz \end{aligned} \tag{A.3}$$

The orthogonal matrix R_b^i defined as

$$R_b^i(\phi,\theta,\psi) = \begin{bmatrix} cos(\theta)cos(\psi) & cos(\theta)cos(\psi) & -sin(\theta) \\ sin(\phi)sin(\theta)cos(\psi) - cos(\phi)sin(\psi) & sin(\phi)sin(\theta)sin(\psi) + cos(\phi)cos(\psi) & sin(\theta)cos(\theta) \\ cos(\phi)sin(\theta)cos(\psi) + sin(\phi)sin(\psi) & cos(\phi)sin(\theta)sin(\psi) - sin(\phi)cos(\psi) & cos(\phi)cos(\theta) \end{bmatrix} \tag{A.4}$$

is the direction cosine matrix (DCM) used to transform from inertial axes to body axes [116].

Appendix B

Octocopter moment of inertia estimation (The Bifilar Experiment)

Based on the Bifilar Pendulum method [130, 42], the moments of inertia around x,y and z axes of the Storm8 octorotor were estimated. In the Bifilar Pendulum experiment, the aircraft is hanged using two wires parallel to the axis that is required to get the inertia around. The used equation relates the period of oscillations T_c with distance between the hanging point and the aircraft center of gravity h as follow

$$I = \frac{m_{kg} g D^2 T^2}{16 h \pi^2} \tag{B.1}$$

where m_{kg} is the aircraft mass, g is the gravitational acceleration and D is the distance between the two wires. Equation B.1 could be re arranged so that

$$h = \frac{m_{kg} g D^2}{16 \pi^2 I} T^2 \tag{B.2}$$

By performing the experiment at different wire lengths h and obtaining the average of the time period of oscillation T_c, the data could be plotted as $h\ vs.\ T^2$. This will get a straight line with gradient $G = \frac{m_{kg} g D^2}{16 \pi^2 I}$ then the inertia around the axis of rotation could be obtained as

$$I = \frac{m_{kg} g D^2}{16 \pi^2 G} \tag{B.3}$$

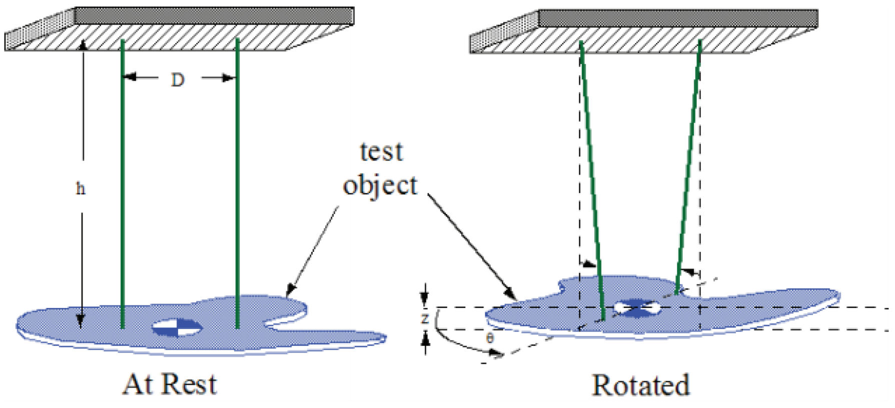


Figure B.1: Bifilar pendulum experiment conguration [42]

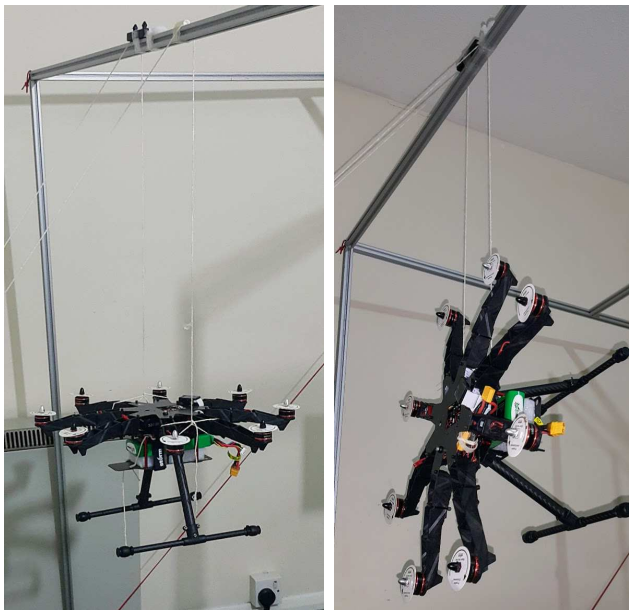


Figure B.2: Inertia estimation experiment: Setup

Table B.1: I_{xx} estimation experiment: Data

Trials	L	Time 10 periods			T_{avg}	T^2_{avg}
		1	2	3		
1	0.99	55.89	55.9	56.1	5.596	31.319
2	0.875	52.73	52.48	52.6	5.260	27.671
3	0.666	45.86	45.86	46.07	4.593	21.096
4	0.5375	40.56	40.24	39.94	4.0247	16.198
5	0.3855	33.51	34.32	34.07	3.397	11.537

Table B.2: I_{yy} estimation experiment: Data

Trials	L	Time 10 periods			T_{avg}	T^2_{avg}
		1	2	3		
1	0.79	0.79	38.77	39.07	38.79	3.8877
2	0.621	0.621	34.04	34.12	34.52	3.4227
3	0.6	0.6	33.18	33.71	34.59	3.3827
4	0.491	0.491	30.33	29.75	29.64	2.9907
5	0.355	0.355	25.42	25.04	25.52	2.5327

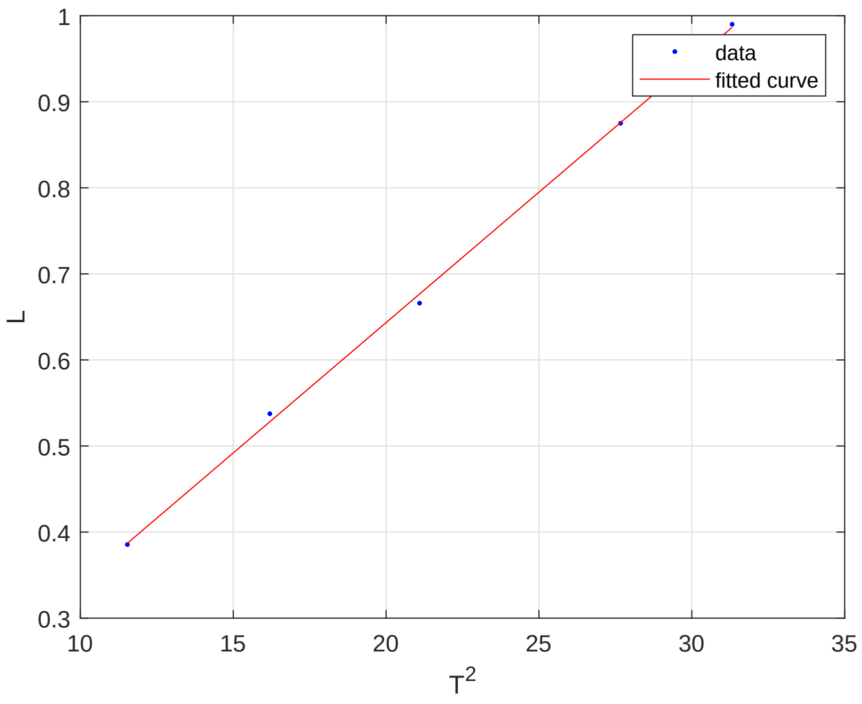


Figure B.3: Ixx estimation experiment: L $vs.$ T^2

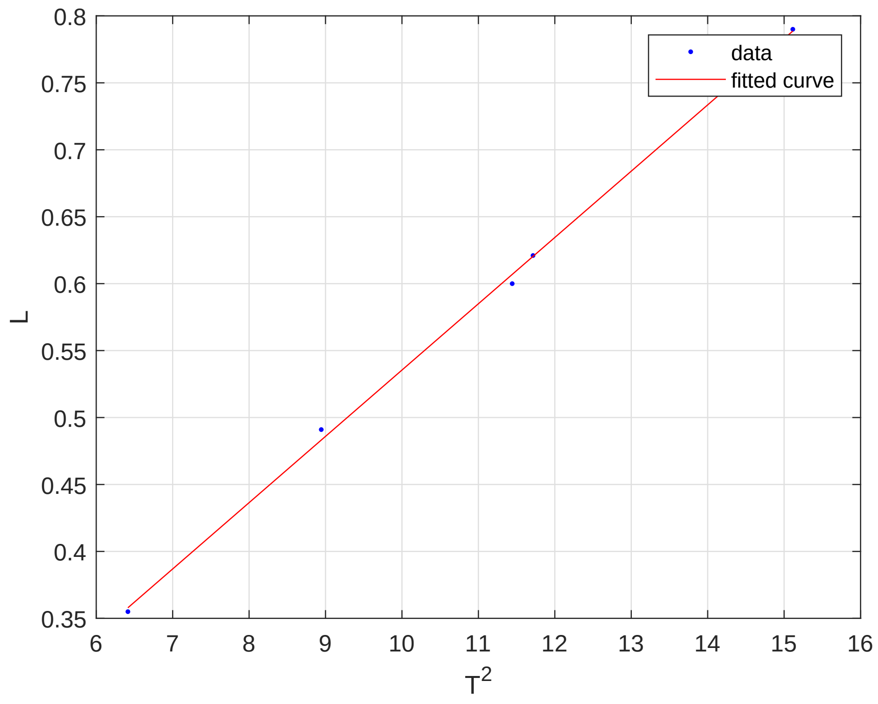


Figure B.4: Iyy estimation experiment: L $vs.$ T^2

Table B.3: I_{zz} estimation experiment : Data

Trials	L	Time 10 periods			T_{avg}	T_{avg}^2
		1	2	3		
1	0.9	0.9	22.9	22.78	23	2.289
2	0.733	0.733	20.56	20.84	20.85	2.075
3	0.47	0.47	16.29	16.33	16.36	1.6327
4	0.35	0.35	14.13	14.24	14.34	1.4237
5	0.6	0.6	18.99	18.73	19.12	1.8947

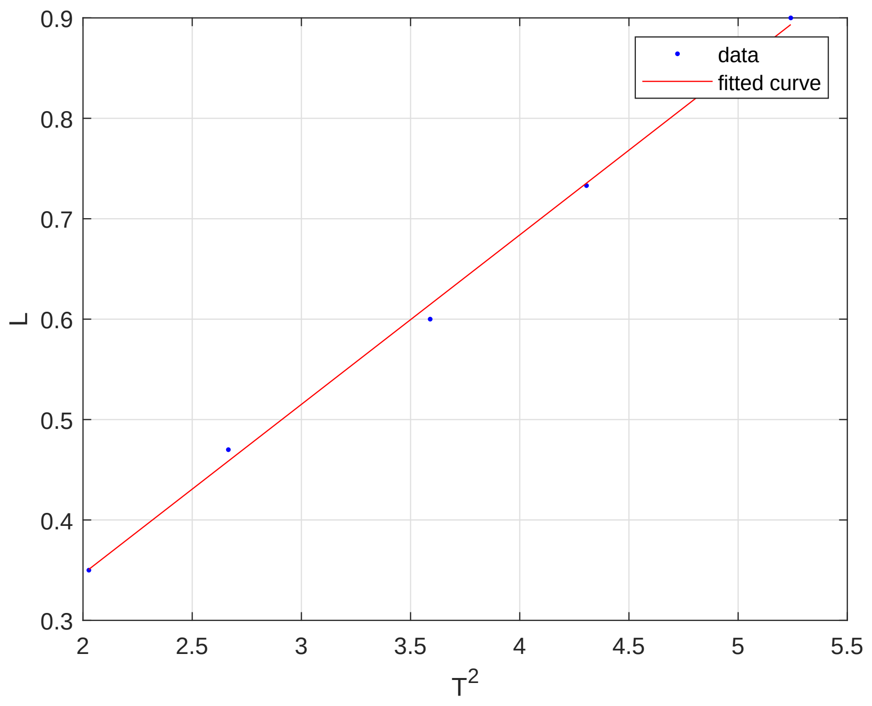


Figure B.5: Izz estimation experiment: L vs. T^2

Table B.4: Storm8 Octorotor physical parameters: Mass and Inertia

Parameter	Value	Unit
m_{kg}	1.214	kg
I_{xx}	0.009565	kgm^2
I_{yy}	0.013746	kgm^2
I_{zz}	0.017866	kgm^2

Finally, The estimated inertia values of the Storm8 Octocopter were summarized as in table B.4.

Appendix C

Matlab codes

C.1 Low super trainer RC model tracking example

Matlab Code for Example in 3.7.2:

```
clear;clc; close all;
A=[-0.0887 0.0011 0  0
-0.0025 -15.15 1 0
0.15 -167.47 -17.63 0
0 0 1 0];
B=[0
0
216.4486
0 ];
Aaa=[0 0 0 1
     0 -0.0887 0.0011 0 %to make B=[0;B2]
  0 -0.0025 -15.15 1
  0 0.15 -167.47 -17.63 ];
Bbb=[0
    0
0
216.4486 ];
A=Aaa;B=Bbb;
C=diag([1,1,1,1]);
D=[0;0;0;0];
ro=1;
Q=diag([4000 1 .1 .1 1  ]);
```

```
Cc=[1  0  0  0];
[Aa,Bb]=intac(Aaa,Bbb,Cc);
[S,E]=lqcf(Aa,Bb,Q);
S=inv(S*Bb)*S;
Phi=diag([-1]);
[L,Lr,Lrdot,Sr,Lam,P]=contlia(A,B,Cc,S,Phi);
 gamma=-1;
```

C.2 ADMIRE fault tolerance example

Matlab Code for Example in 4.3.1:

```
clear;clc; close all;
A=[     -0.5432          0.0137        0              0.9778     0
        0                -0.1179        0.2215       0            -0.9661
        0               -10.5128       -0.9967       0             0.6176
        2.6221          -0.003          0            -0.5057      0
        0                0.7075        -0.0939       0            -0.2127  ];
B=[  0.0069      -0.0866     -0.0866     0.0004
        0             0.0119     -0.0119     0.0287
        0            -4.2423      4.2423     1.4871
        1.6532       -1.2735     -1.2735     0.0024
        0            -0.2805      0.2805    -0.8823  ];
C=diag([1,1,1,1,1]);
D=zeros(5,4);
B1=B(1:2,:);
B2=B(3:5,:);
Ahat=A;
Bhat=[zeros(2,3);eye(3)];
ro=10;
delta=0.05;
Cc=[eye(3),  zeros(3,2)  ];
[Aa,Bb]=intac(Ahat,Bhat,Cc);
Q=diag([2000  2000   2000  1  1  1  1  1  ]);
[S,E]=lqcf(Aa,Bb,Q);
W=diag([1  1  1  1  ]);
Phi=diag([-1  -1  -1]);
```

```
[L,Lr,Lrdot,Sr,Lam,P]=contlia(Ahat,Bhat,Cc,S,Phi );
gamma=2*diag([-1 -1 -1]);
sim('fault_tolerance_11_2017')
```

C.3 IRIS+ 3DR quadcopter example

Matlab Code for Example in 4.3.2:

```
clear all;clc; close all;
A=zeros(8,8);
A(1:4,5:8)=eye(4);
m=1.6;
Ixx=0.024;
Iyy=0.011;
Izz=0.031;
b=2.8*10^-8;
l_1=.23;
l_2=.21;
l_3=.13;
l_4=.13;
bl=2.8*10^-8*0.23;
d=7.5*10^(-7);
Bt=[zeros(4,4);diag([1/m,1/Ixx,1/Iyy,1/Izz ])];
Bw=[   b       b      b      b
      -b*l_1     b*l_2   b*l_1  -b*l_2
       b*l_3   -b*l_4   b*l_3  -b*l_4
       d       d    -d    -d      ];
B=Bt*Bw;
C=eye(8);
D=zeros(8,4);
B1=B(1:4,:);
B2=B(5:8,:);
Ahat=A;
Bhat=[zeros(4,4);eye(4)];
ro=6; %1;
delta=.05;
Cc=[eye(4), zeros(4,4)];
```

```
[Aa,Bb]=intac(Ahat,Bhat,Cc);
Q= diag([100,1000*[ 1 .1 .1], 1,(60/60)* [ 1 1 1],.5, (60/120)*[
    1 1 1]]);
W=diag([1 1 1 1 ]);
[S,E]=lqcf(Aa,Bb,Q);
Fault=[1 1 1 1 ];
Phi=1*diag([-1 -1 -1 -1]);
[L,Lr,Lrdot,Sr,Lam,P]=contlia(Ahat,Bhat,Cc,S,Phi );
gamma=diag([-1 -5 -5 -2]);
```

C.4 IRIS+ 3DR quadcopter implemenation

Matlab Code for Example in 5.4.4:

```
clear all; close all;clc;
A=zeros(6,6);
A(1:3,4:6)=eye(3);
% m=1.482; %with black battery
m=1.506; % with green battery
g=9.807;
Ixx=0.0219 +.017*4*.07^2+m*.07^2;
Iyy=0.0109 +2*.075*.45^2+8*(.075/2)*(.45/2)^2+m*.07^2;
Izz= 0.0306+2*.075*.45^2+4*(.075/2)*(.45/2)^2 ;
A(4,1)=-(m*g*.07/Ixx);
A(5,2)=-(m*g*.07/Iyy);
A(4,4)=0;
A(5,5)=0;
A(6,6)=0;
trans=((60/(2*pi))^2);
b= .78*(1*10^(-7))* trans;
d=(1.8065*10^(-9))* trans;
l_1=.23;
l_2=.21;
l_3=.13;
l_4=.13;
Bt=[zeros(3,3);diag([1/Ixx,1/Iyy,1/Izz ])];
Bw=[ -b*l_1    b*l_2   b*l_1 -b*l_2
```

```
        b*l_3   -b*l_4   b*l_3 -b*l_4
        d          d        -d        -d        ];
B  =  Bt*Bw;
b_2=  b  ;
d_2=  d  ;
m_2    = m   ;
Ixx_2 = Ixx;
Iyy_2 = Iyy;
Izz_2 = Izz;
Bt_2=[zeros(3,3);diag([1/Ixx_2,1/Iyy_2,1/Izz_2 ])];
Bw_2=[ -b_2*l_1    b_2*l_2   b_2*l_1 -b_2*l_2
        b_2*l_3   -b_2*l_4   b_2*l_3 -b_2*l_4
        d_2      d_2   -d_2   -d_2     ];
B_2=Bt_2*Bw_2;
C=eye(6);
D=zeros(6,4);
B1=B(1:3,:);
B2=B(4:6,:);
Ahat=A;
multi=100;
BT=diag([1/Ixx,1/Iyy,1/Izz ]);
Bhat=(multi)*[zeros(3,3);eye(3)];
Btoinv=B2;
Phi1=-3;
Phi2=-1;
Phi3=-1  ;
Phi   =        diag(     [   Phi1  Phi2  Phi3   ]);
ro1=5;
ro2=1;
ro3=1;
ro     =     diag(     [    ro1  ro2  ro3]);
delta1=.12;
delta2=.17;
delta3=.1;
delta=                    [     delta1    delta2    delta3  ];
Qq1=13;Qq2=13;Qq3=13;
```

```
Qq4=2;Qq5=2;Qq6=2;
Qq7=.02;Qq8=.02;Qq9=.17;
Q=    diag([    [ Qq1 Qq2 Qq3],[ Qq4 Qq5 Qq6],[Qq7 Qq8 Qq9]]);
%%
Cc=[eye(3), zeros(3,3)];
[Aa,Bb]=intac(Ahat,Bhat,Cc);
Wdiag= [1  1  1  1  ];
[S,E]=lqcf(Aa,Bb,Q);
Fault=[1  1  1  1  ];
S=inv(S*Bb)*S;
gamma=diag([   -1  -1  -1]);
%% TUNEING VALUES
trim1=1000;
trim2=1000;
trim3=1000;
trim4=1000;
%% SD LOGGING
sampling=.004;
SD_sample=.04;
a=44;
%sim('forsimulation6states.slx')
maxrec=int32(120000);
wn=abs(E(1));
x0=0*[5*pi/180;5*pi/180;0;0;0;0];
```

C.5 Spherical UAV FTC

Matlab Code for Example in 6.4:

```
clear all;clc; close all;
%% add path
addpath 'C:\Users\ha281\OneDrive\My Documents EG1080\SMC m files
   \mfiles5'
addpath 'C:\Users\h\OneDrive\My Documents EG1080\SMC m files\
   mfiles5'
%%
 m=.59*2;
 g=9.81;
 Ixx=3393*10^-6*2;
 Iyy=3918*10^-6*2;
 Izz=2745*10^-6*2;
%% Parameters
ht=0.2*1;
hb=0.2*1;
dt=.2*1;
db=.2*1;
St=300*10^-4*1;
Sb=300*10^-4*1;
Dt=dt;
Db=db;
Cla=2*pi;
Cda=1.28;
rho=1.225;
rprop=.01*12*2.54/2; %propeller radius 12"/2 --> to meter
Aprop=pi*rprop^2;
Kt=3*10^-5;
Km=7.5*10^-7;
%% Trim Condition
omegatrim1= 310.5882; %222 if both working and 444 if only one
    working
omegatrim2= 310.5882;
T1=Kt*omegatrim1^2;
T2=Kt*omegatrim2^2;
V=sqrt((T1+T2)/(2*rho*Aprop));
q=.5*rho*V^2;
%%
a=q*St*Cla;
b=q*Sb*Cla*cos(45*pi/180);
c=q*St*Cda;
d=q*Sb*Cda;
e=q*St*Cda*Dt;
f=q*Sb*Cda*Db*cos(45*pi/180);
%% A matrix
```

```
A          =zeros(9,9);
A(1:3,7:9)=eye(3);
A(4,2)     =-g;
A(5,1)     = g;
%% B matrix
Bu1=zeros(3,10);
Bu2=[0            a/m           0             a/m           -b/m                -b/m                -b/m                -b/m                0             0
     a/m          0             a/m           0             -b/m                b/m                 -b/m                b/m                 0             0
     0            0             0             0             0                   0                   0                   0                   -Kt/m         -Kt/m
     a*ht/Ixx     0             a*ht/Ixx      0             b*hb/Ixx            -b*hb/Ixx           b*hb/Ixx            -b*hb/Ixx           0             0
     0            a*ht/Iyy      0             a*ht/Iyy      b*hb/Iyy            b*hb/Iyy            b*hb/Iyy            b*hb/Iyy            0             0
     a*dt/Izz     -a*dt/Izz     -a*dt/Izz     a*dt/Izz      -b*db/(cos(45*pi/180)*Izz)   b*db/(cos(45*pi/180)*Izz)   b*db/(cos(45*pi/180)*Izz)   -b*db/(cos(45*pi/180)*Izz)   -Km/Izz       Km/Izz];
Bu=[Bu1;Bu2];
 Motorscalling=1e6;
Bu=[Bu(:,1:8) Bu(:,9:10)*Motorscalling]; %% IMPR: scaled the last 2 col for design pusposes
 %% SLIDING MODE CONTROL
 C=eye(9);
 D=zeros(9,10);
Ahat=A;
Bhat=[zeros(5,4);eye(4)];
Cc=[zeros(4,2), eye(4), zeros(4,3)];
[Aa,Bb]=intac(Ahat,Bhat,Cc);
T=eye(13);
T0=eye(9);
BuT=[zeros(4,10);Bu];
Q=diag([ 100 10 10 10     1      1      1   1    1 1   1   1   1]); %
    diag([  1 10 10 100     1      1      1   1    1 1   1   1   1]);
Phi=10*diag([-1 -1 -1 -1 ]); % 10*diag([-.001 -1 -1 -1 ]);
gamma=10*diag([-1 -1 -1 -1 ]);
ro=1*diag([2 1 1 1 ]); %diag([.01 1 1 1 ]); %1;
delta=0.01; %0.01;
[A11,A12,B2,T,Td,Tr,b1td,b2td]=regforCA(Aa,BuT,Bb,T);
[S,E,M]=lqcfCA(Aa,BuT,Bb,Q,T);
Sr=zeros(4,4);
BvTtd=[b1td*b2td' ; b2td*b2td'];
BvTtr=Tr*BvTtd;
Tt=Tr*Td*T;
BvTactual=inv(Tt)*BvTtr;
S=(S*BvTactual)\S;
[L,Lr,Lrdot,Sr,Lam,P,Mca2]=contliaCAphi2(Ahat,Bhat,BvTactual
    (5:13,:),T0,Cc,S,Phi,Sr);% BvTactual(3:6,:),T0??
%% fault and effectiveness level
 Wca=diag([1 1 1 1  1 1 1 1  1 1]);
W=Wca; %control allocation weight
tfail=30;
tsample=0.01;
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%% stability analysis
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% in regfor CA format.
% [A11,A12,B2,T,Td,Tr,b1td,b2td]=regforCA(AT,BuT,BvT,T)
% B2 is equal to b2td.
      b1=b1td;
      b2=b2td; %in this example b2=B2?
    ATtrr=Tt*Aa/(Tt);
%%%%%%%%% transform to [x1; x2]-->[x1; s] %%%%%%%%%
    Ts = [eye(9) zeros(9,4); M eye(4)]; %% its M not MCA2
    ATts = Ts*ATtrr/(Ts);
    a11ts = ATts(1:9,1:9);
    a21ts = ATts(10:13,1:9);
    Eiga11ts=eig(a11ts);
    fprintf('both of the above matrices should be the same')
    fprintf('full CL poles')
    eig(Aa+BvTactual*L)
%%%%%%%%%%%%%%%%%% prepare for loops %%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
checkstab=2; %0:= no check, 1:= loop check, 2:= manual check
if checkstab==1
    stepsize=-0.1;
    endstep=0;
    startstep= 1;
    counter=0;
    MAXnorm=0;
    stabilitytest=0;
    counterfail=0;
tic
    for qq=startstep:stepsize:endstep
        for rr=startstep:stepsize:endstep
            for ii=startstep:stepsize:endstep
                for jj=startstep:stepsize:endstep
                    for kk=startstep:stepsize:endstep
                        for ll=startstep:stepsize:endstep
                            for mm=startstep:stepsize:endstep
                                for nn=startstep:stepsize:
                                    endstep
                                    for oo=startstep:stepsize:
                                        endstep
                                        for pp=startstep:
                                            stepsize:endstep
```

```
Wtest=diag([ii jj kk ll mm nn oo pp qq rr]);
Wf=diag([ii jj kk ll mm nn oo pp]);
W_1=diag([qq rr]);
ww=diag(Wtest);
tt=diag(W_1);
wff=diag(Wf);
[ss,~]=size(find(ww==0));
[vv,~]=size(find(tt==0));
[bbv,~]=size(find(wff==0));
b2W2b2t=b2*(Wtest^2)*b2';
if ss<=6 && vv<2
    if bbv>4 && vv==0
        counter=counter+1;
        pinvvb2 = (Wtest^2)*b2'*inv(b2W2b2t);
        NORMtest4 = norm(pinvvb2,2);
        if NORMtest4>MAXnorm
            MAXnorm = NORMtest4
```

```
                                                                              ;
                                                                          Wtest1=
                                                                              ww';
                                                                     end
                                                                 else
                                                                     counterfail=
                                                                        counterfail
                                                                        +1;
                                                             end
                                                         end

                                                     end
                                                 end
                                             end
                                         end
                                     end
                                 end
                             end
                         end
                     end
                 end
toc
gamma0=MAXnorm;
elseif checkstab==2
    %%% manual test
    % problem combonations
    %Wtest=diag([0 0 1 1  1 0 0 0  0 1]) % stabilitytest1=1.4445
    %Wtest=diag([1 1 0 0  1 0 0 0  0 1]) % stabilitytest1=1.4445
    %Wtest=diag([1 1 0 0  1 0 0 0  1 0]) % stabilitytest1=1.4445
    %Wtest=diag([1 0 0 0  1 1 0 0  0 1])
    Wtest=W;
    b2W2b2t=b2*(Wtest^2)*b2';
    detinv=det(b2W2b2t); %must be nonzero
    pinvvb2 = (Wtest^2)*b2'*inv(b2W2b2t);
    gamma0=norm(pinvvb2,2);

```

```
else
      gamma0=0;
end
gamma1 = norm( M*b1*(eye(10)-(b2'*b2))  );
btest = -b1*( eye(10)-(b2'*b2)  );
Ghat1 = ltisys(a11ts, btest, a21ts  );%%%%%
[gain,freq]=norminf(Ghat1);
gamma2 = gain;
stabilitytest1 = gamma0*gamma2 / ( 1-gamma1*gamma0  );
```

C.6 Parrot UAV FTC

Matlab Code for Example in 7.4:

```
clear;clc;
%% inputs
m=.068;  % with green battery
g=9.807;
Ixx= .0686*10^-3;Iyy= .092*10^-3;Izz= .1366*10^-3;Jr=90e-6; %6e
    -5 %propeller inertia (Kgm2)
L=0.044; %Stormnn Arm Length
b= 6.5328*10^-4;d= 1.6*10^-6; %% calculated torque=Drag*Diam
bl=b*L;
Ts=.005;
%%
Wdiag=[1 1 1 0 ];
W=diag(Wdiag);
%% 1motor lost
signn=-1;
Bomega=[   bl   -bl   -bl     bl
      bl     bl   -bl   -bl    ];
Bww=[  signn*b  signn*b  signn*b  signn*b
      bl         -bl         -bl          bl
      bl           bl         -bl         -bl
      -d             d          -d            d              ]; % for nonlinear
          simulation
ntrim=.1;
```

```
r_trimc=10*sum( Wdiag.*Bww(4,:)/d);   %(-Wdiag(1)+Wdiag(2)-Wdiag
    (3)+Wdiag(4));%18.89;
nxbar=ntrim*sum( Wdiag.*Bww(3,:)/bl); %*( Wdiag(1)+Wdiag(2)-
    Wdiag(3)-Wdiag(4));
nybar=ntrim*sum(-Wdiag.*Bww(2,:)/bl); %*( Wdiag(2)+Wdiag(3)-
    Wdiag(1)-Wdiag(4));
nzbar=-sqrt(1-nxbar^2-nybar^2);
r=0;
ravg=r;
rtrim=0;
ptrim=0;
qtrim=0;%5.69;
c2=(Iyy-Izz)/Ixx;
c5=(Izz-Ixx)/Iyy;
c4=1/Ixx;
c7=1/Iyy;
A0lpv=[     0       r         0         -nzbar
     -r       0       nzbar          0
     0         0              0          c2*r
     0         0         c5*r         0         ];

A1lpv=[   0    1    0    0
     -1    0    0    0
     0    0    0    c2
     0    0    c5   0    ];

A2lpv=[  0       0       0      -1
     0       0       1       0
     0       0       0       0
     0       0       0       0    ];
Btau0lpv=[  0    0
     0    0
     c4   0
     0    c7   ];

Btau1lpv=zeros(4,2);
```

```
Btau2lpv=zeros(4,2);
scale_smc=1;


Btau2=diag([c4  c7  ]);
Btau=[zeros(2,2);  Btau2];
Btoinv_yaw=Bww(4,:);
B=Btau*Bomega*scale_smc;
B02lpv=Bomega*scale_smc;
B12lpv=Btau1lpv(3:4,:)*Bomega*scale_smc;
B22lpv=Btau2lpv(3:4,:)*Bomega*scale_smc;

%% C and D matrices
C=eye(4);
D=zeros(4,4);
Cc=[eye(2),  zeros(2,2)];
%% Ahat and Bhat
B1=B(1:2,:);
B2=B(3:4,:);
multi=1;
Bhat=multi*Btau;
Btoinv=Bomega;
%%  SLIDING MODE CONTROL
%% matching codes
Bv=Bhat;
[pc,~]=size(Cc);
[nc,mc]=size(Bv);
Aa0=[zeros(pc,pc) -Cc   ;zeros(nc,pc)  A0lpv];
Aa1=[zeros(pc,pc) -Cc*0;zeros(nc,pc)  A1lpv];
Aa2=[zeros(pc,pc) -Cc*0;zeros(nc,pc)  A2lpv];
Bva=[zeros(pc,mc);Bv];
nn =4;  %states x
mm =2;  %input u
qq =1;  %disturbance w?
pph=1 ; %output of zinf?
pp =1;  %output y (z2)?
```

```
b1lpv=[0 0 0 0]';
c1lpv=zeros(pph,nn);
c2lpv=zeros(pp,nn);
d11lpv=zeros(qq,qq);
d12lpv=zeros(qq,mm);
d21lpv=zeros(pp,qq);
d22lpv=zeros(pp,mm);
clpv=[c1lpv;c2lpv];
dlpv=[d11lpv d12lpv ; d21lpv d22lpv]; %zeros(8,5)
elpv=eye(nn);
s0lpv=ltisys(Aa0(1:4,1:4),[b1lpv Aa0(1:4,5:6)],clpv ,dlpv ,
    elpv ); % using system A11+A12M
s1lpv=ltisys(Aa1(1:4,1:4),[b1lpv Aa1(1:4,5:6)],clpv*0,dlpv*0,
    elpv*0);
s2lpv=ltisys(Aa2(1:4,1:4),[b1lpv Aa2(1:4,5:6)],clpv*0,dlpv*0,
    elpv*0);
pv=pvec('box',[-40 40; 0 0.1;]);
pds=psys(pv,[s0lpv s1lpv s2lpv ]);
psinfo(pds)
pvinfo(pv)
% disp('c -1 pi/2.1 q')
disp('h r -6 q')
region=lmireg;
tol=1e-2;
[GOPT,H2OPT,Klpv,Pcl,Xlyap] = msfsyn(pds,[pp mm],[0 0 0 0],
    region,tol);
Malpv=-Klpv; %since u=klpv x
Elpv=eig( (Aa0(1:4,1:4)+Aa1(1:4,1:4)*r +Aa2(1:4,1:4)*nzbar )-((
    Aa0(1:4,5:6)+Aa1(1:4,5:6)*r +Aa2(1:4,5:6)*nzbar )*Malpv) ); %
    u=-M x
Sa=[Malpv eye(2)];
Sa=inv(Sa*Bva)*Sa;
Sa*Bva; %check muct be I
Phi1=0;
Phi2= 0;
Phi_smc=diag([Phi1 Phi2]);
```

```
Br=[eye(2);  zeros(2,2)];
Snew=Sa;
S_smc=Sa;
Lam_smc=Sa*Bva;
P_smc=eye(2);
dunv_smc=0.1;
Sr_smc=-Sa(1:2,3:4);
L_smc=-inv(Lam_smc)*(Sa*Aa0-Phi_smc*Sa);
L1_smc=-inv(Lam_smc)*(Sa*Aa1);
L2_smc=-inv(Lam_smc)*(Sa*Aa2);
Lr_smc=-inv(Lam_smc)*(Phi_smc*Sr_smc+Snew(:,1:2));
Lrdot_smc=inv(Lam_smc)*Sr_smc;
L=L_smc;
Lr=Lr_smc;
Lrdot=Lrdot_smc;
Sr=Sr_smc;
S=S_smc;
P=P_smc;
Phi=Phi_smc;
Lam=Lam_smc;
rho_smc     = [.1  .1];
ro1=rho_smc(1);
ro2=rho_smc(2);
delta= [0.001  .001];
delta1=delta(1);
delta2=delta(2);
dunv_smc=0.1;
zeta=[.7  .7  .7     ]  ; %r=2
wn=[1.5  1.5  4  ]  ;
zeta1=zeta(1);zeta2=zeta(2);zeta3=zeta(3);
wn1=wn(1);wn2=wn(2);wn3=wn(3);
ao=2;
h = tf(ao,[1  ao]);
disc_delay = c2d(h,Ts);
%% SD LOGGING
tt=30;
```

```
awe=10;
```

C.7 STORM8 UAV FTC

Matlab Code for Example in 8.6:

```
clear all; close all;clc;
%% PHYSICAL PARAMETERS
% m= 1.482; %with black battery
m  = 1.214;  % with green battery
g  = 9.807;
Ixx= 0.013746026;
Iyy= 0.013746026;
Izz= 0.017865809;

%% Motor Parameters
trans=1;
b = 0.78*(1*10^(-7))* trans ;
d =(1.8065*10^(-9)) * trans ; %% calculated torque=Drag*Diam
% b  = 6.5328*10^-4 ;
% d  = 1.6*10^-5;
L=0.21; %Storm8 Arm Length
L1=L*sind(22.5);
L2=L*cosd(22.5);
bl1=b*L1;
bl2=b*L2;
E1=0.7428472907901847;
E2=0.6732746071678157;
%% A matrix
A=zeros(8,8);
A(1:4,5:8)=eye(4);
A(4,4)=0;
A(5,5)=0;
A(6,6)=0;
A=[ 0 0 1 0
    0 0 0 1
    0 0 0 0
```

```
       0  0  0  0];
%% B matrix
sign=-1;
Bt=[zeros(2,2);diag([1/Ixx,1/Iyy])];
Bw_FLMN=[          sign*b  sign*b   sign*b   sign*b   sign*b   sign*b
    sign*b   sign*b
       -bl1    bl1    bl2    bl2     bl1    -bl1    -bl2    -bl2
        bl2    bl2    bl1  -bl1    -bl2    -bl2    -bl1     bl1
        d     -d       d    -d        d      -d        d      -d    ];
Bw_F    = Bw_FLMN  (1,:)    ;
Bw_LM   = Bw_FLMN  (2:3,:);
Bw_FLM  = Bw_FLMN  (1:3,:);
Bw_N    =  Bw_FLMN(4,:)    ;
B   =   Bt*Bw_LM;
%% C and D matrices
C=eye(4);
D=zeros(4,2);
%% Ahat and Bhat
B1=B(1:2,:);
B2=B(3:4,:);
Ahat=A;
multi=1;
BT=diag([  1/Ixx,1/Iyy]);
Bhat=(multi)*[zeros(2,2);eye(2)];%%% should add W?
Btoinv=B2;
%%  SLIDING MODE CONTROL      Parameters
Phi_in      = -1;        % -20;
Phi_out_1  =-.1;        %-1  ;
Phi_out_2  =-1;
ro          = 3.1;      %100;
delta       = .2;        %0.01;
ro_out      =[.2  20]; %[.1  .1];
delta_out  =[.1  .1];
Qq1         = .8;      %0.6;
Qq2         = 20;      %40;
Qq3         = 1;
```

```
Qq1_out     = 0.0001;
Qq2_out     = .5;      %1;
Qq3_out     = 1;
%%
Q=    diag([    [      Qq1  Qq1    ],[     Qq2  Qq2    ],[  Qq3  Qq3    ]]);
Phi   =         diag(     [   Phi_in  Phi_in    ]);
Phi_out   =         diag(     [   Phi_out_1  Phi_out_1  Phi_out_2    ]);

%%
Cc=[eye(2),  zeros(2,2)];
[Aa,Bb]=intac(Ahat,Bhat,Cc);
[S,E]=lqcf(Aa,Bb,Q);
Fault=[1  1  1  1  1  1  1  1];
Wdiag=Fault;
S=inv(S*Bb)*S;
[L,Lr,Lrdot,Sr,Lam,P]=contlia(Ahat,Bhat,Cc,S,Phi  );
gamma=diag([       -1  -1    ]);
%%
Aout=[zeros(3,3),eye(3)
        zeros(3,6)      ];
Bout=[zeros(3,3)
        eye(3)      ];
C_out=[  eye(3),zeros(3,3)];
D_out=zeros(3,3);
B1_out=Bout(1:3,:);
B2_out=Bout(4:6,:);
Ahat_out=Aout;
Bhat_out=Bout;
Btoinv_out=B2_out;
Q_out=    diag([    [      Qq1_out  Qq1_out  Qq1_out    ],[     Qq2_out
    Qq2_out  Qq2_out    ],[  Qq3_out  Qq3_out  Qq3_out    ]]);
Cc_out=[eye(3),  zeros(3,3)];
[Aa_out,Bb_out]=intac(Ahat_out,Bhat_out,Cc_out);
[S_out,E_out]=lqcf(Aa_out,Bb_out,Q_out);
Fault=[1  1  1  ];
Wdiag=Fault;
```

```
S_out=inv(S_out*Bb_out)*S_out;
[L_out,Lr_out,Lrdot_out,Sr_out,Lam_out,P_out]=contlia(Ahat_out,
    Bhat_out,Cc_out,S_out,Phi_out  );
gamma_out=diag([       -1  -1    -1  ]);
```

Appendix D

List of publications

[1] A. Khattab, H. Alwi, and C. Edwards, "Fault tolerant control of a spherical UAV," in *2019 4th Conference on Control and Fault Tolerant Systems (SysTol)*. IEEE, 2019, pp. 92–97.

[2] ——, "Mitigating total rotor failure in quadrotor using LPV based sliding mode control scheme," in *2019 4th Conference on Control and Fault Tolerant Systems (SysTol)*. IEEE, 2019, pp. 98–103.

[3] ——, "Implementation of sliding mode fault tolerant control on the IRIS+ quadrotor," in *2018 IEEE Conference on Control Technology and Applications (CCTA)*. IEEE, 2018, pp. 1724–1729.

[4] C. Edwards, L. Chen, A. Khattab, H. Alwi, and M. Sato, "Flight evaluations of sliding mode fault tolerant controllers," in *2018 15th International Workshop on Variable Structure Systems (VSS)*. IEEE, 2018, pp. 180–185.

Appendix E

Author's biography

Ahmed Khattab

Education

Master of Science in Aeronautical and Aerospace Engineering	**2015**
Cairo University	*Cairo, Egypt*
Bachelor of Science in Aeronautical and Aerospace Engineering	**2011**
Cairo University	*Cairo, Egypt*

Teaching experience

Teaching Assistant	**2011-2016**
Aeronautical and Aerospace Engineering , Cairo University	*Cairo, Egypt*
Teaching Assistant	**2017-2019**
College of Engineering , University of Exeter	*Exeter, UK*

Bibliography

[1] Makezine. Watch two quadcopters broadcast live tv footage - inside an active volcano, make. http://makezine.com/2015/02/03/watch-two-quadcopters-broadcast-live-tv-footage-inside-an-active-volcano/, Nov 2015. [Online; accessed 16-Apr-2020].

[2] AUVSI. Commercial UAS Exemptions, Interactive Report. http://www.auvsi.org/advocacy/exemptions70, August 2016. [Online; accessed 8-Apr-2020].

[3] Lcady. Hovering over the drone patent landscape. https://www.ificlaims.com/news/view/blog-posts/hovering-over-the-drone.htm, November 2014. [Online; accessed 8-Apr-2020].

[4] Kelly Shores. Multirotors prevail in agriculture. http://dronereview.com/2016/09/15/multirotors-prevail-in-agriculture/, Sep 2016. [Online; accessed 8-Apr-2020].

[5] Amazon. Amazon Prime Air. https://www.amazon.com/Amazon-Prime-Air/b?node=8037720011, December 2016. [Online; accessed 8-Apr-2020].

[6] Dieter Moormann. DHL parcelcopter research flight campaign 2014 for emergency delivery of medication. In *Proceedings of the ICAO RPAS Symposium*, 2015.

[7] Sean Farrell. Facebook's solar-powered internet plane takes flight. https://www.theguardian.com/business/2016/jul/21/facebook-solar-powered-internet-plane-test-flight-aquila, Jul 2016. [Online; accessed 8-Apr-2020].

[8] Mark Prigg. The ambulance drone that could save your life, flying defibrillator can reach speeds of 60mph. http://www.dailymail.co.uk/sciencetech/article-2811851/The-ambulance-drone-save-life-Flying-defibrillator-reach-speeds-60mph.html, Oct 2014. [Online; accessed 8-Apr-2020].

[9] Matthew Williams. The Evolution of UAVs: The Ambulance Drone. https://herox.com/news/189-the-evolution-of-uavs-the-ambulance-drone, April 2015. [Online; accessed 8-Apr-2020].

[10] Leo Kelion. Drone-based blood deliveries in Tanzania to be funded by UK. http://www.bbc.co.uk/news/technology-38450664, Dec 2016. [Online; accessed 8-Apr-2020].

[11] Matt Burgess. Unicef is creating a drone corridor in Malawi to deliver HIV treatments. http://www.wired.co.uk/article/unicef-drone-corridor-malawi-africa, Dec 2016. [Online; accessed 8-Apr-2020].

[12] Ellie Zolfagharifard. The megadrone big enough to carry a passenger. http://www.dailymail.co.uk/sciencetech/article-3387542/The-MEGADRONE-big-carry-passenger-Chinese-firm-says-self-flying-craft-used-smart-taxi.html, Jan 2016. [Online; accessed 8-Apr-2020].

[13] Schiebel. Schiebel Camcopter UAV Rescue Refugees Again. http://www.helis.com/database/news/s-100_refugees2/, May 2015. [Online; accessed 8-Apr-2020].

[14] BBC. Exeter fire: Drone footage shows smouldering hotel. http://www.bbc.co.uk/news/uk-37811958, December 2017. [Online; accessed 5-Apr-2020].

[15] Intel PressRoom. Intel and Airbus Demo Drone Visual Inspection Of Passenger Airliner. https://newsroom.intel.com/chip-shots/intel-airbus-demo-drone-inspection-of-passenger-airliners/, Jul 2016. [Online; accessed8-Apr-2020].

[16] Matt Lavin. Aerial Photo UAV Industrial Inspection. http://aerography.net, January 2016. [Online; accessed 5-Apr-2020].

[17] Rudi Krcma. Pipeline inspections with thermal diagnostics. http://www.workswell.eu, December 2019. [Online; accessed 5-Apr-2020].

[18] Carlos Hazbun. Drones in real estate marketing. http://www.ecamsecure.com/drones-real-estate-marketing/, Dec 2016. [Online; accessed 8-Apr-2020].

[19] Dave Burke. Drone crashes into a packed Boeing 737 passenger jet tearing holes in the plane nose as it comes into land at an airport in Mozambique. http://www.dailymail.co.uk/news/article-4097180/Drone-crashes-packed-Boeing-737-passenger-jet-tearing-holes-plane-s-nose-comes-land-airport-Mozambique.html, Jan 2017. [Online; accessed 5-Apr-2020].

[20] BBC. Toddler eyeball sliced in half by drone propeller. http://www.bbc.co.uk/news/uk-england-hereford-worcester-34936739, Nov 2015. [Online; accessed 8-Apr-2020].

[21] Ben Grubb. River of blood' after drone 'hits' australian athlete. https://www.smh.com.au/technology/river-of-blood-after-drone-hits-australian-athlete-20140407-zqruh.html, April 2014. [Online; accessed 8-Apr-2020].

[22] BBC. Skier Marcel Hirscher almost hit by camera drone in World Cup slalom, BBC Sport. http://www.bbc.co.uk/sport/winter-sports/35164700, December 2015. [Online; accessed 8-Apr-2020].

[23] Graham Warwick. Google owned Titan Solara 50 is being developed to stay aloft for months. http://aviationweek.com/technology/facebook-s-uav-flies-builds-developments-solar-power, March 2015. [Online; accessed 15-Apr-2020].

[24] Stephan Wilkinson. The 10 greatest emergency landings. *Aviation History Magazine*, 2016.

[25] H. Alwi, C. Edwards, and C.P. Tan. *Fault Detection and Fault-Tolerant Control Using Sliding Modes.* Advances in Industrial Control. Springer London, 2011.

[26] Ron J Patton. Fault-tolerant control. *Encyclopedia of systems and control*, pages 422–428, 2015.

[27] T. Tucker. *Touchdown: the development of propulsion controlled aircraft at NASA Dryden.* Monographs in Aerospace History, Number 6, 1999.

[28] John Bosworth and Peggy Williams-Hayes. Flight test results from the NF-15B intelligent flight control system (IFCS) project with adaptation to a simulated stabilator failure. In *AIAA Infotech@ Aerospace 2007 Conference and Exhibit*, page 2818, 2007.

[29] Ten-Huei Guo and Jonathan Litt. Resilient propulsion control research for the NASA integrated resilient aircraft control (IRAC) project. In *AIAA Infotech@ Aerospace 2007 Conference and Exhibit*, page 2802, 2007.

[30] Hafid Smaili, Jan Breeman, and Thomas Lombaerts. *Tool-Based Design and Evaluation of Resilient Flight Control Systems.* INTECH Open Access Publisher, 2012.

[31] Philippe Goupil and Andres Marcos. Industrial benchmarking and evaluation of ADDSAFE FDD designs. *IFAC Proceedings Volumes*, 45(20):1131–1136, 2012.

[32] Christopher Edwards, Lejun Chen, Ahmed Khattab, Halim Alwi, and M. Sato. Flight evaluations of sliding mode fault tolerant controllers. In *2018 15th*

International Workshop on Variable Structure Systems (VSS), pages 180–185. IEEE, 2018.

[33] Yoko Watanabe, Augustin Manecy, Antal Hiba, Sho Nagai, and Shin Aoki. Vision-integrated navigation system for aircraft final approach in case of GNSS/SBAS or ILS failures. In *AIAA Scitech 2019 Forum*, page 0113, 2019.

[34] Youmin Zhang, V Sivasubramaniam Suresh, Bin Jiang, and Didier Theilliol. Reconfigurable control allocation against aircraft control effector failures. In *Control Applications, 2007. CCA 2007. IEEE International Conference on*, pages 1197–1202. IEEE, 2007.

[35] 3DR. 3DR robotics, inc | Drone & UAV technology. https://3dr.com/, 2017. [Online; accessed 28-Feb-2017].

[36] Wei Zhong Fum. Implementation of simulink controller design on IRIS+ quadrotor. Master's thesis, Monterey, California: Naval Postgraduate School, 2015.

[37] Mathworks. Simulink pixhawk support package. https://uk.mathworks.com/hardware-support/px4-autopilots.html, March 2020. [Online; accessed 8-Apr-2020].

[38] Henrik Sönnerlind. Modeling the dynamics of a gyroscope. https://www.comsol.com/blogs/modeling-the-dynamics-of-a-gyroscope/, December 2015. [Online; accessed 5-Apr-2020].

[39] Felipe Jaramillo Gómez. Quadcopter test bench (gyroscope). https://www.youtube.com/watch?v=5caO5zPTjhs, July 2013. [Online; accessed 8-Apr-2020].

[40] Rs components | industrial, electronic products and solutions, 2020. [Online; accessed 10-Aug-2020].

[41] Helipal. Storm8 drone. http://www.helipal.com/storm-drone-8-gps-flying-platform-body-only.html, June 2015. [Online; accessed 8-Apr-2020].

[42] Matt R. Jardin and Eric R. Mueller. Optimized measurements of unmanned-air-vehicle mass moment of inertia with a bifilar pendulum. *Journal of Aircraft*, 46(3):763–775, 2009.

[43] AirbusNewsRoom. Future of urban mobility. https://www.airbus.com/newsroom/news/en/2016/12/My-Kind-Of-Flyover.html, December 2016. [Online; accessed 8-Apr-2020].

[44] Tae H. Ha, Keunseok Lee, and John T. Hwang. Large-scale design-economics optimization of eVTOL concepts for urban air mobility. In *AIAA Scitech 2019 Forum*, page 1218, 2019.

[45] Dominic Lenton. The measure of volocopter flying taxi. *Engineering & Technology*, 13(7/8):10–11, 2018.

[46] Sandy Ong. Electric air taxi flies over Singapore-[news]. *IEEE Spectrum*, 56(12):7–8, 2019.

[47] Nicholas Polaczyk, Enzo Trombino, Peng Wei, and Mihaela Mitici. A review of current technology and research in urban on-demand air mobility applications. In *8th Biennial Autonomous VTOL Technical Meeting and 6th Annual Electric VTOL Symposium*, 2019.

[48] Mark Williamson. Surefly takes helicopters-[news paris air show]. *Engineering & Technology*, 12(7-8):14–15, 2017.

[49] Mark Huber. Surefly makes first flight. *Aviation International News*, 2018.

[50] Stephen Fitzpatrick. Vertical Aerospace Company. https://www.vertical-aerospace.com/, January 2020. [Online; accessed 5-Apr-2020].

[51] Yuri Kaheyama. Japan's NEC shows 'flying car' hovering steadily for minute. https://apnews.com/2d4ce8955bc04032928afdf97ed42818, August 2019. [Online; accessed8-Apr-2020].

[52] Rick Chen. NASA and Uber test system for future urban air transport. https://www.nasa.gov/feature/ames/nasa-and-uber-test-system-for-future-urban-air-transport, October 2019. [Online; accessed 5-Apr-2020].

[53] Baykarmakina. Flying car. https://www.baykarsavunma.com/sayfa-Ucan-Araba.html, September 2019. [Online; accessed 5-Apr-2020].

[54] Aston Martin. The Aston Martin Volante Vision Concept. https://www.astonmartin.com/en-au/models/future-models/the-aston-martin-volante-vision-concept-2019, September 2019. [Online; accessed 5-Apr-2020].

[55] David Reid. Rolls-Royce joins the race to develop a flying taxi. https://www.cnbc.com/2018/07/16/rolls-royce-joins-the-race-to-develop-a-flying-car.html, 2019. [Online; accessed 5-Apr-2020].

[56] EmbraerX. EmbraerX unveils new flying vehicle concept for future urban air mobility. https://embraer.com/global/en/news?slug=1206601-embraerx-unveils-new-flying-vehicle-concept-for-future-urban-air-mobility, November 2019. [Online; accessed 15-Apr-2020].

[57] OPENER. BlackFly. https://www.opener.aero/, December 2019. [Online; accessed 8-Apr-2020].

[58] eVTOL. CityAirbus performs first untethered flight. https://evtol.com/news/cityairbus-evtol-untethered-flight-test/, February 2020. [Online; accessed 1-Apr-2020].

[59] Chris Hughes and Ruth Halkon. Heathrow British Airways passenger plane- hit by suspected drone. http://www.getwestlondon.co.uk/news/west-london-news/heathrow-british-airways-passenger-plane-11199930, Apr 2016. [Online; accessed 8-Apr-2020].

[60] Leith Huffadine. Shocking moment a drone falls from the sky and smashes into a moving car on the Sydney Harbour Bridge. http://www.dailymail.co.uk/news/article-3754290/Drone-hits-moving-car-Sydney-Harbour-Bridge.html, Aug 2016. [Online; accessed 8-Apr-2020].

[61] Alice Ross. Woman dies in police pursuit of suspect car after drone flew near jail. https://www.theguardian.com/uk-news/2016/aug/09/woman-dies-police-pursuit-crash-after-drone-over-wandsworth-prison, Aug 2016. [Online; accessed 8-Apr-2020].

[62] Adam Lusher. London woman dies in possibly the first drone-related accidental death. http://www.independent.co.uk/news/uk/home-news/drones-fatal-road-accident-first-non-military-drone-death-accident-car-crash-surveillance-safety-a7180576.html, Aug 2016. [Online; accessed 8-Apr-2020].

[63] CBS. Man, 19, killed by remote controlled helicopter. https://www.youtube.com/watch?v=4e6WZ_dEOwY, Sep 2013. [Online; accessed 8-Apr-2020].

[64] Michael Zennie. Horror as remote-control helicopter stunt pilot, 19, partially-decapitates himself with his aircraft after he lost control. http://www.dailymail.co.uk/news/article-2413231/Roman-Pirozek-Jr-Man-decapitates-remote-control-helicopter.html, Sep 2013. [Online; accessed 8-Apr-2020].

[65] Alex Hern. Facebook solar-powered drone under investigation after 'accident'. https://www.theguardian.com/technology/2016/nov/22/facebook-solar-powered-aquila-drone-under-investigation, Nov 2016. [Online; accessed 8-Apr-2020].

[66] BBC. Google confirms end of internet drone project. http://www.bbc.co.uk/news/technology-38596974, Jan 2017. [Online; accessed 5-Apr-2020].

[67] Sherisse Pham. Facebook gives up on building internet drones. https://money.cnn.com/2018/06/27/technology/facebook-aquila-drone-abandon/index.html, 2018. [Online; accessed5-Apr-2020].

[68] Youmin Zhang and Jin Jiang. Bibliographical review on reconfigurable fault-tolerant control systems. *Annual reviews in control*, 32(2):229–252, 2008.

[69] Wen Chen, Afef Fekih, and Zehui Mao. Fault Detection, Estimation/Reconstruction, and Fault-Tolerant Control: Theory and Practice. *Mathematical Problems in Engineering*, 2016, 2016.

[70] Zhixiang Liu, Chi Yuan, Youmin Zhang, and Jun Luo. A learning-based fault tolerant tracking control of an unmanned quadrotor helicopter. *Journal of Intelligent & Robotic Systems*, 84(1-4):145–162, 2016.

[71] Halim Alwi, Christopher Edwards, Olaf Stroosma, and JA Mulder. Evaluation of a sliding mode fault-tolerant controller for the El Al incident. *Journal of guidance, control, and dynamics*, 33(3):677–694, 2010.

[72] Abbas Chamseddine, Youmin Zhang, Camille-Alain Rabbath, Cameron Fulford, and Jacob Apkarian. Model reference adaptive fault tolerant control of a quadrotor UAV. *AIAA Infotech@ Aerospace, St. Louis, Missouri, USA*, 2931, 2011.

[73] Hojjat A Izadi, Youmin Zhang, and Brandon W Gordon. Fault tolerant model predictive control of quad-rotor helicopters with actuator fault estimation. *IFAC Proceedings Volumes*, 44(1):6343–6348, 2011.

[74] A. Merheb, H. Noura, and F. Bateman. Active fault tolerant control of quadrotor UAV using sliding mode control. In *International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 156–166. IEEE, 2014.

[75] A. Merheb, H. Noura, and F. Bateman. A novel emergency controller for quadrotor UAVs. In *IEEE Conference on Control Applications (CCA)*, pages 747–752. IEEE, 2014.

[76] Abdel-Razzak Merheb, Hassan Noura, and François Bateman. Design of passive fault–tolerant controllers of a quadrotor based on sliding mode theory. *International Journal of Applied Mathematics and Computer Science*, 25(3):561–576, 2015.

[77] Tong Li, Youmin Zhang, and Brandon W Gordon. Passive and active nonlinear fault-tolerant control of a quadrotor unmanned aerial vehicle based on the sliding mode control technique. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 227(1):12–23, 2013.

[78] Remus C Avram. *Fault Diagnosis and Fault-Tolerant Control of Quadrotor UAVs.* PhD thesis, Wright State University, 2016.

[79] Mark W Mueller and Raffaello D'Andrea. Stability and control of a quadrocopter despite the complete loss of one, two, or three propellers. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 45–52. IEEE, 2014.

[80] Mark W Mueller and Raffaello D Andrea. Relaxed hover solutions for multicopters: Application to algorithmic redundancy and novel vehicles. *The International Journal of Robotics Research*, 35(8):873–889, 2015.

[81] Thomas Schneider, Guillaume Ducard, Konrad Rudin, and Pascal Strupler. Fault-tolerant control allocation for multirotor helicopters using parametric programming. In *International Micro Air Vehicle Conference and Flight Competition (IMAV)*, 2012.

[82] Aryeh Marks, James F. Whidborne, and Ikuo Yamamoto. Control allocation for fault tolerant control of a VTOL octorotor. *Proceedings of UKACC International Conference on Control*, pages 357–362, 2012.

[83] Halim Alwi and Christopher Edwards. Sliding mode fault-tolerant control of an octorotor using linear parameter varying-based schemes. *IET Control Theory & Applications*, 9(4):618–636, 2015.

[84] Majd Saied, Benjamin Lussier, Isabelle Fantoni, Clovis Francis, Hassan Shraim, and Guillaume Sanahuja. Fault diagnosis and fault-tolerant control strategy for rotor failure in an octorotor. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5266–5271, 2015.

[85] Majd Saied, Benjamin Lussier, Isabelle Fantoni, Clovis Francis, and Hassan Shraim. Fault tolerant control for multiple successive failures in an octorotor: Architecture and experiments. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 40–45, 2015.

[86] Ahmed Khattab, Halim Alwi, and Christopher Edwards. Implementation of sliding mode fault tolerant control on the IRIS+ quadrotor. In *2018 IEEE Conference on Control Technology and Applications (CCTA)*, pages 1724–1729. IEEE, 2018.

[87] Ahmed Khattab, Halim Alwi, and Christopher Edwards. Fault tolerant control of a spherical UAV. In *2019 4th Conference on Control and Fault Tolerant Systems (SysTol)*, pages 92–97. IEEE, 2019.

[88] Ahmed Khattab, Halim Alwi, and Christopher Edwards. Mitigating total rotor failure in quadrotor using LPV based sliding mode control scheme. In *2019 4th Conference on Control and Fault Tolerant Systems (SysTol)*, pages 98–103. IEEE, 2019.

[89] Mirza Tariq Hamayun, Christopher Edwards, and Halim Alwi. *Fault tolerant control schemes using integral sliding modes*. Springer, 2016.

[90] Rudaba Khan, Paul Williams, Paul Riseborough, Asha Rao, and Robin Hill. Designing a nonlinear model predictive controller for fault tolerant flight control. *arXiv preprint arXiv:1609.01529*, 2016.

[91] Rudaba Khan, Paul Williams, Paul Riseborough, Asha Rao, and Robin Hill. Active fault tolerant flight control system design-a UAV case study. *arXiv preprint arXiv:1610.03162*, 2016.

[92] Christopher Edwards, Thomas Lombaerts, and Hafid Smaili. *Fault tolerant flight control*, volume 399. Springer, 2010.

[93] Halim Alwi, Christopher Edwards, and Andrés Marcos. Actuator and sensor fault reconstruction using an LPV sliding mode observer. In *AIAA guidance, navigation, and control conference*, page 8157, 2010.

[94] Halim Alwi and Christopher Edwards. Robust actuator fault reconstruction for LPV systems using sliding mode observers. In *Decision and Control (CDC), 2010 49th IEEE Conference on*, pages 84–89. IEEE, 2010.

[95] Halim Alwi and Christopher Edwards. Oscillatory failure case detection for aircraft using an adaptive sliding mode differentiator scheme. In *American Control Conference (ACC), 2011*, pages 1384–1389. IEEE, 2011.

[96] Halim Alwi and Christopher Edwards. Application of second order sliding mode observers for fault reconstruction on the ADDSAFE benchmark. In *AIAA Guidance, Navigation, and Control Conference*, page 6682, 2011.

[97] Halim Alwi and Christopher Edwards. Evaluation of sliding mode observers for fault reconstruction on the ADDSAFE functional engineering simulator. *SAE International Journal of Aerospace*, 4(2011-01-2802):1485–1499, 2011.

[98] Halim Alwi, Lejun Chen, and Christopher Edwards. Reconstruction of simultaneous actuator and sensor faults for the reconfigure benchmark using a sliding mode observer. *IFAC Proceedings Volumes*, 47(3):3497–3502, 2014.

[99] Paulo Rosa, José Vasconcelos, and Murray Kerr. A mixed-μ approach to the integrated design of an FDI/FTC system applied to a high-fidelity industrial Airbus nonlinear simulator. *IFAC-PapersOnLine*, 48(21):988–993, 2015.

[100] Lejun Chen, Halim Alwi, Christopher Edwards, and M. Sato. Hardware-in-the-loop evaluation of an LPV sliding mode fixed control allocation scheme on the MuPAL-α research aircraft. In *2017 IEEE Conference on Control Technology and Applications (CCTA)*, pages 590–595. IEEE, 2017.

[101] Christopher Edwards and Sarah Spurgeon. *Sliding Mode Control: Theory and Applications.* CRC Press, 1998.

[102] Vadim I. Utkin. *Sliding modes in control and optimization.* Springer Science, 2013.

[103] VI Utkin. Methods for constructing discontinuity planes in multidimensional variable structure systems. *Automation and Remote Control*, 39:1466–1470, 1978.

[104] Ahmed Khattab, Ayman Hamdy Kassem, and Gamal Baioumi. Low cost framework for parameter identification of unmanned aerial vehicles. Master's thesis, Cairo University, 2015.

[105] Hassan K Khalil. *Nonlinear control.* Prentice Hall, 2014.

[106] PX4DevTeam. *PX4 Autopilot User Guide (master).* https://docs.px4.io/, April 2020.

[107] Michael Oborne. *Mission Planner (software).* http://ardupilot.org/planner/, April 2019.

[108] Adam Polak. *PX4 Development Kit for Simulink.* http://polakiumengineering.com/px4-development-kit-for-simulink/, Jun 2015. [Online; accessed 8-Apr-2020].

[109] Robert Hartley. APM2 Simulink Blockset. *MATLAB Central*, 13, 2012.

[110] Kun Li, Swee King Phang, Ben M. Chen, and Tong Heng Lee. Platform design and mathematical modeling of an ultralight quadrotor micro aerial vehicle. In *International Conference on Unmanned Aircraft Systems (ICUAS).* IEEE, 2013.

[111] Kang Hou, Han Xu Sun, Qing Xuan Jia, Yan Heng Zhang, Nan Zhe Wei, and Liu Meng. Analysis and design of spherical aerial vehicle's motion modes. In *Applied Mechanics and Materials*, volume 411, pages 1836–1839. Trans Tech Publ, 2013.

[112] O. C. Carholt, E. Fresk, G. Andrikopoulos, and G. Nikolakopoulos. Design, modelling and control of a single rotor UAV. In *2016 24th Mediterranean Conference on Control and Automation (MED)*, pages 840–845, June 2016.

[113] K Malandrakis, Roland Dixon, A Savvaris, and A Tsourdos. Design and development of a novel spherical UAV. *IFAC-PapersOnLine*, 49(17):320–325, 2016.

[114] Tommaso Matassini, Hyo-Sang Shin, Antonios Tsourdos, and Mario Innocenti. Adaptive control with neural networks-based disturbance observer for a spherical UAV. *IFAC-PapersOnLine*, 49(17):308–313, 2016.

[115] Ben Loh and Jamey D Jacob. Modeling and attitude control analysis of a spherical VTOL aerial vehicle. *51st AIAA Aerospace Sciences Meeting, Dallas, TX*, 2013.

[116] Robert C Nelson. *Flight stability and automatic control*, volume 2. WCB/McGraw Hill New York, 1998.

[117] The MathWorks. *Global Optimization Toolbox User's Guide.* The MathWorks, Inc, 3 Apple Hill Drive Natick, MA, R2018a edition, 2018.

[118] Cara McGoogan. Self-flying taxi to transport passengers in Dubai. http://www.telegraph.co.uk/technology/2017/02/14/self-flying-taxi-transport-passengers-dubai/, Feb 2017. [Online; accessed 5-Apr-2020].

[119] YM Zhang, A Chamseddine, CA Rabbath, BW Gordon, C-Y Su, S Rakheja, C Fulford, J Apkarian, and P Gosselin. Development of advanced FDD and FTC techniques with application to an unmanned quadrotor helicopter testbed. *Journal of the Franklin Institute*, 350(9):2396–2422, 2013.

[120] Francois Bateman, Hassan Noura, and Mustapha Ouladsine. Actuators fault diagnosis and tolerant control for an unmanned aerial vehicle. In *Control Applications, 2007. CCA 2007. IEEE International Conference on*, pages 1061–1066. IEEE, 2007.

[121] Alessandro Freddi, Alexander Lanzon, and Sauro Longhi. A feedback linearization approach to fault tolerance in quadrotor vehicles. *IFAC Proceedings Volumes*, 44(1):5413–5418, 2011.

[122] Johannes Stephan, Lorenz Schmitt, and Walter Fichter. Linear parameter-varying control for quadrotors in case of complete actuator loss. *Journal of Guidance, Control, and Dynamics*, pages 1–15, 2018.

[123] Cédric De Crousaz, Farbod Farshidian, Michael Neunert, and Jonas Buchli. Unified motion control for dynamic quadrotor maneuvers demonstrated on slung load and rotor failure tasks. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 2223–2229. IEEE, 2015.

[124] Peng Lu and Erik-Jan van Kampen. Active fault-tolerant control for quadrotors subjected to a complete rotor failure. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 4698–4703. IEEE, 2015.

[125] Ashutosh Simha, Sharvaree Vadgama, and Soumyendu Raha. A geometric approach to rotor failure tolerant trajectory tracking control design for a quadrotor. *arXiv preprint arXiv:1704.00327*, 2017.

[126] P. Gahinet, A. Nemirovski, A. J. Laub, and M. Chilali. The LMI Control Toolbox. For Use with Matlab. User's Guide. Natick, MA: The MathWorks, 1995.

[127] Parrot. Parrot official. https://www.parrot.com/, April 2019. [Online; accessed 5-Apr-2020].

[128] Jeff Ferrin, Robert Leishman, Randy Beard, and Tim McLain. Differential flatness based control of a rotorcraft for aggressive maneuvers. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2688–2693. IEEE, 2011.

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

[130] Joseph Habeck and Peter Seiler. Moment of inertia estimation using a bilar pendulum. Technical report, University of Minnesota, April 2016.