

**AUTONOMOUS SENSOR DEPLOYMENT USING AN
UNMANNED AERIAL VEHICLE**

BY
ABDULMAJEED MUHAMMAD KABIR

A Thesis Presented to the
DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

SYSTEMS & CONTROL ENGINEERING

DECEMBER 2016

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN 31261, SAUDI ARABIA

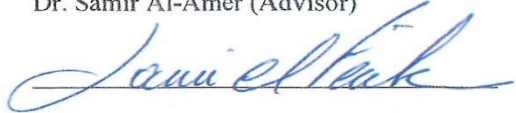
DEANSHIP OF GRADUATE STUDIES

This thesis, written by **ABDULMAJEED MUHAMMAD KABIR** under the direction of his thesis advisor and approved by his thesis committee, has been presented to and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE IN SYSTEMS & CONTROL ENGINEERING**.

Thesis Committee



Dr. Samir Al-Amer (Advisor)



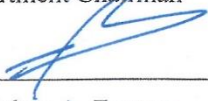
Dr. Sami El Ferik (Member)



Dr. Abdul Wahid Al-Saif (Member)



Dr. Hesham K. Al-Fares
Department Chairman



Dr. Salam A. Zummo
Dean of Graduate Studies



16/1/17

Date

© Abdulmajeed Muhammad Kabir

December 2016

*Dedicated to my beloved parents, lovely siblings, nephews and nieces, unmarried spouse
and unborn kids*

ACKNOWLEDGEMENTS

This thesis is the result of work done for a Masters degree from January 2015 to December 2016 at the department of Systems Engineering, King Fahd University under the supervision of Dr. Samir Al-Amer. I would like to thank my advisor, Dr. Samir Al-Amer for his kindness and guidance throughout the period of my research, I also appreciate Dr. Sami El-Ferik for selecting me in his research team for this exciting work. I would also like to thank Dr. Abdul Wahid Al-Saif for being in my thesis committee.

Table of Contents

ACKNOWLEDGEMENTS	v
LIST OF TABLES	xii
LIST OF FIGURES	xiii
LIST OF ABBREVIATIONS	xx
ABSTRACT (ENGLISH)	xxi
ABSTRACT (ARABIC)	xxii
CHAPTER 1 INTRODUCTION	1
1.1 Unmanned Aerial Vehicles	2
1.2 Micro Aerial Vehicles	3
1.3 The Quadrotor	6
1.4 Problem Statement and Application	7
1.5 Thesis Organization	10
1.6 Scope and Objectives of Study	10
1.7 Problem Formulation and Contribution	11
CHAPTER 2 LITERATURE REVIEW	12
2.1 Introduction	12
2.2 History, Overview and Applications of Unmanned Aerial Vehicles	12
2.3 Quadrotor Control Methods	17
2.3.1 Literature on Control Structures	18

2.3.2	Quadrotor Performance and Autonomy	20
2.3.3	Applications in Payload Delivery	22
CHAPTER 3 QUADROTOR SYSTEM MODELING		25
3.1	Introduction.....	25
3.2	Kinematic Model	27
3.3	Dynamics Modeling	28
3.3.1	Rotational Dynamics.....	29
3.3.2	Translational Equations of Motion	31
3.4	State Space Model	31
3.4.1	Rotational Dynamics.....	33
3.4.2	Translational Equations of Motion	34
3.4.3	State Space Representation	35
3.5	Expanded Model	36
3.5.1	Aerodynamic Drag Forces and Moments	37
3.5.2	Ground Effect	37
3.5.3	Wind Gust Effect	41
3.5.4	Sensor Noise	41
3.5.5	Modeling Prismatic Arm	42
3.5.6	Change in Moment of Inertia	43
3.5.7	Change in Center of Mass	46

3.5.8	Dynamic Effects of Change in Center of Mass	47
3.5.9	Complete Model	49
CHAPTER 4 AUTONOMOUS QUADROTOR CONTROL.....		50
4.1	Introduction.....	50
4.2	Backstepping Control	53
4.2.1	Controller Development for Roll Channel.....	55
4.2.2	Controller Development for Altitude Channel	57
4.2.3	Controller Development for Other Channels.....	60
4.2.4	Controller Development for Under-Actuated Channel	61
4.2.5	Backstepping Controller for Plant B	62
4.3	Active Disturbance Rejection Control	66
4.3.1	Controller Formulation	68
4.3.2	Controller Design for Roll Dynamics.....	73
4.3.3	Controller Design for Altitude Dynamics (Landing).....	75
4.4	Numerical Optimal Control	77
4.4.1	Direct and Indirect Methods	79
4.4.2	Literature on Trajectory Optimization.....	82
4.4.3	Trajectory Optimization Control.....	83
4.4.4	Solution of the Trajectory Optimization Problem	84
4.4.5	Direct Transcription (Hermite-Simpson Collocation).....	86

4.5	Proportional Derivative Control of Prismatic Robotic Arm	89
4.6	Proportional Optimization of Controllers	90
CHAPTER 5 HYBRID SYSTEM.....		92
5.1	Hybrid System	92
5.2	Hybrid Control Structure	94
5.2.1	Sensor Deployment.....	96
5.2.2	Sensor Retrieval	97
5.3	Robotic Hand	99
5.4	Quadrotor Vision System	100
5.4.1	Introduction to the Vision System	101
5.4.2	Camera Calibration	101
5.4.3	Projection Transformation	105
5.4.4	Image Processing	109
5.4.5	Vision Feedback to Controller	112
CHAPTER 6 SIMULATION, RESULTS AND VIRTUALIZATION		115
6.1	Introduction.....	115
6.2	Simulation Tools	115
6.2.1	MATLAB.....	116
6.2.2	VREP	116
6.2.3	OpenCV	116

6.3	Results: Backstepping Controller	117
6.3.1	Quadrotor Backstepping Control (Plant A)	118
6.3.2	Quadrotor-Manipulator Backstepping Control I (Plant B)	122
6.3.3	Quadrotor-Manipulator Sensor Capsule Handling	125
6.4	Results: Active Disturbance Rejection Controller	128
6.4.1	Quadrotor-Manipulator ADRC Control I	129
6.4.2	Quadrotor-Manipulator ADRC Control II	134
6.4.3	Quadrotor Landing with ADRC.....	139
6.4.4	PD Control of 1-DOF Prismatic Arm.....	142
6.5	Results: Trajectory Optimization Controller	144
6.5.1	Trajectory Optimization Control	144
6.6	Comments on Controller Performance	149
6.6.1	Backstepping Controller	149
6.6.2	Active Disturbance Rejection Controller (ADRC).....	150
6.6.3	Trajectory Optimization Controller	150
6.7	Robotics Simulation using VREP, OpenCV, Python and Matlab	151
CHAPTER 7 SUMMARY AND CONCLUSIONS.....		157
7.1	Summary of Work Done.....	157
7.2	Summary of Contributions	158
7.3	Future Work	159

7.4 Appreciation	159
REFERENCES.....	160
VITAE.....	172

LIST OF TABLES

Table 1.1	UAV Classification Based on Operational Characteristics	2
Table 1.2	Aerodynamic Configuration	4
Table 1.3	Comparison of VTOL Concepts	5
Table 1.4	Advantages and Disadvantages of Quadrotors	7
Table 2.1	Properties of Different Control Schemes	24
Table 3.1	Influence of Ground Effect	40
Table 6.1	Quadrotor Model Parameters	117
Table 6.2	Backstepping Controller Parameters.....	118
Table 6.3	ADRC Controller Parameters	128
Table 6.4	ADRC Controller Parameters for Quadrotor Landing.....	140
Table 6.5	Prismatic Arm Model and Control Parameters.....	143

LIST OF FIGURES

Figure 1.1	Classification of Aircrafts	3
Figure 1.2	Aerodynamic Configurations.....	5
Figure 1.3	Reflection Seismology and Vibroseis	8
Figure 1.4	Typical Middle-Eastern Field with no Foliage and Open Skies	8
Figure 1.5	Geophone Sensor and Capsule.....	9
Figure 2.1	Historical Aircraft Types	15
Figure 2.2	Application of UAV, UAS, Quadrotors.....	16
Figure 3.1	Quadrotor with Reference Frames	26
Figure 3.2	Quadrotor with Landing Skid; Height h_{lz} (m).....	39
Figure 3.3	Ground Effect Variation with Altitude Decrease.....	40
Figure 3.4	Prismatic Joint, Barret Hand and Combined Free-body Diagram	42
Figure 3.5	Geometric Description of Shapes under Analysis	44
Figure 3.6	Central-Axis Mass Alignment	47
Figure 4.1	Controller Architecture for the ADRC Controller	67
Figure 4.2	Quadrotor with Landing Skids of height h_{lz}	76
Figure 4.3	Quadrotor Landing Profiles	76
Figure 4.4	Major Components and Classes of Optimal Control Methods	81
Figure 4.5	Transcription	86
Figure 5.1	Hybrid System	93

Figure 5.2	Phases and Modes of Operation.....	94
Figure 5.3	Finite State Automata	95
Figure 5.4	Sensor Deployment Phase.....	97
Figure 5.5	Sensor Retrieval Phase.....	98
Figure 5.6	Barret Hand with Reach Radius.....	99
Figure 5.7	Barret Hand Ellipse.....	99
Figure 5.8	Sensor Capsule Cap	101
Figure 5.9	Calibration Chessboard Scene in VREP.....	102
Figure 5.10	Camera Calibration Scene with a Perspective Vision Sensor.....	103
Figure 5.11	Checkerboard Images at Different Perspectives.....	103
Figure 5.12	Camera Calibration Toolbox in Matlab	104
Figure 5.13	Quadrotor Visual Feedback System in Pinhole Camera Form	105
Figure 5.14	Image-Camera-World Transformation	107
Figure 5.15	Vision Sensor Image of Sensor Capsule.....	110
Figure 5.16	Pixel Intensities and Color of Object Blue Pixel Region.....	110
Figure 5.17	Red, Green and Blue Components of the Scene	111
Figure 5.18	Color Segmentation, Binarization and Centroid Detection	111
Figure 5.19	Color Segmentation, Binarization and Centroid Detection in the Absence of Perspective Distortion.....	111
Figure 5.20	Vector Between Center of Image Plane and Object Centroid	112

Figure 5.21	Quadrotor with Vision Sensor Detecting Sensor Capsule	113
Figure 5.22	Quadrotor Stabilized with Barret Hand Zone of Reach Coincident with Sensor Capsule.....	113
Figure 5.23	Quadrotor-Decoupled Barret Hand Mechanism in VREP.....	114
Figure 6.1	Simulation Tools.....	117
Figure 6.2	Altitude Dynamics (Backstepping).....	119
Figure 6.3	Roll Dynamics Stabilized for y Translation (Backstepping).....	119
Figure 6.4	Pitch Dynamics Stabilized for x Translation (Backstepping).....	119
Figure 6.5	Yaw Dynamics (Backstepping)	120
Figure 6.6	Dynamics of x Translation (Backstepping)	120
Figure 6.7	Dynamics of y Translation (Backstepping)	120
Figure 6.8	U_1 Control Input (Backstepping)	121
Figure 6.9	U_2 Control Input (Backstepping)	121
Figure 6.10	U_3 Control Input (Backstepping)	121
Figure 6.11	U_4 Control Input (Backstepping)	122
Figure 6.12	Altitude Dynamics (Backstepping Plant B).....	123
Figure 6.13	Roll Dynamics (Backstepping Plant B)	123
Figure 6.14	Pitch Dynamics (Backstepping Plant B).....	123
Figure 6.15	Yaw Dynamics (Backstepping Plant B)	124
Figure 6.16	U_1 Control Input (Backstepping Plant B)	125

Figure 6.17 U_2 Control Input (Backstepping Plant B)	125
Figure 6.18 U_3 Control Input (Backstepping Plant B)	125
Figure 6.19 U_4 Control Input (Backstepping Plant B)	125
Figure 6.20 Effect of Sensor Capsule Drop on Altitude Dynamics at $t = 5s$	126
Figure 6.21 U_1 Control Input During Drop (Backstepping Plant B).....	126
Figure 6.22 Effect of Lifting the Sensor Capsule on the Quadrotor Dynamics	127
Figure 6.23 U_1 Control Input during Drop (Inset: Initial Reaction).....	127
Figure 6.24 Altitude Dynamics (ADRC)	129
Figure 6.25 Estimation of Total Disturbance in Altitude Dynamics (ADRC).....	129
Figure 6.26 U_1 Control Input (ADRC).....	130
Figure 6.27 Roll Dynamics (ADRC).....	130
Figure 6.28 Estimation of Total Disturbance in Roll Dynamics (ADRC)	130
Figure 6.29 U_2 Control Input (ADRC).....	131
Figure 6.30 Pitch Dynamics (ADRC)	131
Figure 6.31 Estimation of Total Disturbance in Pitch Dynamics (ADRC).....	131
Figure 6.32 U_3 Control Input (ADRC).....	132
Figure 6.33 Yaw Dynamics (ADRC).....	132
Figure 6.34 Estimation of Total Disturbance in Yaw Dynamics (ADRC)	132
Figure 6.35 U_4 Control Input (ADRC).....	133
Figure 6.36 Dynamics of x Translation (ADRC).....	133

Figure 6.37 Dynamics of y Translation (ADRC).....	133
Figure 6.38 Altitude Dynamics (ADRC Plant B).....	134
Figure 6.39 Estimation of Total Disturbance in Altitude Dynamics (ADRC Plant B)	135
Figure 6.40 U_1 Control Input (ADRC Plant B).....	135
Figure 6.41 Roll Dynamics (ADRC Plant B).....	135
Figure 6.42 Estimation of Total Disturbance in Roll Dynamics (ADRC Plant B)	136
Figure 6.43 U_2 Control Input (ADRC Plant B).....	136
Figure 6.44 Pitch Dynamics (ADRC Plant B)	136
Figure 6.45 Estimation of Total Disturbance in Pitch Dynamics (ADRC Plant B)....	137
Figure 6.46 U_3 Control Input (ADRC Plant B).....	137
Figure 6.47 Yaw Dynamics (ADRC Plant B).....	137
Figure 6.48 Estimation of Total Disturbance in Yaw Dynamics (ADRC Plant B)....	138
Figure 6.49 U_4 Control Input (ADRC Plant B).....	138
Figure 6.50 Effect of Integrator on Noise Signal	139
Figure 6.51 Ground Effect during Quadrotor Landing	139
Figure 6.52 Quadrotor Landing (ADRC True and Estimated Altitude State).....	140
Figure 6.53 True and Estimated Quadrotor Velocity during Landing	141
Figure 6.54 ESO Estimation Error in Altitude and Velocity	141
Figure 6.55 Estimation of Total Disturbance during Landing	141

Figure 6.56 U_1 Control Input during Landing	142
Figure 6.57 Prismatic Arm Position Dynamics (PD Controller).....	143
Figure 6.58 Velocity Dynamics (PD Controller)	143
Figure 6.59 Control Input U_τ	144
Figure 6.60 Altitude Dynamics (Trajectory Optimization).....	145
Figure 6.61 Altitude Control Input (Trajectory Optimization)	145
Figure 6.62 Yaw Dynamics (Trajectory Optimization)	146
Figure 6.63 Yaw Control Input (Trajectory Optimization).....	146
Figure 6.64 Roll Dynamics (Trajectory Optimization)	147
Figure 6.65 Roll Control Input (Trajectory Optimization).....	147
Figure 6.66 Pitch Dynamics (Trajectory Optimization).....	148
Figure 6.67 Pitch Control Input (Trajectory Optimization)	148
Figure 6.68 Simulink Diagram for Quadrotor Backstepping Control.....	151
Figure 6.69 Simulink Diagram for Backstepping Control of Quadrotor with Disturbances.....	151
Figure 6.70 Simulink Diagram for ADRC Control of Quadrotor	152
Figure 6.71 Simulink Diagram for Full ADRC Control and Parameter Optimization	152
Figure 6.72 Simulink Diagram for Quadrotor Landing with Sensor Noise	153
Figure 6.73 Simulink Diagram for Hybrid System	153

Figure 6.74 Simulink Diagram for Control of Prismatic Arm	154
Figure 6.75 Simulink Diagram for ADRC Control and Parameter Optimization.....	154
Figure 6.76 VREP Scene of Quadrotor Hovering Above Sensor Capsule	155
Figure 6.77 VREP Scene Showing Camera Field of View	155
Figure 6.78 VREP Scene Emulating Sensor Capture.....	156
Figure 6.79 VREP Scene Emulating Sensor Pick-Up.....	156

LIST OF ABBREVIATIONS

ADRC Active disturbance rejection control

CoM Center of Mass effect

ESO Extended State Observer

FOV Field of View

FSM Finite State Machine

GPS Global Positioning System

HSV Hue-Saturation-Value color space

NLP Nonlinear Programming

QM Quadrotor Manipulator

RGB Red-Green-Blue color space

UAV Unmanned Aerial Vehicle

UAS Small Unmanned Aircraft

VTOL Vertical Take-off and Landing

ZOR Zone of Reach

THESIS ABSTRACT

NAME: Abdulmajeed Muhammad Kabir

TITLE OF STUDY: Autonomous Sensor Deployment using an Unmanned
Aerial Vehicle

MAJOR FIELD: Systems and Control Engineering

DATE OF DEGREE: December, 2016

This thesis presents the development of an autonomous geophone-sensor deployment system using quadrotors. The objective is to present control structures for the deployment and retrieval of the geophone sensor capsule at a desired location. A model is developed for the quadrotors-manipulator system taking into account aerodynamic drag, ground effect, center of mass effect and wind disturbance. Thereafter, the backstepping control, active disturbance rejection control and trajectory optimization control techniques are proposed to solve the autonomous control task. Finally, a hybrid system comprising vision-assist subsystem and some finite state automata is proposed for the precision landing and task completion for the quadrotor system. Results were verified using Matlab and VREP.

ملخص الرسالة

الإسم:

عبد المجيد محمد كبير

عنوان الدراسة:

نشر حساسات تلقائي باستخدام طائرات بدون طيار

التخصص:

هندسة نظم وتحكم

تاريخ درجة:

ربيع الأول 1438هـ

تعرض هذه الرسالة تطوير نظام تلقائي لنشر حساسات جيوفونية باستخدام كوادروتر ، الهدف هنا عرض هيكل تحكم لنشر و إسترجاع كبسولات الحساسات الجيوفونية الى أو من موقع محدد. تم تطوير نموذج رياضي لنظام يشمل الكوادروتر و ذراع آلية يأخذ في الاعتبار مقاومة الهواء و تأثير الأرض ومركز الكتلة و اضطراب الرياح . بعدها تم إقتراح نظام تحكم الخطوات الخلفية ونظام تحكم مضاد فعال للإضطراب ونظم إختيار المسار الأفضل لحل مشكلة التحكم التلقائي . وفي الختام تم إقتراح نظام مركب يشمل نظام مبني على الرؤية ونظام أتمتة حالات محددة للهبوط الدقيق للكوادروتر . وتم تدقيق النتائج بإستخدام برنامجي MATLAB و VREP .

CHAPTER 1

INTRODUCTION

1.1 Unmanned Aerial Vehicles

An Unmanned Aerial Vehicle (UAV) can simply be defined as a space-traversing vehicle that flies without a human pilot or crew and can be remotely controlled or can fly autonomously. There are fixed wing and rotorcraft UAVs, classified based on the way ‘lift’ is generated. Fixed-wing UAVs are similar to passenger carrier aircraft in which the wings are used to provide lift while a jet engine provides the forward thrust for take-off. Rotorcraft UAVs on the other hand have different configurations such as: Helicopters, Cyclocopters, Autogiros, Gyrodynes, and Quadrotors. They generate lift using ‘vertical-thrusting’ rotary blades attached to the aircraft’s frame. This gives them their natural ability of vertical take-off and landing (VTOL) while fixed-wing UAVs require tilt rotor modification or thrust vectoring to achieve VTOL. The ‘Quadrotor’ belongs to the ‘rotorcraft’ family of aircrafts and a group commonly known as ‘Multi-Rotor Micro-UAVs.’

According to a 2015 press release on UAVs by ‘*Teal Group*’, a \$93billion UAV Market was forecasted for the next decade [1]. Parrot Inc., a popular manufacturer of social quadrotors sold 500,000 ‘*AR. Drone*’ units in a span of three years [2]. These depict the

enormous potential of the industry. Classification of different unmanned aerial vehicles was done in [3]. Table 1.1 summarizes different classes of UAVs based on criteria such as region of operation, altitude, payload, endurance and application.

Table 1.1: UAV Classification Based on Operational Characteristics [4].

UAV	Category	Maximum Take-Off Weight (kg)	Max Flight Altitude (km)	Endurance (hrs.)	Typical Application
Micro/Mini	Micro (MAV)	0.1	0.25	1	SC, ES, SU, IH
	Mini	< 30	0.15 - 0.3	< 2	EN, AG, SU, EV
Tactical	Close Range	150	3	2 - 4	MD, S&R
	Short range	200	3	3 - 6	MD
	Medium range	150 - 500	3 - 5	6 - 10	MD
	Long range	-	5	6 - 13	CR
	Endurance	500 – 1,500	5 - 8	12 - 24	CR
	MALE	1,000 – 15,000	5 - 8	24 - 48	WD and CR
Strategic	HALE	2,500-12,500	15 - 20	24 - 48	CR, AS, IV
Special Task	Lethal	250	3-4	3 - 4	ARSA
	Decoys	250	0.05-5	< 4	AND
	Stratospheric	-	20 – 30	> 48	-
	Exo-stratospheric	-	> 30	-	-
MALE: Medium-Altitude Long-Endurance, HALE: High-Altitude Long-Endurance. SC - Scouting, ES - Espionage, SU - Surveillance, IH - Indoor Hobbyists, EN - Entertainment, AG - Agriculture, EV - Environmental, MD - Mine Detection, S&R - Search and Rescue, CR - Communications Relay, WD - Weapons Delivery, AS - Airport Security, IV - Interception Vehicle, ARSA - Anti-radar/ship/aircraft, AND - Aerial Navigation and Naval Deception.					

1.2 Micro Aerial Vehicles

Aerial vehicles can be divided into two broad categories: ‘Lighter-Than-Air’ Vehicles and ‘Heavier-Than-Air’ Vehicles. Figure 1.1 and Table 1.2 presents the classification structure obtained from [5], based on the flying principle and type of propulsion used. One of the major advantages of VTOL and other rotorcrafts over other aircraft types is in their ability to hover at a place and low speed flight. Blimps on the other hand have the advantage of ‘auto-lift’ due to their helium filled bodies.

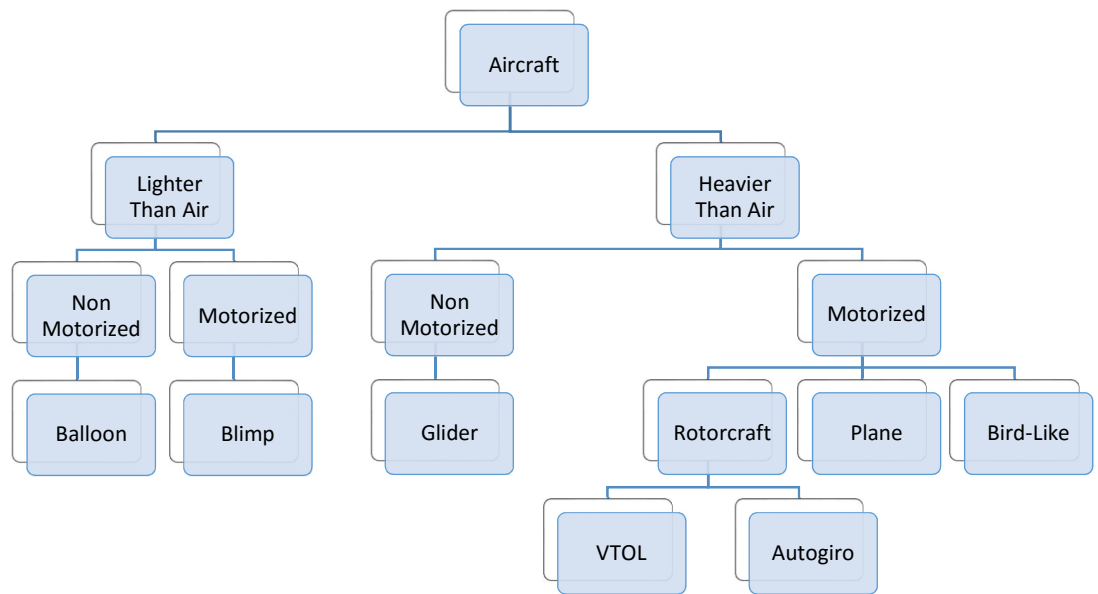


Figure 1.1: Classification of Aircrafts.

Table 1.2: Aerodynamic Configuration.

Aircraft Configuration	Advantages	Disadvantages	Description
Fixed-Wing	Simple mechanics, Silent operation	No hovering, Requires launching or runway	Engine thrust parallel to ground. Aerodynamic surfaces for control
Single rotor	Good controllability, Good manoeuvrability	Complex mechanics, Large rotor, Long tail boom	One rotor at top and one at tail like helicopter
Coaxial Rotors	Simple mechanics, Compact	Complex aerodynamics	Two rotors in counter rotation but same shaft
Tandem Rotors	Good controllability, Simple aerodynamics	Complex mechanics, Large size	Two rotors on different shafts at both ends of the vehicle
Quadrotor	Good manoeuvrability, Simple mechanics, Increased payload	High energy consumption, Large size	Four rotors in cross- frame
Blimp	Low power, Long flight operation, Auto-lift	Large size, Weak manoeuvrability	Light weight, Helium filled body
Hybrid quadrotor-blimp	Good survivability	Large size, Weak manoeuvrability	Light weight body with four rotor actuators
Bird-like	Good manoeuvrability, Compactness	Complex mechanics, Complex control	Avian body, small size, miniature actuators
Multirotor	Good manoeuvrability, Higher payload	Large size, High energy consumption	Even number of arms and rotors (> 4), Sturdy structure



Figure 1.2 Aerodynamic Configurations (Top-Left to Bottom-Right): Blimp, Coaxial, Fixed-Wing, Quadrotor, Tandem rotors, Bird-like, Single rotor, Hybrid [6].

A brief comparison of different VTOL aircraft properties according to [7] is provided in Table 1.3 wherein it is clear that the coaxial rotor and the quadrotor are the most promising MAV systems for practical duties. The next section details the quadrotor mode of operation and characteristics.

Table 1.3: Comparison of VTOL Concepts

ATTRIBUTE	A	B	C	D	E	F	G	H
Power Cost	2	2	2	2	1	4	3	3
Control Cost	1	1	4	2	3	3	2	1
Payload/Volume	2	2	4	3	3	1	2	1
Manoeuvrability	4	3	2	2	3	1	3	3
Mechanics Simplicity	1	2	3	1	4	4	1	1
Aerodynamics Complexity	1	1	1	1	4	3	1	1
Low speed flight	4	3	4	3	4	4	2	2
High speed flight	2	4	1	2	3	1	3	3
Miniaturization	2	3	4	2	3	1	2	4
Survivability	1	3	3	1	1	3	2	3
Stationary flight	4	4	4	4	4	3	1	2
Total	24	28	32	23	33	28	22	24
<i>Score: 1 = Bad, 4 = Very good; Type: A = Conventional helicopter, B = Axial rotor, C = Coaxial rotor, D = Tandem rotors, E = Quadrotor, F = Blimp, G = Bird-like, H = Insect-like</i>								

1.3 The Quadrotor

There has been a sharp rise in the development, control and applications of quadrotors in recent years and the field of aerial robotics is a fast growing one with engineers and hobbyists actively involved. Quadrotors are small-sized, simple and highly manoeuvrable vehicles. They find applications in transportation, construction, meteorology, archaeology, entertainment, security, agriculture and so on. They have been shown to perform tasks such as transporting simple loads, object assembly and construction, environmental mapping, monitoring construction sites, ball juggling and catching, flips and acrobatics [8], reconnaissance defence missions, disaster relief, providing first-aid material, air ambulance, fruit and pest monitoring.

A typical quadrotor, sometimes called ‘Quadcopter’ or ‘Quadrocopter’ and rarely called ‘Tetra-copter’ has four independently controlled fixed-pitch propellers - from which lift is generated, attached to a rigid cross airframe. Opposite propeller pairs are designed to rotate clockwise while the other two anticlockwise. This is done to cancel out the net torque generated on the center of the aircraft’s frame if all rotors were rotating in the same direction. This simple approach prevents the quadrotor from spinning on its own axis and also cancels the need for tail rotors found in helicopter systems. Moving the quadrotor is achieved by relative variation in the speeds of the rotors. The quadrotor vehicle gains motion by inducing differences in the speeds of one, two, three or all four rotors. For instance, to climb in altitude, the quadrotor has to increase the speed of its four rotors; creating thrusts that acts in opposite direction to the weight of the vehicle. To move to the right or left, the quadrotor has to be tilted at an angle and this means there is no direct control of the $x - y$ transverse dynamics. The quadrotor operates in three dimensional

space and is capable of translations and rotations along x , y , and z -axes resulting in six degrees of freedom. It is worthy of note here that with four rotors as input actuators and six degrees of freedom, the quadrotor is an under-actuated system, thus, there exists constraints in manoeuvring. One major drawback in quadrotor operation is power consumption. Hobbyist versions can remain airborne between ten and thirty minutes, however, there are new approaches to powering these vehicles with hybrid supply such as solar energy and combustion engines leading to airborne times of more than one hour.

Table 1.4: Advantages and Disadvantages of Quadrotors

	Advantages	Disadvantages
1	Simple Mechanics	Low Payload Capacity
2	Agile Manoeuvrability	Low Endurance
3	Reduced Gyroscopic effects	-
4	Easy to construct and maintain	-

1.4 Problem Statement and Application

Advances in technology has led to the widespread presence of quadrotor systems making them find applications in numerous fields. In this work, a quadrotor system is developed to deploy sensors used in oil and gas exploration to their required locations autonomously.

In the oil and gas industry, a technique known as “reflection seismology” uses established principles of seismology in the estimation of the properties of earth’s subsurface by studying reflected seismic waves. As depicted in Figure 1.3, seismic wave energy is applied to the surface of the earth using a *vibroiseis* and the reflected waves are observed by an array of sensors known as geophones strategically placed in the field. Data collected from

the geophones is thereafter analysed and processed to obtain an image of the earth's interior.

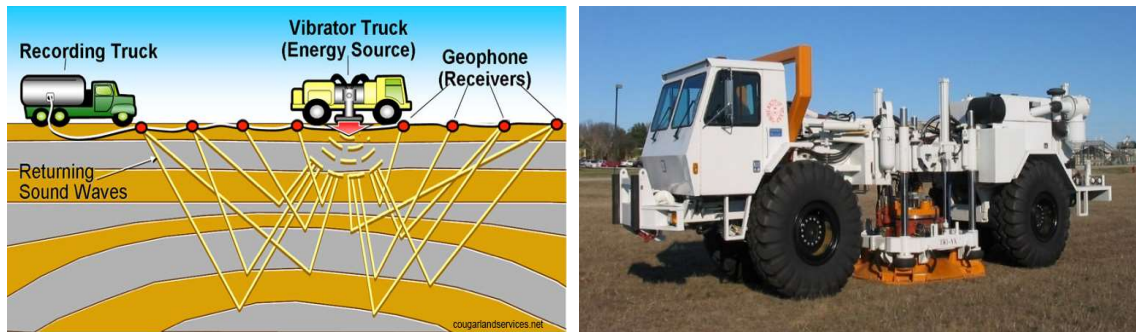


Figure 1.3: Reflection Seismology(left) [9], Vibroseis(right) [10].



Figure 1.4: Typical Middle-Eastern Field with no Foliage and Open Skies.

The geophone is a sensor that converts reflected seismic waves to electric voltage signals. The load for the quadrotor is the sensor capsule shown in Figure 1.5 and is used to house the geophone sensor for ease of transportation. It consists of two main parts; the cap – which is grabbed by the quadrotor and the body which houses the sensor.

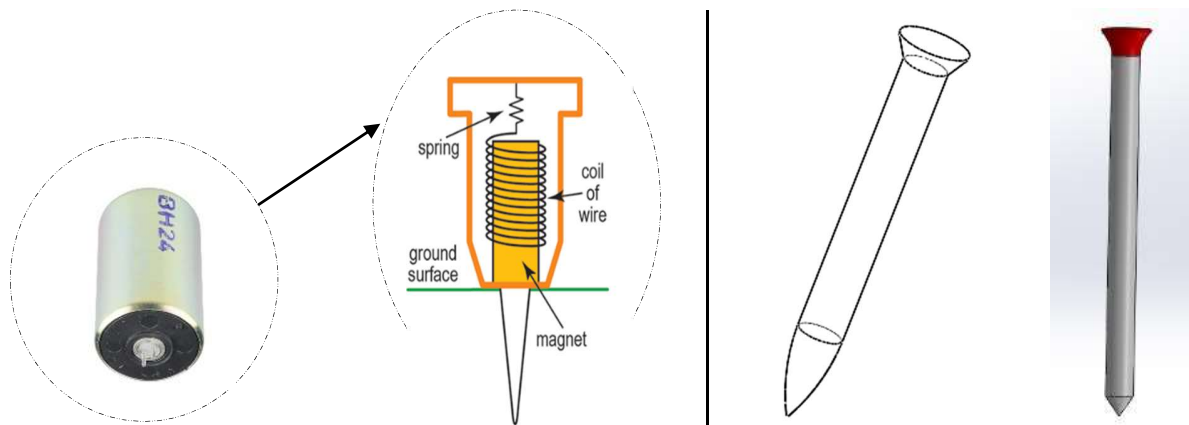


Figure 1.5 (Left to Right) Geophone Sensor and Capsule

Normal procedures involve human operators and engineers, scanning the field and inserting sensors to the ground manually or with the use of a vehicle. A new method can be employed wherein an operator is able to provide a mission plan to an autonomous sensor deployment system. This solves the problem of deploying the sensor arrays manually. The benefits of autonomous sensor deployment include: flexibility in planning sensor deployment, multiple sensor deployments at single mission, increased speed, accuracy, precision and repeatability, continuous deployment in harsh weather conditions, utilization of available technical resources, remote real-time monitoring of sensor deployment, accessibility to areas risky to humans, reduced workloads on engineers and cost savings.

In this thesis work, the core of the system described above would be treated. The main objectives are to model the quadrotor with an attached arm and sensor capsule, develop the appropriate control methods to achieve the goal and to validate the design via simulations. This is done under the assumption of an obstacle free open field as shown in Figure 1.4.

1.5 Thesis Organization

The thesis is organized as follows:

Chapter 2: Contains the literature review on subjects of interest related to the thesis work. Abstracts of previous publications on quadrotors and different control methodologies are briefly summarized.

Chapter 3: Contains development of the quadrotor system's equations of motion including the added prismatic robotic arm and disturbances.

Chapter 4: Contains the development of the quadrotor's control system.

Chapter 5: Presents a hybrid structure and a vision subsystem for the complete quadrotor system.

Chapter 6: Contains the results, simulation and visualization of the system.

Chapter 7: Summary and Conclusion of the research work.

1.6 Scope and Objectives of the Study

The main focus of this thesis is the development of an appropriate control structure that solves the quadrotor problem highlighted in Section 1.4.

1.7 Problem Formulation and Contribution

- The quadrotor-manipulator system (QM) comprises a quadrotor and a one degree of freedom prismatic robotic arm. The induced dynamics due to the coupling between the parts need to be analysed in an extended model. External disturbances are approximated using functions that mimic the true disturbance. The overall changes in mass, moment of inertia and center of mass are studied.
- A backstepping controller is designed for the quadrotor-manipulator model in the presence of aerodynamic drag, ground effect, wind disturbance and the center of mass effect.
- An active disturbance rejection controller based on linear extended state observer is developed for the quadrotor-manipulator model in the presence of aerodynamic drag, ground effect, wind disturbance and the center of mass effect.
- Trajectory optimization control based on Hermite-Simpson technique is presented for the quadrotor dynamics.
- Vision-Assist landing system is developed for precision landing of the quadrotor on sensor capsule during the retrieval phase.
- A hybrid structure based on finite state machine is used to model the different modes of operation for the autonomous system.
- Gradient Descent - Sequential Quadratic Programming optimization algorithm is used to optimize controller parameters.
- Results are validated in Matlab and VREP is used for virtualization.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

In this chapter, a comprehensive collection of existing literature on the subject of quadrotors is presented. Quadrotors as discussed earlier have attracted a large audience with interests from engineers and hobbyists. Firstly, the interesting history, developments and applications of aircraft and unmanned aerial vehicles are presented in Section 2.2. Preliminary information and brief summary of existing control techniques and their performances as reported in numerous literatures are highlighted in Section 2.3.

2.2 History, Overview and Applications

The helical screw by Leonardo da Vinci is probably the first evidenced attempt towards the creation of a flying helicopter. An art work of a flying machine with a rotating wing or air screw based on Archimedean screw was found dating back to 1486. The word ‘Helicopter’ itself was coined from the Greek words: ‘Helix’ (spiral curve) and ‘Petron’ (wings) was first used by Ponton d’Amécourt in 1863 [5]. It has been reported that Austria sent unmanned bomb-filled balloons to attack Venice in 1849 [11]. Although balloons cannot be thought of as unmanned aerial vehicles today, it was an interesting technology back then and is now regarded as the earliest use of an unmanned aerial vehicle in warfare.

The interests of the military would henceforth lead to further development of aircrafts and UAVs as would be shown in the coming paragraphs. The creation of the first unmanned helicopter in 1877 without active stabilization or steering is credited to Enrico Forlanini [12] while the Gyroplane, regarded as an early French experimental quadrotor was a rotary-wing aircraft developed by Breguet Aviation. It had an open steel frame and opposite pairs of double-bladed rotors were driven in opposing directions to cancel out net torque. On the 29th of September 1907, the Gyroplane-I achieved flight to an altitude of 0.6 meters (2ft) for a minute. It was neither steerable nor controllable. In November 1907, another French inventor in person of Paul Cornu flew his ‘Cornu helicopter’ with counter-rotating rotors and an 18kW (24hp) Antoinette Engine to a height of 0.3 meters (1ft) for a duration of 20 seconds.

The first pilotless aircrafts were built shortly after World War I. English man Archibald Montgomery Low was to build a remotely controlled aircraft for the Royal Flying Corps. Although the design of the ‘radio gear’- the major component for success was working, the noise (radio noise) produced by the rotary engines made the deployment of the aircraft unreliable [11]. The development of the automatic gyroscopic stabilizer by Dr. Cooper and Elmer few years after the first manned airplane flight was vital to the stability of aircrafts by making them fly straight and level. The automatic gyroscope was used to convert the US Navy Curtiss N-9 trainer aircraft to the first radio-controlled UAV. The first UAVs were tested in the US during WWI but never deployed in combat. In World War II however, Germany took a serious advantage and demonstrated the potential of UAVs on the battle field. Most if not all of the UAVs in deployment at this time were fixed wing aircrafts [13]. The first rotary-wing UAV, the Gyrodyne QH-50 D.A.S.H (Drone Anti-Submarine

Helicopter) is reported to have entered service in 1962 and remained in operation until 1997 [14]. These were precursor to the development of advanced systems today such as the Taranis, EQ9 Reaper drone, Predator and the compact Sky Saker H300.

Fast forward to the relatively modern era; in the 1950s, the advancement in UAV missions, propulsion and guidance systems led to the development of purpose built jet propelled target drones (Ryan Firebee UAV series). In 1960, Northrup SM-62 Snark (Figure 2.1), one of the first nuclear armed UAVs was developed. Most of the old autonomous aircraft systems suffered from reliability issues due to the absence of technology known to us today.

UAVs for commercial and civilian use has been on the rise. Petitions for the abrogation of the ban on commercial UAVs were received by the Federal Airports Authority (FAA) of the United States of America (USA) in May 2014. A press release in June 2014 mentions that the FAA had granted the first commercial small unmanned aircrafts (UAS) license for checking pipelines and similar infrastructure in Alaska [15]. Subsequently, the FAA in February 2015 released regulations and guidelines for commercial UAV systems and in December 2015, the FAA announced registration rules for every owner of a UAS weighing between 250 grams and 25 kilograms. This was regarded as necessary since sightings of UASs by pilots had doubled and posed security risks. The commercial use of UAS include aerial photography, public safety, wildlife, oil infrastructure and wind farm inspections, agriculture, mining, bridge inspection, asset tracking, mapping, environmental monitoring, situational awareness, logistics, restaurants, health care, earth study, search and rescue, wild fire suppression, law enforcement, reconnaissance and border surveillance.

In a report on Biosystems and Agricultural Engineering by the College of Agriculture, University of Kentucky [16], the UAS was described as an agricultural *tractor* which can carry some tools useful in the collection of aerial imagery with mounted cameras while more sophisticated tools enable it to perform more laborious works such as seeding, fertilizer and chemical application. A growing number of companies such as HoneyComb, PrecisionHawk and many others have invested and developed solutions for precision farming. Apart from agriculture, companies such as Airware provide solutions for the inspection of pipeline infrastructure, drilling and refining facilities and environmental management. A list of FAA approved commercial drone companies and services to be rendered can be found online [15]. The Figure 2.2 is a predicted forecast of the evolution of commercial UAS by Helen Greiner of CyPhyWorks for a 5-year period ranging 2014 to 2019 and beyond [17].



Figure 2.1: Historical Aircraft Types [14], [18], [6]

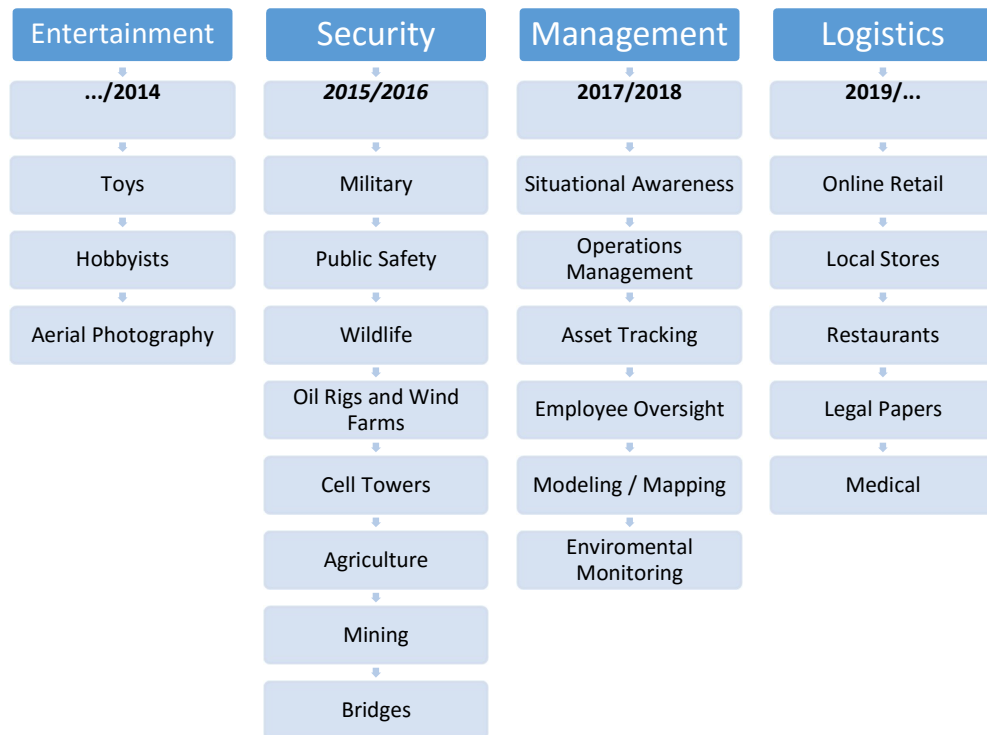


Figure 2.2: Applications of UAV, UAS and Quadrotors.

The quadrotor is arguably the most common UAS platform. Simple to build and easy to fly, the increasing success of this vehicle can be attributed to advances in computational power, extensive scientific research and development of new manufacturing techniques such as 3D printing. The earliest quadrotors include the Oemichen by Etienne Oemichen in 1920, The Convertawings by Dr. George de Bothezat and Ivan Jerome in 1956 and the Curtis Wright VZ-7 by Curtis Wright company in 1958. Numerous universities have research themes related to aerial robotics and many have set up dedicated laboratories. For instance, the Flying Machine Arena [8] is a multi-vehicle aerial robotics testbed belonging to ETH Zurich just like the Raven is to Massachusetts Institute of Technology (MIT) and the GRASP lab, belonging to University of Pennsylvania. In the paper [8], the authors described the modularity, networking and communication systems, sensing and navigation

systems, aerial vehicles and mentioned the acrobatic abilities achieved in the Arena. From [3], top universities such as MIT, Harvard, EPFL, ETH Zurich, Carnegie Mellon, Georgia Institute of Technology, Purdue, National University of Singapore, KAIST and Queensland University in addition to twenty others were mentioned as having research focus in UAV technology.

2.3 Quadrotor Control Methods

An extensive study by [3] was done to describe the recent advances and future development trends of small scale UAVs. It was suggested by [19], that the mainstream control techniques applied in UAV design are in three groups:

- Linear control with model,
- Nonlinear control with model,
- Model-free control.

In model based linear control, the quadrotor model is first linearized leading to a narrow envelope of controller performance. After linearization, a classical controller such as Proportional-Integral-Derivative (PID) is applied or optimal control methods such as Linear-Quadratic-Regulation (LQR) or Linear-Quadratic-Gaussian control [20]. The \mathcal{H}_∞ robust control technique has also been applied previously [21]. In model-based nonlinear control, a nonlinear quadrotor model is used and a nonlinear control technique having wider envelope of operation is applied. These nonlinear control techniques include: gain scheduling [22], backstepping control [23], composite nonlinear feedback [24], model predictive control [25], feedback linearization[19] and adaptive control [26]. The model-free techniques on the other hand are related to intelligent control systems. Here, flight

records are used to train the unmanned control system and thereafter, fuzzy logic [27], [28], learning control [29] or other intelligent methods are used for flight control.

In [30], a broad expose to control structures and groups the structures into three: linear flight controllers, model-based nonlinear flight controllers and learning based control methods. In the linear flight controllers, there are the PID, LQR/LQG and H_∞ . For the model based nonlinear flight controllers, there are the sliding mode control, feedback linearization, backstepping control, model predictive control, robust control, adaptive control and nested saturation technique. In learning based control methods, fuzzy logic-based (FLB) controllers, Artificial neural network (ANN)-based controllers, Reinforcement learning (RL) based controllers, Iterative learning (IL) based controllers, Memory based (MB) controllers and Brain emotional learning based intelligent controllers (BELBIC) were discussed. In another comprehensive survey by [3], the control structures were grouped into three categories; Model based linear control (comprising of PID, optimal: LQR/LQG, robust control: H_∞), model based nonlinear control (comprising adaptive, backstepping, composite nonlinear feedback (CNF), feedback linearization, gain scheduling and MPC), and model free control (comprising: Fuzzy logic and learning control).

2.3.1 Literature on Control Structures

In [31], a nonlinear controller was developed for quadrotor micro UAVs. The development was based on decomposition into a nested structure and feedback linearization. In [32], description and detailed modeling of the quadrotor system was presented including aerodynamic effects. Control of an OS4 quadrotor was achieved by Lyapunov based control design and experimental verification was performed. In [33], a vision system was

proposed as the primary sensor for control of a quadrotor. The vision system comprises a ground camera to estimate the pose of the vehicle while mode-based feedback linearization controllers and a backstepping-like control law were developed. In [34], two nonlinear control structures were proposed for attitude and altitude quadrotor control in the presence of disturbances while position control was achieved using feedback linearization. The first control structure presented was a sliding mode control structure which deals with the whole MIMO system rather than subsystems while in the second case, a block-backstepping controller was developed. The closed loop system was capable of dealing with uncertainties and tracking a trajectory in Cartesian coordinates. In [35], a hybrid control technique was developed for the stabilization of inherently unstable quadrotor vehicle. After deriving the nonlinear model, an integral backstepping controller (IBS) was developed and reinforced with a fuzzy system for tuning the coefficients of the IBS. The authors claimed the Fuzzy IBS system performed slightly better than the classical IBS controllers. In [36], image based servoing with an adaptive backstepping controller was developed. The vision system was used to provide reference velocities by using Image-Jacobian matrix and errors of image features. The overall system enables the quadrotor to achieve flight by minimizing the errors of image feature positions. In [37], an altitude P+D+DD controller for quadrotor UAVs was developed under the assumption of a well-designed attitude loop. Due to the difficulty in measuring altitude velocity, a Kalman filter based ascending/descending velocity estimators using accelerometer and barometers as sensors for velocity feedback was used. The technique was able to successfully achieve desired altitude references. In [38], a novel technique based on MIMO radial basis function – autoregressive exogenous input RBF_ARX model of a quadrotor was developed for control and LQR controller

synthesis was applied to the linearized model ARX model. The validity of the control technique was tested in real time. In [39], integral sliding mode and reinforcement learning were presented for altitude control of a quadrotor and comparison was made with established classical control approaches. The proposed controllers were shown to be capable of handling nonlinear disturbances and had better performance over classical methods. In [40], quadrotor was treated as a case study in the application of recurrent neural networks (RNNs). A modular deep RNN was developed to learn the altitude dynamics of the quadrotor. The results foretell great potential in the accurate modeling of quadrotor dynamics. In [41], a Type-2 Fuzzy system was developed for the control of a quadrotor UAV. The controller was implemented and its performance compared with both the Type-1 Fuzzy system and the PID control techniques. In [42], the authors proposed a backstepping control technique augmented with a sliding surface. The chattering effect induced by sliding mode techniques was minimized by a fuzzy system. Simulation was used to verify the performance of the controller.

2.3.2 Quadrotor Performance and Autonomy

In [43], the authors designed a constrained finite time optimal controller (CFTOC) for a quadrotor under heavy wind disturbances. The quadrotor model was linearized at different operating points and the controller developed for set-point manoeuvres. Experiments were performed to verify the attenuation of wind disturbances while performing set-point manoeuvres. In [44], the authors discussed tangent linearization and feedback linearization for the control of quadrotor. In [45], a switching model predictive attitude controller for an unmanned quadrotor helicopter was developed. The controller was developed based on a set of piecewise affine models (PWAs) with additive atmospheric disturbances. The rate

of rotation angles was used as the rule for switching between the PWA models and controllers. Experimental validation was used to prove that the control structure was able to reject induced wind disturbances. In [46], the authors presented the formulation of the quadrotor model and designed two nonlinear controllers; a feedback linearization controller and a backstepping controller based on Lyapunov functions. Simulations were used to verify the stability and tracking performance of the controller. In [47], the lightweight structure of the quadrotor was mentioned as a factor exposing the vehicle to external disturbances. In order to achieve robust trajectory tracking in the presence of wind disturbances, a feedback linearization controller and a backstepping controller was developed. Simulation was used to verify disturbance rejection by the controllers. In [48], a model aided visual-inertial fusion technique was developed for quadrotor vehicles to withstand wind disturbances. A model was developed for simultaneously estimating the vehicle pose and two components of the wind velocity vector using a monocular camera and an inertial measurement unit. Experimental verification with *vicom* motion capture system were used to prove the effectiveness of the proposed method in dealing with wind disturbances. In [49], a controller was designed for a quadrotor in the presence of external wind field disturbances. An input-output feedback linearization controller that estimates a parametric model of the wind field using a recursive Bayesian filter. The individual rotors experience different wind effects and the moment induced were compensated by the controller. In [50], a robust tracking controller based on feedback linearization and sliding mode techniques were developed for trajectory tracking and disturbance rejection. In [51], a simplified model was used to develop a linear active disturbance rejecting controller (LADRC). The LADRC was shown to be capable of estimating and compensating

generalized disturbances and reduce the closed loop system to a unity gain double integrator which is easy to implement and also robust to external disturbances.

2.3.3 Applications in Payload Delivery

In [52], both estimation and control was developed for an aerial manipulator with 2 DOF arm for payload handling. The estimated payload parameters were thereafter utilized in the flight control of the aerial manipulator. The performance of the system was compared with the adaptive sliding mode control technique. In [53], a novel hierarchical control structure was developed for an aerial vehicle with a manipulator. In the first layer, an inverse kinematics algorithm computes motion references while in the second layer, a motion control algorithm is used to track the computed motion references. In [54], the stabilization of a hexa-copter with arm was solved using nonlinear control laws. An attitude controller was first developed to stabilize the quadrotor with consideration of the manipulator arm, thereafter, a nonlinear controller was developed for reference tracking. In [55], the quadrotor system with slung load was treated as a differentially-flat hybrid system. The flatness property was used to generate trajectories that enable minimum load swing and can also cause large load swings to compliment agile manoeuvres. In [56], a control scheme was developed to achieve stability in aerial vehicles with multiple degree of freedom manipulators. The controller was developed such that the manipulator and the load itself contributed to overall stability of the system. Simulation results in Matlab was used to verify that the modified PID controller developed was capable of stabilizing a quadrotor interacting with objects. In [57], adaptive control of aerial manipulation vehicle was developed. It was mentioned that a quadrotor with robotic arm has highly nonlinear coupled dynamics which introduces additional forces and moments on the quadrotor.

Controllers for the quadrotor - Adaptive and PD and arm - PID, were developed separately. Simulation results was used to show that the strategy successfully handles hover stabilization and tracking of a trajectory.

In [58], an adaptive controller was developed to transport a cable suspended payload whose cable was modelled as serially-connected rigid links and the control objective is to transport the load while keeping the links vertical. A fixed gain nonlinear PD controller was presented and a cost adaptive controller was used to compensate for the uncertainty in the payload mass. In [59], a generalized approach using an iterative optimal control algorithm was developed to simultaneously stabilize and generate trajectory for a quadrotor with slung load. In [60], an image based visual servoing system was introduced and then implemented on a quadrotor with aerial manipulator for the purpose of stabilization and positioning tasks. The dynamics and kinematics of the quadrotor with arm was analyzed, thereafter, the cases of aerial manipulation with visual servoing, positioning using visual sensor and vision based stabilization during manipulation were developed. In [61], the control of a quadrotor equipped with a 2 DOF robot arm useful for picking up and load delivery in remote spaces was discussed. Kinematic and dynamic model of the quadrotor and robotic arm were merged into a combined system and adaptive sliding mode control was developed afterwards. In [62], a case for the control of a helicopter under external forces and interacting objects was treated. The mechanics between the helicopter and the contact surfaces were modelled as elastic couplings. Finally, it was shown that the PD and PID controllers used in free flight was capable of stabilizing the system under contact forces, displacements and stiffness. In [63], two problems were highlighted. The first being extending UAV capabilities to achieve interaction and manipulation of objects while the

second was to ensure stable and controllable flight. A passive mechanical compliance and adaptive under actuation to a gripper was developed to mitigate the need for precision flight to enable object grasping. In [64], an outdoor Octo-quad with a 7 DOF manipulator for handling payloads was developed. A backstepping controller was developed for the flight control using full dynamic model of the quadrotor while an admittance controller was presented for control of the manipulator. Table 2.1 describes the performance characteristics of different quadrotor control schemes with some descriptions obtained from [30].

Table 2.1: Properties of Different Control Schemes

Control Scheme	Advantages	Disadvantages
Intelligent PID Controller	Simple controller structure, good robustness and reliability, self-learning and adaptability.	In Fuzzy type, steady state error exists while in Neural network type, there is a low speed of convergence.
LQR Controller	Easy to design	Linear algorithm lacking robustness
H_{∞} Loop forming Controller	High robustness	Limited range of control
Sliding Mode Controller	Quick response, robustness to external disturbances	Chattering phenomenon
Feedback Linearization Controller	Provides the avenue for flexible controller design	Precise modelling is required and can't handle external disturbances
Adaptive Controller	Can compensate for effects generated by parameter change	Mismatch between theory and practice
Backstepping Controller	Ability to handle external disturbances and under-actuation, fast convergence	Lacks robustness
Robust Controller	Resists disturbances and uncertainties	Poor tracking ability

CHAPTER 3

QUADROTOR SYSTEM MODELING

3.1 Introduction

The quadrotor system comprises a quadrotor, a one degree of freedom (1 DOF) prismatic robotic arm and a sensor capsule. The model of the quadrotor has been developed previously by different authors. Articles [65], [66], have approached this by using the Euler-Lagrange formulation. In [33], [67], [68] however, the Newton-Euler equations were used. In both cases, the choice of Euler angles was arbitrary. In [33], the ZYX-Euler angles was used while [65] made use of the XYZ-Euler angles which results in simplified model equations. The quadrotor model consists of coupled nonlinear set of differential equations. A lot of work has been done in deriving accurate mathematical models that capture the realistic dynamics and aerodynamics of the quadrotor vehicle. The model presented in this chapter was obtained through study of previous literature; [33], [68], [65], [69], [70], [71]. In order to model the quadrotor system, some simplifications and assumptions have to be taken into consideration. Such assumptions include: the quadrotor has a rigid structure and symmetric body, center of gravity coincides with the body fixed frame, propellers are rigid, neglect of blade flapping phenomena and the thrust and drag forces are proportional to the

square of the propeller's speed. It is assumed that the robotic arm is light weight and preserves the quadrotor's symmetry. The attachment of a robotic arm and capsule affects the center of mass of the quadrotor system, the moment of inertia and induces dynamic interactions.

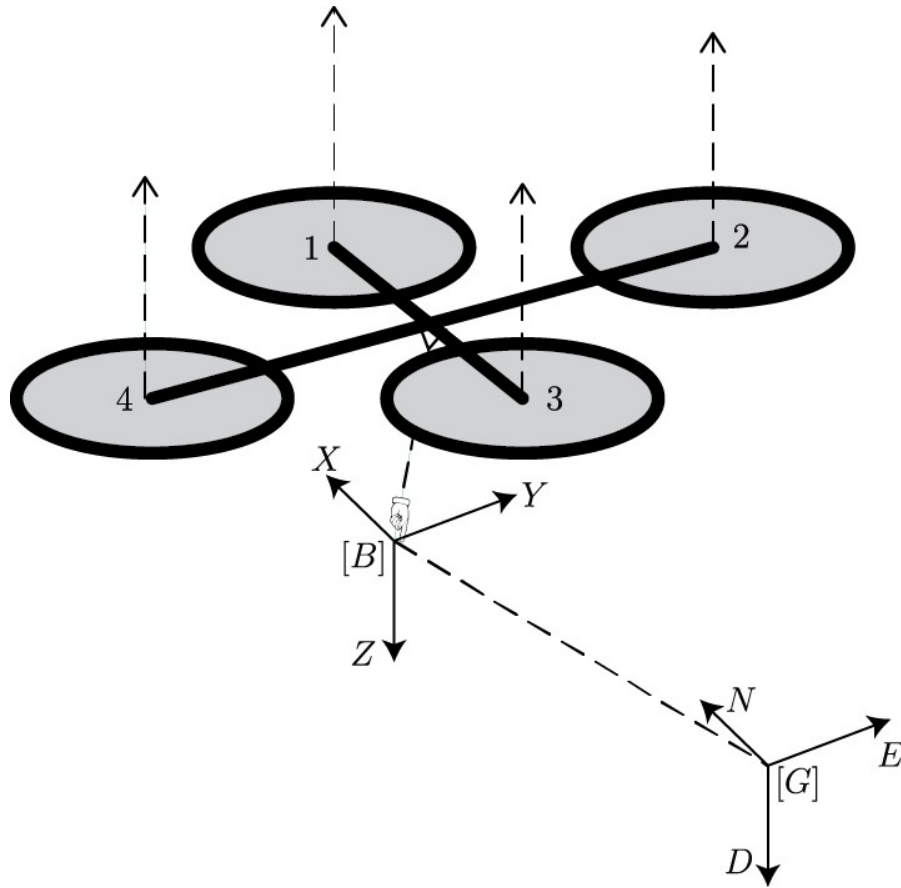


Figure 3.1: Quadrotor with Reference Frames

The axes of the quadrotor are defined in Figure 3.1 following the aerospace convention [70], the x -axis points in the forward direction (along first motor), the z -axis points downward while the y -axis is orthogonal to the x and z axes. The notation [B] represents the body frame of the quadrotor (Figure 3.1 shows [B] at an offset for clarity) while [G] is the global frame also known as earth or inertial frame. There are four rotor blades powered

by four independent motors (brushless direct current motors - BLDCs) and electronic speed control systems (ESCs). Each rotor is represented with an index 1 through 4 with corresponding thrusts: F_1 , F_2 , F_3 and F_4 . The motors are identical and symmetrically installed on the cross-axis of the quadrotor.

For the vehicle to climb in altitude, it needs to increase the thrusts to all four of its motors. The thrust produced creates an upward reaction in the $-z$ direction based on Newton's law. The maximum altitude attainable is subject to the maximum total thrusts generated by the motors. In order to move forward ($+x$) and backwards ($-x$), the quadrotor has to pitch forward or backwards about the y -axis. This is achieved by decreasing the thrust of F_1 , increasing that of F_3 and keeping F_2 and F_4 constant. Motion to the right ($+y$) or left ($-y$) directions is achieved by rolling about the x -axis; that is, increasing thrust F_4 , decreasing F_2 while keeping the rest constant. The quadrotor is an under-actuated mechanical system as there is no direct control for the $x - y$ translational dynamics.

3.2 Kinematic Model

The detailed kinematic model of the quadrotor in this section was adapted from works by [5], [71]. In order to derive the kinematic model, a body and a global coordinate frame has been defined. The frames are three dimensional representations that would be used to model the quadrotor's translational and rotational dynamics. The transformation of the body frame of the quadrotor to the global frame is obtainable by the $S\mathcal{O}3$ (3-dimensional special orthogonal group) rotation matrix; the ZYX Euler angles also known as Fick angles [33], [72] given as:

$$\mathbf{R}_{zyx} = \begin{pmatrix} \cos \theta \cos \psi & \cos \psi \sin \theta \sin \phi - \sin \psi \cos \phi & \cos \phi \sin \theta \cos \psi + \sin \phi \sin \psi \\ \sin \psi \cos \theta & \sin \phi \sin \theta \sin \psi + \cos \phi \cos \psi & \sin \psi \sin \theta \cos \phi - \cos \psi \sin \theta \\ -\sin \theta & \cos \theta \sin \phi & \cos \theta \cos \phi \end{pmatrix}$$

\mathbf{R}_{zyx} is used to switch the body [B] frame to the global frame [G] while its inverse is used to switch back to the [B] frame. The transformations are necessary since some of the states such as propeller thrusts in the dynamic model are measured with respect to the body frame while gravitational forces and quadrotor position is modelled with respect to the global frame. The properties below hold for the $\mathcal{SO}(3)$ group of transformation matrices and clearly depict the preservation of physical characteristics when used in modelling:

$$[\mathbf{R}_{zyx}]^{-1} = [\mathbf{R}_{zyx}]^T \text{ and } |\mathbf{R}_{zyx}| = 1.$$

The Inertial Measurement Unit (IMU) provides angular velocity information in the body frame. In order to relate the Euler rates $\dot{\eta} = [\dot{\phi}, \dot{\theta}, \dot{\psi}]^T$ measured in the inertial frame with the body angular rates $\omega = [p, q, r]^T$, the transformation $\omega = R_r \dot{\eta}$ is used with;

$$R_r = \begin{bmatrix} 1 & 0 & -\sin \theta \\ 0 & \cos \phi & \sin \phi \cos \theta \\ 0 & -\sin \phi & \cos \phi \cos \theta \end{bmatrix}$$

Around hover position, small angles approximation can be used to simplify the R_r matrix to an identity matrix by using the relations: $\cos(*) \approx 1$ and $\sin(*) \approx 0$.

3.3 Dynamics Modeling

The quadrotor as mentioned earlier is capable of translations and rotations. The translations are in the x , y and z directions while the rotations ϕ , θ and ψ are the roll, pitch and yaw angles. This part presents the formulation of the dynamic model of the quadrotor.

3.3.1 Rotational Dynamics

The Newton-Euler formalism below is used to model the rotational dynamics of mechanical systems.

$$J\dot{\omega} + \omega \times J\omega + M_g = M_b \quad (3.1)$$

J is quadrotor's diagonal inertia matrix, ω is the angular velocity, M_g is the gyroscopic moment due to rotor inertia and M_b is the moments acting on the quadrotor. The gyroscopic moments are defined as $M_g = \omega \times [0 \ 0 \ J_r \Omega_r]^T$. Thus, Equation 3.1 can be rewritten as:

$$J\dot{\omega} + \omega \times J\omega + \omega \times [0 \ 0 \ J_r \Omega_r]^T = M_b \quad (3.2)$$

where J_r is the rotor inertia and $\Omega_r = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4$ is the rotor's relative speed. The gyroscopic moment is a moment that attempts to align the spin axis of a rotor along its inertial z-axis.

Inertia Matrix (J)

The inertia matrix of a body is defined as J . Assuming a symmetric quadrotor body, J is diagonal. I_{xx} , I_{yy} , I_{zz} are the moments of inertia about the x , y and z axes respectively.

$$J = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}$$

Moments Acting on the Quadrotor (M_b)

The aerodynamic forces and moments produced by the rotors are two main physical effects acting on the quadrotor. The aerodynamic forces (thrusts) created by the rotation of the rotor blades creates the lifting force required for the quadrotor to climb. Consequently, the

four rotating blades induce aerodynamic moment about the center axis of the quadrotor. In order to cancel out the induced moment on the central axis of the vehicle, each pair of rotor blades is made to rotate in opposite direction to the other pair. If this is not done, the quadrotor will spin around its own axis in hover because the net induced moment is positive and greater than air resistance. The aerodynamic forces F_i and aerodynamic moments M_i for each rotor is defined as Equation 3.3:

$$\begin{aligned} F_i &= \frac{1}{2} \rho A C_T r^2 \Omega_i^2 \approx k_f \Omega_i^2 \\ M_i &= \frac{1}{2} \rho A C_D r^2 \Omega_i^2 \approx k_m \Omega_i^2 \end{aligned} \quad (3.3)$$

where ρ is the density of air, A is the total area swept by a blade, C_T and C_D are aerodynamic constants, r is the radius of the blade and Ω_i is the rotational velocity of the i 'th rotor. k_f and k_m are the aerodynamic force and aerodynamic moment constants. Since the maximum altitude of the quadrotor is constant, the density of air is approximately constant and the aerodynamic forces and moments depend on the geometry of the propeller and the air density, thus, F_i and M_i can be simplified as in the right hand side of Equation 3.3.

The total moments about the x -axis (rolling torque), the y -axis (pitching torque) and the z -axis (yawing torque) can be expressed in Equation 3.4 respectively as:

$$\begin{aligned} M_x &= -F_2 l + F_4 l = -(k_f \Omega_2^2) l + (k_f \Omega_4^2) l = k_f l (-\Omega_2^2 + \Omega_4^2) \\ M_y &= F_1 l - F_3 l = (k_f \Omega_1^2) l - (k_f \Omega_3^2) l = k_f l (\Omega_1^2 - \Omega_3^2) \\ M_z &= M_1 - M_2 + M_3 - M_4 \\ M_z &= (k_m \Omega_1^2) - (k_m \Omega_2^2) + (k_m \Omega_3^2) - (k_m \Omega_4^2) \\ M_z &= k_m (\Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2) \end{aligned} \quad (3.4)$$

and l is the distance from the rotor shaft to the central axis of the quadrotor.

3.3.2 Translational Equations of Motion

Using Newton's second law of motion, the translational equations of motion can be derived as follows in the global frame.

$$m\dot{\mathbf{r}} = [0 \ 0 \ mg]^T + \mathbf{R}_{zyx}F_b \quad (3.5)$$

$\mathbf{r} = [x \ y \ z]^T$ is the quadrotor's translational position, m is the total mass, g is the gravitational acceleration constant $9.81ms^{-2}$ and F_b is the non-gravitational forces acting on the quadrotor in the body frame.

Non-Gravitational Forces Acting on Quadrotor

When the quadrotor is in hover, the non-gravitational forces acting on it is the thrust provided by the rotation of the propellers and is proportional to the square of the angular velocity of the motors. The total gravitational forces acting on quadrotor is given as:

$$F_b = \begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -k_f(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \end{bmatrix} \quad (3.6)$$

The f_x and f_y forces are zero while the f_z is defined as above since the motors are parallel to the z -axis. f_z counters the gravitational force and compliments the weight of the quadrotor when it is in hover. In order to represent the translation in the global frame, F_b is multiplied by the \mathbf{R}_{zyx} transform.

3.4 State Space Model

In the process of designing controllers for the quadrotor, it is desirable to put the quadrotor model in a state space model. The quadrotor states are defined in the vector:

$$X = [\phi \ \dot{\phi} \ \theta \ \dot{\theta} \ \psi \ \dot{\psi} \ z \ \dot{z} \ x \ \dot{x} \ y \ \dot{y}]^T$$

$$X = [x_1 \ x_2 \ x_3 \ x_4 \ x_5 \ x_6 \ x_7 \ x_8 \ x_9 \ x_{10} \ x_{11} \ x_{12}]^T$$

A vector of control inputs U_i is defined consisting of four inputs U_1 through U_4

$$U = [U_1, U_2, U_3, U_4]^T$$

$$\begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} k_f & k_f & k_f & k_f \\ 0 & -k_f & 0 & k_f \\ k_f & 0 & -k_f & 0 \\ k_m & -k_m & k_m & -k_m \end{bmatrix} \begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} \quad (3.7)$$

U_1 is responsible for altitude and the required altitude control signal, U_2 is responsible for the roll and U_3 for pitch while U_4 is the required control signal for the desired yaw (heading) angles. k_f and k_m are constants of aerodynamic force and moments respectively.

U_1 is the resulting upwards force of the four rotors which is responsible for altitude of the quadrotor and its rate of change (z, \dot{z}). U_2 is the difference between thrusts of rotor 2 and 4 and is responsible for the roll rotation and its rate of change ($\phi, \dot{\phi}$). U_3 on the other hand, represents the difference in thrust between rotor 1 and 3 thus generating the pitch rotation and its rate of change ($\theta, \dot{\theta}$). Finally, U_4 is the difference in torques between two clockwise running motors and two anti-clockwise running motors generating yaw and subsequently its rate of change ($\psi, \dot{\psi}$). The rotor velocity relationship to the control inputs $U_{1...4}$ can be obtained by inverting Equation 3.7 to obtain the respective rotor speeds as a function of control input U_i .

$$\begin{aligned}
\Omega_1 &= \sqrt{\frac{1}{4k_f}U_1 + \frac{1}{2k_f}U_3 + \frac{1}{4k_m}U_4} & \Omega_2 &= \sqrt{\frac{1}{4k_f}U_1 - \frac{1}{2k_f}U_2 - \frac{1}{4k_m}U_4} \\
\Omega_3 &= \sqrt{\frac{1}{4k_f}U_1 - \frac{1}{2k_f}U_3 + \frac{1}{4k_m}U_4} & \Omega_4 &= \sqrt{\frac{1}{4k_f}U_1 + \frac{1}{2k_f}U_2 - \frac{1}{4k_m}U_4}
\end{aligned} \tag{3.8}$$

3.4.1 Rotational Equations of Motion

Substituting respective U_2 to U_4 values from Equation 3.7 into Equations 3.4, the equation of the total moments acting on the quadrotor becomes:

$$M_b = [U_2l \quad U_3l \quad U_4]^T \tag{3.9}$$

Substituting Equation 3.9 into the rotational equation of motion Equation 3.1 and expanding using the right cross-products and arithmetic, the following relations can be derived:

$$\begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} + \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \times \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \times \begin{bmatrix} 0 \\ 0 \\ J_r \Omega_r \end{bmatrix} = \begin{bmatrix} U_2l \\ U_3l \\ U_4 \end{bmatrix}$$

Upon expansion;

$$\begin{bmatrix} I_{xx}\ddot{\phi} \\ I_{yy}\ddot{\theta} \\ I_{zz}\ddot{\psi} \end{bmatrix} + \begin{bmatrix} \dot{\phi}I_{zz}\dot{\psi} - \dot{\psi}I_{yy}\dot{\theta} \\ \dot{\psi}I_{xx}\dot{\phi} - \dot{\phi}I_{zz}\dot{\psi} \\ \dot{\phi}I_{yy}\dot{\theta} - \dot{\theta}I_{xx}\dot{\phi} \end{bmatrix} + \begin{bmatrix} \dot{\theta}J_r\Omega_r \\ -\dot{\phi}J_r\Omega_r \\ 0 \end{bmatrix} = \begin{bmatrix} U_2l \\ U_3l \\ U_4 \end{bmatrix} \tag{3.10}$$

This can be rewritten in terms of the main states as;

$$\begin{aligned}
\ddot{\phi} &= \frac{l}{I_{xx}}U_2 - \frac{J_r}{I_{xx}}\Omega_r\dot{\theta} + \frac{I_{yy}}{I_{xx}}\dot{\psi}\dot{\theta} - \frac{I_{zz}}{I_{xx}}\dot{\theta}\dot{\psi} \\
\ddot{\theta} &= \frac{l}{I_{yy}}U_3 - \frac{J_r}{I_{yy}}\Omega_r\dot{\phi} + \frac{I_{zz}}{I_{yy}}\dot{\psi}\dot{\phi} - \frac{I_{xx}}{I_{yy}}\dot{\phi}\dot{\psi} \\
\ddot{\psi} &= \frac{1}{I_{zz}}U_4 + \frac{I_{xx}}{I_{zz}}\dot{\phi}\dot{\theta} - \frac{I_{yy}}{I_{zz}}\dot{\theta}\dot{\phi}
\end{aligned} \tag{3.11}$$

$$a_1 = \frac{I_{yy} - I_{zz}}{I_{xx}}, a_2 = \frac{J_r}{I_{xx}}, a_3 = \frac{I_{zz} - I_{xx}}{I_{yy}}, a_4 = \frac{J_r}{I_{yy}}, a_5 = \frac{I_{xx} - I_{yy}}{I_{zz}}, a_6 = \frac{l}{I_{xx}}, a_7 = \frac{l}{I_{yy}}, a_8 = \frac{1}{I_{zz}}$$

Equation 3.11 can be written in terms of state space variables as:

$$\begin{aligned}\ddot{\phi} &= a_6 U_2 - a_2 x_4 \Omega_r + a_1 x_4 x_6 \\ \ddot{\theta} &= a_7 U_3 + a_4 x_2 \Omega_r + a_3 x_2 x_6 \\ \ddot{\psi} &= a_8 U_4 + a_5 x_2 x_4\end{aligned}\tag{3.12}$$

Each of ϕ , θ and ψ states have a corresponding control input U_2 , U_3 , U_4 and thus, fully actuated.

3.4.2 Translational Equations of Motion

Using the value of U_1 from Equation 3.7, Equation 3.6 can be simply stated as:

$$F_b = [0 \quad 0 \quad -U_1]^T\tag{3.13}$$

which when combined with the translational equations of motion in Equation 3.5 and expanding, we get;

$$m \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} + \mathbf{R}_{zyx} \begin{bmatrix} 0 \\ 0 \\ -U_1 \end{bmatrix}$$

The negative U_1 sign here is due to the initially assumed direction of the z –vector altitude direction with the gravitational vector points downwards.

$$m \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} + \begin{bmatrix} -U_1(\cos \phi \sin \theta \cos \psi + \sin \phi \sin \psi) \\ -U_1(\sin \psi \sin \theta \cos \phi - \cos \psi \sin \theta) \\ -U_1(\cos \theta \cos \phi) \end{bmatrix}\tag{3.14}$$

This can be re-written in expanded form as:

$$\begin{aligned}
\ddot{x} &= -\frac{U_1}{m}(\cos \phi \sin \theta \cos \psi + \sin \phi \sin \psi) \\
\ddot{y} &= -\frac{U_1}{m}(\sin \psi \sin \theta \cos \phi - \cos \psi \sin \theta) \\
\ddot{z} &= g - \frac{U_1}{m}(\cos \phi \cos \theta)
\end{aligned} \tag{3.15}$$

Equation 3.15 can be expressed in terms of state variables defined in Section 3.4:

$$\begin{aligned}
\ddot{x} &= -\frac{U_1}{m}(\cos x_1 \sin x_3 \cos x_5 + \sin x_1 \sin x_5) \\
\ddot{y} &= -\frac{U_1}{m}(\sin x_5 \sin x_3 \cos x_1 - \cos x_5 \sin x_1) \\
\ddot{z} &= g - \frac{U_1}{m}(\cos x_1 \cos x_3)
\end{aligned} \tag{3.16}$$

3.4.3 Final State Space Representation

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \\ \dot{x}_6 \\ \dot{x}_7 \\ \dot{x}_8 \\ \dot{x}_9 \\ \dot{x}_{10} \\ \dot{x}_{11} \\ \dot{x}_{12} \end{bmatrix} = \begin{bmatrix} x_2 \\ U_2 a_6 - a_2 x_4 \Omega_r + a_1 x_4 x_6 \\ x_4 \\ U_3 a_7 + a_4 x_2 \Omega_r + a_3 x_2 x_6 \\ x_6 \\ U_4 a_8 + a_5 x_2 x_4 \\ x_8 \\ g - \frac{U_1}{m}(\cos x_3 \cos x_1) \\ x_{10} \\ -\frac{U_1}{m}(\sin x_1 \sin x_5 + \sin x_3 \cos x_1 \cos x_5) \\ x_{12} \\ \frac{U_1}{m}(-\cos x_1 \sin x_3 \sin x_5 + \sin x_1 \cos x_5) \end{bmatrix} \tag{3.17}$$

3.5 Expanded Model

Quadrotor control systems are usually developed with simplified models for the sake of controller development. It is worthy of note to pinpoint the various nonlinear aerodynamic burden on the vehicle. [73]–[75] have discussed nonlinearities in UAV vehicles. [75] described blade-flapping and thrust variation and designed a control structure based on feedback linearization to mitigate the effects. It was noted that both factors disturb proper altitude and attitude control. [74] mentions different types of drag as aerodynamic forces to be considered. Also mentioned were ground effects and vortex ring dynamics which have so far rarely been considered. [73] studied the effects of external disturbances such drag force and wind disturbance on the flight performance of a 2D quadrotor model and thereafter implemented a linear PID controller which followed a specified trajectory successfully.

In this section, the additional dynamics on the quadrotor are developed to modify the already formulated model of Section 3.4.3. As an airspace traversing vehicle, the quadrotor is subject to aerodynamic forces such as drag, ground effect and wind gusts. The inclusion of these elements makes the overall model realistic. It was already mentioned that the quadrotor system consists of a robotic arm and sensor capsule which imposes induced dynamics on the quadrotor. The resulting change in mass, change in moment of inertia, change in center of mass and dynamic coupling are discussed in subsequent sections.

3.5.1 Aerodynamic Drag Forces and Moments

Drag is a resisting force induced on the quadrotor vehicle due to friction with air. The magnitude of drag force is proportional to the velocity of the quadrotor. The drag force is approximated by the relation:

$$F_D = k_r \dot{\mathbf{r}} \quad (3.18)$$

where k_r is a constant referred to as the aerodynamic drag coefficient matrix and $\dot{\mathbf{r}}$ is the time derivative of the translation vector $\mathbf{r} = [x, y, z]$. By adding the drag term, a realistic model for the translational dynamics is given by:

$$m\ddot{\mathbf{r}} = [0 \ 0 \ mg]^T + \mathbf{R}_{zyx}F_b - F_D. \quad (3.19)$$

Air friction also introduces a drag moment on the quadrotor body. This drag moment can be expressed similarly as:

$$M_D = k_r \dot{\boldsymbol{\eta}} \quad (3.20)$$

Thus the rotational equation of motion can be expressed as:

$$J\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times J\boldsymbol{\omega} + \boldsymbol{\omega} \times [0 \ 0 \ J_r\Omega_r]^T = M_b - M_D \quad (3.21)$$

3.5.2 Ground Effect

Ground effect is an important aerodynamic effect that needs to be taken into consideration when landing quadrotor vehicles. As highlighted in [62], ground effect is a phenomenon wherein the *wake* of a helicopter rotor pushes a cushion of air resisting the helicopter's descent, thereby, creating a form of damping or cushion effect. This makes smooth descent of helicopter vehicles including quadrotors difficult to achieve. Failure to compensate for this effect will cause plunging or bouncing off the helicopter's desired landing target [63]. In [76], study was performed on canted-rotor quadrotor platforms and it was mentioned

that the ground effect can be treated as a simple spring with its spring constant proportional to the effect of ground wake. In [77] however, the difficulty in modeling ground effect analytically was clearly stated. They proposed an empirical formula based on system identification. In [78], significant work was done on ground effect for four-bladed quadrotors unlike previous papers which were based on single-rotor helicopters, canted rotor vehicles and the dragon-x8 flyer. In their work, they aimed to extend the ground effect model from helicopters to quadrotors. The model for ground effect (3.22) presented by Cheeseman and Bennett in [79] is:

$$\frac{T_{\text{ige}}}{T_{\text{oge}}} = \frac{1}{1 - \left(\frac{r}{4z_r}\right)^2} \quad (3.22)$$

where r is the radius of the rotor propeller, z_r is the vertical distance of the rotor from the ground, T_{oge} is the thrust generated by the rotor outside of ground effect. T_{ige} is the thrust generated by the same rotor in ground effect. This model was developed for single rotors but often used for quadrotors too. For a quadrotor with four rotors however, the aerodynamic interaction prompts further investigation. The authors in [78] presented a modified model (3.23):

$$\frac{T_{\text{input}}}{T_{\text{output}}} = 1 - \rho \left(\frac{r}{4z_r}\right)^2 \quad (3.23)$$

T_{input} is the input thrust command provided by the user and T_{output} is the actual thrust generated by quadrotor and $\rho = 8.6$ was determined experimentally with a quadrotor where $r = 0.1905m$ is the radius of a 15" propeller. Outside the ground effect zone,

$T_{\text{output}} = T_{\text{input}}$ while inside the ground effect zone, $T_{\text{output}} > T_{\text{input}}$. This implies that the power required to hover in ground effect zone is lower than when out of it [80]. Applying the results of [78], the relationship between quadrotor vertical thrust in and out of the ground effect zone can be expressed with the relation:

$$U_1 = U_z \left(1 - \rho \left(\frac{r}{4z_r} \right)^2 \right) \quad (3.24)$$

U_z is now regarded as a virtual input, U_1 is the actual control input, z_r is the vertical distance from rotor to ground, r is the rotor radius. A new variable G is introduced which

$$G = \frac{1}{\left(1 - \rho \left(\frac{r}{4z_r} \right)^2 \right)} \quad (3.25)$$

is a scaling factor for the altitude thrust. G comes to live at a particularly low altitude determined by r and z_r . Beyond this range, $G \approx 1$.

The effect of G can be observed in Figure 3.3 as influential at a particular range of values for $z_r < 5\text{m}$. $z_r = 0$ is avoided due to presence of landing skids (Figure 3.2) which constrains $z_r > 0$.

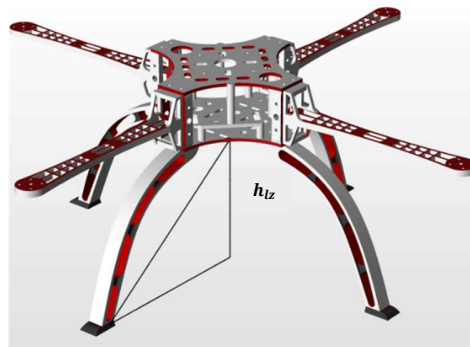


Figure 3.2: Quadrotor with Landing Skid; Height h_{LZ} (m)

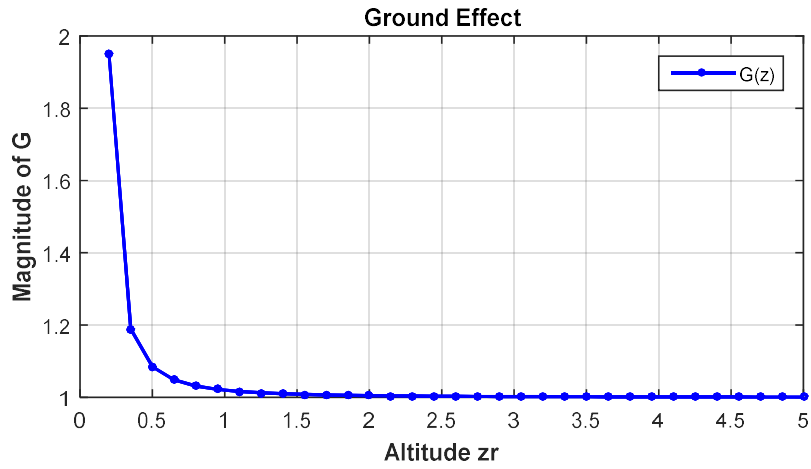


Figure 3.3: Ground Effect Variation with Altitude Decrease

Table 3.1 Influence of Ground Effect.

No.	Type	Range (z)	Reference
1	Quadrotor	$0.43 < z < 4$	[78]
2	Helicopter	$0.5 < \frac{zr}{r} < 2$	[79]
3	Dragon X8 Flyer	$0.5 < \frac{zr}{r} < 3$	[81]

In [82], the authors mentioned automatic landing as a challenging problem because of the presence of model uncertainties, external disturbances and ‘ground effect’. The authors remarked that the ground effect was more pronounced at higher altitudes than previously thought.

3.5.3 Wind Gust

In [83], the authors discussed the estimation of wind effects on quadrotor using wind tunnel tests. From the paper [84], it was mentioned that wind effect on quadrotor flight control can be significant and lead to instabilities, thereafter, a Dryden wind-gust model was developed. In order to improve the position control of the quadrotor, wind effects be modelled approximately and compensated in the controller design. The Dryden wind-gust model - a summation of sinusoidal excitations, is commonly used in this regard. In this work however, the wind disturbance is modelled using a simple sinusoidal disturbance model as presented in [85]:

$$\varpi_i(t) = \alpha_i + \beta_i \sin(nt) \quad (3.26)$$

where $\varpi_i(t)$ is a time dependent estimate of the wind vector at time t , α_i and β_i represent the corresponding wind disturbance magnitude.

3.5.4 Sensor Noise

Noise is a well-known phenomenon. It exists in different forms and magnitudes and is usually unwanted; although noise is useful in jamming radio communications networks. In [86], several noise types affecting the *MaxSonar* ultrasonic sensor were mentioned. These include: air turbulence, propeller acoustic noise, conducted electrical noise, radiated electrical noise and frame vibrations. Noise disturbance exist in GPS, accelerometers and gyroscopes. The noise signal in the sensor feedback in this work is formed as a random noise n_i with zero mean and variance of magnitude σ_i^2 .

3.5.5 Modeling 1-DOF Prismatic Arm

The attached robotic arm consists of a prismatic joint Figure 3.4 and a Barret end-effector. The prismatic joint gives the end-effector a vertical reach to the sensor capsule during retrieval phase – given that the quadrotor lands vertically above the sensor capsule with its central axis coincident to the central axis of the capsule. This implies that the landing must be done with high precision to ensure that the sensor capsule remains in the reaching-zone of the Barret end-effector. The end effector has individually actuated fingers thus capable of grasping the capsule with efficient form-closure even if the load is slightly displaced from the central axis. Figure 3.4 shows the prismatic joint with two sides. The upper side (base) is attached to the end (underneath) of the quadrotor. The lower side has a screw slot to hold the Barret end-effector in place. The quadrotor must land with a precision that ensures that the sensor capsule always remains in the end-effector's grasp-reach.

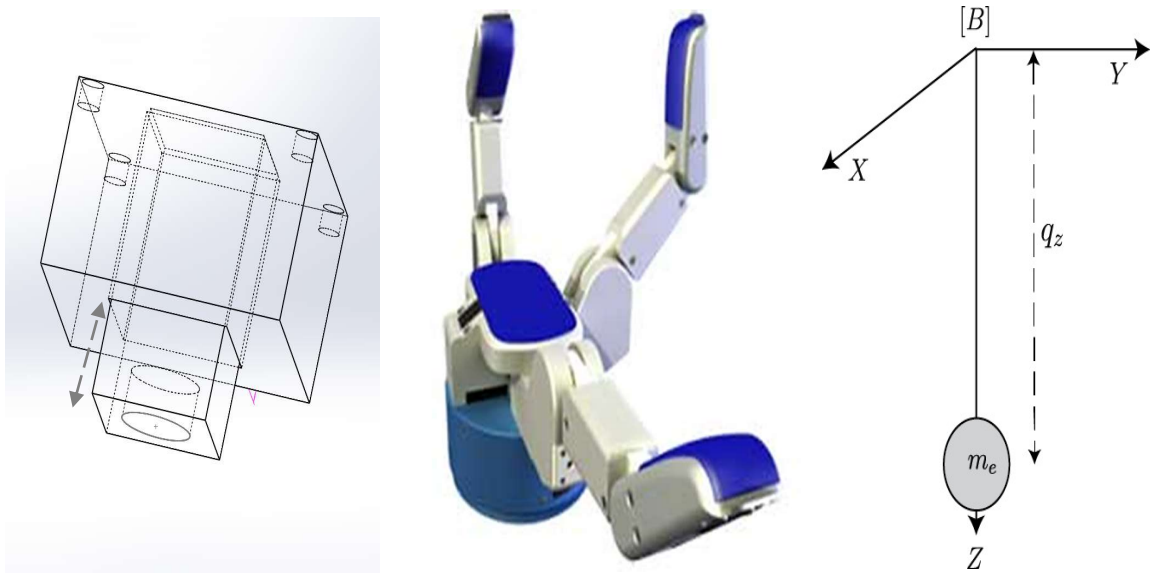


Figure 3.4 (Left to Right): Prismatic Joint, Barret Hand [87], Combined Free-Body Diagram.

Dynamic Model of Prismatic Joint

Using Lagrangian formulation, with generalized coordinates $q = q_z$, the one degree of freedom robotic arm is modelled using the free-body diagram of Figure 3.4:

Kinetic Energy $K(q, \dot{q}) = \frac{1}{2}mv^2 = \frac{1}{2}m\dot{q}^2$, Potential Energy $U(q) = mgh = mgq$

$$\mathcal{L}(q, \dot{q}) = K(q, \dot{q}) - U(q) = \frac{1}{2}m\dot{q}^2 - mgq \quad (3.27)$$

$$\frac{\partial \mathcal{L}(q, \dot{q})}{\partial q} = -mg, \quad \frac{\partial \mathcal{L}(q, \dot{q})}{\partial \dot{q}} = m\dot{q}, \quad \frac{d}{dt} \left(\frac{\partial \mathcal{L}(q, \dot{q})}{\partial \dot{q}} \right) = m\ddot{q}$$

Formulation:

$$\frac{d}{dt} \left(\frac{\partial \mathcal{L}(q, \dot{q})}{\partial \dot{q}} \right) - \frac{\partial \mathcal{L}(q, \dot{q})}{\partial q} = \tau \quad (3.28)$$

$$m\ddot{q} + mg = \tau$$

which can be represented as a simple second order system:

$$\frac{d}{dt} \begin{bmatrix} q \\ \dot{q} \end{bmatrix} = \begin{bmatrix} \dot{q} \\ \frac{\tau}{m} - g \end{bmatrix}$$

And in state space form:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ \frac{U_\tau}{m} - g \end{bmatrix} \quad (3.29)$$

x_1 is the translational state (position) of the end-effector while x_2 is the time derivative of position and $U_\tau = \tau$ is the torque-control input.

3.5.6 Change in Moment of Inertia

The contributors to the change in moment of inertia include the prismatic joint, the end effector, the sensor capsule and the landing skids. Although the model in Section 3.4.3

highlights the quadrotor's inertia matrix, this section of the report develops the inertia matrix for the quadrotor-manipulator system (combination of all parts).

According to [88] adding a quadrotor arm and payload changes three main features of the quadrotor's dynamics. These are: net mass of the vehicle, moment of inertia and centre of mass. Analysing the change in inertia properties of the quadrotor-manipulator systems is made simpler by the use of parallel axis theorem. With parallel axis theorem, one can calculate the net moment of inertia of a series of bodies aligned at the same central axis (Figure 3.5) by using simple arithmetic addition.

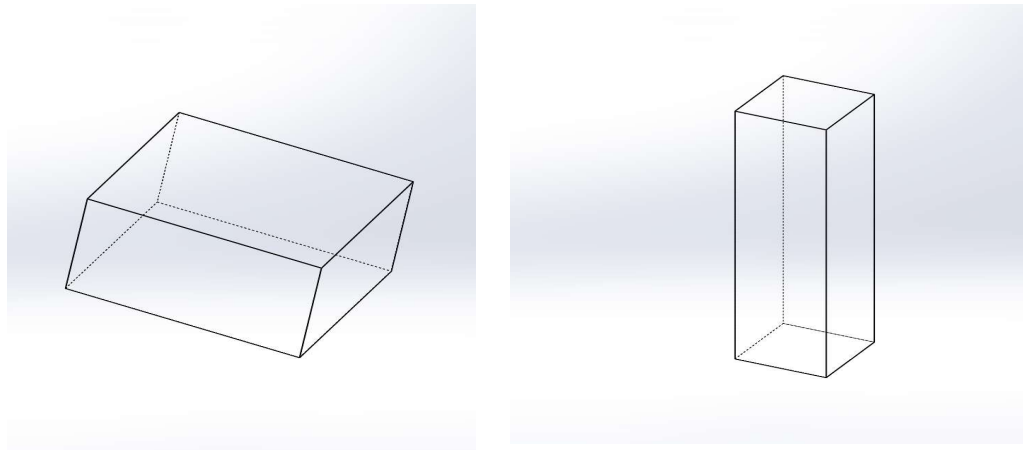


Figure 3.5: Geometric Description of Shapes under Analysis

The moment of inertia of the quadrotor cross-axis frame (cylinder - q) is represented as two cylinders with the following moment of inertia properties, dimensions: $R_q = 5 \times 10^{-3} \text{m}$, $L_q = 45 \times 10^{-3} \text{m}$ and mass $m_q = 1 \text{kg}$

$$I_{xq} = m_q \left(\frac{R_q^2}{4} + \frac{L_q^2}{12} + \frac{R_q^2}{2} \right); \quad I_{yq} = m_q \left(\frac{R_q^2}{4} + \frac{L_q^2}{12} + \frac{R_q^2}{2} \right); \quad I_{zq} = m_q \left(\frac{R_q^2}{4} + \frac{L_q^2}{12} + \frac{R_q^2}{4} + \frac{L_q^2}{12} \right).$$

The moment of inertia of the robotic arm linker-rod (cuboid - l) is represented as a cuboid with the following moment of inertia properties, dimensions: $W_l = 5 \times 10^{-3} \text{m}$, $H_l = 10 \times 10^{-3} \text{m}$, $D_l = 0$, $L_l = 5 \times 10^{-3} \text{m}$ and mass $m_l = 0.15 \text{kg}$

$$I_{xl} = m_l \left(\frac{W_l^2}{12} + \frac{H_l^2}{12} + D_l^2 \right); I_{yl} = m_l \left(\frac{L_l^2}{12} + \frac{H_l^2}{12} + D_l^2 \right); I_{zl} = m_l \left(\frac{L_l^2}{2} + \frac{W_l^2}{2} \right).$$

The end effector is modelled as a cube (c) with the following inertia properties and sides: $L_c = 10 \times 10^{-3} \text{m}$ and mass $m_c = 0.1 \text{kg}$.

$$I_{xc} = m_c \left(\frac{L_c^2}{6} + D_c^2 \right); I_{yc} = m_c \left(\frac{L_c^2}{6} + D_c^2 \right); I_{zc} = m_c \left(\frac{L_c^2}{6} + D_c^2 \right).$$

Two landing skids are modelled as two cylinders (s) with the following inertia properties and dimensions: $R_s = 3 \times 10^{-3} \text{m}$, $H_s = 20 \times 10^{-3} \text{m}$, $D_s = 10 \times 10^{-3} \text{m}$, $L_s = 22.5 \times 10^{-3} \text{m}$ and mass $m_s = 0.05 \text{kg}$ each.

$$I_{xs} = m_s \left(\frac{R_s^2}{4} + \frac{H_s^2}{12} + D_s^2 \right); I_{ys} = m_s \left(\frac{R_s^2}{4} + \frac{H_s^2}{12} + D_s^2 + L_s^2 \right); I_{zs} = m_s \left(\frac{R_s^2}{2} + L_s^2 \right).$$

The sensor capsule is modelled as a cylinder (k) with the following inertia properties and dimensions: $R_k = 3 \times 10^{-3} \text{m}$, $H_k = 10 \times 10^{-3} \text{m}$, $D_k = 37.5 + \epsilon \times 10^{-3} \text{m}$, $L_k = 0$ and mass $m_k = 0.2 \text{kg}$. ϵ is a small number representing the shift in center of mass of the sensor capsule due to the added mass of inserted geophone. $\epsilon = 0.0333$.

$$I_{xk} = m_k \left(\frac{R_k^2}{4} + \frac{H_k^2}{12} + D_k^2 \right); I_{yk} = m_k \left(\frac{R_k^2}{4} + \frac{H_k^2}{12} + D_k^2 + L_k^2 \right); I_{zk} = m_k \left(\frac{R_k^2}{2} + L_k^2 \right).$$

R_i -radius, L_i -length, W_i -width, H_i -height, D_i -displacement from the quadrotor's central axis, are the respective geometric dimensions of the quadrotor-manipulator system's parts and m_i for mass.

Thus the resulting change in the moment of inertia (combined moment of inertia) while keeping the assumption of symmetry true can be described by the Inertial tensor \bar{J} is obtainable as follows:

$$\bar{J} = \begin{pmatrix} \bar{I}_{xx} & 0 & 0 \\ 0 & \bar{I}_{yy} & 0 \\ 0 & 0 & \bar{I}_{zz} \end{pmatrix}$$

The new moments of inertia are now described by adding the calculated inertias to the primary inertia matrix J developed in Section 3.3 where, $I_{xq}, I_{yq}, I_{zq} \Leftarrow I_{xx}, I_{yy}, I_{zz}$ respectively.

$$\bar{I}_{xx} = I_{xq} + I_{xl} + I_{xc} + I_{xs} + I_{xk}$$

$$\bar{I}_{yy} = I_{yq} + I_{yl} + I_{yc} + I_{ys} + I_{yk}$$

$$\bar{I}_{zz} = I_{zq} + I_{zl} + I_{zc} + I_{zs} + I_{zk}$$

(3.30)

3.5.7 Change in Center of Mass (CoM)

Making sure the quadrotor, robotic arm, end-effector and capsule are vertically aligned on the same z-axis, the quadrotor system's Center of Mass/Gravity (x_G, y_G, z_G) can be calculated using the simple formula in Equation 3.31. The total mass of the quadrotor-manipulator system is simply the sum of the masses of the individual components.

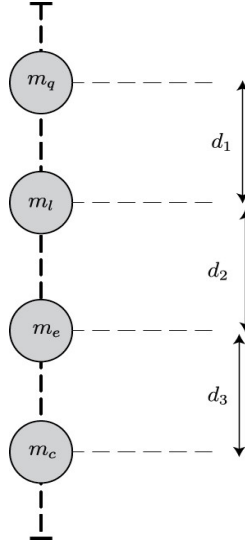


Figure 3.6: Central-Axis Mass Alignment.

The CoM of the quadrotor system in Figure 3.6 can be calculated using the formula:

$$X_{\text{CoM}[z_G]} = \frac{\sum_{i=0}^3 m_i d_i}{\sum_{i=0}^3 m_i} \quad (3.31)$$

m_q , is the quadrotor's mass, m_l , is the mass of the robotic arm, m_e , is the mass of the end-effector, m_c , is the mass of the sensor capsule. $d_0 = 0$, is taken as the reference point at the quadrotor's centre of mass with magnitude of zero. d_1, d_2, d_3 , are the distances of the respective centres of mass to d_0 .

3.5.8 Dynamic Effects of Change in CoM

The study of quadrotor-manipulator systems was done in several publications [62], [63], [76], [88]–[95]. The effect of the added manipulator and payload on the Center of Mass (CoM) of the quadrotor was studied in [96] and [97]. It was mentioned that the shift in CoM introduces additional accelerations and velocities sensed by the inertial sensors. Although mass trimming can be done to balance out the CoM, this procedure is very

tedious. Also, if sensors are not exactly aligned to the CoM, they would provide erroneous data. The effect of the added manipulator and load as described in [96] is the introduction of additional dynamics as highlighted in equations below:

Given the CoM dimensions at position x_G, y_G, z_G , the following represent the induced dynamics on the quadrotor along the translation and orientation dynamics.

Along the x -axis:

$$\mathcal{U}_x = x_G(\dot{\theta}^2 + \dot{\psi}^2) - y_G(\dot{\phi}\dot{\theta} - \ddot{\psi}) - z_G(\dot{\phi}\dot{\psi} - \ddot{\theta}) \quad (3.32a)$$

Along the y -axis:

$$\mathcal{U}_y = -x_G(\dot{\phi}\dot{\theta} + \ddot{\psi}) + y_G(\dot{\psi}^2 + \dot{\phi}^2) - z_G(\dot{\theta}\dot{\psi} - \ddot{\phi}) \quad (3.32b)$$

Along the z -axis:

$$\mathcal{U}_z = -x_G(\dot{\phi}\dot{\psi} - \ddot{\theta}) - y_G(\dot{\theta}\dot{\psi} + \ddot{\phi}) + z_G(\dot{\phi}^2 - \dot{\theta}^2) \quad (3.32c)$$

Along the ϕ -axis:

$$\mathcal{U}_\phi = mh_1; \quad h_1 = x_G(\dot{z}\dot{\psi} + \dot{y}\dot{\theta}) + y_G(\ddot{z} - \dot{x}\dot{\theta}) - z_G(\ddot{y} + \dot{x}\dot{\psi}) \quad (3.32d)$$

Along the θ -axis:

$$\mathcal{U}_\theta = mh_2; \quad h_2 = -x_G(\ddot{z} + \dot{y}\dot{\phi}) + y_G(\dot{x}\dot{\phi} + \dot{z}\dot{\psi}) + z_G(\ddot{x} - \dot{y}\dot{\psi}) \quad (3.32e)$$

Along the ψ -axis:

$$\mathcal{U}_\psi = mh_3; \quad h_3 = -x_G(\ddot{y} + \dot{z}\dot{\phi}) - y_G(\ddot{x} - \dot{z}\dot{\theta}) + z_G(\dot{y}\dot{\theta} + \dot{x}\dot{\phi}) \quad (3.32f)$$

Since the parts are assumed to be aligned along the central z-axis, $x_G = 0$ and $y_G = 0$ thus, the change in CoM is effected only on z_G (calculated using Equation 3.31) thereby simplifying the above sets of equation to:

$$\mathcal{U}_{x \Leftrightarrow x_{10}} = -z_G(\dot{\phi}\dot{\psi} - \ddot{\theta}); \mathcal{U}_{y \Leftrightarrow x_{12}} = -z_G(\dot{\theta}\dot{\psi} - \ddot{\phi}); \mathcal{U}_{z \Leftrightarrow x_8} = z_G(\dot{\phi}^2 - \dot{\theta}^2)$$

$$\mathcal{U}_{\phi \Leftrightarrow x_2} = -mz_G(\ddot{y} + \dot{x}\dot{\psi}); \mathcal{U}_{\theta \Leftrightarrow x_4} = mz_G(\ddot{x} - \dot{y}\dot{\psi}); \mathcal{U}_{\psi \Leftrightarrow x_6} = mz_G(\dot{y}\dot{\theta} + \dot{x}\dot{\phi})$$

3.6 Complete Model

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \\ \dot{x}_6 \\ \dot{x}_7 \\ \dot{x}_8 \\ \dot{x}_9 \\ \dot{x}_{10} \\ \dot{x}_{11} \\ \dot{x}_{12} \end{bmatrix} = \begin{bmatrix} x_2 \\ U_2 a_6 - a_2 x_4 \Omega_r + a_1 x_4 x_6 - \mathcal{U}_{x_2} - k_{x_2} x_2 - \overline{\omega}_{x_2} \\ x_4 \\ U_3 a_7 + a_4 x_2 \Omega_r + a_3 x_2 x_6 - \mathcal{U}_{x_4} - k_{x_4} x_4 - \overline{\omega}_{x_4} \\ x_6 \\ U_4 a_8 + a_5 x_2 x_4 + \mathcal{U}_{x_6} - k_{x_6} x_6 - \overline{\omega}_{x_6} \\ x_8 \\ g - G \frac{U_1}{m} (\cos x_3 \cos x_1) - \mathcal{U}_{x_8} + k_{x_8} x_8 - \overline{\omega}_{x_8} \\ x_{10} \\ -\frac{U_1}{m} (\sin x_1 \sin x_5 + \sin x_3 \cos x_1 \cos x_5) - \mathcal{U}_{x_{10}} - k_{x_{10}} x_{10} - \overline{\omega}_{x_{10}} \\ x_{12} \\ \frac{U_1}{m} (-\cos x_1 \sin x_3 \sin x_5 + \sin x_1 \cos x_5) - \mathcal{U}_{x_{12}} - k_{x_{12}} x_{12} - \overline{\omega}_{x_{12}} \end{bmatrix} \quad (3.33)$$

CHAPTER 4

AUTONOMOUS QUADROTOR CONTROL

4.1 Introduction

In this work, the nonlinear class of controllers would be investigated for the control of the quadrotor-manipulator system dynamics. This chapter presents the formulation and development of controllers for the purpose of deploying the sensor capsule to the target location. Usually, key performance indices (KPIs) are set in the design of controllers for dynamic plants. These include but not limited to: energy efficiency, response time, settling time, stability margins, overshoot, steady-state error, robustness and adaptation. The controllers developed are based on the assumption of available full state feedback. The following would be discussed in this chapter:

- Backstepping Control
- Active Disturbance Rejection Control
- Trajectory Optimization Control

In order to design the control system, it is essential to take a look at the quadrotor model again. The two models presented in Chapter 3 include a model for the quadrotor (Section 3.4.3) and another for the quadrotor-manipulator system (Section 3.6).

Quadrotor model without manipulator (to be referred as Plant A):

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \\ \dot{x}_6 \\ \dot{x}_7 \\ \dot{x}_8 \\ \dot{x}_9 \\ \dot{x}_{10} \\ \dot{x}_{11} \\ \dot{x}_{12} \end{bmatrix} = \begin{bmatrix} x_2 \\ U_2 a_6 - a_2 x_4 \Omega_r + a_1 x_4 x_6 \\ x_4 \\ U_3 a_7 + a_4 x_2 \Omega_r + a_3 x_2 x_6 \\ x_6 \\ U_4 a_8 + a_5 x_2 x_4 \\ x_8 \\ g - \frac{U_1}{m} (\cos x_3 \cos x_1) \\ x_{10} \\ -\frac{U_1}{m} (\sin x_1 \sin x_5 + \sin x_3 \cos x_1 \cos x_5) \\ x_{12} \\ \frac{U_1}{m} (-\cos x_1 \sin x_3 \sin x_5 + \sin x_1 \cos x_5) \end{bmatrix} \quad (4.1)$$

Quadrotor model with, aerodynamic drag, center of mass induced dynamics (CoM) effect, wind disturbance and ground effect (to be referred as Plant B):

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \\ \dot{x}_6 \\ \dot{x}_7 \\ \dot{x}_8 \\ \dot{x}_9 \\ \dot{x}_{10} \\ \dot{x}_{11} \\ \dot{x}_{12} \end{bmatrix} = \begin{bmatrix} x_2 \\ U_2 \bar{a}_6 - \bar{a}_2 x_4 \Omega_r + \bar{a}_1 x_4 x_6 - \mathcal{U}_{x_2} - k_{x_2} x_2 + \varpi_{x_2} \\ x_4 \\ U_3 \bar{a}_7 + \bar{a}_4 x_2 \Omega_r + \bar{a}_3 x_2 x_6 - \mathcal{U}_{x_4} - k_{x_4} x_4 + \varpi_{x_4} \\ x_6 \\ U_4 \bar{a}_8 + \bar{a}_5 x_2 x_4 + \mathcal{U}_{x_6} - k_{x_6} x_6 + \varpi_{x_6} \\ x_8 \\ g - G \frac{U_1}{\bar{m}} (\cos x_3 \cos x_1) - \mathcal{U}_{x_8} + k_{x_8} x_8 + \varpi_{x_8} \\ x_{10} \\ -\frac{U_1}{\bar{m}} (\sin x_1 \sin x_5 + \sin x_3 \cos x_1 \cos x_5) - \mathcal{U}_{x_{10}} - k_{x_{10}} x_{10} + \varpi_{x_{10}} \\ x_{12} \\ \frac{U_1}{\bar{m}} (-\cos x_1 \sin x_3 \sin x_5 + \sin x_1 \cos x_5) - \mathcal{U}_{x_{12}} - k_{x_{12}} x_{12} + \varpi_{x_{12}} \end{bmatrix} \quad (4.2)$$

x_1 is the ϕ -roll angle, x_3 is the θ -pitch angle, x_5 is the ψ -yaw angle, x_7 is the z -altitude state, x_9 is the x -translation, x_{11} is the y -translation. \dot{x}_i is the derivative of the state x_i representing respective state velocities.

$k_{x_i} x_i$, $i = 2,4,6,8,10,12$ represents the aerodynamic drag where x_i , $i = 2,4,6,8,10,12$ are the state velocities and k_{x_i} is the drag coefficient. m is the quadrotor's mass. ϖ_{x_i} is the sinusoidal model for wind disturbance:

$$\varpi_i(t) = \alpha_i + \beta_i \sin(nt) \quad (4.3)$$

\mathcal{U}_{x_i} represents the induced dynamic effect due to change in CoM:

$$\begin{aligned} \mathcal{U}_{x \leftrightarrow x_9} &= -z_G(x_2 x_6 - \dot{x}_4), \mathcal{U}_{y \leftrightarrow x_{11}} = -z_G(x_4 x_6 - \dot{x}_2), \mathcal{U}_{z \leftrightarrow x_7} = z_G(x_2^2 - x_4^2) \\ \mathcal{U}_{\phi \leftrightarrow x_1} &= -\bar{m} z_G(\dot{x}_{12} + x_{10} x_6), \mathcal{U}_{\theta \leftrightarrow x_3} = \bar{m} z_G(\dot{x}_{10} - x_{12} x_6), \\ \mathcal{U}_{\psi \leftrightarrow x_5} &= \bar{m} z_G(x_{12} x_4 + x_4 x_2) \end{aligned} \quad (4.4)$$

a_i, \bar{a}_i contain inertial model parameters;

$$\begin{aligned} a_1 &= \frac{I_{yy} - I_{zz}}{I_{xx}}, a_2 = \frac{J_r}{I_{xx}}, a_3 = \frac{I_{zz} - I_{xx}}{I_{yy}}, a_4 = \frac{J_r}{I_{yy}}, a_5 = \frac{I_{xx} - I_{yy}}{I_{zz}}, a_6 = \frac{l}{I_{xx}}, a_7 = \frac{l}{I_{yy}}, a_8 = \\ &\frac{1}{I_{zz}}. \text{ Similarly; } \bar{a}_1 = \frac{\bar{I}_{yy} - \bar{I}_{zz}}{\bar{I}_{xx}}, \bar{a}_2 = \frac{J_r}{\bar{I}_{xx}}, \bar{a}_3 = \frac{\bar{I}_{zz} - \bar{I}_{xx}}{\bar{I}_{yy}}, \bar{a}_4 = \frac{J_r}{\bar{I}_{yy}}, \bar{a}_5 = \frac{\bar{I}_{xx} - \bar{I}_{yy}}{\bar{I}_{zz}}, \bar{a}_6 = \frac{l}{\bar{I}_{xx}}, \\ \bar{a}_7 &= \frac{l}{\bar{I}_{yy}}, \bar{a}_8 = \frac{1}{\bar{I}_{zz}}. \end{aligned}$$

\bar{m} is the total mass of the quadrotor-manipulator system. Ω_r is the relative rotor speeds, U_i are the control inputs, G is the ground effect.

$$G = \frac{1}{\left(1 - \rho \left(\frac{r}{4z_r}\right)^2\right)} \quad (4.5)$$

$z_r \Leftrightarrow z = x_7$ is the vertical distance from rotor to ground and r is the rotor radius, ρ is a constant. The model of the prismatic arm is given by:

$$\begin{bmatrix} \dot{x}_{1p} \\ \dot{x}_{2p} \end{bmatrix} = \begin{bmatrix} x_{2p} \\ \frac{U_\tau}{m_p} - g \end{bmatrix} \quad (4.6)$$

x_{1p} is the position of the end-effector, x_{2p} is the velocity of the joint, m_p represents mass while U_τ is the torque-control input.

4.2 Backstepping Control

The backstepping controller is a very useful tool in nonlinear control design and is based on Lyapunov theory. It has been established that with the existence of a Lyapunov function whose derivative is negative definite within a domain, all state trajectories starting within the domain will converge to the same equilibrium point [98]. The backstepping method can also be modified with adaptive and robust techniques that improve the overall performance.

In order to utilize the backstepping control design technique, the plant has to be in lower triangular form. Pure-feedback form systems and strict-feedback systems can be handled perfectly [99]. Although many systems cannot be written in the lower triangular form, with transformations and neglect of some physical properties, it is possible to write such systems in feedback form. Verification via analysis or simulation is necessary to ensure that the manipulations do not affect the stability of the closed loop systems. As stated in [71], stability of a system is guaranteed if the time derivative of the associated Lyapunov function (V) is negative semi-definite.

In backstepping control design, nonlinear systems or subsystems of the form Equation (4.7) are controlled:

$$\begin{aligned}\dot{x} &= f(x) + g(x)\xi \\ \dot{\xi} &= u\end{aligned}\tag{4.7}$$

where $x \in \mathbb{R}^n$, $u \in \mathbb{R}$ is the control input, $\xi \in \mathbb{R}$, $f: \mathcal{D} \rightarrow \mathbb{R}^n$, $g: \mathcal{D} \rightarrow \mathbb{R}^n$. Assume that there exists a state feedback $\lambda(x)$, $\lambda(0) = 0$ such that the origin of the system:

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

is asymptotically stable. Also, if a Lyapunov function $V(x)$ is known such that:

$$\frac{\partial V(x)}{\partial x} [f(x) + g(x)\lambda(x)] \leq -W(x), \quad \forall x \in \mathcal{D} \quad (4.8)$$

where $W(x)$ is a positive definite bounding function; then, the controller:

$$u = \frac{\partial \lambda}{\partial x} [f(x) + g(x)\xi] - \frac{\partial V}{\partial x} g(x) - k(\xi - \lambda(x)) \quad (4.9)$$

makes the origin of the system asymptotically stable [23]. Furthermore, the derivative with respect to time of the Lyapunov function:

$$\tilde{V} = V(x) + \frac{1}{2}(\xi - \lambda(x))^2$$

along the trajectories of the system is given by:

$$\dot{\tilde{V}} \leq -W(x) - k(\xi - \lambda(x))^2 \quad (4.10)$$

As would be shown in the next section, the backstepping control design utilizes a step-by-step recursive approach which brings individual subsystems in the blocks into a stabilized form until it gets to the last subsystem where the control input U appears. U is then solved for the appropriate value that stabilizes the entire system. It is thus intuitive that the number of Lyapunov functions to be evaluated and the complexity of the developed controller is dependent on the order and complexity of the controlled plant.

In the next sections, backstepping controllers are designed for quadrotor Plants A and B.

4.2.1 Controller Development for Roll Channel

The roll controller was developed based on the roll subsystem ($\phi = x_1$):

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= x_4 x_6 a_1 - x_4 \Omega_r a_2 + U_2 a_6\end{aligned}\tag{4.11}$$

An error variable e_1 is defined based on the difference between the desired reference and actual roll $\phi = x_1$ angle:

$$e_1 = x_{1r} - x_1\tag{4.12}$$

A positive definite Lyapunov function V_1 is defined in terms of the error variable:

$$V_1 = \frac{1}{2} e_1^2\tag{4.13}$$

If the time-derivative of a Lyapunov function in error variables (V_1) is negative definite, it implies that the error would decay to zero – this is desirable in Lyapunov-based controller design. Thus the time derivative of the Lyapunov function Equation 4.13 is

$$\begin{aligned}\dot{V}_1 &= e_1 \dot{e}_1 \\ \dot{V}_1 &= e_1 (\dot{x}_{1r} - \dot{x}_1) = e_1 (\dot{x}_{1r} - x_2)\end{aligned}\tag{4.14}$$

x_2 is now regarded as a virtual input on the first subsystem coming from the second subsystem in the feedback plant. Based on Krasovski-Lasalle principle [100], let $x_2 \rightarrow \tilde{x}_2$ where \tilde{x}_2 is the required value of x_2 chosen so that $\dot{V}_1 \leq -W_1$ and W_1 is chosen as a positive definite function $W_1 = p_1 e_1^2$ with $p_1 > 0$.

Thus:

$$\dot{e}_1 = \dot{x}_{1r} - x_2$$

and therefore;

$$\dot{V}_1 = e_1 (\dot{x}_{1r} - \tilde{x}_2)\tag{4.15}$$

Choosing $\tilde{x}_2 = \dot{x}_{1r} + p_1 e_1$ guarantees that $\dot{V}_1 = -W_1$ which satisfies $\dot{V}_1 \leq -W_1$.

A new error variable e_2 is defined based on the difference between x_2 and its required value.

$$e_2 = x_2 - \tilde{x}_2$$

From which, $\tilde{x}_2 = \dot{x}_{1r} + p_1 e_1$, and substituting into the e_2 equation above;

$$e_2 = x_2 - \dot{x}_{1r} - p_1 e_1 \implies x_2 = e_2 + \dot{x}_{1r} + p_1 e_1 \quad (4.16)$$

Therefore, the Lyapunov expression in Equation 4.15 can be rewritten as:

$$\dot{V}_1 = e_1 \dot{e}_1 = e_1 (\dot{x}_{1r} - x_2)$$

$$\dot{V}_1 = e_1 \dot{e}_1 = e_1 (\dot{x}_{1r} - (e_2 + \dot{x}_{1r} + p_1 e_1))$$

$$\dot{V}_1 = -e_1 e_2 - p_1 e_1^2 \quad (4.17)$$

Thus \dot{V}_1 is always negative definite for $\forall e_1 e_2 > 0$. A new Lyapunov function V_2 is defined. V_2 incorporates the second part of the plant while making a case for the appearance of the controller input U_2 which need to be evaluated. Let:

$$V_2 = V_1 + \frac{1}{2} e_2^2 \quad (4.18)$$

As previously done, the time-derivative of V_2 is:

$$\dot{V}_2 = \dot{V}_1 + e_2 \dot{e}_2 \quad (4.19)$$

From Equation 4.16, $e_2 = x_2 - \dot{x}_{1r} - p_1 e_1$ and $\dot{e}_2 = \dot{x}_2 - \ddot{x}_{1r} - p_1 \dot{e}_1$; substituting in Equation 4.19 above:

$$\dot{V}_2 = -e_1 e_2 - p_1 e_1^2 + e_2 \dot{e}_2$$

$$\dot{V}_2 = -e_1 e_2 - p_1 e_1^2 + e_2 (\dot{x}_2 - \ddot{x}_{1r} - p_1 \dot{e}_1) \quad (4.20)$$

As previously, a second positive definite bounding function augmenting the first is defined:

$$W_2 = p_1 e_1^2 + p_2 e_2^2, \quad p_1, p_2 > 0 \quad (4.21)$$

So that $\dot{V}_2(e) \leq -W_2$ and thus:

$$\dot{V}_2 = -e_1 e_2 - p_1 e_1^2 + e_2(\dot{x}_2 - \ddot{x}_{1r} - p_1 \dot{e}_1) \leq -p_1 e_1^2 - p_2 e_2^2 \quad (4.22)$$

Now, we replace \dot{x}_2 with its appropriate value on the right hand side of Equation 4.11.

The control input U_2 appears and the Equation 4.23 is solved for the value of U_2 that makes

$\dot{V}_2 = -p_1 e_1^2 - p_2 e_2^2$ which is always negative definite for $\forall e_1 e_2 \neq 0$.

$$\dot{V}_2 = -e_1 e_2 - p_1 e_1^2 + e_2(x_4 x_6 a_1 - x_4 \Omega_r a_2 + U_2 a_6 - \ddot{x}_{1r} - p_1 \dot{e}_1) \leq -p_1 e_1^2 - p_2 e_2^2 \quad (4.23)$$

The backstepping controller input for the roll channel $x_1 = \phi$ which stabilizes the system is thus solved as:

$$U_2 = \frac{1}{a_6} (-x_4 x_6 a_1 + x_4 \Omega_r a_2 + \ddot{x}_{1r} + p_1 \dot{e}_1 + e_1 - p_2 e_2) \quad (4.24)$$

4.2.2 Controller Development for Altitude Channel

The altitude controller was developed based on the altitude subsystem ($z = x_7$):

$$\begin{aligned} \dot{x}_7 &= x_8 \\ \dot{x}_8 &= g - \frac{U_1}{m} (\cos x_3 \cos x_1) \end{aligned} \quad (4.25)$$

An error variable e_7 is defined based on the difference between the desired reference and actual altitude $z = x_7$ state.

$$e_7 = x_{7r} - x_7 \quad (4.26)$$

A positive definite Lyapunov function is defined in terms of the error variable:

$$V_7 = \frac{1}{2} e_7^2 \quad (4.27)$$

The time derivative of the Lyapunov function Equation 4.27 is:

$$\dot{V}_7 = e_7 \dot{e}_7 \quad (4.28)$$

$$\dot{V}_7 = e_7(\dot{x}_{7r} - \dot{x}_7) = e_7(\dot{x}_{7r} - \dot{x}_8)$$

x_8 is now regarded as a virtual input on the first subsystem coming from the next subsystem in the feedback plant. Let $x_8 \rightarrow \tilde{x}_8$ where \tilde{x}_8 is the required value of x_8 chosen so that $\dot{V}_7 \leq -W_7$ and W_7 is a positive definite function $W_7 = p_7 e_7^2$ with $p_7 > 0$.

Thus:

$$\dot{e}_7 = \dot{x}_{7r} - \dot{x}_8$$

and therefore;

$$\dot{V}_7 = e_7(\dot{x}_{7r} - \dot{\tilde{x}}_8) \quad (4.29)$$

Choosing $\tilde{x}_8 = \dot{x}_{7r} + p_7 e_7$ guarantees that $\dot{V}_7 = -W_7$ which satisfies $\dot{V}_7 \leq -W_7$.

A new error variable e_8 is defined based on the difference between x_8 and its required value.

$$e_8 = x_8 - \tilde{x}_8$$

where, $\tilde{x}_8 = \dot{x}_{7r} + p_7 e_7$, and substituting into the error equation above:

$$e_8 = x_8 - \dot{x}_{7r} - p_7 e_7 \implies x_8 = e_8 + \dot{x}_{7r} + p_7 e_7 \quad (4.30)$$

Therefore, the Lyapunov expression in Equation 4.29 can be written as:

$$\dot{V}_7 = e_7 \dot{e}_7 = e_7(\dot{x}_{7r} - \dot{x}_8)$$

$$\dot{V}_7 = e_7 \dot{e}_7 = e_7(\dot{x}_{7r} - (\dot{e}_8 + \dot{x}_{7r} + p_7 \dot{e}_7))$$

$$\dot{V}_7 = -e_7 \dot{e}_8 - p_7 e_7^2 \quad (4.31)$$

With \dot{V}_7 always negative definite for $\forall e_7 e_8 > 0$. A new Lyapunov function V_8 is defined. V_8 incorporates the second part of the plant while making a case for the appearance of the controller input U_1 which need to be evaluated. Let:

$$V_8 = V_7 + \frac{1}{2} e_8^2 \quad (4.32)$$

As previously done, the time-derivative of V_8 is:

$$\dot{V}_8 = \dot{V}_7 + e_8 \dot{e}_8 \quad (4.33)$$

having $e_8 = x_8 - \dot{x}_{7r} - p_7 e_7$ and $\dot{e}_8 = \dot{x}_8 - \ddot{x}_{7r} - p_7 \dot{e}_7$; substituting in the Equation 4.33:

$$\begin{aligned} \dot{V}_8 &= -e_7 e_8 - p_7 e_7^2 + e_8 \dot{e}_8 \\ \dot{V}_8 &= -e_7 e_8 - p_7 e_7^2 + e_8 (\dot{x}_8 - \ddot{x}_{7r} - p_7 \dot{e}_7) \end{aligned} \quad (4.34)$$

As previously, a second positive definite bounding function augmenting the first is defined:

$$W_8 = p_7 e_7^2 + p_8 e_8^2 \quad (4.35)$$

So that $\dot{V}_8 \leq -W_8$ and thus:

$$\dot{V}_8 = -e_7 e_8 - p_7 e_7^2 + e_8 (\dot{x}_8 - \ddot{x}_{7r} - p_7 \dot{e}_7) \leq -p_7 e_7^2 - p_8 e_8^2 \quad (4.36)$$

Now, replacing \dot{x}_8 with its appropriate value on the right hand side of Equation 4.25. The control input U_1 appears and Equation 4.36 is solved for the value of U_1 that makes

$\dot{V}_8 = -p_7 e_7^2 - p_8 e_8^2$ which is always negative definite for $\forall e_7 e_8 \neq 0$.

$$\dot{V}_8 = -e_7 e_8 - p_7 e_7^2 + e_8 \left(g - \frac{U_1}{m} (\cos x_3 \cos x_1) - \ddot{x}_{7r} - p_7 \dot{e}_7 \right) \leq -p_7 e_7^2 - p_8 e_8^2 \quad (4.37)$$

The backstepping controller input for the altitude channel $x_7 = z$ which stabilizes the system is thus solved as:

$$U_1 = \frac{m}{(\cos x_3 \cos x_1)} (g - \ddot{x}_{7r} - p_7 \dot{e}_7 - e_7 + p_8 e_8) \quad (4.38)$$

4.2.3 Controller Development for Other Channels

Following the conventions developed in Section 4.2.1 and Section 4.2.2, the controllers developed for the pitch and yaw subsystems are developed in this section. In the next section we show how to handle the control of the under-actuated part of the system dynamics using simple PD controller as a stabilizer. Consequently, we conclude that with the proposed control structure in Section 4.2.4, the controller development for subsequent controllers; Section 4.3 and Section 4.4, would be based on controlling only the ϕ, θ, ψ and z channels only.

The pitch ($\theta \rightarrow x_3$) controller U_3 is given as:

$$\begin{aligned} e_3 &= x_{3r} - x_3 \\ e_4 &= x_4 - \dot{x}_{3r} - p_3 e_3 \\ U_3 &= \frac{1}{a_7} (-a_4 x_2 \Omega_r - a_3 x_2 x_6 + \ddot{x}_{3r} + p_3 \dot{e}_3 + e_3 - p_4 e_4) \end{aligned} \quad (4.39)$$

The yaw ($\psi \rightarrow x_5$) controller U_4 is given as:

$$\begin{aligned} e_5 &= x_{5r} - x_5 \\ e_6 &= x_6 - \dot{x}_{5r} - p_5 e_5 \\ U_4 &= \frac{1}{a_8} (-a_5 x_2 x_4 + \ddot{x}_{5r} + p_5 \dot{e}_5 + e_5 - p_6 e_6) \end{aligned} \quad (4.40)$$

4.2.4 Controller Development for Under-Actuated $x - y$ Channel Dynamics

The position subsystem (x and y) represents the under-actuated part of the quadrotor dynamics. A quadrotor has to change its roll and pitch angles to move in the x and y directions. Since we have developed stable controllers for the roll and pitch subsystems (4.12) – (4.39) and our direct relation to the world is in $x - y - z$ coordinates rather than angles, it is intuitive to invert the system by augmenting the controller such that the roll and pitch angles are automatically stabilized to drive the quadrotor to the desired x and y positions. A proportional-derivative controller is proposed for this inner-loop system. The x and y controllers developed in this section would be used throughout the rest of the work. It is clear that full control of the quadrotor is achievable with the stabilization and control of the roll, pitch, yaw and altitude dynamics.

The position controller is based on the position subsystem:

$$\begin{aligned}\ddot{x} &= -\frac{U_1}{m}(\cos x_1 \sin x_3 \cos x_5 + \sin x_1 \sin x_5) \\ \ddot{y} &= -\frac{U_1}{m}(\sin x_5 \sin x_3 \cos x_1 - \cos x_5 \sin x_1)\end{aligned}\quad (4.41)$$

which is now written as:

$$\begin{aligned}\ddot{x}_r &= -\frac{U_1}{m}(\cos x_{1r} \sin x_{3r} \cos x_5 + \sin x_{1r} \sin x_5) \\ \ddot{y}_r &= -\frac{U_1}{m}(\cos x_{1r} \sin x_{3r} \sin x_5 - \sin x_{1r} \cos x_5)\end{aligned}\quad (4.42)$$

by small angles approximation $\sin(\varepsilon) = \varepsilon$ and $\cos(\varepsilon) = 1$, where ε ($\varepsilon = \phi, \theta$) is a small number (preferably $|\varepsilon| \leq 20^\circ$). Thus:

$$\begin{aligned}\ddot{x}_r &= -\frac{U_1}{m}(x_{3r} \cos x_5 + x_{1r} \sin x_5) \\ \ddot{y}_r &= -\frac{U_1}{m}(x_{3r} \sin x_5 - x_{1r} \cos x_5)\end{aligned}\quad (4.43)$$

The above can be written in matrix form:

$$\begin{aligned} \left(-\frac{m}{U_1}\right) \begin{bmatrix} \ddot{x}_r \\ \ddot{y}_r \end{bmatrix} &= \begin{bmatrix} \sin x_5 & \cos x_5 \\ -\cos x_5 & \sin x_5 \end{bmatrix} \begin{bmatrix} x_{1r} \\ x_{3r} \end{bmatrix} \\ \begin{bmatrix} -\sin x_5 & -\cos x_5 \\ \cos x_5 & -\sin x_5 \end{bmatrix} \begin{bmatrix} x_{1r} \\ x_{3r} \end{bmatrix} &= \frac{m}{U_1} \begin{bmatrix} \ddot{x}_r \\ \ddot{y}_r \end{bmatrix} \\ \begin{bmatrix} x_{1r} \\ x_{3r} \end{bmatrix} &= \begin{bmatrix} -\sin x_5 & -\cos x_5 \\ \cos x_5 & -\sin x_5 \end{bmatrix}^{-1} \frac{m}{U_1} \begin{bmatrix} \ddot{x}_r \\ \ddot{y}_r \end{bmatrix} \end{aligned} \quad (4.44)$$

and setting the controller:

$$\begin{aligned} \ddot{x}_r &= k_{px}(x_{9r} - x) + k_{dx}(\dot{x}_{9r} - \dot{x}) \\ \ddot{y}_r &= k_{py}(y_{11r} - y) + k_{dy}(\dot{y}_{11r} - \dot{y}) \end{aligned} \quad (4.45)$$

Thus, the $\phi \rightarrow x_{1r}$ and $\theta \rightarrow x_{3r}$ obtained are the stable automatic set-points that drives the quadrotor to the desired $x = x_{9r}$ and $y = y_{11}$ states.

4.2.5 Backstepping Controller for Plant B

Having designed a backstepping controller for the plain quadrotor dynamics, in this section, the same technique is applied to the quadrotor-manipulator system under the assumption that the disturbance parameters (z_G, G, ϖ) are known. The controllers for the ϕ, θ, ψ, z states are designed as follows:

4.2.5.1 Controller Development for Roll Channel

The roll controller was developed based on the roll subsystem ($\phi = x_1$):

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= U_2 \bar{a}_6 - \bar{a}_2 x_4 \Omega_r + \bar{a}_1 x_4 x_6 - \bar{U}_{x_2} - k_{x_2} x_2 + \bar{\varpi}_{x_2} \end{aligned} \quad (4.46)$$

An error variable e_1 is defined based on the difference between the desired reference and actual roll $\phi = x_1$ angle:

$$e_1 = x_{1r} - x_1 \quad (4.47)$$

A positive definite Lyapunov function is defined in terms of the error variable:

$$\bar{V}_1 = \frac{1}{2} e_1^2 \quad (4.48)$$

If the time-derivative of a Lyapunov function in error variables (V_1) is negative definite, it implies that the error would decay to zero – this is a desirable cause in Lyapunov-based controller design. Thus the time derivative of the Lyapunov function (4.48) is

$$\begin{aligned} \dot{\bar{V}}_1 &= e_1 \dot{e}_1 \\ \dot{\bar{V}}_1 &= e_1(\dot{x}_{1r} - \dot{x}_1) = e_1(\dot{x}_{1r} - x_2) \end{aligned} \quad (4.49)$$

x_2 is now regarded as a virtual input on the first subsystem coming from the next subsystem in the feedback plant. Thus choosing, $x_2 \rightarrow \tilde{x}_2$ where \tilde{x}_2 is the required value of x_2 chosen so that $\dot{\bar{V}}_1 \leq -W_1$ and W_1 is chosen as a positive definite function $W_1 = p_1 e_1^2$ with $p_1 > 0$.

Thus:

$$\dot{e}_1 = \dot{x}_{1r} - x_2$$

and therefore;

$$\dot{\bar{V}}_1 = e_1(\dot{x}_{1r} - \tilde{x}_2) \quad (4.50)$$

Choosing $\tilde{x}_2 = \dot{x}_{1r} + p_1 e_1$ guarantees that $\dot{\bar{V}}_1 = -W_1$ which satisfies $\dot{\bar{V}}_1 \leq -W_1$.

A new error variable e_2 is defined based on the difference between x_2 and its required value.

$$e_2 = x_2 - \tilde{x}_2 \quad (4.51)$$

knowing that $\tilde{x}_2 = \dot{x}_{1r} + p_1 e_1$, and substituting into the error equation above:

$$e_2 = x_2 - \dot{x}_{1r} - p_1 e_1 \implies x_2 = e_2 + \dot{x}_{1r} + p_1 e_1$$

Therefore, the Lyapunov equation in Equation (4.50) can be written as:

$$\dot{\bar{V}}_1 = e_1 \dot{e}_1 = e_1 (\dot{x}_{1r} - x_2)$$

$$\dot{\bar{V}}_1 = e_1 \dot{e}_1 = e_1 (\dot{x}_{1r} - (e_2 + \dot{x}_{1r} + p_1 e_1))$$

$$\dot{\bar{V}}_1 = -e_1 e_2 - p_1 e_1^2 \quad (4.52)$$

$\dot{\bar{V}}_1$ is always negative definite for $\forall e_1 e_2 > 0$. A new Lyapunov function \bar{V}_2 is defined. \bar{V}_2 incorporates the second part of the plant while making a case for the appearance of the control input U_2 which need to be evaluated. Let:

$$\bar{V}_2 = \bar{V}_1 + \frac{1}{2} e_2^2 \quad (4.53)$$

As previously done, the time-derivative of \bar{V}_2 is:

$$\dot{\bar{V}}_2 = \dot{\bar{V}}_1 + e_2 \dot{e}_2 \quad (4.54)$$

where $e_2 = x_2 - \dot{x}_{1r} - p_1 e_1$ and $\dot{e}_2 = \dot{x}_2 - \ddot{x}_{1r} - p_1 \dot{e}_1$; and substituting into Equation (4.54):

$$\dot{\bar{V}}_2 = -e_1 e_2 - p_1 e_1^2 + e_2 \dot{e}_2$$

$$\dot{\bar{V}}_2 = -e_1 e_2 - p_1 e_1^2 + e_2 (\dot{x}_2 - \ddot{x}_{1r} - p_1 \dot{e}_1) \quad (4.55)$$

As previously, a second positive definite bounding function augmenting the first is defined:

$$W_2 = p_1 e_1^2 + p_2 e_2^2 \quad (4.56)$$

So that $\dot{\tilde{V}}_2 \leq -W_2$ and thus:

$$\dot{\tilde{V}}_2 = -e_1 e_2 - p_1 e_1^2 + e_2 (\dot{x}_2 - \ddot{x}_{1r} - p_1 \dot{e}_1) \leq -p_1 e_1^2 - p_2 e_2^2 \quad (4.57)$$

Replacing \dot{x}_2 with it's appropriate value, the control input U_2 appears and the Equation 4.57 is solved for the value of U_2 that makes $\dot{\tilde{V}}_2 = -p_1 e_1^2 - p_2 e_2^2$ which is always negative definite for $\forall e_1 e_2 \neq 0$.

$$\begin{aligned} \dot{\tilde{V}}_2 &= -e_1 e_2 - p_1 e_1^2 \\ &+ e_2 (U_2 \bar{a}_6 - \bar{a}_2 x_4 \Omega_r + \bar{a}_1 x_4 x_6 - \mathcal{U}_{x_2} - k_{x_2} x_2 + \varpi_{x_2} - \ddot{x}_{1r} - p_1 \dot{e}_1) \\ &\leq -p_1 e_1^2 - p_2 e_2^2 \end{aligned} \quad (4.58)$$

The backstepping controller input for the roll channel $x_1 = \phi$ which stabilizes the system is thus solved as:

$$U_2 = \frac{1}{\bar{a}_6} (\bar{a}_2 x_4 \Omega_r - \bar{a}_1 x_4 x_6 + \mathcal{U}_{x_2} + k_{x_2} x_2 - \varpi_{x_2} + \ddot{x}_{1r} + p_1 \dot{e}_1 + e_1 - p_2 e_2) \quad (4.59)$$

Following the same procedure, the controllers for the Pitch and Yaw dynamics respectively is defined as follows:

$$U_3 = \frac{1}{\bar{a}_7} (-\bar{a}_4 x_2 \Omega_r - \bar{a}_3 x_2 x_6 + \mathcal{U}_{x_4} + k_{x_4} x_4 - \varpi_{x_4} + \ddot{x}_{3r} + p_3 \dot{e}_3 + e_3 - p_4 e_4) \quad (4.60)$$

$$U_4 = \frac{1}{\bar{a}_8} (\bar{a}_5 x_2 x_4 - \mathcal{U}_{x_6} + k_{x_6} x_6 - \varpi_{x_6} + \ddot{x}_{5r} + p_5 \dot{e}_5 + e_5 - p_6 e_6) \quad (4.61)$$

The Altitude controller is given as :

$$U_1 = \frac{\bar{m}}{G(\cos x_3 \cos x_1)} (g - \mathcal{U}_{x_8} + k_{x_8} x_8 + \varpi_{x_8} - \ddot{x}_{7r} - p_7 \dot{e}_7 - e_7 + p_8 e_8) \quad (4.62)$$

4.3 Active Disturbance Rejection Control

Active disturbance rejection control (ADRC) is a relatively new control technique that has gained attention of the control community. The novel ADRC technique developed by Jinqing Han was reportedly first introduced to English literature in [101]. ADRC control technique is based on a unique type of disturbance observer known as extended state observer (ESO). State observers, also known as estimators, are crucial in the design of modern control systems. They estimate the internal variables of a physical plant (sometimes immeasurable with sensors) using the plant's input and output data only. An obvious need for observers arise in flux estimation for A.C. induction motors. An interesting discussion on disturbance observers can be found in [102]. In [103], the ESO was classified as an efficient observer requiring the least amount of plant information for its operation compared to other disturbance observers.

Obtaining accurate mathematical models for highly complex and nonlinear systems is usually a challenge. This is partly because dynamics could be coupled, nonlinear and or even stochastic. More so, the attenuation, compensation and elimination of disturbances from physical system is usually a key criterion in control design. The main operation of the ADRC is to estimate (using the ESO) and compensate for the effects of unknown dynamics and disturbances. This transforms a system normally in control-affine form to a simple linear-feedback double-integrator. Due to the ability to estimate overall disturbance and uncertainty in plant model using just input and output data signals, the ADRC is capable of compensating for large amounts of intrinsic and extrinsic uncertainties.

Stability analysis of the ADRC controller was studied by Qing Zheng in [103] and Wankun Zhou et al. in [104]. The analytical results presented asserts the stability of the system which enhances confidence for its real-time implementation. The ADRC has been applied in robot motion control [105], industrial heater [106], boiler units [107], electrical voltage regulation [108], marine steam turbine [109] and autonomous aerial vehicles [110]. In [103], the stability analysis and performance analytics of ADRC was developed. The following was concluded:

- Estimation error converges to the origin asymptotically when the model of the plant is given.
- Estimation error is bounded and the error upper bound monotonously decreases with bandwidth of the observer when the plant model is mostly unknown.
- Asymptotic stability is achieved when the plant dynamics is completely known.
- The closed-loop tracking error upper bounds monotonously decrease with the bandwidth of the controller and the observer.

Figure 4.1 shows the architecture of the ADRC controller.

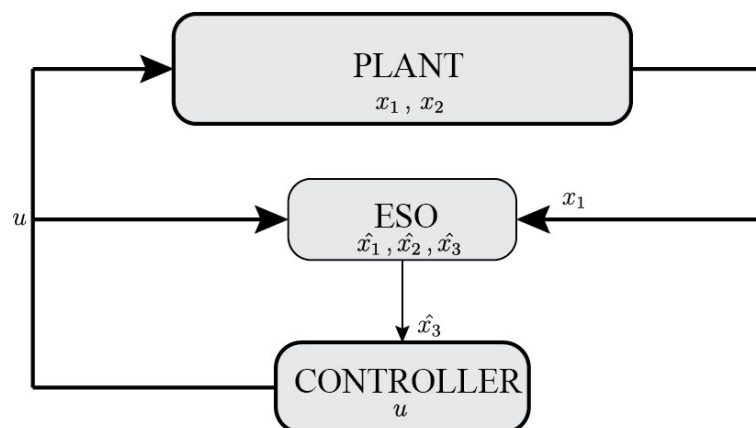


Figure 4.1: Controller Architecture for the ADRC controller.

4.3.1 Controller Formulation

The most important subsystem of the ADRC is the ESO. The formulation of the ADRC as obtained in [111], [103] and [105] is treated in this section. Given a second order single-input-single-output (SISO) system affine in control:

$$\ddot{x} = f(x, \dot{x}, \omega(t), t) + bu \quad (4.63)$$

In state space form:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= f(x, \dot{x}, \omega(t), t) + bu \\ y &= x_1 \end{aligned} \quad (4.64)$$

where $y \in \mathbb{R}$ is the plant output, measurable and to be controlled, $u \in \mathbb{R}$ is the input, and $f(x, \dot{x}, \omega(t), t) = F(t)$ is a function of the plant's states: $x_i \in \mathbb{R}$, external disturbances ω , and time t . $F(t)$ is regarded to as the total disturbance and assumed to be differentiable. The goal is to make y track a desired signal or reference by manipulating u . Taking $F(t)$ as an additional state variable $x_3 = F(t)$ and denoting $\dot{F}(t) = \bar{G}(t)$, with $\bar{G}(t)$ unknown, the original plant in Equation 4.64 is now described in extended or augmented form Equation 4.65 and the ESO is presented in Equation 4.66:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= x_3 + bu \\ \dot{x}_3 &= \bar{G}(t) \end{aligned} \quad (4.65)$$

$$\begin{aligned} \dot{\hat{x}}_1 &= \hat{x}_2 + p_1(x_1 - \hat{x}_1) \\ \dot{\hat{x}}_2 &= \hat{x}_3 + p_2(x_1 - \hat{x}_1) + \hat{b}u \\ \dot{\hat{x}}_3 &= p_3(x_1 - \hat{x}_1) \end{aligned} \quad (4.66)$$

\hat{x}_1 is the estimate of x_1 , \hat{x}_2 is the estimate of x_2 , \hat{x}_3 is the estimate of $F(t)$. $p_{i:1,2,3}$ are the observer gains (bandwidth) to be tuned. The observer gains can be tuned manually such that the characteristic polynomial $s^3 + p_1s^2 + p_2s + p_3$ is Hurwitz. As expected, larger observer gains result in more accurate state estimations. This however comes at the detriment of increased noise sensitivity. Thus, the appropriate observer bandwidth should be selected as a compromise between the tracking performance and noise tolerance. In summary, the stability margins, performance characteristics and noise sensitivity of the system can be put in check by an optimization approach to tuning the gains. Note that the ground effect factor G is not the same as the extended state derivative \bar{G} .

Using the control structure Equation 4.67 below in Equation 4.64 eliminates the total disturbance from the plant

$$u = \frac{u_0 - F(t)}{\hat{b}} \quad (4.67)$$

thereby reducing it to a simple linear feedback double-integrator; Equation 4.68.

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= u_0 \\ \text{Desired output: } y &= x_1 \end{aligned} \quad (4.68)$$

It is now possible to control the system described by Equation 4.64 via Equation 4.67 using linear control theory for u_0 . In this work, the PD controller was employed at this stage. The backstepping controller was also explored.

The closed loop system when Equation 4.67 is used in Equation 4.63 is therefore:

$$\ddot{x} = (f(x, \dot{x}, \omega(t), t) - F(t)) + u_0 \quad (4.69)$$

For a properly designed and tuned ESO,

$$f(x, \dot{x}, \omega(t), t) - F(t) \approx 0. \quad (4.70)$$

The closer the left hand side of Equation 4.70 is to zero, the closer the plant Equation 4.63 is to an ideal double integrator Equation 4.68. This closeness can be enforced by optimizing the ESO gain parameters. As seen from the formulations, the ADRC requires very little information about quadrotor parameters.

4.3.1.1 Estimation Error Dynamics

Let the estimation error be defined as $\epsilon = (x_1 - \hat{x}_1) = e_1$; The estimation error dynamics of the system can be obtained by subtracting the estimated states dynamics in Equation 4.65 from the actual states dynamics Equation 4.66. The estimation error \dot{e}_i is thus:

$$\begin{aligned} \dot{e}_1 &= \dot{x}_1 - \dot{\hat{x}}_1 = x_2 - \hat{x}_2 - p_1\epsilon = e_2 - p_1\epsilon \\ \dot{e}_2 &= \dot{x}_2 - \dot{\hat{x}}_2 = x_3 - \hat{x}_3 - p_1\epsilon = e_3 - p_2\epsilon \\ \dot{e}_3 &= \dot{x}_3 - \dot{\hat{x}}_3 = \ddot{G} - p_3\epsilon \end{aligned} \quad (4.71)$$

The estimation error dynamics is thus:

$$\begin{aligned}
 \dot{e}_3 &= \dot{\tilde{G}} - p_3 \epsilon \\
 \dot{\tilde{G}} - p_3 \epsilon &= \ddot{e}_2 + p_2 \dot{e}_2 \\
 \dot{\tilde{G}} - p_3 \epsilon &= \ddot{e}_1 + p_1 \dot{e}_1 + p_2 \dot{e}_2 \\
 \dot{\tilde{G}} &= \ddot{e}_1 + p_1 \dot{e}_1 + p_2 \dot{e}_2 + p_3 \epsilon \\
 \dot{\tilde{G}} &= \ddot{\epsilon} + p_1 \dot{\epsilon} + p_2 \dot{\epsilon} + p_3 \epsilon
 \end{aligned} \tag{4.72}$$

The above equation shows how the estimation dynamics depends on observer gains p_1, p_2, p_3 .

4.3.1.2 Modification for Noisy Feedback

Choosing small observer gains might lead to loss of stability whereas gains that are too large amplify noise. Derivative action is also known to amplify noise. Using noisy signals in feedback is capable of destabilizing a plant especially when the noise gets over-amplified. It is also very likely for this to happen since ADRC systems are usually built with high observer gains. The solution to this problem as mentioned in [112] is the use of integrators. Integrators are known to attenuate noise, remove offsets or steady state error but also reduce plant response speed. In the modified ADRC, an extra fictitious state variable is introduced to the ESO for the purpose of noise decoupling. Consider the output of a plant corrupted by noise so that:

$$y = x_1 + n_z \tag{4.73}$$

where n_z is the unwanted noise that corrupts the fed-back sensor signal. In order to deal with noisy measurements, the ESO structure should be augmented with a new integrator fictitious state variable, defined as:

$$x_0 = \int_0^t y(\tau) d\tau = \int_0^t [x_1(\tau) + n_z(\tau)] d\tau \quad (4.74)$$

The combined plant and ESO dynamics for the modified ADRC is given below:

$$\begin{aligned} \dot{x}_0 &= x_1 + n_z & \dot{\hat{x}}_0 &= \hat{x}_1 + p_1(x_0 - \hat{x}_0) \\ \dot{x}_1 &= x_2 & \dot{\hat{x}}_1 &= \hat{x}_2 + p_2(x_0 - \hat{x}_0) \\ \dot{x}_2 &= x_3 + bu & \dot{\hat{x}}_2 &= \hat{x}_3 + p_3(x_0 - \hat{x}_0) + \hat{b}u \\ \dot{x}_3 &= \bar{G}(t) & \dot{\hat{x}}_3 &= \quad + p_4(x_0 - \hat{x}_0) \end{aligned} \quad \begin{array}{l} (4.75) \\ (4.76) \end{array}$$

4.3.1.3 Estimation Error Dynamics with Noisy Feedback

Given the modification in Section 4.3.1.2 above, the estimation error dynamics is derived in this section. Given the definitions in Equations 4.77;

$$\begin{aligned} \epsilon_0 &= x_0 - \hat{x}_0 \\ e_1 &= x_1 - \hat{x}_1 \end{aligned} \quad (4.77)$$

The error dynamics is thus:

$$\begin{aligned}
\dot{e}_0 &= \dot{\hat{x}}_0 - \dot{x}_0 = x_1 + n_z - \hat{x}_1 - p_0 \epsilon_0 = e_1 - p_0 \epsilon_0 + n_z \\
\dot{e}_1 &= \dot{\hat{x}}_1 - \dot{x}_1 = x_2 - \hat{x}_2 - p_1 \epsilon_0 = e_2 - p_1 \epsilon_0 \\
\dot{e}_2 &= \dot{\hat{x}}_2 - \dot{x}_2 = x_3 - \hat{x}_3 - p_2 \epsilon_0 = e_3 - p_2 \epsilon_0 \\
\dot{e}_3 &= \dot{\hat{x}}_3 - \dot{x}_3 = \dot{G} - p_3 \epsilon_0 = \dot{G} - p_3 \epsilon_0
\end{aligned}$$

which implies:

$$\begin{aligned}
\dot{e}_3 &= \dot{G} - p_3 \epsilon_0 \\
\ddot{e}_2 + p_2 \dot{\epsilon}_0 &= \dot{G} - p_3 \epsilon_0 \\
\ddot{e}_1 + p_1 \ddot{\epsilon}_0 + p_2 \dot{\epsilon}_0 &= \dot{G} - p_3 \epsilon_0 \\
e_0^{(4)} + p_0 \ddot{\epsilon}_0 - \ddot{n}_z + p_1 \ddot{\epsilon}_0 + p_2 \dot{\epsilon}_0 &= \dot{G} - p_3 \epsilon_0 \\
\epsilon_0^{(4)} + p_0 \ddot{\epsilon}_0 + p_1 \ddot{\epsilon}_0 + p_2 \dot{\epsilon}_0 + p_3 \epsilon_0 &= \dot{G} + \ddot{n}_z \tag{4.78}
\end{aligned}$$

$(*)^{(r)}$ is the r 'th derivative of $(*)$. It is clear from the above Equation 4.78 that the observer gains do not amplify the noise n_z and thus, the observer gains $(p_i, i = 0,1,2,3)$ are decoupled from the noise characteristics.

4.3.2 Controller Design for Roll Dynamics

The roll controller was developed based on the roll subsystem ($\phi = x_1$):

$$\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= U_2 \bar{a}_6 - \bar{a}_2 x_4 \Omega_r + \bar{a}_1 x_4 x_6 - \mathcal{U}_{x_2} - k_{x_2} x_2 + \varpi_{x_2} \tag{4.79}
\end{aligned}$$

The ADRC controller is formulated:

$$\begin{aligned}
 \dot{x}_0 &= x_1 + n_z & \dot{\hat{x}}_0 &= \hat{x}_1 + p_1(x_0 - \hat{x}_0) \\
 \dot{x}_1 &= x_2 & \dot{\hat{x}}_1 &= \hat{x}_2 + p_2(x_0 - \hat{x}_0) \\
 \dot{x}_2 &= x_3 + bu & \dot{\hat{x}}_2 &= \hat{x}_3 + p_3(x_0 - \hat{x}_0) + \hat{b}u \\
 \dot{x}_3 &= \bar{G}(t) & \dot{\hat{x}}_3 &= \hat{x}_3 + p_4(x_0 - \hat{x}_0)
 \end{aligned}
 \tag{4.80} \tag{4.81}$$

where x_3 represents the total disturbance $\{-\bar{a}_2 x_4 \Omega_r + \bar{a}_1 x_4 x_6 - \bar{U}_{x_2} - k_{x_2} x_2 + \bar{\omega}_{x_2}\}$ with b as a gain on the input signal corresponding to \bar{a}_6 and $u = U_2$. x_0 the integral of the sensor feedback signal corrupted with noise. Two cases; $n_z = 0$ and $n_z \neq 0$ would be considered. $\hat{x}_i, i = 0,1,2,3$ are the estimates of x_i .

As mentioned in Section 4.3.1, the control structure Equation 4.67, reduces the plant Equation 4.79 to a double integrator of Equation 4.68 which can now be controlled using a simple PD controller. Defining the roll error as Equation 4.82 where x_{1r} is the desired roll angle.

$$e = x_{1r} - x_1, \tag{4.82}$$

The PD controller is given as:

$$u_0 = k_1 e + k_2 \dot{e} \tag{4.83}$$

4.3.3 Controller Design for Altitude Dynamics (Landing)

$$\begin{aligned}\dot{x}_7 &= x_8 \\ \dot{x}_8 &= g - G \frac{U_1}{m} (\cos x_3 \cos x_1) - \mathcal{U}_{x_8} + k_{x_8} x_8 + \overline{\omega}_{x_8}\end{aligned}\quad (4.84)$$

Since the ADRC controller has been developed, it is applied to the quadrotor landing problem. The double integrator equivalent of the nonlinear system Equation 4.84 can be controlled using different techniques. It is clear that G is coupled with the control input U_1 . In this case, the ESO parameter \hat{b} is crucial for compensation. Thus $\hat{b} = G$. The Quadrotor starts from an altitude of z (meters) and landing procedure is initiated. The landing command could be initiated by a set-point reference. This method however leads to abrupt control actions during flight. As a result, the function in Equation 4.85 is developed to generate smooth landing profiles.

$$z_L = \alpha - \left(\frac{\alpha}{1 + e^{-\beta t + c} + \gamma} \right) \quad (4.85)$$

Here, α is the initial height during hover, β is the rate of descent, c determines how long it stays in hover before landing (time delay before landing action) and γ is a constant tuned to set the desired height of the quadrotor above ground after landing. Landing skid height - h_{Lz} (Figure 4.2) should be considered when tuning γ . In this work, $h_{Lz} = 0.15\text{m}$ which implies that the quadrotor needs to compensate for the large ground effects at such low heights. The conventions for these parameters should be followed. $\alpha, \gamma > 0$, preferably $\beta = \frac{\alpha}{2}$ for $\alpha < 20$ and $c \geq \beta$.

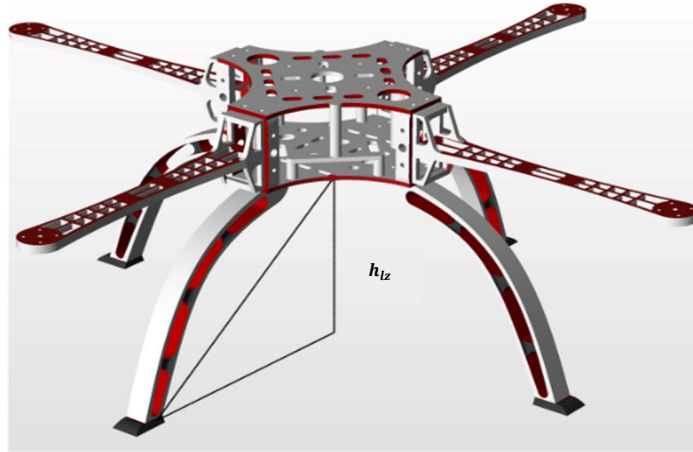


Figure 4.2: Quadrotor with Landing Skid of height h_{lz}

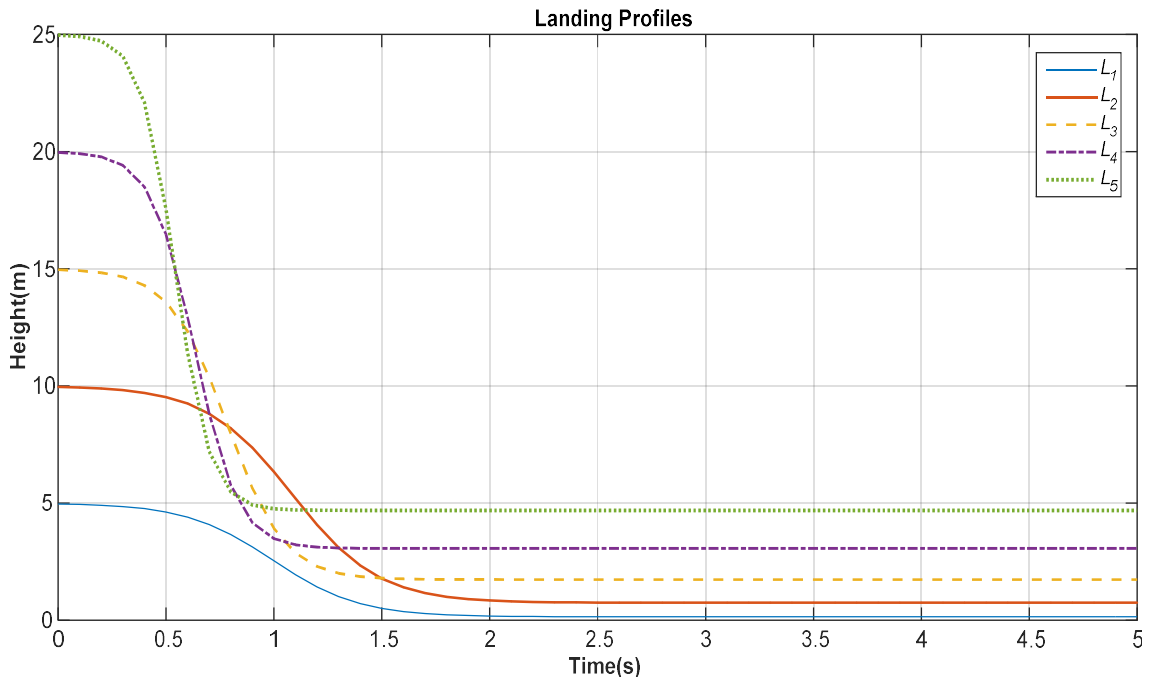


Figure 4.3: Quadrotor Landing Profiles

The respective landing profiles in Figure 4.3 were generated using the following values: $L_1: \alpha = 5, \beta = 5, c = 5, \gamma = 0.031$. $L_2: \alpha = 10, \beta = 5, c = 1, \gamma = 0.1$. $L_3: \alpha = 15, \beta = 7.5, c = 1.5, \gamma = 0.15$. $L_4: \alpha = 20, \beta = 10, c = 2, \gamma = 0.2$. $L_5: \alpha = 25, \beta = 12.5, c = 2.5, \gamma = 0.25$. $L_i \Leftrightarrow x_{7r}$ represents the tracked landing profile for the altitude dynamics.

The ADRC formulation of Equations 4.65 to 4.68 and Equation 4.83 are applied to the pitch and yaw subsystems.

4.4 Numerical Optimal Control

Optimal control is the process of determining control and state trajectories for a dynamic system, over a period of time in order to optimize a given performance index. Historically, optimal control is an extension of the calculus of variations as related to the famous ‘*brachistochrone*’ problem, [113]. Due to increase in variables and complexity, optimal control problems can no longer be solved analytically and, consequently, numerical methods are required. According to [114], three methodologies for solving optimal control problems were highlighted. These include: Dynamic programming, Indirect methods and Direct methods.

Dynamic programming methods give an excellent solution to optimal control problem. They work well for low order systems but too cumbersome for high order systems due to the required discretization of full state. Indirect methods are said to be numerically unstable, difficult to implement and initialize. Indirect methods involve solving a calculus problem dealing with Jacobi-Hamiltonians and Lagranges. Direct methods on the other hand; the solution to an optimal control problem is obtained by transcription and scales well to high dimensional systems, but yields a single trajectory through state and control space, rather than a global policy like in dynamic programming. Direct methods consist of discretizing the optimal control problem, reducing it to a nonlinear constrained optimization problem while Indirect methods are based on Pontryagin maximum principle which reduces the problem to a boundary value problem. Basically, two methods of solving

optimal control problems numerically exist; the direct and indirect methods. On the other hand, direct approaches involve numerical methodologies.

Application of optimization based control to complex high-order nonlinear systems is computationally tasking and tedious. It requires high performance processors and top notch numerical solving approaches. The situation becomes worse when the dynamics is of high order complex or multi-phase. Trajectory optimization in problems are usually implemented off-line. This is intuitive as the computational time and possibilities of the solution not converging or having no solution at all (for instance if the constraints are too tight or have to be violated for the existence of a solution) would represent catastrophic consequences if implemented on-line and in real-time without a backup plan. Transcription basically converts a dynamic system (which is solved under the space of functions) to nonlinear programming problem (NLP) (which is solved under the space of real-valued numerals). In [115], transcription methods for solving an optimal control problem works by converting a continuous problem into a nonlinear programming problem. In this form, the system can be implemented with a generic solver. The direct approach to solving the optimal control problem is the subject of this section. The Bolza's description of the optimal control problem is given in Equation 4.86:

$$\begin{aligned} \max_{x \in X, u \in U} J[x(\cdot), u(\cdot)] &= \phi(t_0, x(t_0), t_f, x(t_f)) + \int_{t_0}^{t_f} g(t, x(t), u(t)) dt \\ \text{Subject to: } \dot{x}(t) &= f(t, x(t), u(t)), \quad x(t_0) = x_0 \end{aligned} \tag{4.86}$$

where ϕ is a continuously differentiable function representing a terminal cost, f is the system dynamics.

4.4.1 Direct and Indirect Methods

The numerical methods to solve optimal control problems can be divided into three main families [116], [117]:

- Dynamic programming (DP): use the Hamiltonian-Jacobi-Bellman optimality criteria.
- Indirect methods: use calculus of variations and Pontryagin's minimum principle to derive the necessary conditions of optimality, this gives rise to a boundary value problem (BVP) that arises from taking the derivative of the Hamiltonian. The BVP is then discretized.
- Direct methods: discretize the continuous control problem and construct a sequence of points. This gives a finite set of variables that can be solved using optimization methods. A typical strategy is to convert the problem into a NLP problem which can be solved using programming techniques. NLP techs use the Karush-Kuhn-Tucker (KKT) conditions to achieve local optimizations. These methods do not require explicit derivation and construction of the necessary conditions. As mentioned, the direct collocation methods are the best in solving aerospace trajectory optimization problems.

There are several difficulties to overcome when an optimal control problem is solved by indirect methods. Firstly, it is necessary to calculate the Hamiltonian, adjoint equations, the optimality and transversality conditions. Indirect method makes use of calculus of variations to determine first-order optimality conditions for the original control problem. This approach leads to a multiple-point BVP that is solved to determine candidate optimal

trajectories called extremals. In indirect method, the optimal solution is found by solving a system of differential equations that satisfies interior point conditions [118]. This approach is not flexible, since each time a new problem is formulated, a new derivation is required. The primary advantage of indirect methods is their high accuracy in the solution and the assurance that the solution satisfies the first-order optimality conditions. Although they suffer from small radii of convergence and the need to analytically derive the Hamiltonian BVP [119]. In contrast, a direct method does not require explicit derivation of necessary conditions. Due to its simplicity, the direct approach has been gaining popularity in numerical optimal control over the past three decades.

A new family of numerical methods for dynamics optimization emerged; referred as direct methods. This was driven by the industrial need to solve large-scale optimization problems and it has also been supported by the rapidly increasing computational power. Here, the state and or control are approximated using an appropriate function approximation e.g. polynomial approximation or piecewise constant parameterization. Simultaneously, the cost functional is approximated as a cost function. Then the coefficients for the function approximations are treated as optimization variables and the problem is reformulated as a standard nonlinear optimization (NLP) problem Equation 4.87. Thus, the main idea of direct methods is to transform the initial problem into a finite dimensional NLP (i.e. first discretize then optimize):

$$\begin{aligned} & \min_{z \in \mathbb{R}^n} f(z) \\ \text{Subject to: } & g(z) \leq 0 ; h(z) = 0 ; z_{low} \leq z \leq z_{upp} \end{aligned} \tag{4.87}$$

Where $h(z)$, $g(z)$ are equality and inequality constraints respectively. The formulation of the optimal control problem in NLP form makes it easier to solve than the BVP because of sparsity in the resulting nonlinear programming problem.

In direct approaches, the optimal problem is transformed or transcribed into a nonlinear programming problem as in Equation 4.87 above. The NLP can be solved using either a penalty function method or methods of augmented or modified Lagrangian functions such as sequential quadratic programming (SQP) methods. Naturally, the resulting NLP is solved numerically by well-developed algorithms which attempt to satisfy a set of conditions called Karush-Kuhn-Tucker conditions (KKT) associated with the NLP. Optimal solution is now found by transcribing (discretizing) an infinite-dimensional optimization problem to a finite dimensional optimization problem. Direct shooting, state and control parametrization methods and pseudo-spectral methods are the techniques to transcribe an optimal control problem into a NLP. The disadvantage of direct methods is that they produce relatively less accurate results compared to indirect methods. More so, the discretized optimal control problems have several minima which results in “pseudo-minima” solutions when direct methods are applied on them. One of the most important advantages of direct compared to indirect methods is that they can easily treat inequality constraints.

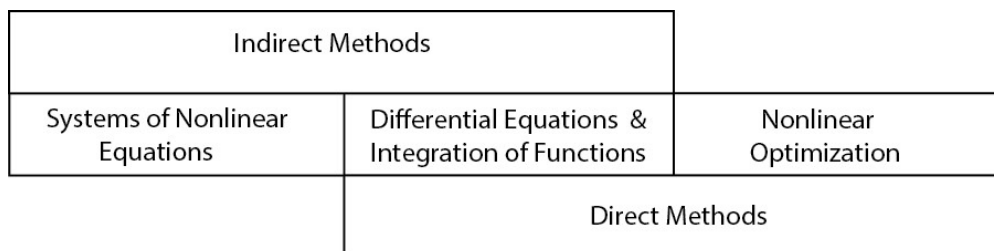


Figure 4.4: Major Components and Classes of Optimal Control Methods [118].

4.4.2 Literature on Trajectory Optimization

A number of applications of optimal techniques in trajectory optimization and control can be found in a number of papers. In [120], the authors presented a performance benchmarking system for optimal control of a 2D quadrotor. The developed manoeuvres were shown to satisfy Pontryagin's minimum principle with respect to time optimality. While the algorithm developed was mentioned as too slow to be used in real-time trajectory generation settings, it offers a valuable reference to benchmark other trajectory generation tools and controllers. In [121], The time-optimal control problem of a hovering quadrotor helicopter was addressed. Instead of utilizing the Pontryagin's Minimum Principle, in which one needs to solve a set of highly nonlinear differential equations, a nonlinear programming (NLP) method was proposed. In their method, the count of control steps is fixed initially and the sampling period is treated as a variable in the optimization process. The optimization objective is to minimize the sampling period such that it will be below a specific minimum value, which is set in advance considering the accuracy of discretization. To generate initial feasible solutions of the formulated NLP problem, genetic algorithms (GAs) were adopted. In summary, time-optimal movements of the helicopter between two configurations can be found as confirmed by simulations. In [122], [123], [124], a time-optimal path-constrained trajectory planning problem was treated. The authors made use of nonlinear change of variables to transform the time optimal trajectory planning problem to a convex optimal control problem with single state. In [125], The paper presented a nice study of direct transcription methods as applied to robotic motion planning. Similarly, [117], [126] and [127] contains a detailed study of direct collocation techniques as applied to trajectory optimization problems. The major advantage of direct collocation over

multiple shooting is their better run-time performance (using relatively small collocation intervals) and a larger convergence radius. In [128], the authors described the direct multiple shooting approach as being capable of solving complex problems without any analytical development by using an off-the-shelf solver. They thereafter applied this to a quadrotor for online trajectory optimization for going through a window and manipulation tasks. Optimal control techniques are also indispensable when it comes to agile motions as it enables the design of system states in different configurations and complex environments.

4.4.3 Trajectory Optimization

In the transcription phase, trajectory optimization is converted to a constrained parameter optimization problem (NLP) (infinite dimensional to finite dimensional). The decision variables change from vector functions to real numbers and differential equations become algebraic equations. The transcription problem is the crux of majority of nonlinear programming and optimization solving programs. A thorough description of the transcription process for trajectory optimization was discussed [115]. Transcription is a crucial process for the use of constrained optimization techniques such as nonlinear programming to solve trajectory optimization problems of dynamical systems. It is the conversion of a function-valued dynamical system to a real number numerical system wherein nonlinear programming can be applied. Once the problem can be represented in the form Equation 4.88 below, it can easily be solved by available software programs.

$$\begin{aligned} & \min_{z \in \mathbb{R}^n} \bar{J}(z) \\ \text{Subject to: } & 1 \leq \begin{pmatrix} z \\ \bar{c}(z) \\ Az \end{pmatrix} \leq u \end{aligned} \quad (4.88)$$

In direct collocation, the input is represented as a piecewise-linear function of time while the state is piecewise-cubic. The decision variables are now the values of the state and control at the knot points and the collocation points are the mid-points of each cubic segment. The slope of state is prescribed by the dynamics at each knot point and the slope of the cubic at the collocation point is constrained to match the system dynamics at that point. Solutions to the NLP problem can be obtained using several software such as IPOPT, SNOPT and FMINCON.

4.4.4 Solution to the Trajectory Optimization Problem

A general framework for a trajectory optimization problem is:

$$\begin{aligned}
 \text{Optimal Trajectory:} & \quad \{x^*(t), u^*(t)\} \\
 \text{System Dynamics:} & \quad \dot{x} = f(t, x, u) \\
 \text{Constraints:} & \quad c_{min} < c(t, x, u) < c_{max} \\
 \text{Boundary Conditions:} & \quad b_{min} < b(t_0, x_0, t_f, x_f) < b_{max} \\
 \text{Cost Functional:} & \quad J = \phi(t_0, x_0, t_f, x_f) + \int_{t_0}^{t_f} g(t, x, u) dt
 \end{aligned} \tag{4.89}$$

In summary, the trajectory optimization problem is posed as Equation 4.89. In setting up the optimization problem, discontinuities need to be avoided to maintain smoothness. The ‘*OptimTraj*’ solver based on Matlab’s ‘*FMINCON*’ function developed by [129] was used in this work. The *OptimTraj* solver is capable of dealing with continuous dynamics, boundary constraints, path constraints, integral cost function and boundary cost functions. As presented in [115], a trajectory optimization problem seeks to find a trajectory for some dynamical system that satisfies some set of constraints while minimizing some cost functional J .

Arguably, the most powerful methods for solving general optimal control problems are direct collocation methods [118]. A direct collocation method is a state and control parameterization method where the state and control are approximated using a specified functional form. Direct collocation method has been successfully applied to difficult constrained optimal control problems such as the minimum accumulated heat descent trajectory of an Apollo capsule with a height constraint [117]. In this work, the Hermite-Simpson Direct Collocation method was used. Figure 4.5 shows the concept of transcription. Finally, we intend to obtain a solution using *OptimTraj* to solve the problem structured in Equation 4.90 below [115]:

$$\min_{t_0, t_f, \mathbf{x}(t), \mathbf{u}(t)} J(t_0, x_0, \mathbf{x}(t_0), \mathbf{x}(t_f)) + \int_{t_0}^{t_f} \omega(\tau, \mathbf{x}(\tau), \mathbf{u}(\tau)) d\tau \quad (4.90)$$

Subject to:

$$\mathbf{g}(t_0, t_f, \mathbf{x}(t_0), \mathbf{x}(t_f)) \leq 0$$

Boundary constraints

$$t_{low} \leq t_0 < t_f \leq t_{upp}$$

Bound on Initial and Final Time

$$\mathbf{x}_{0,low} \leq \mathbf{x}(t_0) \leq \mathbf{x}_{0,upp}$$

Bound on Initial State

$$\mathbf{x}_{f,low} \leq \mathbf{x}(t_f) \leq \mathbf{x}_{f,upp}$$

Bound on Final State

$$\dot{\mathbf{x}}(t) = f(t, \mathbf{x}(t), \mathbf{u}(t))$$

Continuous Dynamics

$$\mathbf{h}(t, \mathbf{x}(t), \mathbf{u}(t)) \leq 0$$

Path Constraints

$$\mathbf{x}_{low} \leq \mathbf{x}(t) \leq \mathbf{x}_{upp}$$

Continuous Bounds on State

$$\mathbf{u}_{low} \leq \mathbf{u}(t) \leq \mathbf{u}_{upp}$$

Continuous Bounds on Control

The left hand side of Equation 4.90 represents the boundary objective function (deals with the beginning and end of the trajectory) while the right hand side stands for the integral objective function (dealing with a quality along the trajectory). The transcription process now converts the trajectory optimization problem to a Nonlinear Program (NLP) of the

form in Equation 4.91 which are now real valued, finite dimensional set of algebraic equations.

$$\begin{aligned} & \min_z f(z) \\ \text{Subject to: } & g(z) \leq 0; \\ & h(z) = 0; \\ & z_{low} \leq z \leq z_{upp} \end{aligned} \tag{4.91}$$

In summary, the direct methods work by discretizing then optimizing, are less accurate but easier to pose and solve. Collocation methods are based on function approximation and better for problems with complicated control and/or path constraints. The techniques could also be improved by using any of h- or p- methods as highlighted in [115].

4.4.5. Direct Transcription (Hermite-Simpson Collocation)

This process converts the continuous time functions to discrete form:

Let N be the number of grid points. $x_k = x(t_k)$, $u_k = u(t_k)$. $t = \{t_0, t_1 \dots t_N\}$, $x(t) = \{x_0, x_1, \dots x_N\}$ and $u(t) = \{u_0, u_1, \dots u_N\}$. Approximations are used to convert (transcribe) the continuous functions in dynamics and objective function to algebraic equivalents.

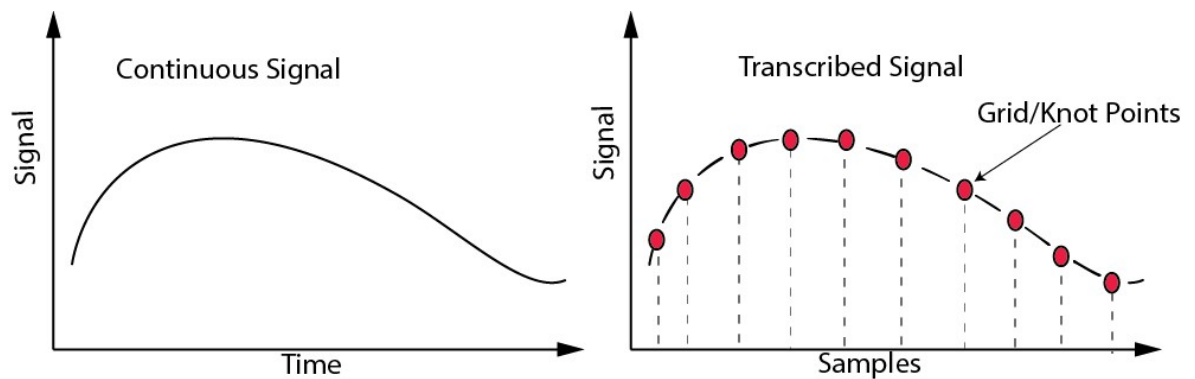


Figure 4.5. Transcription.

The Hermite-Simpson method was used in this part as inspired by work done in [129]. In the Hermite-Simpson's (*HS*) method, the dynamics \mathbf{x} and controls \mathbf{u} are assumed to be quadratic between grid points. The state however has a cubic profile and this is because the state is the integral of the dynamics. Interval points between knot points are also introduced and the decision variables become:

In controls: $u_0, u_{0+1/2}, u_1, u_{1+1/2}, \dots, u_N$

In state: $x_0, x_{0+1/2}, x_1, x_{1+1/2}, \dots, x_N$

The objective function is approximated using Simpson quadrature as:

$$\int_{t_0}^{t_F} \omega(\tau) d\tau \approx \sum_{k=0}^{N-1} \frac{h_k}{6} (\omega_k + 4\omega_{k+1/2} + \omega_{k+1}) \quad (4.92)$$

The system dynamics is also approximated by a Hermite interpolant combined with Simpson collocation:

$$\dot{\mathbf{x}}(t) = f(t, \mathbf{x}(t), \mathbf{u}(t)) \approx$$

$$\text{Hermite Interpolant: } x_{k+1} = \frac{1}{2}(x_k + x_{k+1}) + \frac{h_k}{8}(f_k - f_{k+1})$$

$$\text{Simpson Collocation: } \frac{h_k}{6}(f_k + 4f_{k+1/2} + f_{k+1}) = x_{k+1} - x_k \quad (4.93)$$

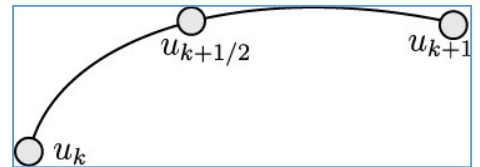
The corresponding method of interpolation is given by a quadratic spline for the controls and a cubic spline for the states.

Quadratic spline for the controls is defined as:

$$\beta_1 = -\frac{1}{h_k} \left(3u_k - 4u_{k+1/2} + u_{k+1} \right)$$

$$\beta_2 = \frac{2}{h_k^2} \left(u_k - 2u_{k+1/2} + u_{k+1} \right)$$

$$u(t) = u_k + \delta_k \beta_1 + \delta_k^2 \beta_2$$



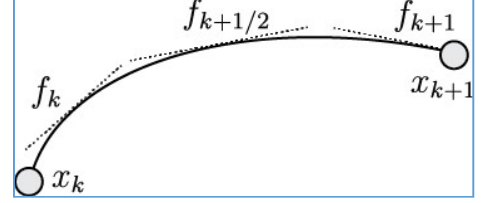
$$(4.94)$$

Cubic spline for the states is defined as:

$$\gamma_2 = -\frac{1}{2h_k} \left(3f_k - 4f_{k+\frac{1}{2}} + f_{k+1} \right)$$

$$\gamma_3 = \frac{2}{3h_k^2} \left(f_k - 2f_{k+\frac{1}{2}} + f_{k+1} \right)$$

$$x(t) = x_k + \delta_k f_k + \delta_k^2 \gamma_2 + \delta_k^3 \gamma_3$$



(4.95)

while $t_k \leq t \leq t_{k+1}$, $h_k = t_{k+1} - t_k$ and $\delta_k = t - t_k$.

The objective function is defined with the aim to be minimize time and control effort,

thus:

$$\min_{x(t), u(t)} \int_0^T (u^2(\tau) + 1) d\tau \quad (4.96)$$

Finally, the trajectory optimization task is solved in the defined structure:

- Create the Dynamics Function for each subsystem
- Create the Objective Function $\min_{x(t), u(t)} \int_0^T (u^2(\tau) + 1) d\tau$
- Initialize Guess for the Optimization Problem
- Call Trajectory Optimization Solver: GPOPSII, OptimTraj
- Define the Hermite-Simpson Direct Collocation Algorithm
- Run the Optimization Task

4.5 Proportional Derivative Control of 1-DoF Prismatic Arm

The model of the prismatic arm is given in Equation 3.29 as:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ \frac{U_\tau}{m_p} - g \end{bmatrix} \quad (4.97)$$

x_1 is the position of the end-effector, x_2 is the velocity of the joint, m_p represents mass while U_τ is the torque-control input.

The system in this section is simple to control. A simple PD controller is proposed by writing the system state in error variables.

Let x_r be the reference position of the prismatic arm.

$$\text{Error in desired position: } e_1 = x_1 - x_r \quad (4.98)$$

$$\text{Error derivative: } \dot{e}_1 = \dot{x}_1 - \dot{x}_r \Rightarrow e_2$$

$$e_2 = \dot{e}_1 = x_2 - \dot{x}_r$$

$$\dot{e}_2 = \ddot{e}_1 = \dot{x}_2 - \ddot{x}_r$$

$$\dot{e}_2 = \frac{U_\tau}{m_p} - g - \ddot{x}_r \quad (4.99)$$

Thus, the error dynamics is written as:

$$\dot{e}_1 = \dot{e}_2$$

$$\dot{e}_2 = \frac{U_\tau}{m_p} - g - \ddot{x}_r$$

The error \mathbf{e} is driven to zero by selecting the controller:

$$U_\tau = m_p g - k_p e_1 - k_d e_2 \quad (4.100)$$

4.6 Parameter Optimization of Controllers

Parameter optimization is very crucial for any tuning problem. Numerous algorithms have been developed for optimization and utilized in a cornucopia of applications; even outside the control field. In [130], the particle swarm algorithm was applied for the optimization of mobile robot controller while [131] applied genetic algorithms to tune PID controllers. Optimization can be applied online (gains are dynamically fitted during process loop) or offline (static gains obtained are used in the process loop). The objective function is set to minimize or maximize a cost function. The use of evolutionary algorithms to optimize system parameters is widespread and often requires a tedious development of code to implement the algorithms.

A powerful and usually overlooked tool for parameter optimization; '*Response Optimization*' lies in Matlab (v.2015b) itself. The Simulink Design Optimization (SDO) software [132] automatically converts design requirements to a constrained optimization problem and then applies optimization techniques. The Simulink model is then iteratively simulated until the design requirements are satisfied. This SDO tool also has a graphical user interface where users can utilize click-programming to set decision variables and signal bounds. It can also be implemented in code. The signal bounds (optimization criteria) can be seen as the cost function and constraints of the problem. They are defined as piecewise linear bounds. Once all is set, the user chooses an appropriate solver such as: gradient descent, pattern search or simplex search. In order to speed up optimization, the parallel computing option can be activated. This tool is powerful and easy to use. The entire optimization task is very flexible.

In this work, Gradient Descent-Sequential Quadratic Programming was used to optimize the controller parameters. This was combined with Matlab's Parallel computing feature for faster computations. The optimized controller parameters are presented in Chapter 6.

CHAPTER 5

HYBRID SYSTEM

5.1 Hybrid System

A hybrid control system is a control system where the plant or the controller contains discrete modes that together with continuous system equations govern the behaviour of the overall system. If these discrete parts are embedded in the control system, then it takes the form of a scheduler or a supervisor control system. Apparently, a lot of real life systems have coupled modes of operation and there is no unified approach to treat them. Consequently, researchers in computer science, mathematics and controls have different terminologies to describe hybrid system phenomena. Terminologies that can be seen to capture hybrid system phenomena include: ‘*timed automata*’, ‘*dynamical systems theory*’, ‘*automata theory*’, ‘*discrete event systems*’, ‘*selector-based control*’, ‘*gain scheduling*’, ‘*fuzzy control*’, ‘*logic control*’, ‘*behaviour based control*’, ‘*supervisory control*’, ‘*switched control*’ and so on. Investigations on hybrid systems are mostly done by simulation and the field is seen as still-developing. Technical details, analysis and applications on hybrid control systems can be found in [133]–[139].

Applications of hybrid system control are numerous. For instance, they can be found in flight control of aircrafts which have different modes as in take-off and landing, different loading and atmospheric conditions. Quadrotor-manipulator systems too require discrete modes in order to complete a specific job. Hybrid systems can be analysed in Matlab *Stateflow*. Since conventional controllers are suitable for a specific design objective, discrete modes of operation require a control structure that can adapt or change with respect to the system mode.

This chapter of the thesis presents the fusion of modes for the autonomous sensor deployment system and presents the following:

- Hybrid Control Structure
- Quadrotor Vision System

The general block model of the hybrid system used in this work is shown in Figure 5.1:

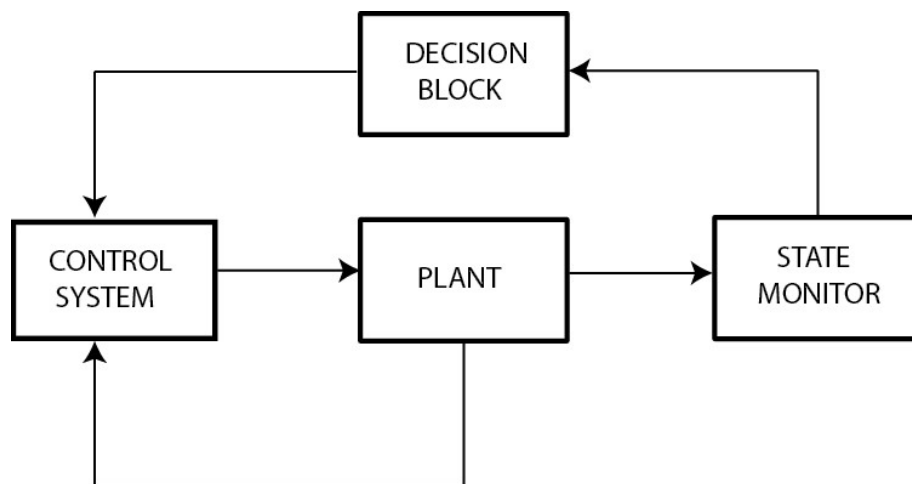


Figure 5.1: Hybrid System

5.2 Hybrid Control Structure

Hybrid systems are characterized by continuous systems with a mode-based operation with different modes corresponding to different continuous dynamics which are described by ordinary differential equations [134]. In this work, the relationship between the different modes of operation is mapped using finite state machines. There are two phases in the overall system with each having specific modes of operation. The overall system is highlighted in Figure 5.2 and a finite state map is defined in Figure 5.3:

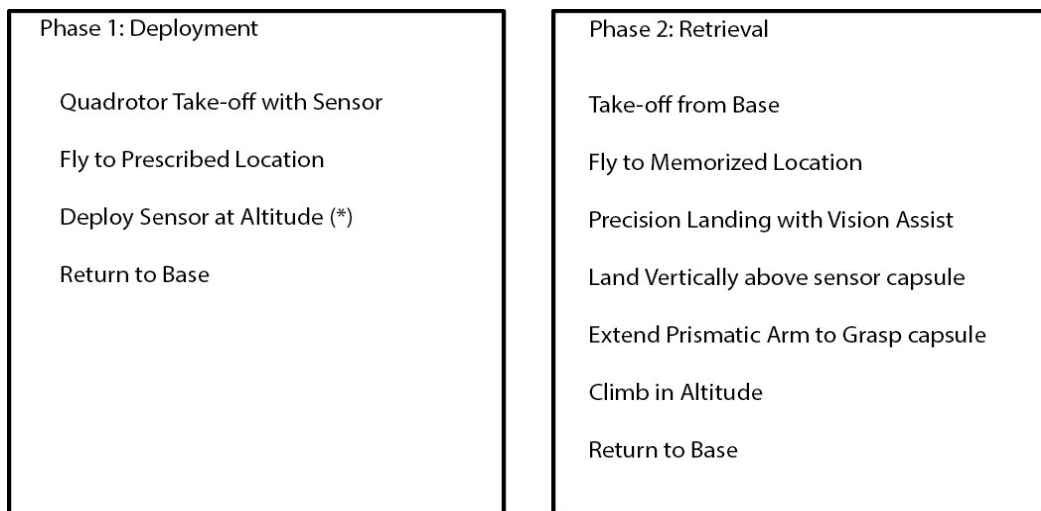


Figure 5.2: Phases and Modes of Operation

QUADROTOR SYSTEM FINITE STATE AUTOMATA

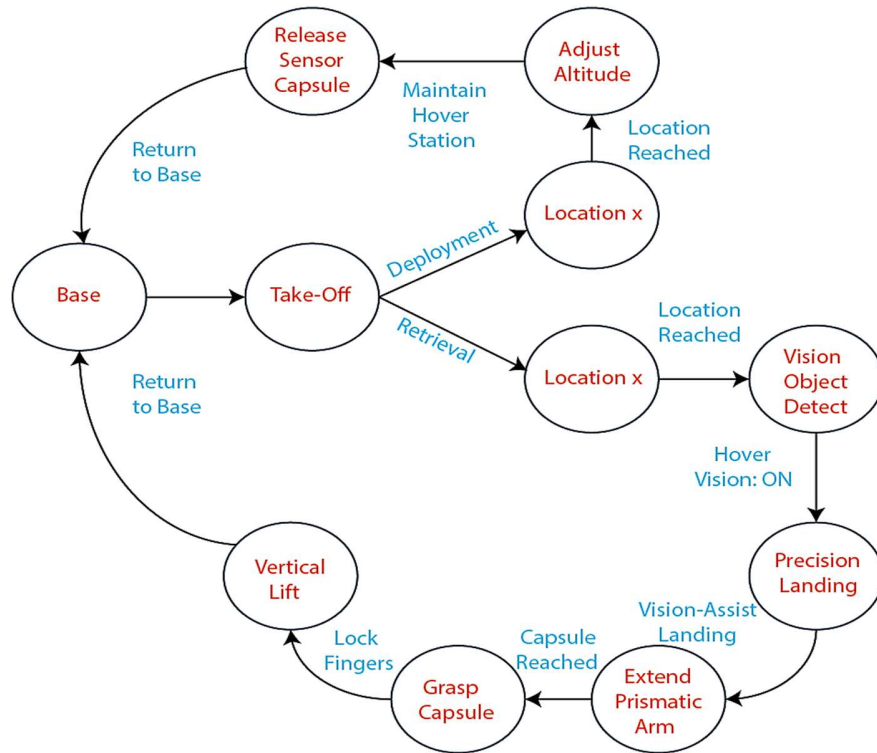


Figure 5.3: Finite State Automata

The hybrid system in Figure 5.1 consist of four blocks: plant, state monitor, decision block and control system, each with a specific function. The plant contains the dynamic model of the quadrotor manipulator system. The input to the plant block is the control commands while it outputs the plant states. The plant states are stored and updated in the state monitoring block. It serves as a look-up cache for the decision block. The finite state automata module is embedded in the decision block. It takes information from the state monitor and checks the guard conditions. Once guard conditions are satisfied, the decision block informs the control system to take appropriate action in terms of set-points and reference profiles. The control system in turn generates command signals to drive the quadrotor-manipulator system (plant) to the desired target. As a result, set-point changes

and switching between different controllers can be used in the hybrid system for specific modes of operation. The basis of using this technique stems from the well-known theorem from switched control systems and as related to the system's dwell-time [140]:

Statement: *A switched system is stable if all individual subsystems are stable and the switching is sufficiently slow, so as to allow transient effects to dissipate after each switch* [140]. The implication of the above statement is that, since each mode takes a considerable amount of time, successful switching between controllers is possible between each mode transition.

5.2.1 Sensor Deployment

In order to interpret the hybrid system, we consider an example. Under the assumption of already attached sensor capsule and no obstacles along flight path, the quadrotor-manipulator (Q-M) flies to location (x, y) at an arbitrary altitude. When the location is reached, the output of the decision block is for the location mode is 'True' and it moves the state automata to the next mode; 'altitude adjust'. At (x, y) and with prior information of the required height (h) of drop to ensure ground penetration, the Q-M adjusts its altitude to $(z = h)$. It then maintains this altitude at hover and opens the fingers of the robotic arm; releasing the sensor capsule and returns to base. In order to penetrate the ground, the geophone sensor has to be released at an altitude that would result in a predetermined potential energy magnitude for ground penetration. The energy requirement can be evaluated with the use of a soil penetrometer. More so, by fixing the geophone at the bottom edge of the sensor capsule, the center of mass shifts towards the bottom, ensuring that the sensor capsule drops vertically. This is also aided by the aerodynamics of the sharp edge.

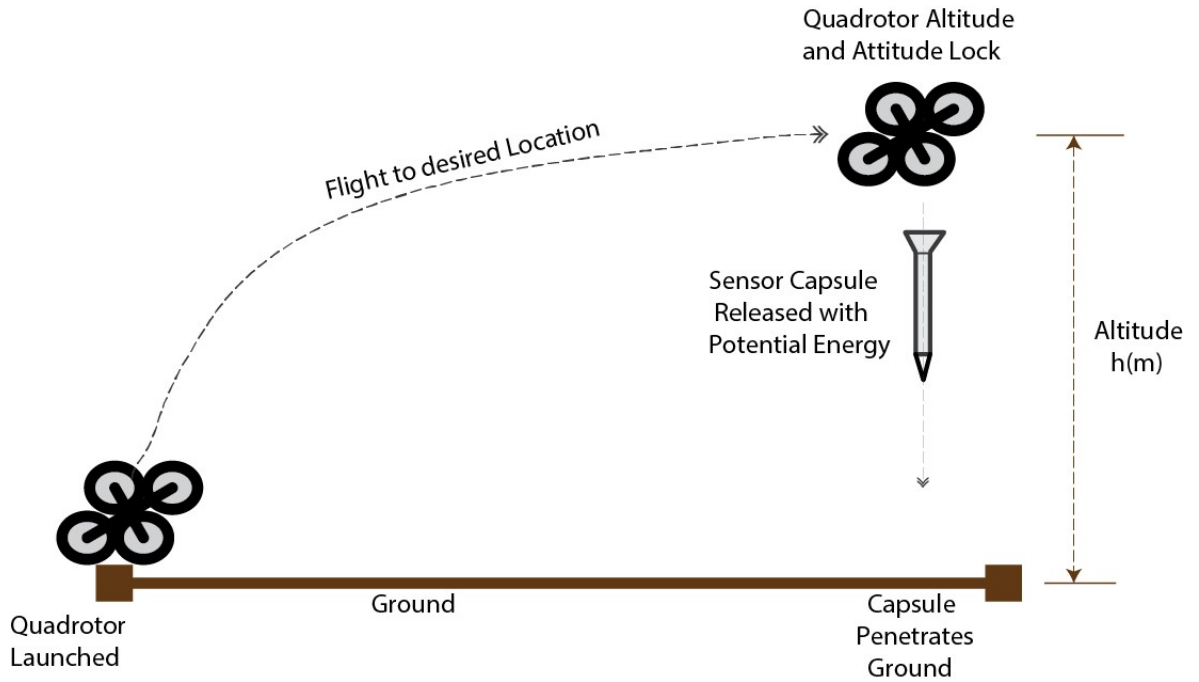


Figure 5.4: Sensor Deployment Phase

5.2.2 Sensor Retrieval

In the retrieval mode, the quadrotor requests stored information of location (x, y) from the state monitor. Due to the presence of sensor uncertainties and noise, the quadrotor actually flies to the location (x^*, y^*) with an error $\varepsilon = (x, y) - (x^*, y^*)$. In order to land vertically above the sensor capsule with better accuracy and precision, location refinement is applied via a visual feedback system. At location (x^*, y^*) , the quadrotor enters hover mode and the camera is switched on. Using visual feedback, the sensor capsule is located as an object on the image plane and its relative position in world coordinates is obtained using relevant transformations as described in Section 5.4. The decision block determines if proper alignment has been made; if true, the quadrotor enters attitude lock and gradually descends for a smooth landing directly above the sensor capsule. After landing, the prismatic arm with open fingers is extended until a proper proximity to the capsule is attained. This is

achieved using a combination of vision and proximity detector. The robotic fingers have enough reach to form an ellipse of radius (ζ) (Section 5.3). A logic command is sent to close the fingers and lock the grasped capsule in place. Thereafter, the quadrotor exerts a vertical thrust (altitude climb) that breaks the friction and force coupling between the capsule and the ground. Achieving lift off, the quadrotor system returns to base.

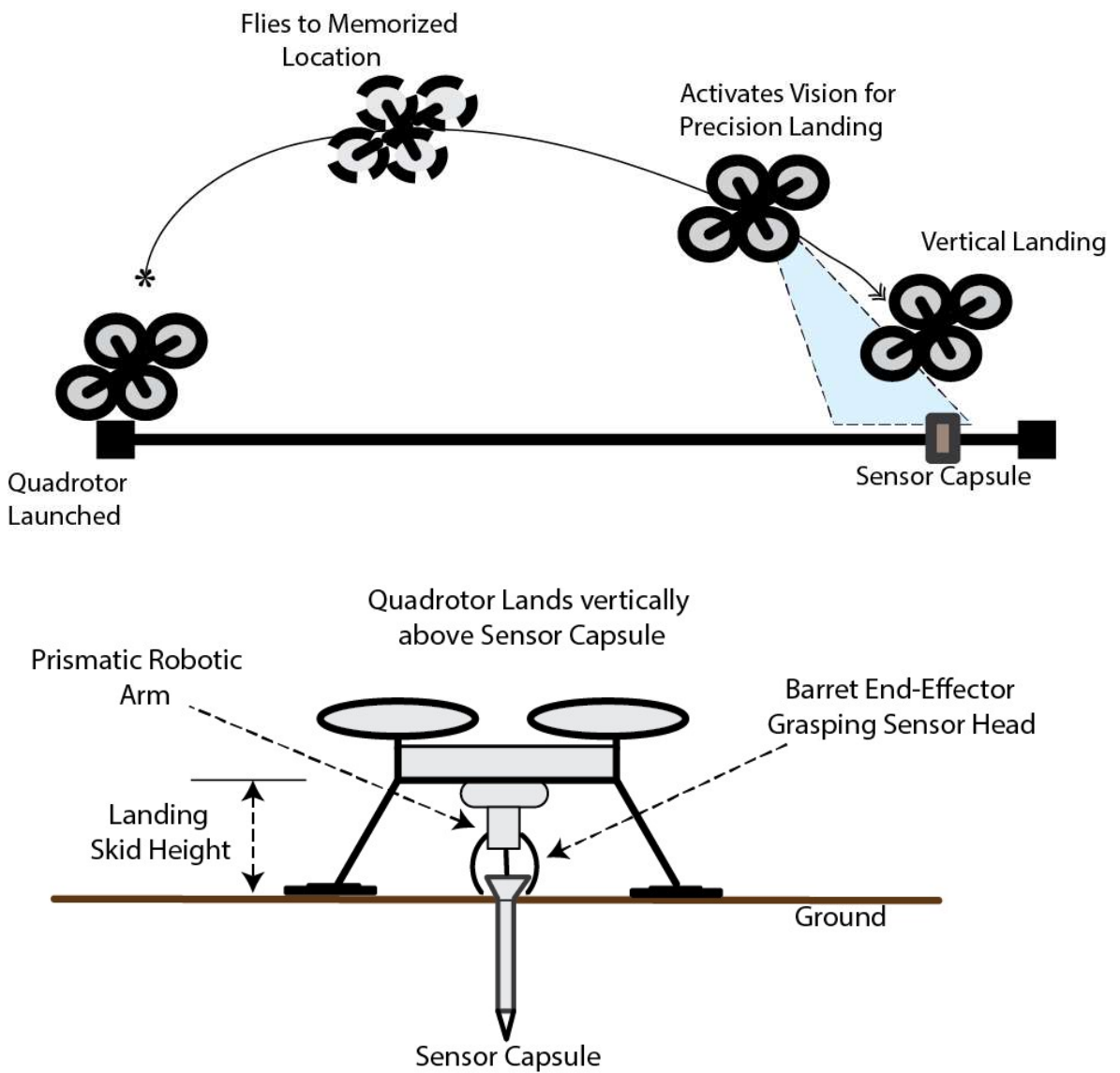


Figure 5.5: Sensor Retrieval Phase

5.3 Robotic Hand

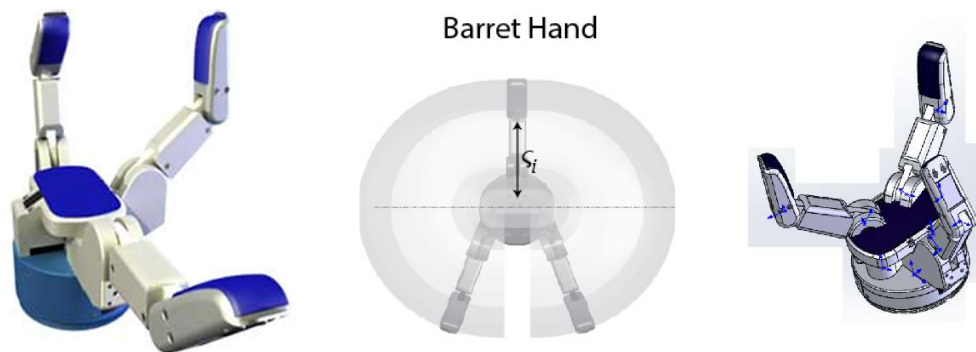


Figure 5.6: Barret Hand [141] with Reach Radius ζ_i .

The Barret hand has three fingers, with length ζ between the axis center and the proximal link of each of the fingers. As shown in Figure 5.6, the geometric zone of reach can be expressed mathematically as a positive definite symmetric matrix in the form Equation 5.1:

$$x^T \Lambda^{-1} x = 1$$

$$\Lambda = \begin{bmatrix} \zeta_1 & 0 \\ 0 & \zeta_2 \end{bmatrix}, \quad \zeta_1 = \zeta_2, \quad \zeta_1, \zeta_2 > 0 \quad (5.1)$$

Since $|\Lambda| > 0$, the zone of reach of the Barret finger is a complete circle with the major and minor axes proportional to ζ with ζ assumed to be a length of 0.12m.

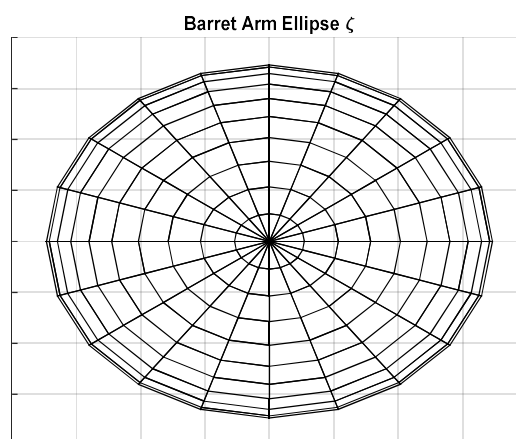


Figure 5.7: Barret Hand Ellipse

The Barret hand ellipse in Figure 5.7 is an envelope that defines the zone of reach (ZOR) or workspace of the hand. The necessary and sufficient conditions for reaching and grasping the sensor capsule is that the sensor capsule must lie inside this ZOR. This is ensured by the use of visual feedback from the vision system.

5.4 Quadrotor Vision System

Computer vision is a broad field of robotics with many institutions actively involved. It is a very useful sensor as it enables measurements and feedback information from the field without contact with the environment. Using visual information to control a robot's pose is known as visual servoing. A very good article on visual servo control can be found in [142]. Cameras are analytically described with projection models. Projection models describe how a scene is represented on the image plane, the model varies with the type or combination of vision sensors in use and transformations are required to relate image features to real world objects. The focal length and optical center are important intrinsic parameters of the camera and these properties are obtained via camera calibration. Camera calibration on its own is a research problem being tackled by academia. Similarly, image processing techniques have been applied to quadrotors, UAVs and autonomous landing in numerous works [33], [143]–[155].

As highlighted in [142], we consider a dynamic look-and-move structure where the vision system provides set-points to the quadrotor's position control subsystem.

In this work, we consider an *eye-in-hand* system, where the camera is attached to the central axis of the Barret hand. The vision system observes only the target object (sensor capsule) in an endpoint open-loop (EOL) configuration. The final goal is to ensure autonomous

precision landing on the target. The main steps required include: camera calibration, image processing and projection transformations.

5.4.1 Introduction to the Vision System

The camera takes series of images of the scene. These images are evaluated using image processing techniques to retrieve desired information for the control subsystem. From an altitude, the camera has a horizontal view on its image plane where all objects in the scene can be analysed. The sensor capsule has a blue colored round cap and is seen from the camera plane as an object as shown in Figure 5.8 with a diameter of $d_s = 0.04\text{m}$.

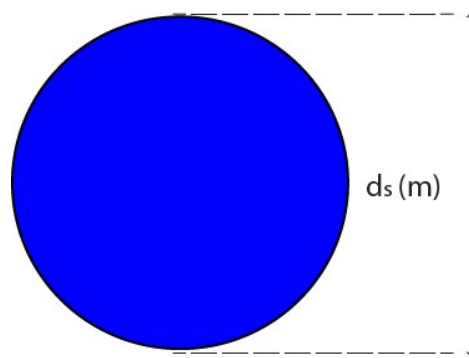


Figure 5.8: Sensor Capsule Cap (*blue*).

5.4.2 Camera Calibration

In order to relate the image information to the real world, the intrinsic parameters (camera characteristics) of the camera need to be known. The process of deriving these parameters is known as camera calibration and is usually done using a standard checkerboard template. In this work, a VREP vision sensor is used to obtain a series of 20 checkerboard image templates to be used in the calibration task. These templates are obtained from different angles and translations relative to the checkerboard to obtain more accurate results. Thereafter, Matlab's Camera Calibration toolbox was used to estimate the vision sensor's

intrinsic parameters by loading the 20 saved images of the checkerboard. This is important since the main parameters that can be predefined for the vision sensor in the VREP platform is the camera field of view (FOV). The FOV is also related to the camera focal length. Figures 5.9, 5.10, 5.11, 5.12 show the respective steps in the camera calibration task.

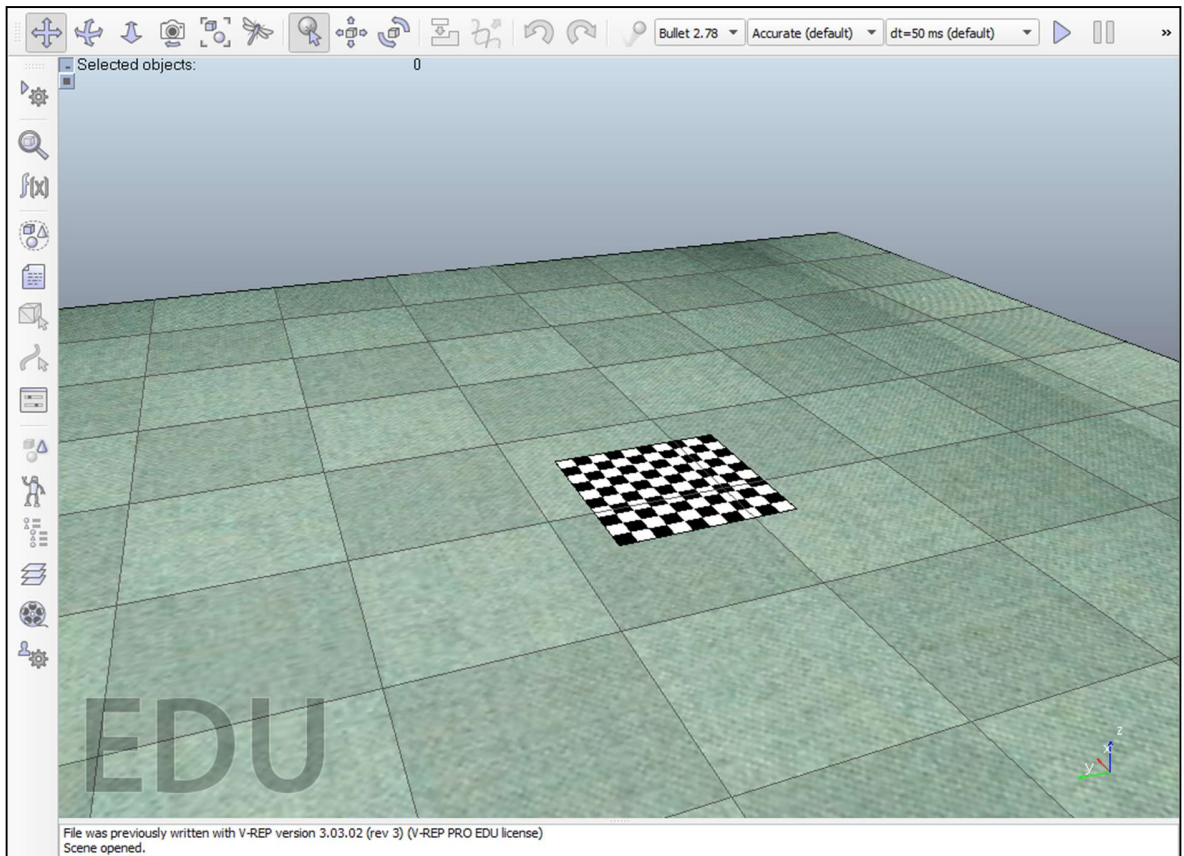


Figure 5.9: Calibration Chessboard Scene in VREP

The checkerboard size is 50×50 cm with 10 by 10 checker-boxes of 5cm each. The camera in use is a perspective-type vision sensor with a perspective angle of 60° (deg.) and an image resolution of 512×512 . It also has a near and far clipping distance of 0.01m and 10m respectively.

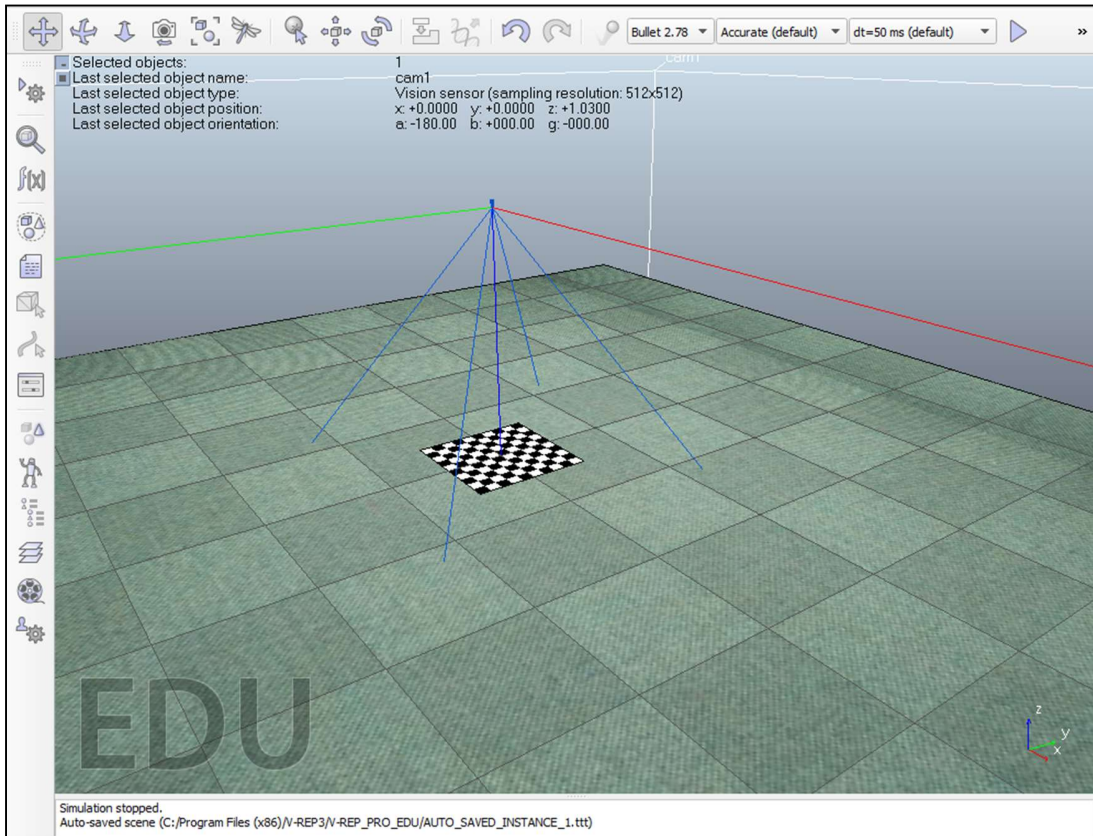


Figure 5.10: Camera Calibration Scene with a Perspective Vision Sensor.

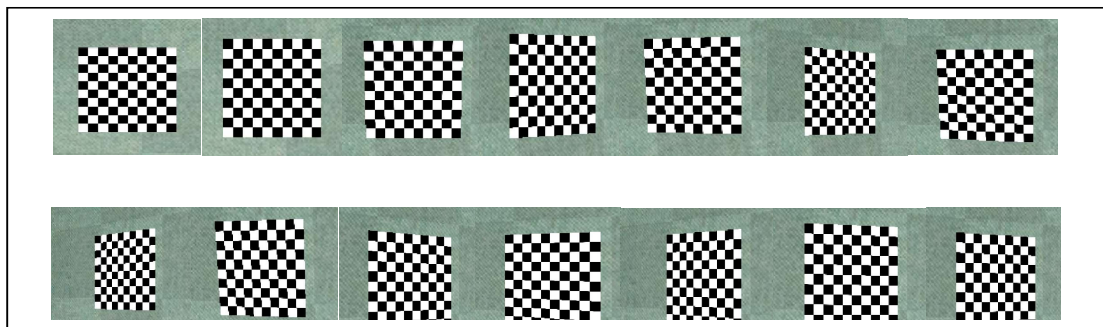


Figure 5.11: Checkerboard Images from Different Perspectives.

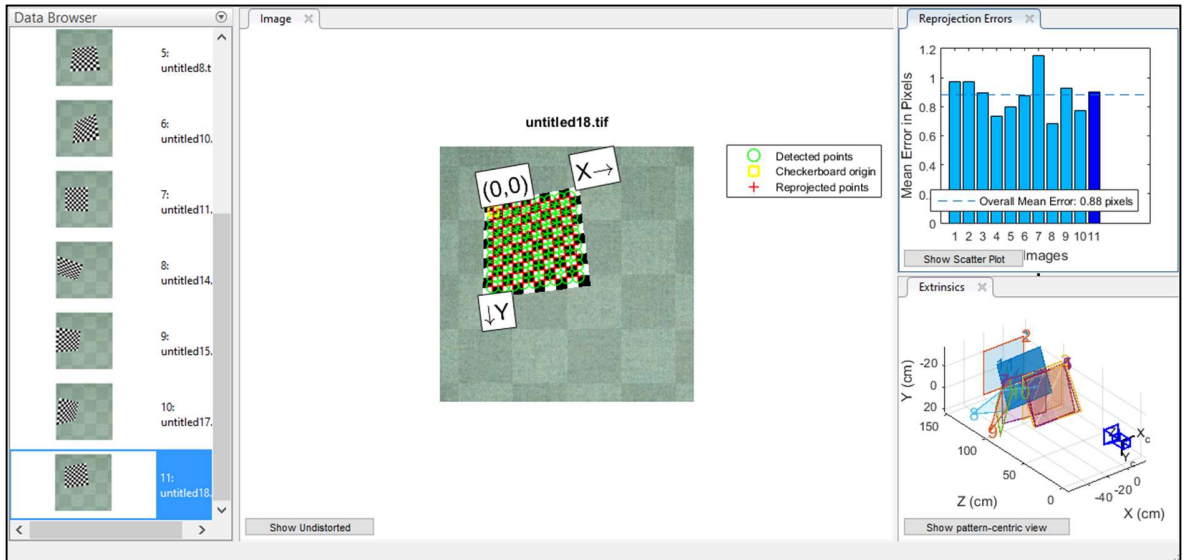


Figure 5.12: Camera Calibration Toolbox in Matlab

5.4.2.1 Calibration Results

Of the 20 calibration images, 9 were rejected while 11 was used for the calibration task.

The results provided are summarized below:

- Radial distortion with 2 coefficients: Radial Distortion: $[-0.0076 \quad 0.0205]$

- Intrinsic 3×3 Camera Matrix: $\begin{bmatrix} 1.3159 & 0 & 0 \\ 0 & 1.3161 & 0 \\ 1.9531 & 0.8462 & 0.0010 \end{bmatrix} \times 10^3$

- The above matrix corresponds to $\mathcal{C} = \begin{bmatrix} f_x & 0 & 0 \\ s & f_y & 0 \\ c_x & c_y & 1 \end{bmatrix}$

c_x, c_y is the optical center (principal point) in pixels, f_x, f_y represents the focal length in pixels. The focal length in world coordinates $F = f_x p_x = f_y p_y$ where $p_{x,y}$ is the pixel dimension in world coordinates while s represents skew.

5.4.3 Projection Transformation

The projection transformation is used to relate the object on the image plane to real world coordinates. Figure 5.13 shows the world-camera-image plane depiction of the computer vision system where $[I]$ is the hypothetical plane where the object from the scene is detected. The quadrotor's body frame and camera are assumed to be perfectly aligned thus, the center of the image plane $[I]$ lies on the same axis with the quadrotor.

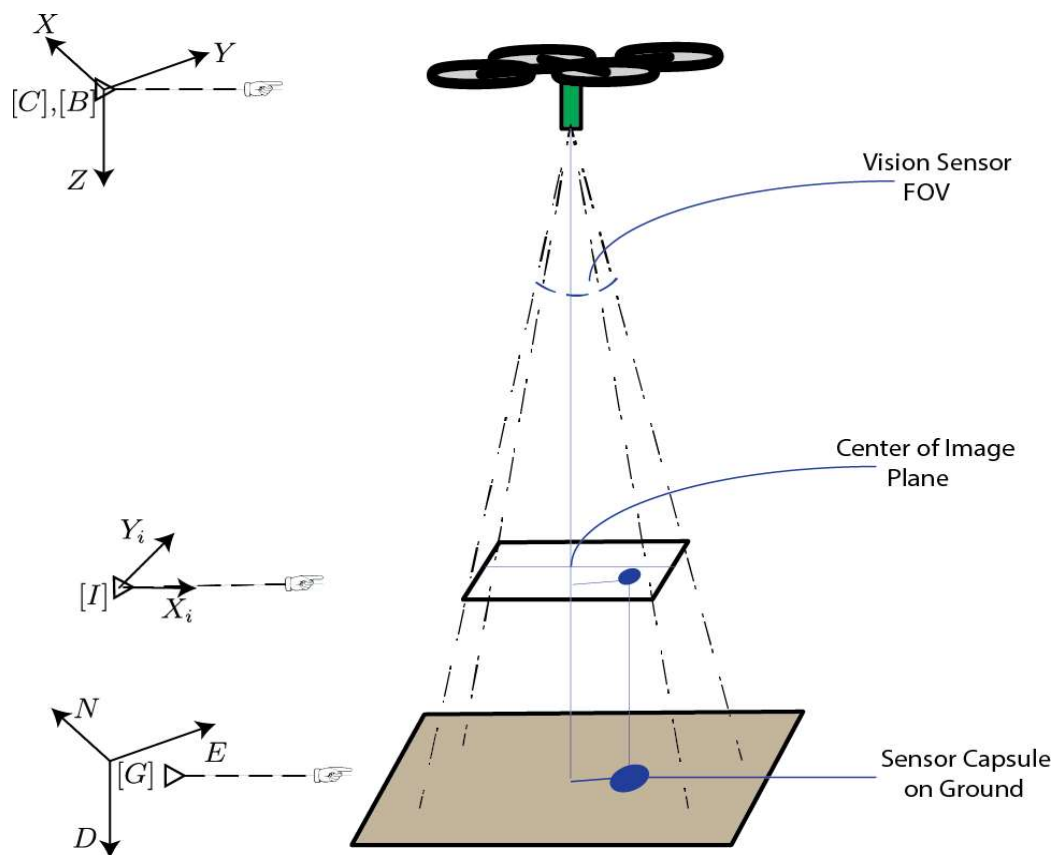


Figure 5.13. Quadrotor Visual Feedback System in Pinhole Camera Form.

The central perspective imaging model as presented in [70] is commonly used in vision systems modeling. Given a camera coordinate frame $[C]$ and image plane coordinate frame

$[I]$, the image of an object $\mathbf{P} = (X, Y, Z)$ in world coordinate frame $[G]$ from the scene is formed on the image plane $\mathbf{p} = (x, y)$ and related by Equation 5.2 using similar triangles.

$$x = f \frac{X}{Z}, y = f \frac{Y}{Z} \quad (5.2)$$

The distance between $[I]$ and $[C]$ along the optical axis is the focal length $z = f$. The image plane principal point (optical center) is located at the center of the image plane. Equation 5.2 above is a projective transformation or perspective transformation with the following properties [70]:

- It performs a mapping from 3D to 2D space $\mathbb{R}^3 \mapsto \mathbb{R}^2$
- Straight lines in the world are projected to straight lines on the image plane
- Parallel lines in the world become lines with a vanishing point on the image plane.
- Conics in the world are projected to conics in the image plane.
- A unique inverse transformation does not exist
- The perspective transformation is not conformal.

The final camera parameters using a pinhole model as described in [156] and [157] is given by a rectangular camera matrix and is useful in mapping the 3D world scene to the image plane. After camera calibration to obtain the extrinsic and intrinsic parameters, the extrinsic parameters represent the location of the camera in 3D scene or world while the intrinsic parameters depict the optical center and focal length of the camera. The transformation process is described in Figure 5.14. World points are transformed to camera coordinates using the extrinsic parameters via a rigid-body-like transformation; $\mathbb{R}^3 \mapsto \mathbb{R}^3$, while the camera coordinates are related to the image plane using the intrinsic parameters with perspective or projective transformation; $\mathbb{R}^3 \mapsto \mathbb{R}^2$.

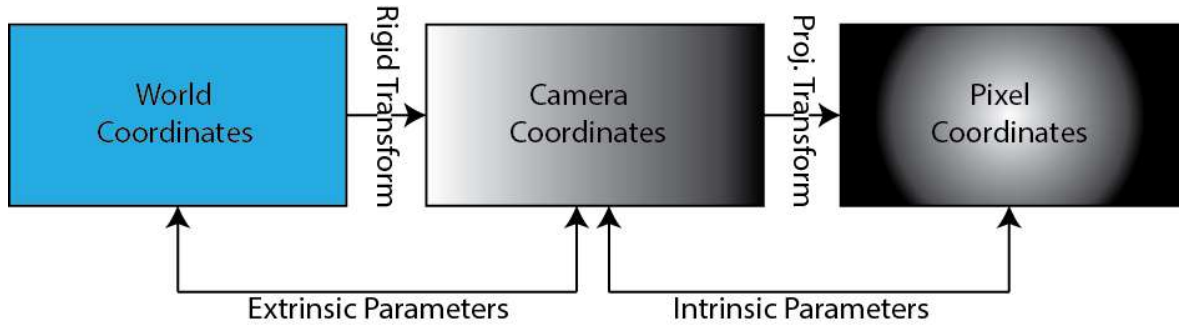


Figure 5.14. Image-Camera-World Transformation

Image Projection (Intrinsic Camera Parameters)

The image point (u, v) is related to the camera frame via the intrinsic camera matrix (\mathcal{C}) also known as the camera internal parameters given as:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathcal{C} \begin{bmatrix} x_c \\ y_c \\ f \end{bmatrix} \quad (5.3)$$

Camera Perspective Projection

This projects a scene's 3D feature point $P = (X_e, Y_e, Z_e)$ to a 2D point $p = (x_c, y_c)$ on the image plane and is given as a linear mapping in homogeneous coordinates as:

$$\begin{bmatrix} x_c \\ y_c \\ f \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_e \\ Y_e \\ Z_e \\ 1 \end{bmatrix} \quad (5.4)$$

World Transformation

This represents the external parameters of the camera system. The relationship between the camera and the world coordinates is given by the extrinsic camera matrix which comprises of conventional rigid body rotation and translation.

Given the world coordinates point of interest as $P = (X_w, Y_w, Z_w)$, this point is projected to the camera frame using:

$$\begin{bmatrix} X_e \\ Y_e \\ Z_e \\ 1 \end{bmatrix} = \begin{bmatrix} R & T \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \quad (5.5)$$

The matrix R is a rotation matrix. In the case where the quadrotor is in hover, R is an identity matrix. T stands for the translation and determines image depth which is related to the quadrotor's altitude.

World-Camera-Image Transformation

Finally, by concatenating the matrices Equations 5.3, 5.4 and 5.5, the image to world projection is given by:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathcal{C} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & T \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \quad (5.6)$$

This can be written in closed form as:

$$\mathbf{x} = \mathbb{P}\mathbf{X} \quad (5.7)$$

where $\mathbf{x} = [u, v, 1]^T$, $\mathbb{P} = \mathcal{C}[R|T]$.

Dimensions X_w and Y_w represent the desired final location of the quadrotor and this position is obtained through transformations from the centroid (u, v) of the sensor capsule's image to the world coordinates.

5.4.4 Image Processing

The image processing phase deals with the extraction of visual feedback information for implementation in the control subsystem. Matlab and OpenCV with Python was used to achieve this task. The color features of the sensor capsule's cap in the *RGB* format gives values $R = 0, G = 0, B = 255$, corresponding to an *HSV* value of $H = 240^\circ, S = 100\%, V = 100\%$. It is easier to perform color filtering and segmentation using *HSV* color space. The feedback from the sensor however (Figure 5.15) would not perfectly retain these values due to lighting and other disturbances. Figure 5.16. highlights the pixel values from the vision sensor. The choice of a blue colored sensor capsule stems from its uniqueness with the environment. Green is associated to foliage and Red is somewhat similar to soil while blue is the remaining primary color with no direct relation with the ground under consideration.

5.4.4.1 Image Processing Algorithm

The image processing algorithm is described below:

- Obtain Image (I_m) from Vision sensor
- Remove distortion and perform Gaussian filtering
- Color Segmentation in *HSV* format
- Image thresholding and Connected component analysis
- Blob Analysis to segment
- Object (sensor cap) detection
- Obtain Centroid of the Object (x_a, y_a) and Center of Image Plane (x_b, y_b)
- Calculate the vector displacement from the center of the image plane.

5.4.4.2 Implementation

The first step in implementing the vision subsystem is to obtain the image (I_m) from the vision sensor (Figure 5.15). Thereafter, I_m is blurred with a 2-D Gaussian kernel of variance $\sigma = 5$ to remove exogenous and random feature noise (as shown in Figure 5.17).

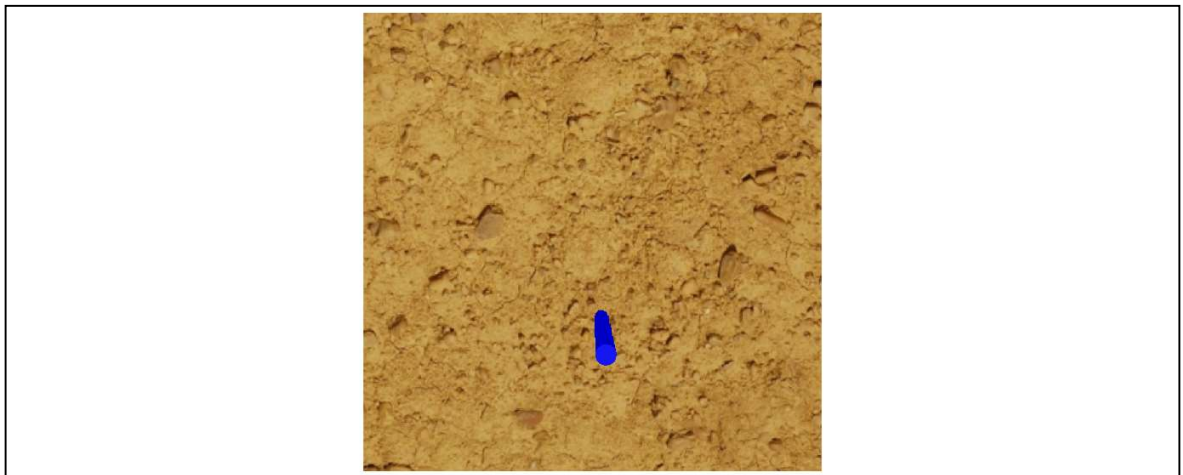


Figure 5.15: Vision Sensor Image of Sensor Capsule

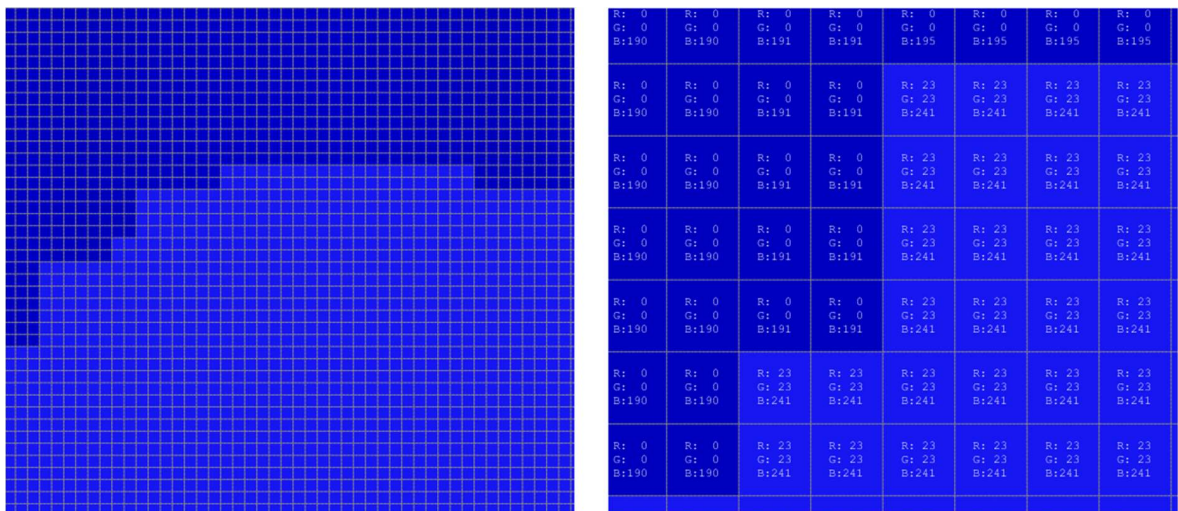


Figure 5.16: Pixel Intensities and Color on Object Blue Pixel Region.

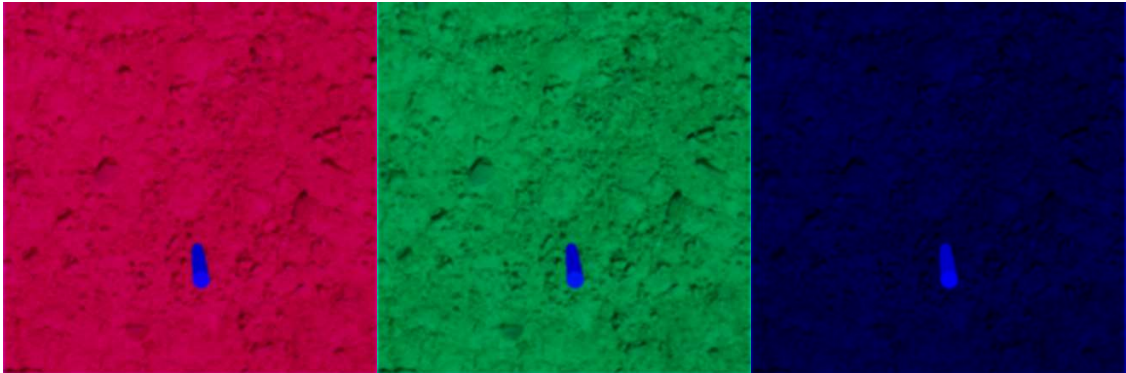


Figure 5.17: Red, Green and Blue Components of the Scene.

After the blurring process, the image is filtered in *HSV* color space (which handles color filtering better) and segmentation of the sensor capsule is achieved (Figure 5.18 and Figure 5.19) with only the blue channels remaining.



Figure 5.18. (Left to Right): Color Segmentation, Binarization and Centroid Detection.

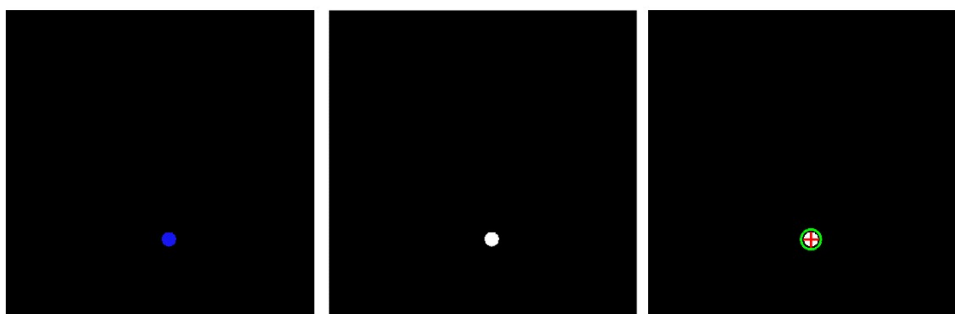


Figure 5.19 (Left to Right): Color Segmentation, Binarization and Centroid Detection in the Absence of Perspective Distortion

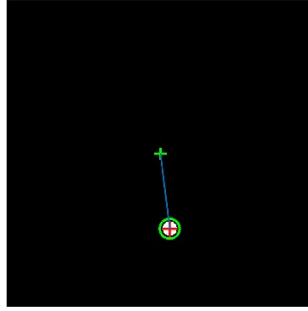


Figure 5.20: Vector Between Center of Image Plane and Object Centroid.

Since the location of the object's centroid has been found in Figure 5.20, the transformations in Section 5.4.3 is used to map the image features from pixels to equivalent world coordinates.

5.4.5 Vision Feedback to Controller

The goal now, is to drive the quadrotor from location (x^*, y^*) to location (x, y) thereby minimizing the displacement ε between the position of the quadrotor from memory and the actual position of the sensor capsule. This error is minimized by transferring the sensor capsule position coordinates as set-points to the control subsystem. Image-to-World transformation is used to specify the object position features in world coordinates. If object not found, send feedback to increase altitude which increases the image view. Similarly, if object still not found, the FSM can be modified with a loiter mode to search vicinity.

In this work, we assume the position sensor error is small enough such that ε is bounded and subsequently, the sensor capsule can be located as an image object while the quadrotor is at location (x^*, y^*) due to its large FOV. In the last step of the visual feedback system, the obtained image features and parameters are converted to world coordinates useful for the quadrotor using the transformations described in Section 5.4.3.

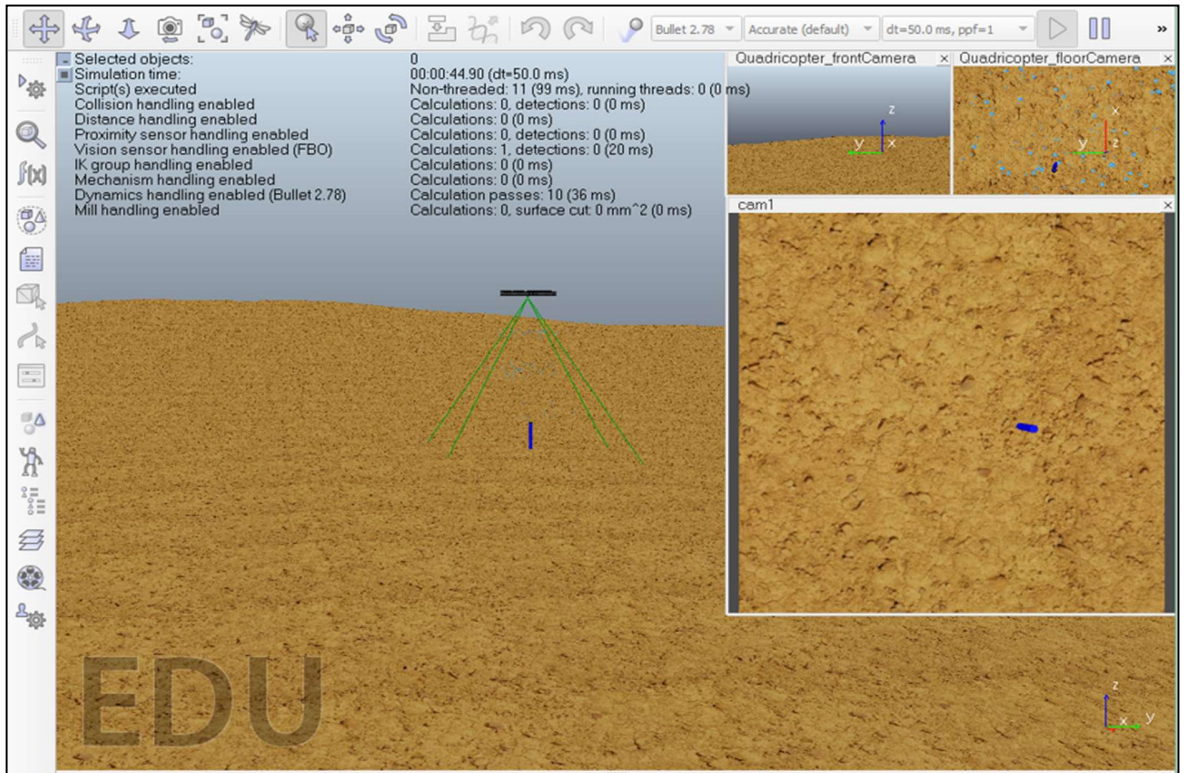


Figure 5.21: Quadrotor with Vision Sensor Detecting Sensor Capsule

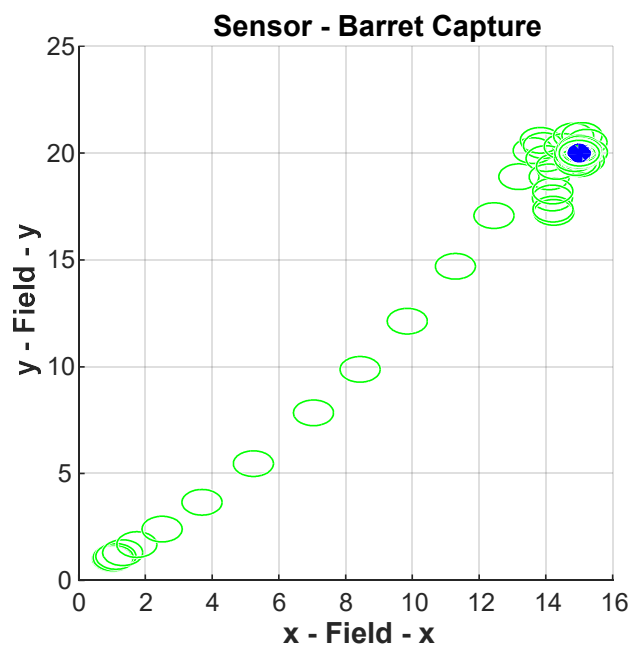


Figure 5.22: Quadrotor Stabilized at $(x = 15, y = 20)$; Barret Hand ZOR (Green Circles) Coincident with the Sensor Capsule (Blue) for Successful Retrieval.

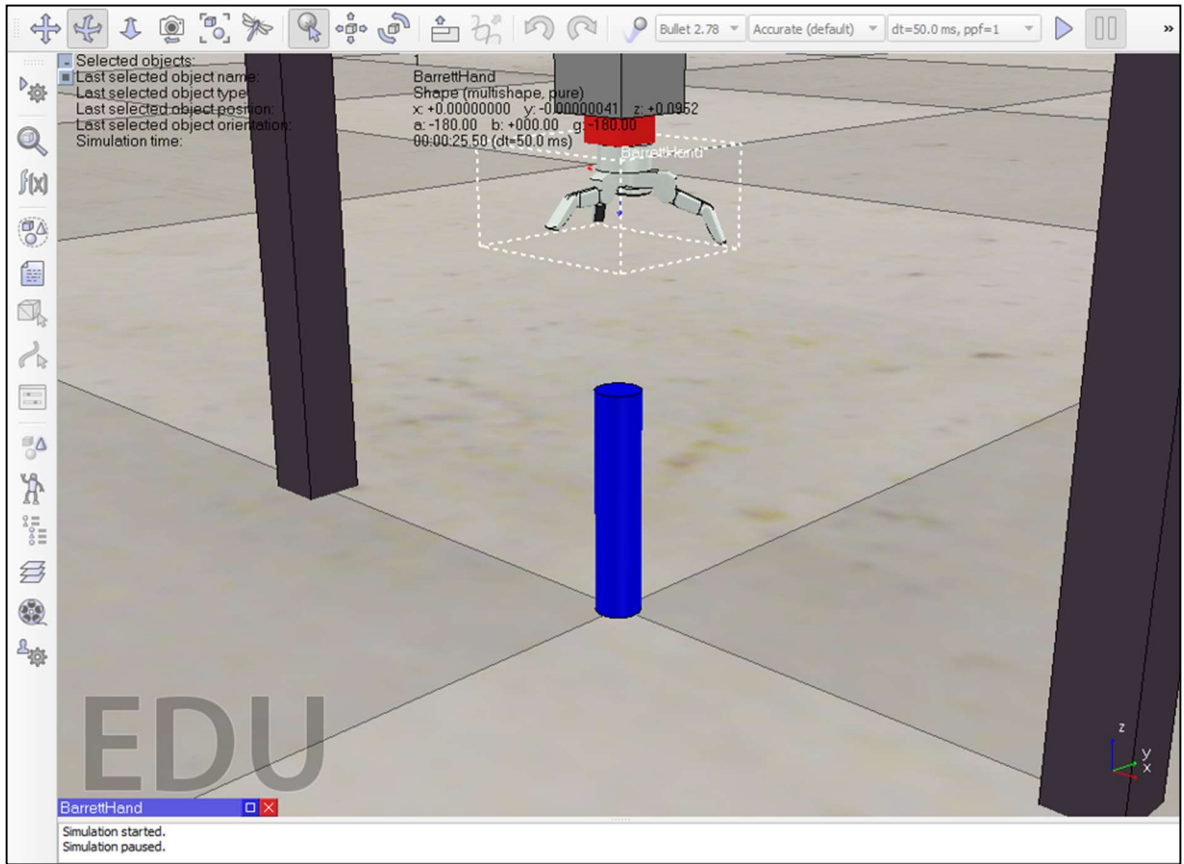


Figure 5.23: Quadrotor-Decoupled Barret Hand Mechanism.

CHAPTER 6

SIMULATION, RESULTS, VIRTUALIZATION

6.1 Introduction

This chapter presents the overall results of this thesis work. Computer simulation tools were used to verify the controllers and visualize the performance of the combined system. The results of controller performance are first presented then visual simulation using a combination of robotics tools are presented.

6.2 Simulation Tools

In this section, the computer tools used in this work are highlighted. These tools include; Matrix laboratory (MATLAB), Virtual robotics experimentation platform (VREP) and OpenCV. These simulation tools were installed on a Windows operating system. Programming in Matlab requires knowledge of Matlab scripting, VREP is based on LUA but control scripts can be written with remote Matlab API. The OpenCV libraries were used under the Python programming framework.

6.2.1 MATLAB

Matlab is short for matrix laboratory and is one of the most powerful tools used by scientists and engineers for computation. The Simulink tool gives users the wide choice of development tools using predefined building blocks. In this chapter, the combination of Matlab m-file scripts, Simulink and Stateflow blocks were used to simulate the autonomous quadrotor system.

6.2.2 VREP

Virtual robotics experimentation platform (VREP) is a versatile robotics simulator developed by ‘coppeliarobotics’ and used in the design, development, verification and simulation of robotic systems. It is composed of scene objects, calculation modules and control mechanisms. Many in-built sensors and algorithms exist to enhance productivity for roboticists. It also has local and remote interfaces which enables modular development. The ability to link Matlab, ROS and VREP over a remote interface would be explored.

6.2.3 OpenCV

Open CV is an open source collection of computer vision libraries with C, C++, Python and Java interfaces. It is used in creative arts, photography and very much applied in robotics. It has an open feature that allows contributions by the computer vision community with tons of functionalities and contributions.



Figure 6.1: Simulation Tools.

The quadrotor is set to track an altitude of $5m$ with x and y positions of $5m$. Likewise, roll, pitch and yaw angles of 5° are to be tracked by the rotational subsystems. The quadrotor parameters are given in Table 6.1.

Table 6.1: Quadrotor Model Parameters.

No.	Quadrotor Model Parameters		
	Parameter	Symbol	Value
1	Mass	$m(kg)$	1.6
2	Rotor radius	$r(m)$	0.1905
3	Sensor noise	$\sigma_z^2(m)$	0.14^2
4	Drag coefficient (Translation)	$k_{t=x,y,z}$	0.3729
5	Drag coefficient (Rotation)	$k_{r=\phi,\theta,\psi}$	0.3729
6	Ground effect coefficient	ρ	8.6
7	Gravitational constant	$g(ms^{-2})$	9.81
8	Principal Inertial Matrix (x)	I_{xx}, \bar{I}_{xx}	0.018, 0.053
9	Principal Inertial Matrix (y)	I_{yy}, \bar{I}_{yy}	0.008, 0.053
10	Principal Inertial Matrix (z)	I_{zz}, \bar{I}_{zz}	0.035, 0.091
11	Change in Center of Mass	z_G	0.08
12	Rotor Inertia	J_r	6×10^{-4}
13	Length of Quadrotor cross-axis	l	0.45
14	Mass of Sensor Capsule	$m(kg)$	0.2

6.3 Results: Backstepping Controller

The results obtained in this section are based on the quadrotor models in Section 3.4.3 and Section 3.6 with backstepping formulation of Section 4.2. The optimized controller parameters are given in Table 6.2.

Table 6.2: Backstepping Controller Parameters.

No.	Backstepping Controller Parameters	
	<i>Parameter</i>	<i>Value</i>
1	p_1	11.52
2	p_2	8.40
3	p_3	8.00
4	p_4	7.50
5	p_5	13.6263
6	p_6	13.5392
7	p_7	2.54
8	p_8	5.49
9	k_{px}	4.9
10	k_{py}	4.9
11	k_{dx}	7.3
12	k_{dy}	8.3

6.3.1 Quadrotor Backstepping Control I

The first part of the results presents the quadrotor dynamics under backstepping control and without disturbances (Plant A). The $x - y$ under-actuated dynamics is also controlled using the formulation presented in Section 4.2.1 - 4.2.4. It is verified that the control of the roll and pitch subsystems is the necessary condition for the control of the $x - y$ dynamics.

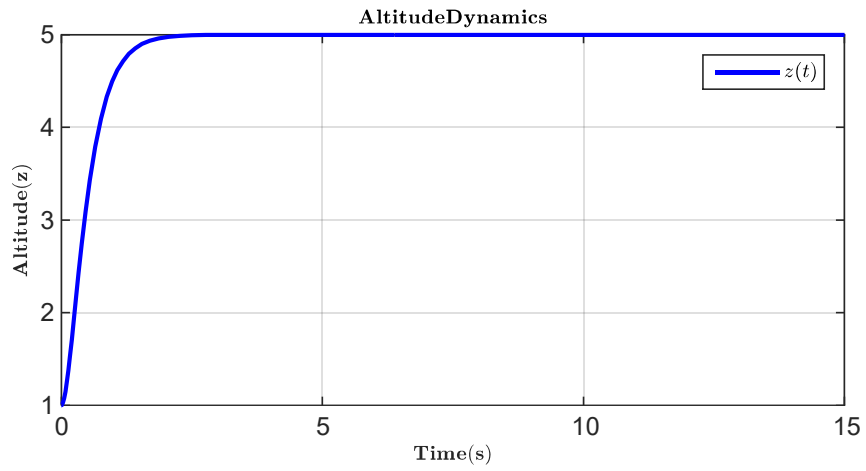


Figure 6.2 Altitude Dynamics (Backstepping)

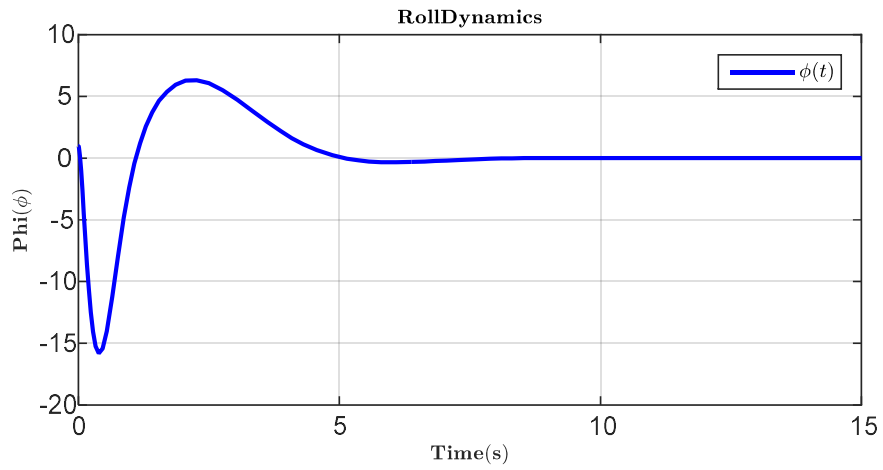


Figure 6.3 Roll Dynamics Stabilized for y Translation (Backstepping)

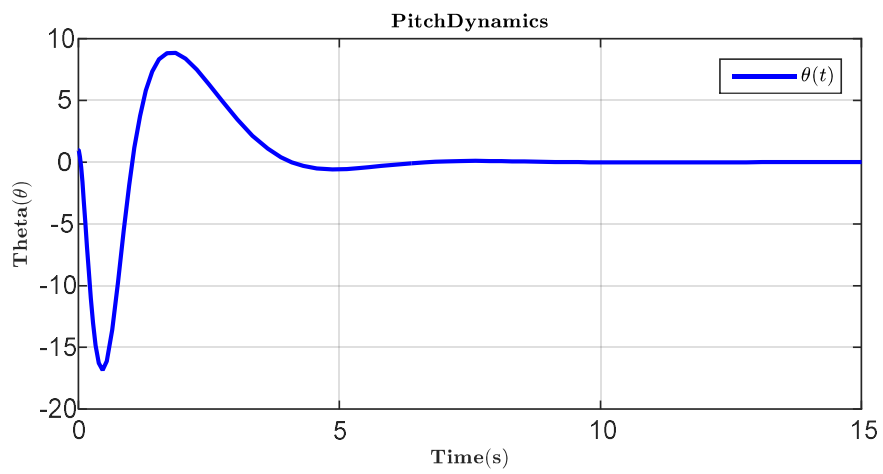


Figure 6.4 Pitch Dynamics Stabilized for x Translation (Backstepping)

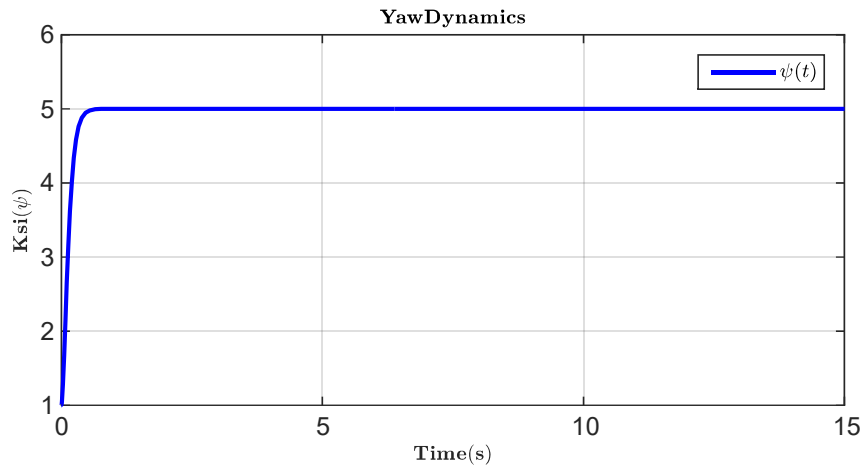


Figure 6.5 Yaw Dynamics (Backstepping)

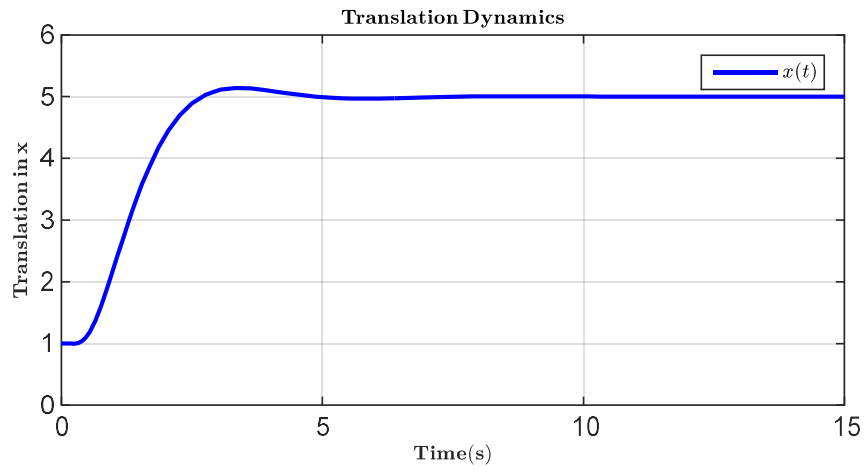


Figure 6.6 Dynamics of x Translation (Backstepping)

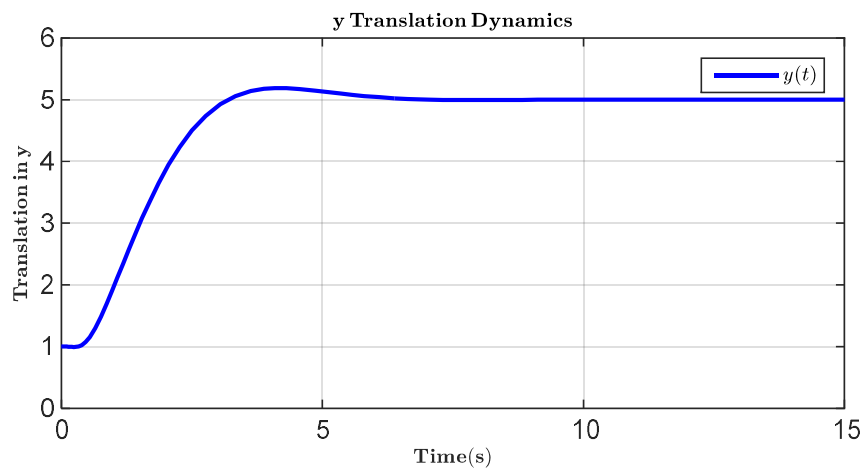


Figure 6.7 Dynamics of y Translation (Backstepping)

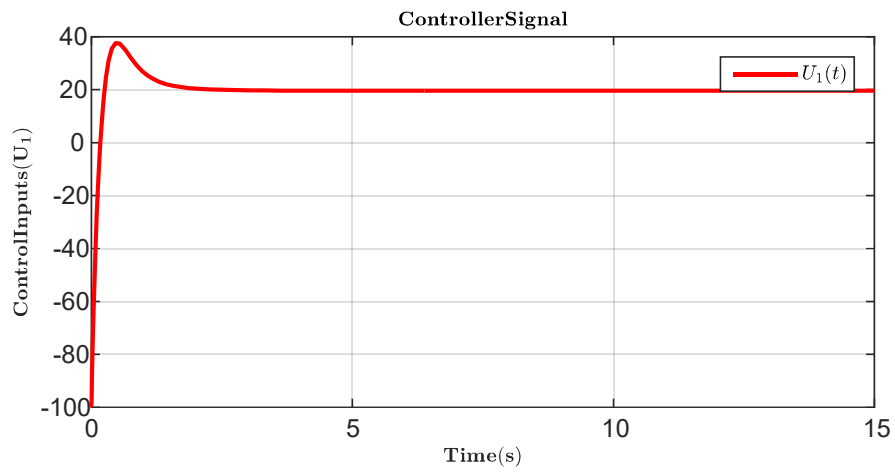


Figure 6.8 U_1 Control Input (Backstepping)

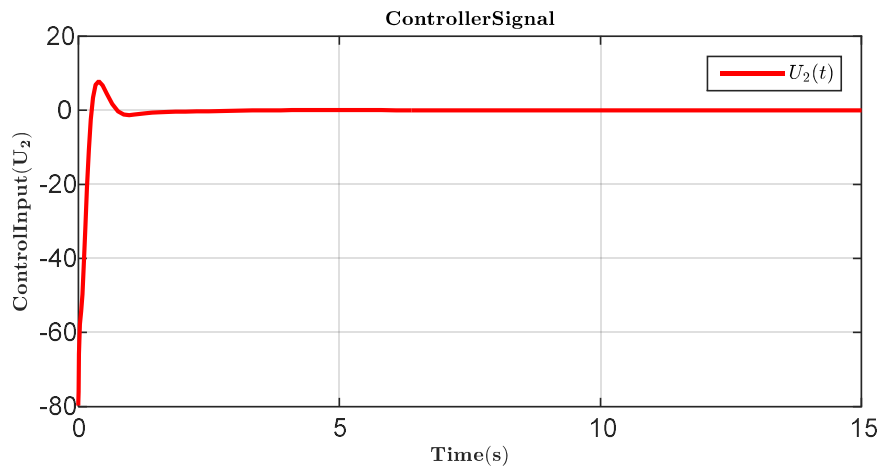


Figure 6.9 U_2 Control Input (Backstepping)

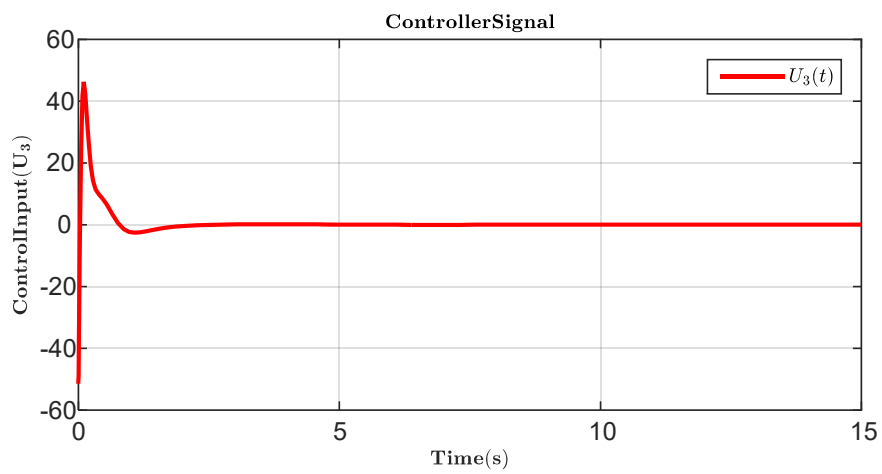


Figure 6.10 U_3 Control Input (Backstepping)

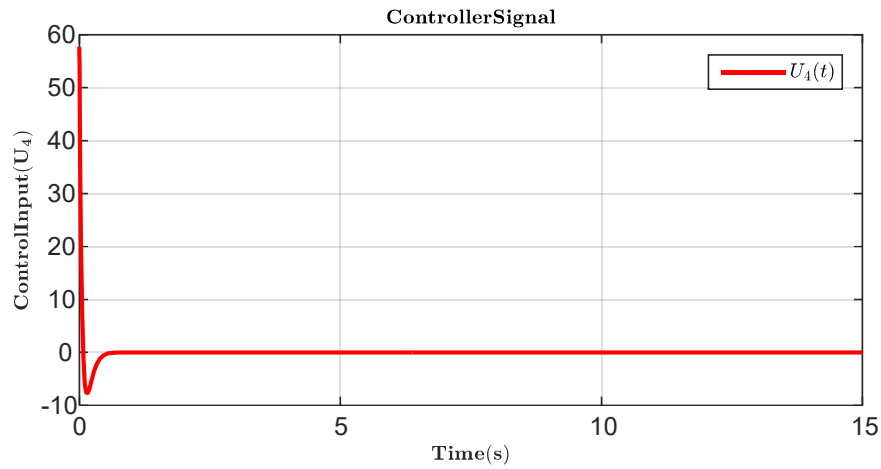


Figure 6.11 U_4 Control Input (Backstepping)

Figure 6.2 to 6.7 shows the quadrotor states. The backstepping controller inputs are presented in Figure 6.8 to 6.11 are the respective control inputs. The tracking ability of the optimized controller is very good. The results also confirm that the stabilizing controller presented in Section 4.2.3 is sufficient to drive the quadrotor to the required $x - y$ states.

6.3.2 Quadrotor Backstepping Control II

This section of the results presents the quadrotor dynamics under backstepping control including dynamic disturbances (Plant B). The $x - y$ under-actuated dynamics is also controlled using the formulation presented in Section 4.2.5. The following plots show the altitude, roll, pitch and yaw quadrotor manipulator dynamics under backstepping control.

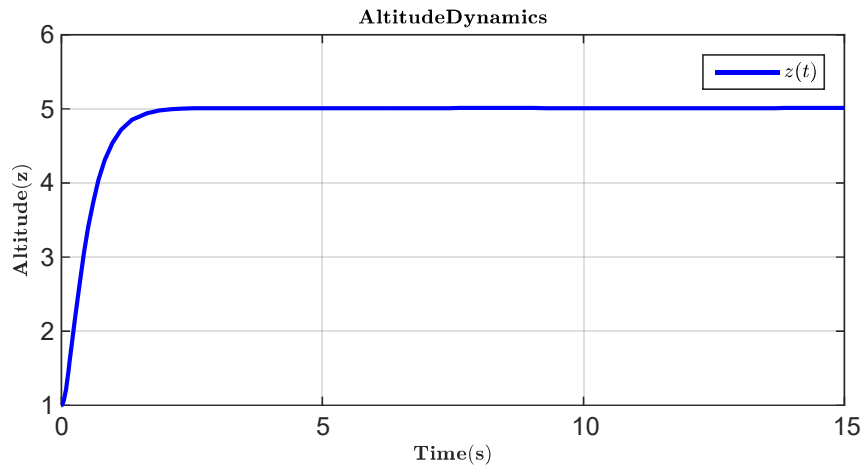


Figure 6.12 Altitude Dynamics (Backstepping Plant B)

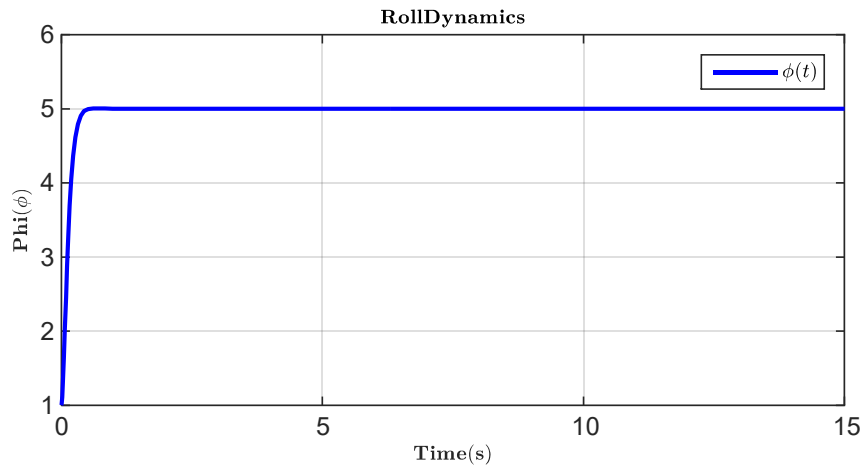


Figure 6.13 Roll Dynamics (Backstepping Plant B)

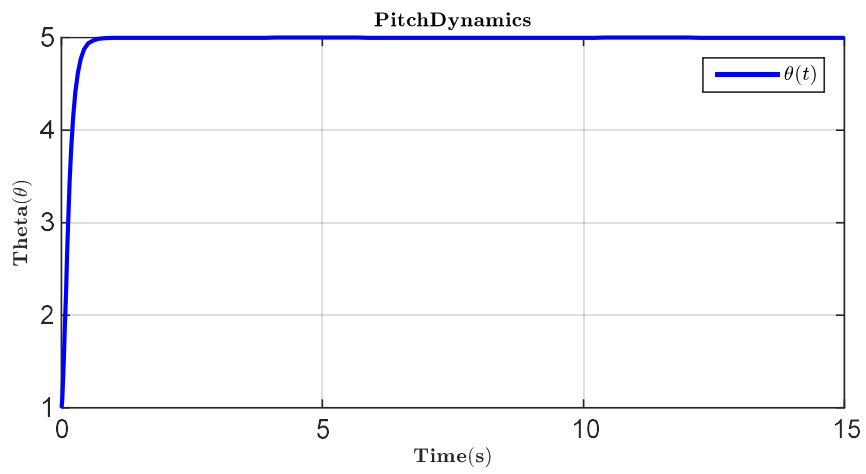


Figure 6.14 Pitch Dynamics (Backstepping Plant B)

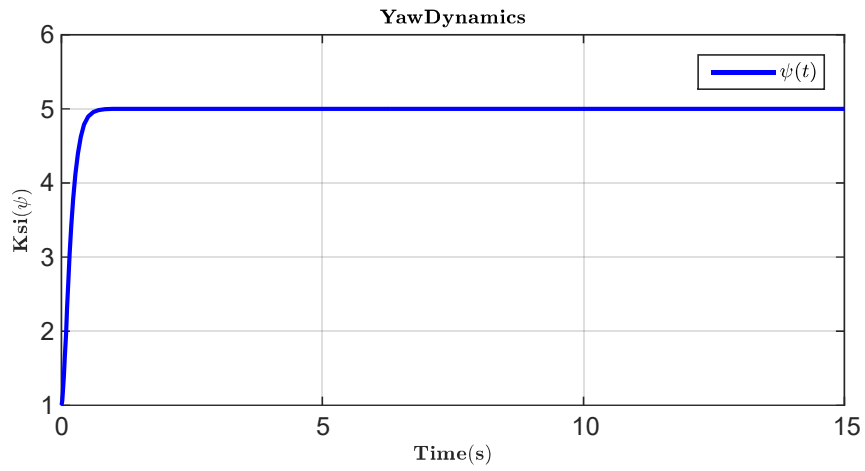


Figure 6.15 Yaw Dynamics (Backstepping Plant B)

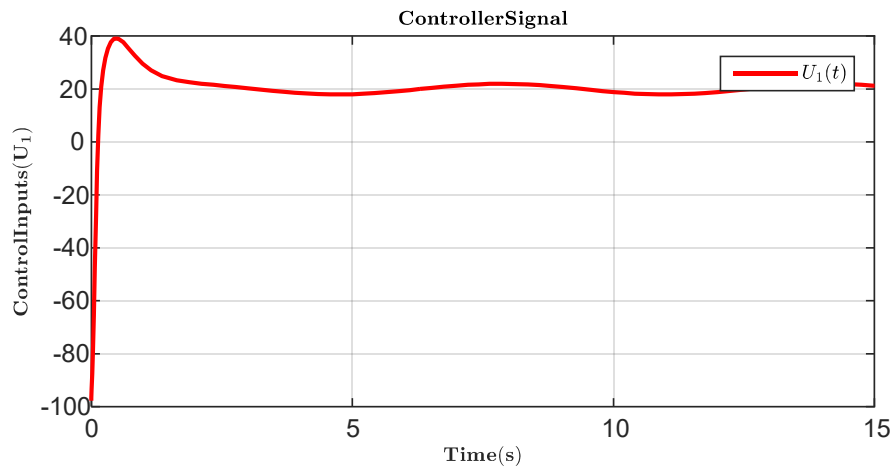


Figure 6.16 U_1 Control Input (Backstepping Plant B)

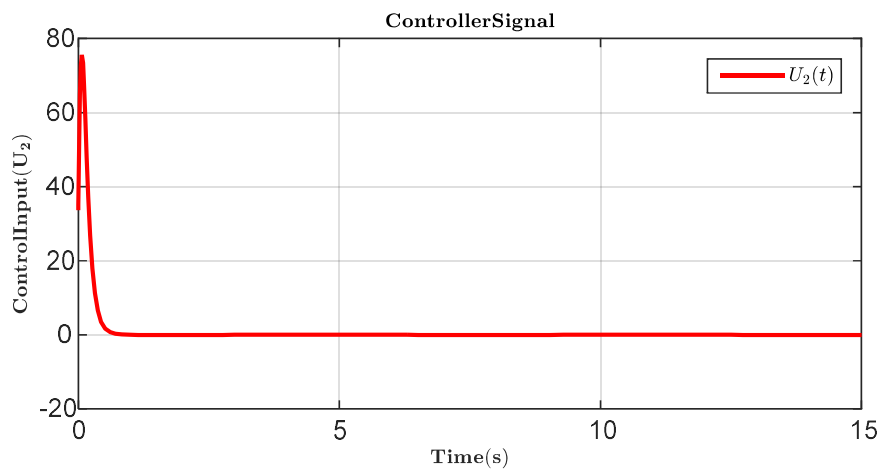


Figure 6.17 U_2 Control Input (Backstepping Plant B)

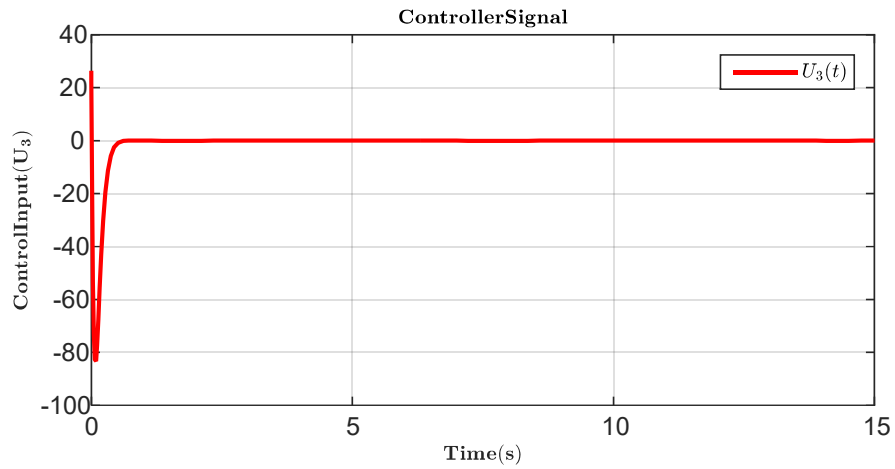


Figure 6.18 U_3 Control Input (Backstepping Plant B)

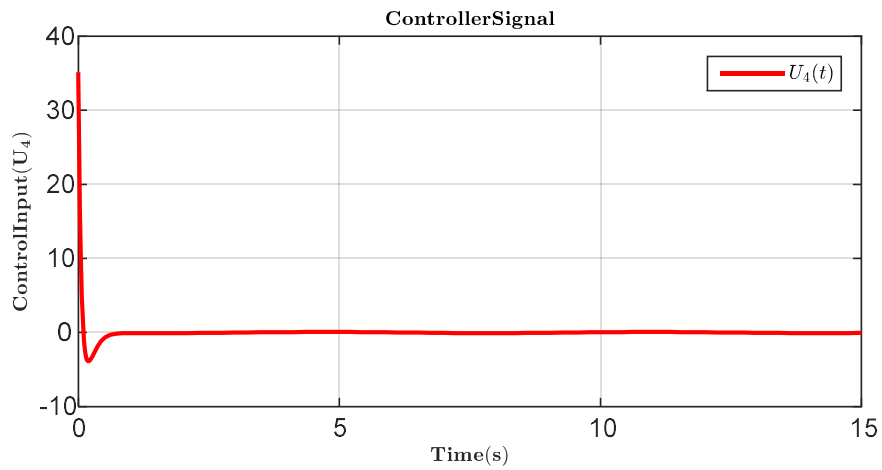


Figure 6.19 U_4 Control Input (Backstepping Plant B)

Figure 6.12 to 6.15 show the fully-actuated states of the quadrotor manipulator system under the influence of previously discussed dynamics disturbances (Plant B). The optimized controller works well in tracking the set-points.

6.3.3 Sensor Capsule Handling

Figure 6.20 and 6.21 show the effect of sensor capsule drop after 5 seconds on the quadrotor-manipulator dynamics. The effect was noticeable on the altitude dynamics as a kink in altitude at $t = 5$ s.

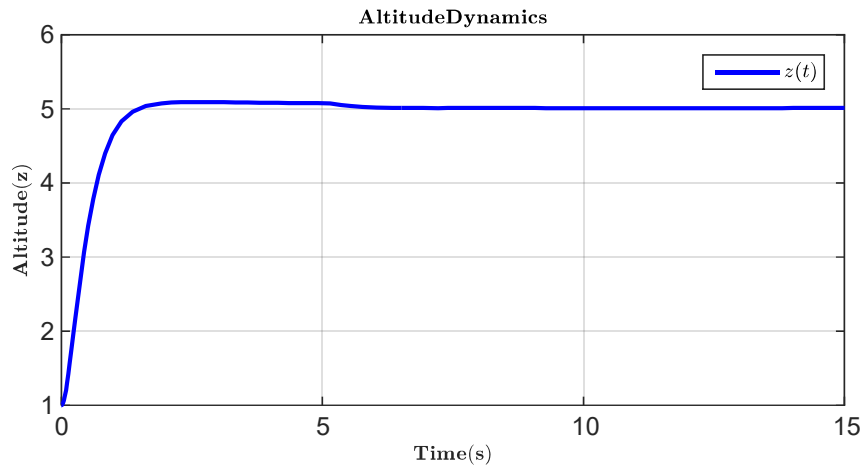


Figure 6.20 Effect of Sensor Capsule drop on Altitude Dynamics at $t = 5$ s

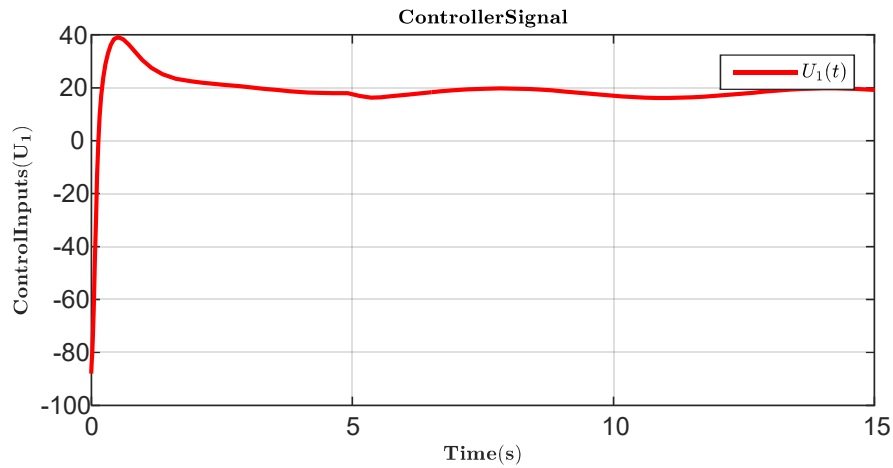


Figure 6.21 U_1 Control Input during drop

Figure 6.20 shows the effect of dropping the sensor capsule after 5 seconds. There is a kink at $t = 5$ s and the quadrotor-manipulator doesn't recover tracking the appropriate set-point. Similar kink can be found in the control signal Figure 6.21.

On the other hand, Figure 6.22 presents the results during load pick-up. The quadrotor lifts off from 0.15m (landing skids). A resisting force of 0.98N (Equation 6.1) was added to represent the breakaway coupling between ground and capsule that the controller must be

overcome. The quadrotor mass changes from 1.8 to 2kg and the inertial properties change too after $t = 1$ s.

$$f_{\text{resistance}} = \begin{cases} 0.98N, & t < 1 \\ 0, & t > 1 \end{cases} \quad (6.1)$$

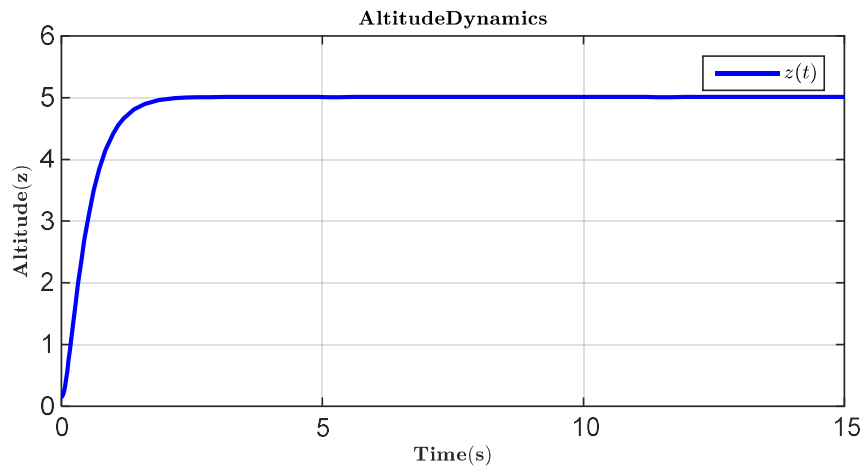


Figure 6.22 Effect of Lifting the Sensor Capsule on the Quadrotor Dynamics

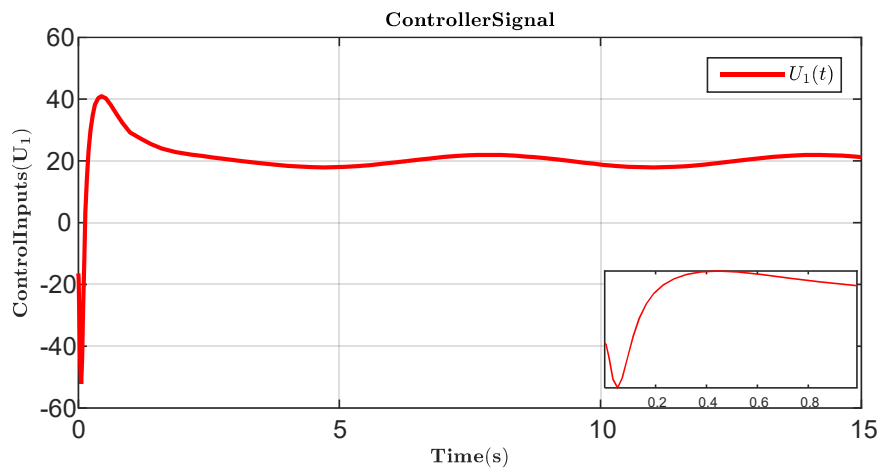


Figure 6.23 U_1 Control Input During Drop (Inset: Initial Reaction)

During the load pick-up phase, the backstepping controller works very well to overcome ground effect at initial height 0.15m of the landing skids and the force coupling between

the sensor capsule and ground. The inset in Figure 6.23 show the reaction of the backstepping controller between times 0 – 1s.

6.4 Results: Active Disturbance Rejection

The results obtained in this section are based on the quadrotor model in Section 3.6 with the ADRC controller formulation of Section 4.3. The ADRC controller parameters are given in Table 6.3.

Table 6.3: ADRC Controller Parameters

No.	ADRC Controller Parameters (Plant B)	
	<i>Parameter</i>	<i>Value</i>
1	p_1	29.5659
2	p_2	2907
3	p_3	3000
4	$k_{1\phi}$	100
5	$k_{2\phi}$	80.4975
6	$k_{1\theta}$	100
7	$k_{2\theta}$	21.5361
8	$k_{1\psi}$	69.8457
9	$k_{2\psi}$	16.8096
10	k_{1z}	10.5246
11	k_{2z}	9.5557
12	k_{1x}	3.9
13	k_{2x}	7.3
14	k_{1y}	3.9
15	k_{2y}	8.3

6.4.1 Active Disturbance Rejection Control I

The first set of results presented here is based on the application of ADRC controller on the quadrotor without disturbances. It is verified that the control of the roll and pitch angles is necessary for the control of the $x - y$ translation. The plots include states and estimated states, estimation of total disturbance and the controller inputs.

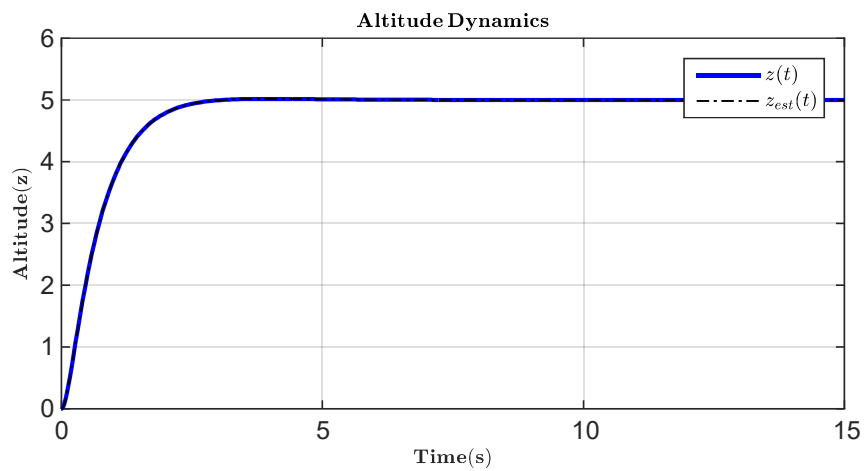


Figure 6.24 Altitude Dynamics (ADRC)

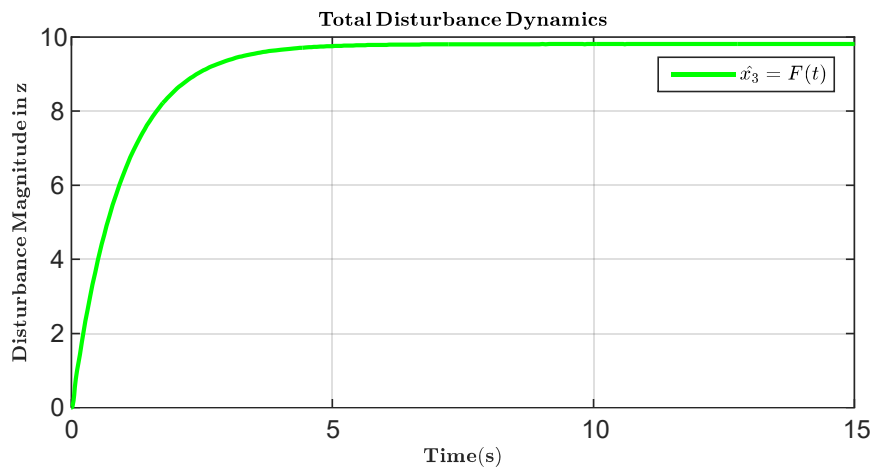


Figure 6.25 Estimation of Total Disturbance in Altitude Dynamics (ADRC)

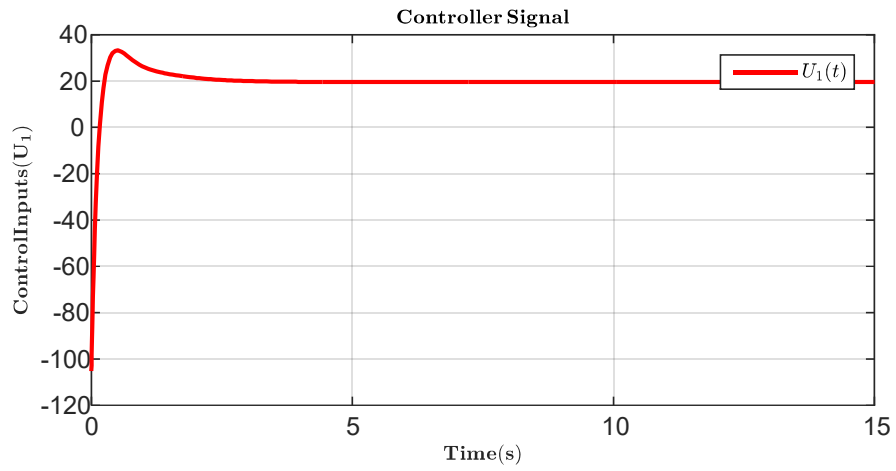


Figure 6.26 U_1 Control Input (ADRC)

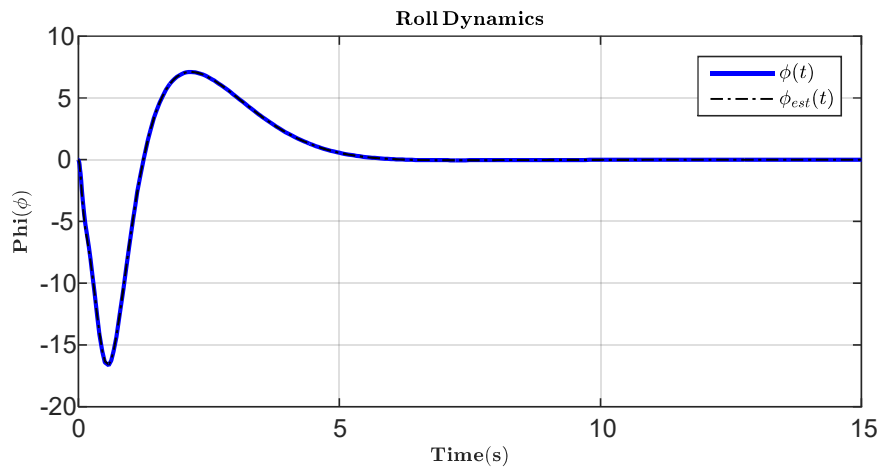


Figure 6.27 Roll Dynamics (ADRC)

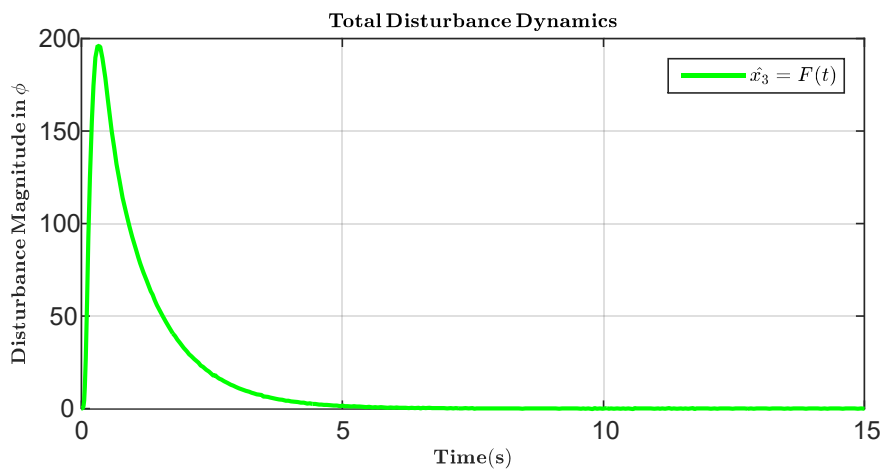


Figure 6.28 Estimation of Total Disturbance in Roll Dynamics (ADRC)

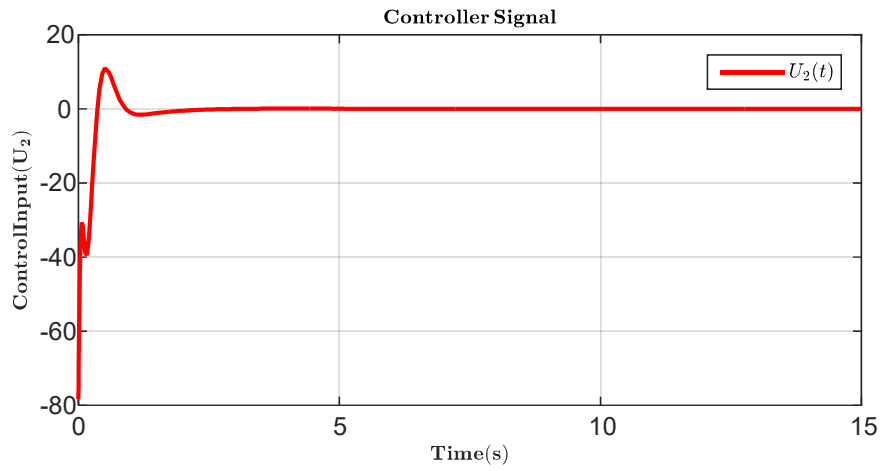


Figure 6.29 U_2 Control Input (ADRC)

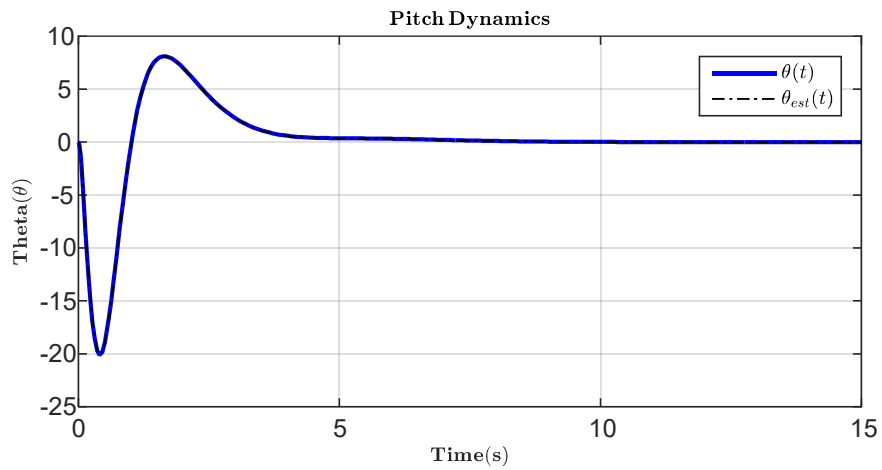


Figure 6.30 Pitch Dynamics (ADRC)

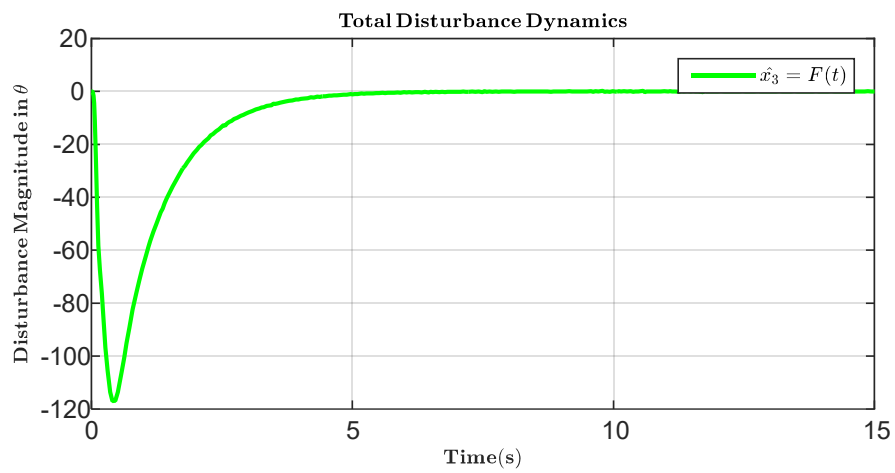


Figure 6.31 Estimation of Total Disturbance in Pitch Dynamics (ADRC)

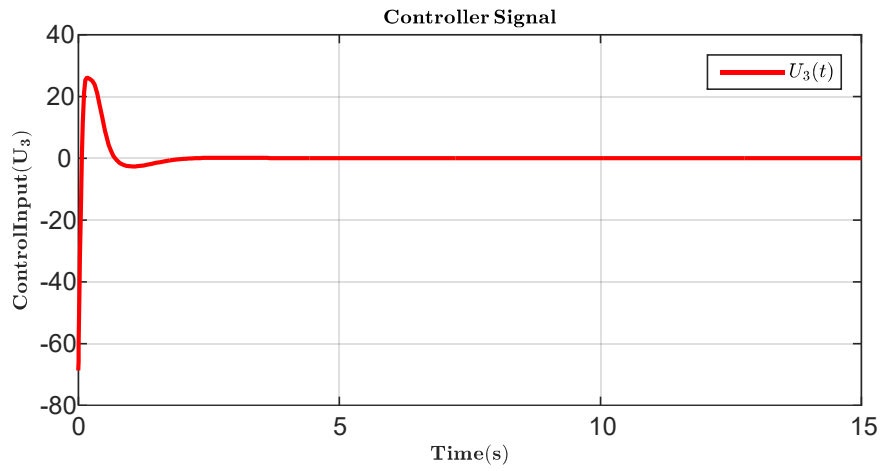


Figure 6.32 U_3 Control Input (ADRC)

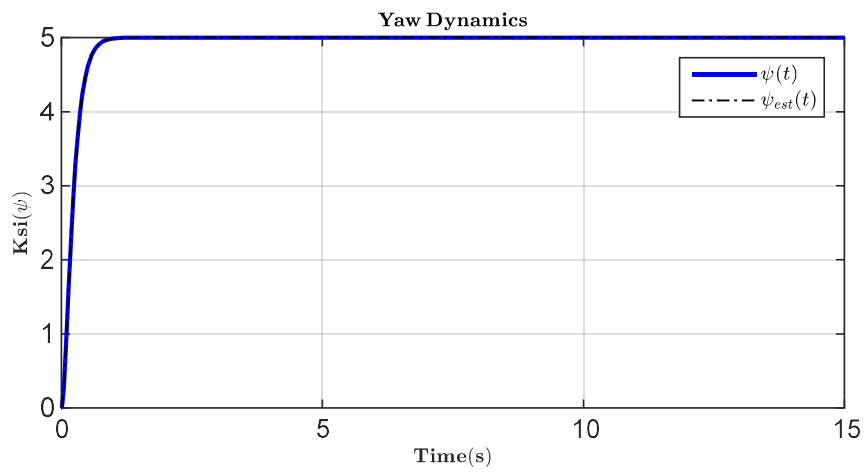


Figure 6.33 Yaw Dynamics (ADRC)

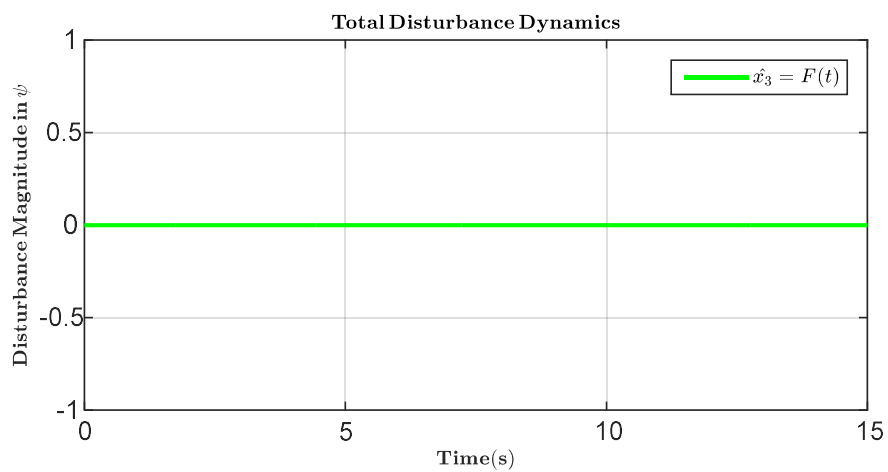


Figure 6.34 Estimation of Total Disturbance in Yaw Dynamics (ADRC)

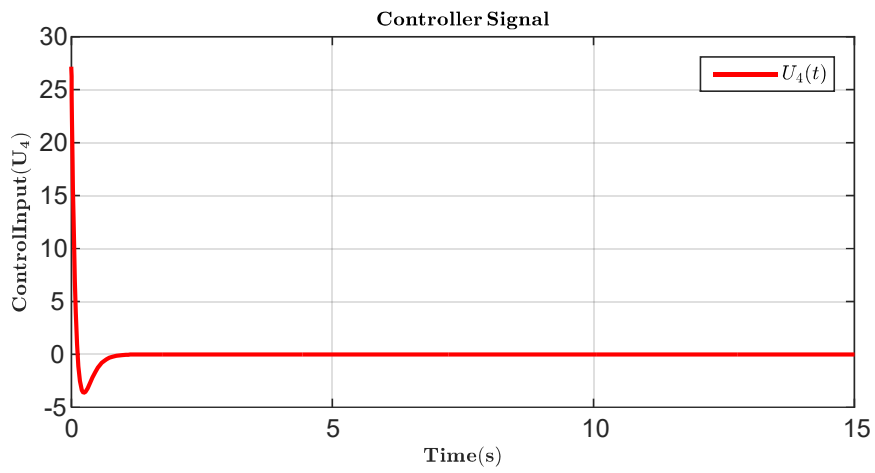


Figure 6.35 U_4 Control Input (ADRC)

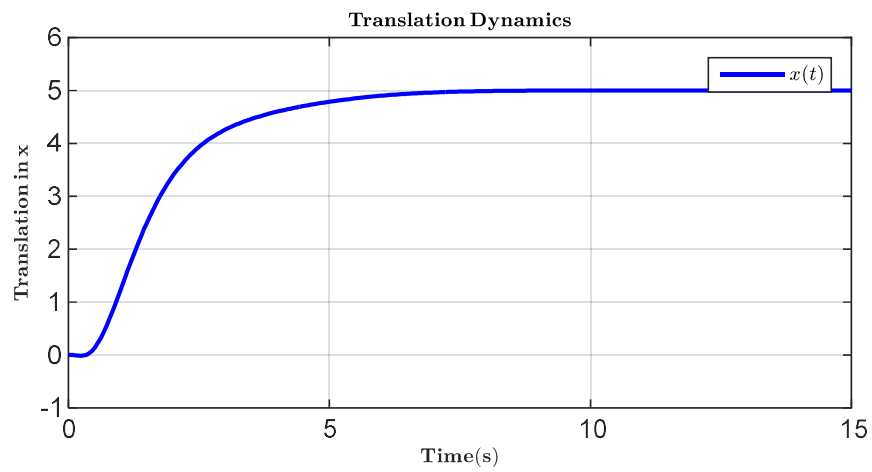


Figure 6.36 Dynamics of x Translation (ADRC)

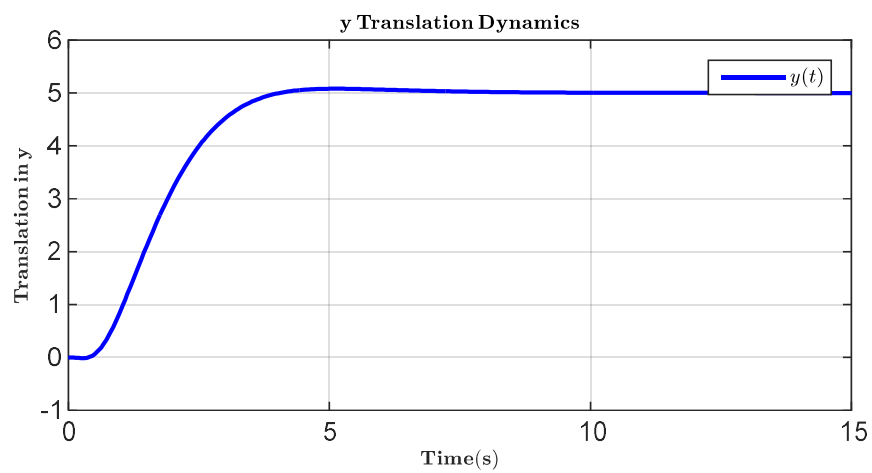


Figure 6.37 Dynamics of y Translation (ADRC)

In Figure 6.24 to 6.37, the results of the ADRC controller on the quadrotor is presented. Figure 6.24 is the altitude and estimated altitude of the quadrotor. Figure 6.25 is the estimate of the total disturbance in the altitude dynamics. It is clear from the equations of motion that this disturbance is equivalent to the effect of gravity. Figure 6.26 is the corresponding control input. The total estimated disturbance in Figure 6.34 is zero because $a_5 = (I_{xx} - I_{yy})/I_{zz} = 0$ in Equation 4.1.

6.4.2 Active Disturbance Rejection Control II

The plots here show the dynamics of the quadrotor manipulator system under ADRC controller in the presence of dynamic disturbances. The resisting force of Equation 6.1 is included. The wind disturbance parameters include $\alpha_i = 0.1, \beta = 0.25, n = 2$.

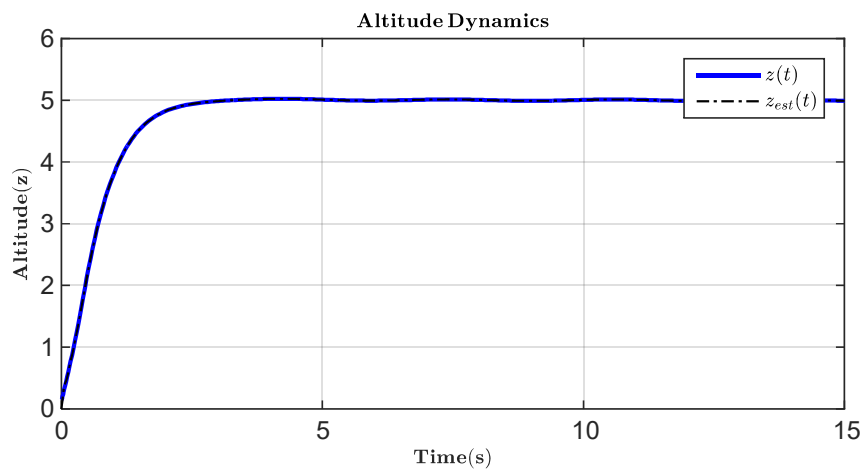


Figure 6.38 Altitude Dynamics (ADRC Plant B)

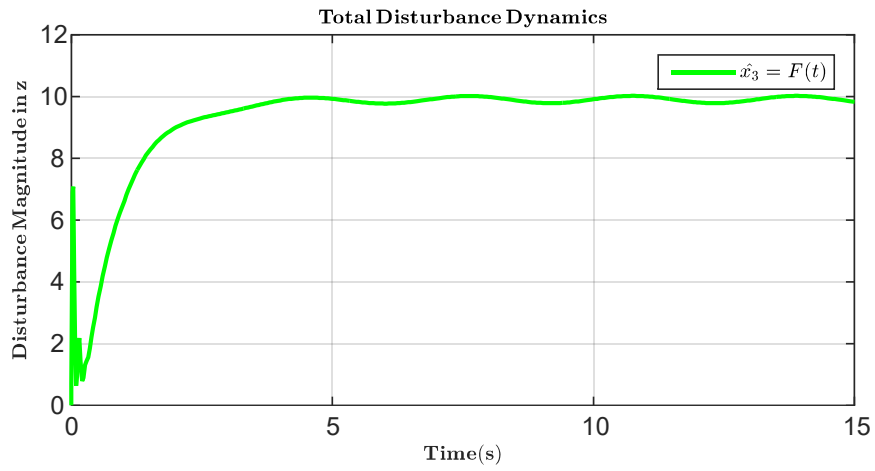


Figure 6.39 Estimation of Total Disturbance in Altitude Dynamics (ADRC Plant B)

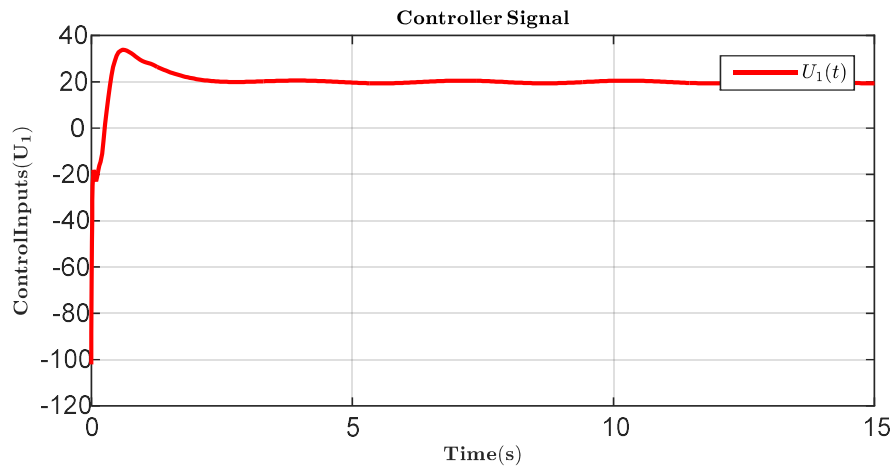


Figure 6.40 U_1 Control Input (ADRC Plant B)

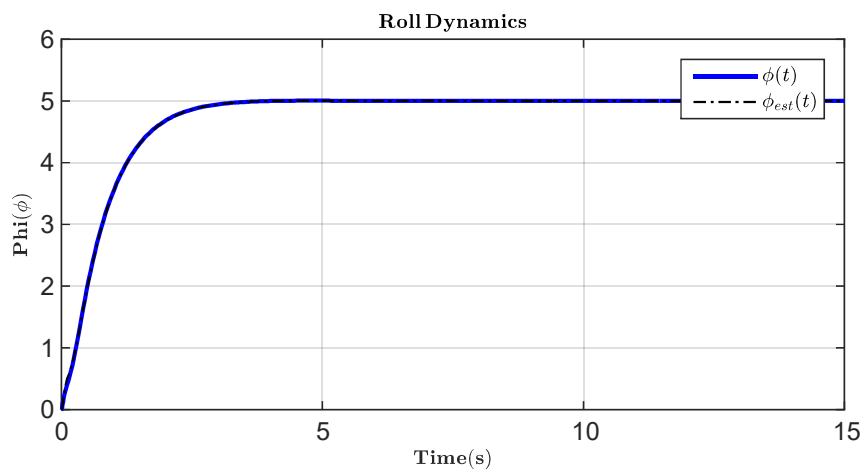


Figure 6.41 Roll Dynamics (ADRC Plant B)

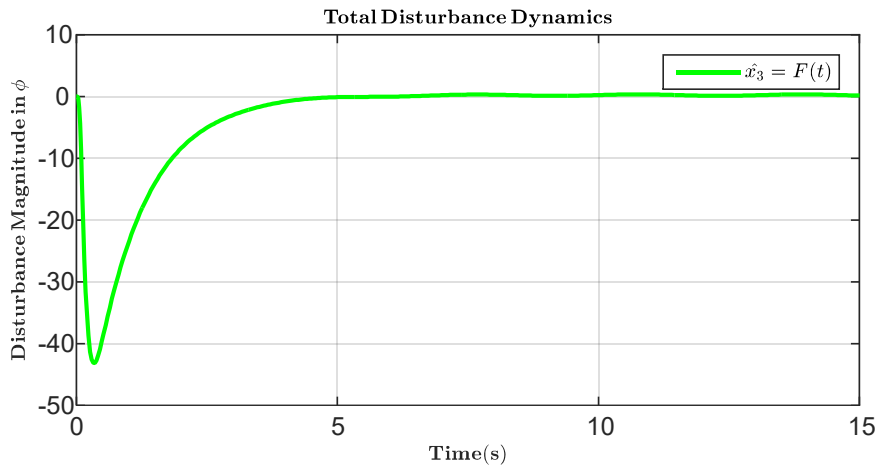


Figure 6.42 Estimation of Total Disturbance in Roll Dynamics (ADRC Plant B)

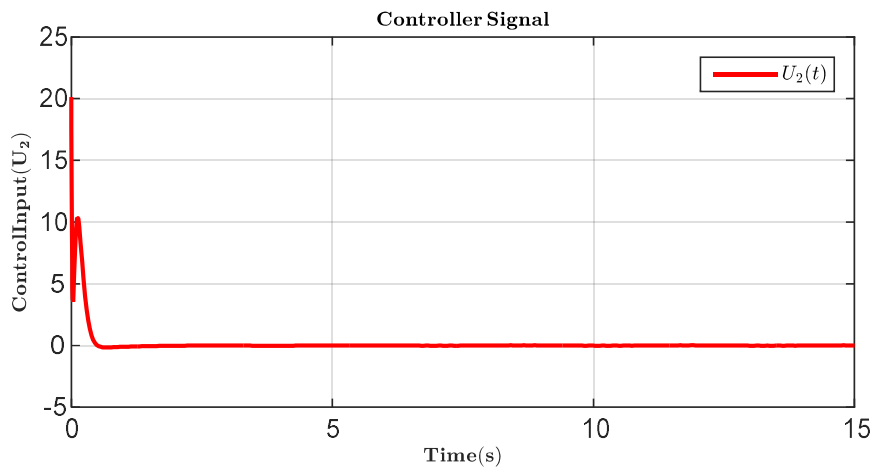


Figure 6.43 U_2 Control Input (ADRC Plant B)

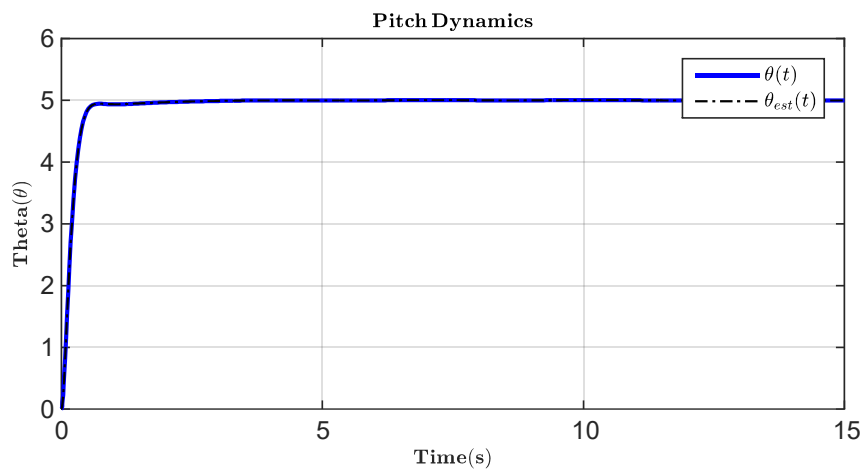


Figure 6.44 Pitch Dynamics (ADRC Plant B)

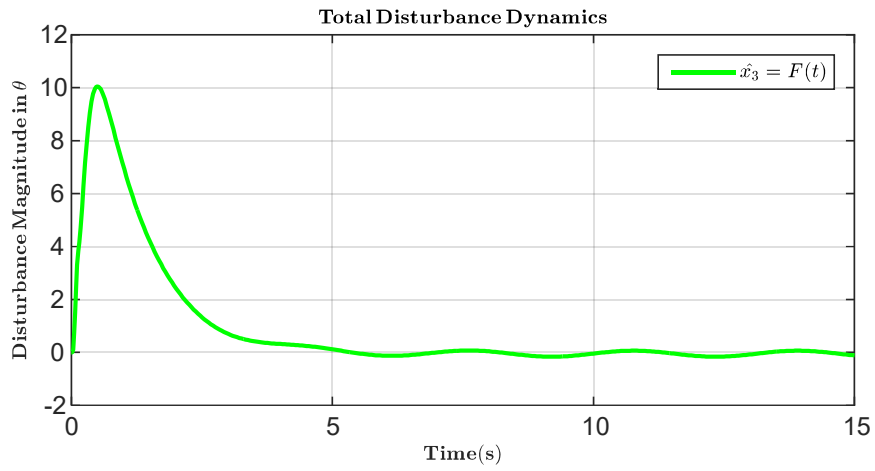


Figure 6.45 Estimation of Total Disturbance in Pitch Dynamics (ADRC Plant B)

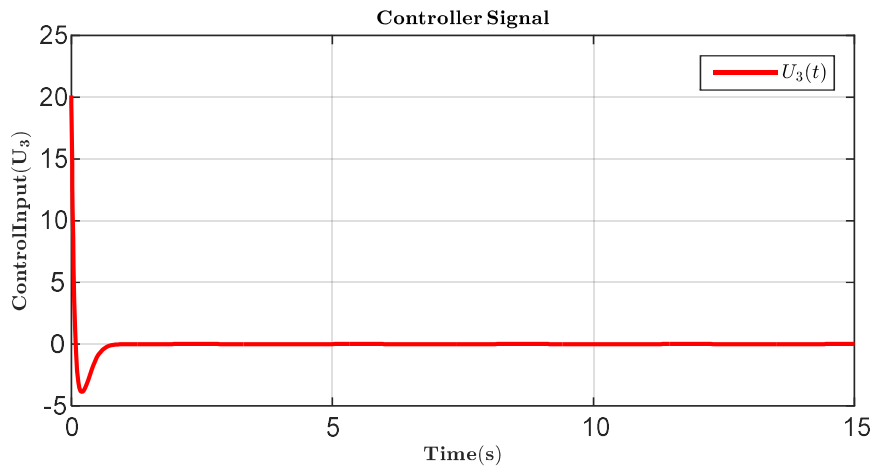


Figure 6.46 U_3 Control Input (ADRC Plant B)

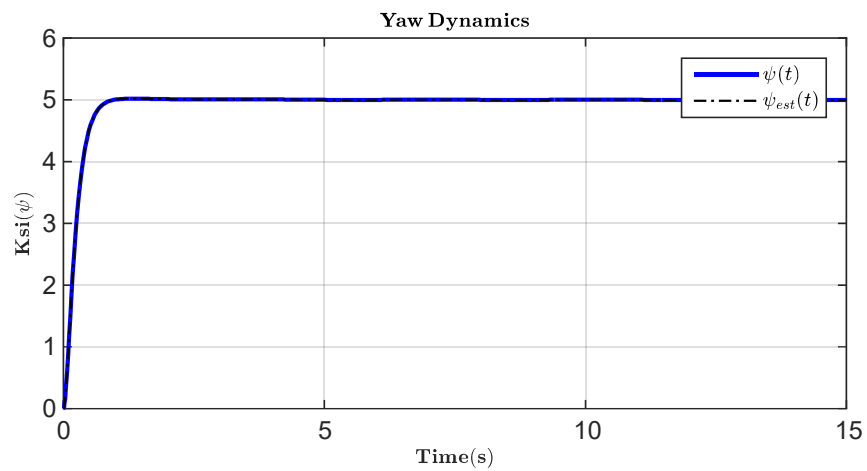


Figure 6.47 Yaw Dynamics (ADRC Plant B)

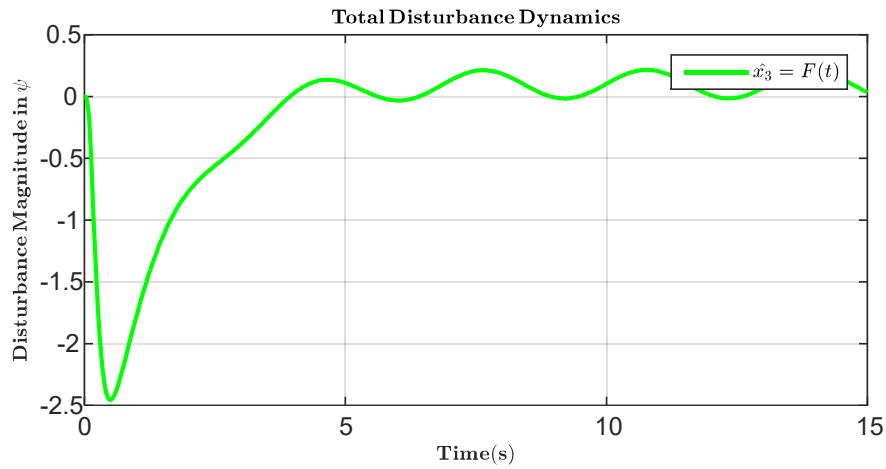


Figure 6.48 Estimation of Total Disturbance in Yaw Dynamics (ADRC Plant B)

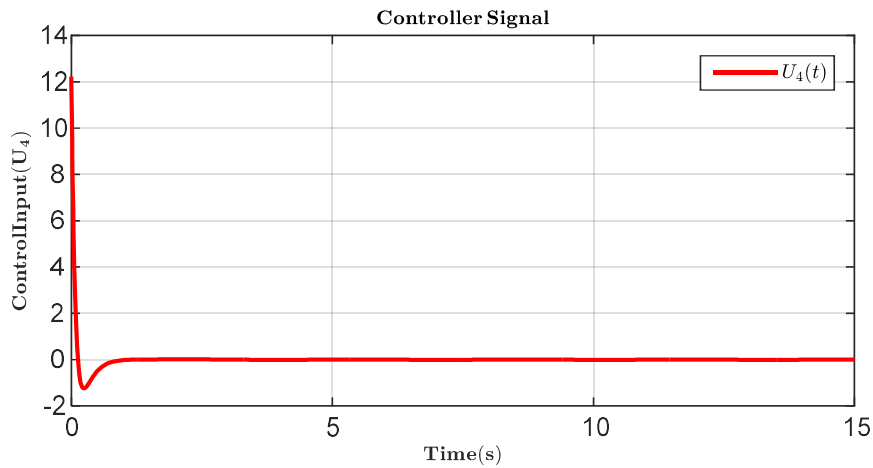


Figure 6.49 U_4 Control Input (ADRC Plant B)

Figures 6.38 to 6.49 are the ADRC controller results for the quadrotor manipulator system in the presence of disturbances (Plant B). Figure 6.38 shows the altitude dynamics of the quadrotor. The controller was able to attenuate sinusoidal wind disturbance. This can be observed in the estimation of total disturbance of Figure 6.39. Similarly, the controller's estimation of the total disturbance (wind, ground effect, drag, CoM effect) enabled good tracking and disturbance free performance of the quadrotor states. The effect of sensor capsule drop and pick up is totally compensated in the system dynamics.

6.4.3 Quadrotor Landing with Active Disturbance Rejection

The need for the quadrotor to pick up the sensor requires appropriate quadrotor landing, however, the ground effect, wind and aerodynamic drag previously discussed prevent smooth landings. The results presented in this section demonstrate quadrotor landing using ADRC on the altitude dynamics of the quadrotor. The noise attenuation modification technique was used to demonstrate the ability of the modified ADRC to handle the system in the presence of bounded noise.

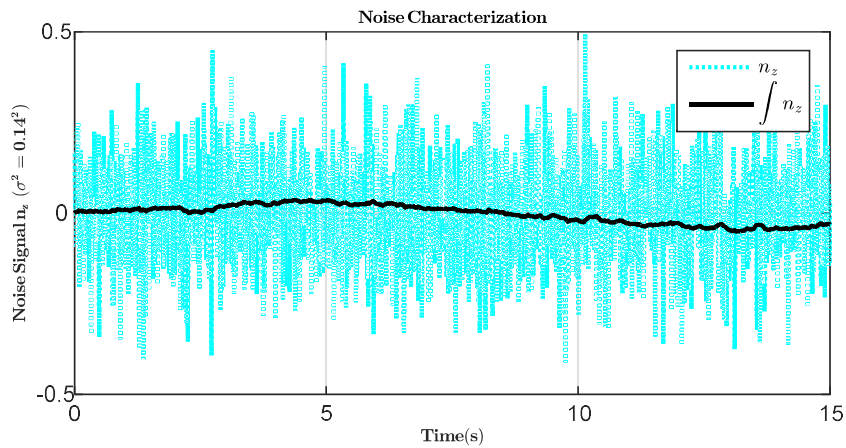


Figure 6.50 Effect of Integrator on Noise Signal

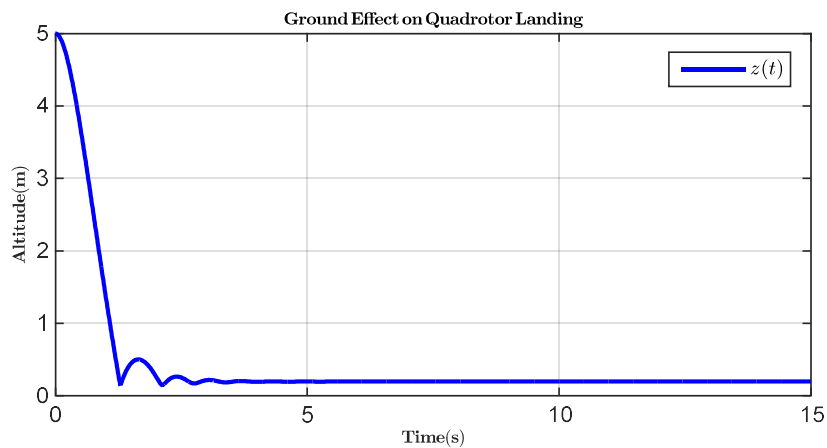


Figure 6.51 Ground Effect during Quadrotor Landing

The optimized ADRC controller parameters for quadrotor landing are given in Table 6.4.

The quadrotor's altitude dynamics tracks the landing profile given in Figure 4.3.

Table 6.4: ADRC Controller Parameters for Quadrotor Landing.

No.	ADRC Controller Parameters for Landing	
	Parameter	Value
1	p_1	32.3993
2	p_2	239.0597
3	p_3	817.1598
4	p_4	2346.6
5	k_1	1.9628
6	k_2	2.8522

The ADRC Controller is thereafter applied to the quadrotor landing problem and the following results in the presence of sensor noise were obtained.

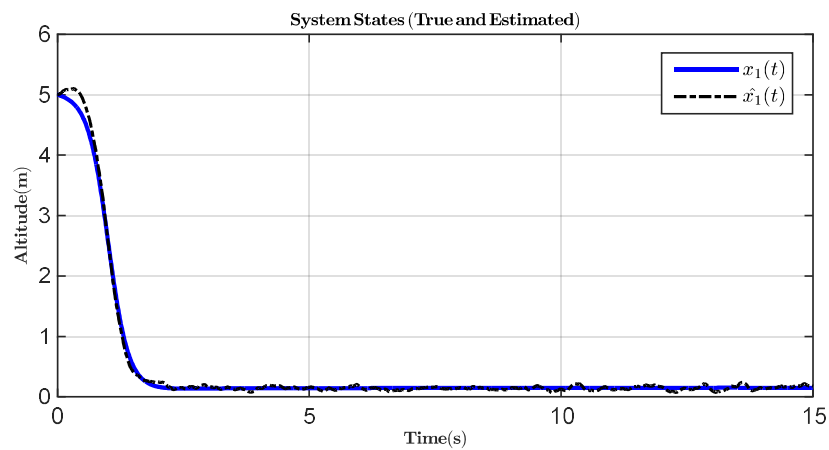


Figure 6.52 Quadrotor Landing (ADRC True and Estimated Altitude State)

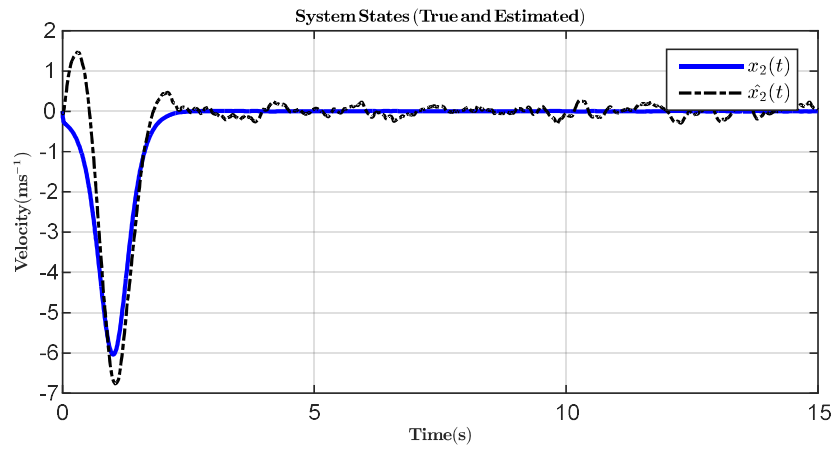


Figure 6.53 True and Estimated Quadrotor Velocity During Landing

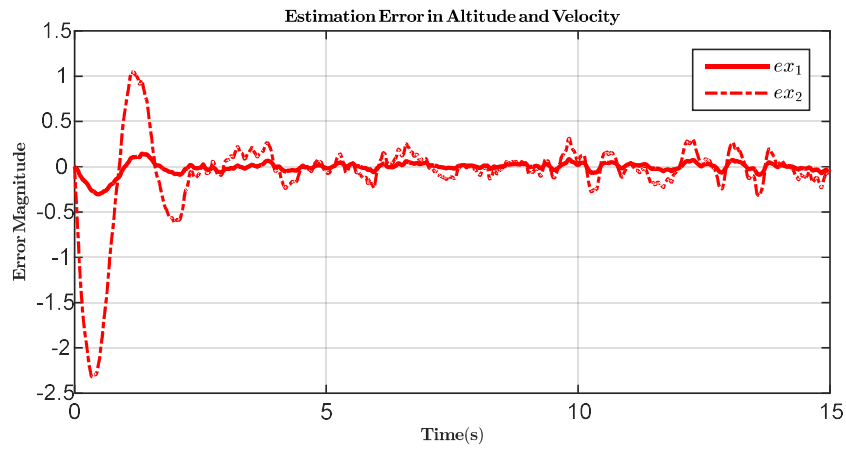


Figure 6.54 ESO Estimation Error in Altitude and Velocity

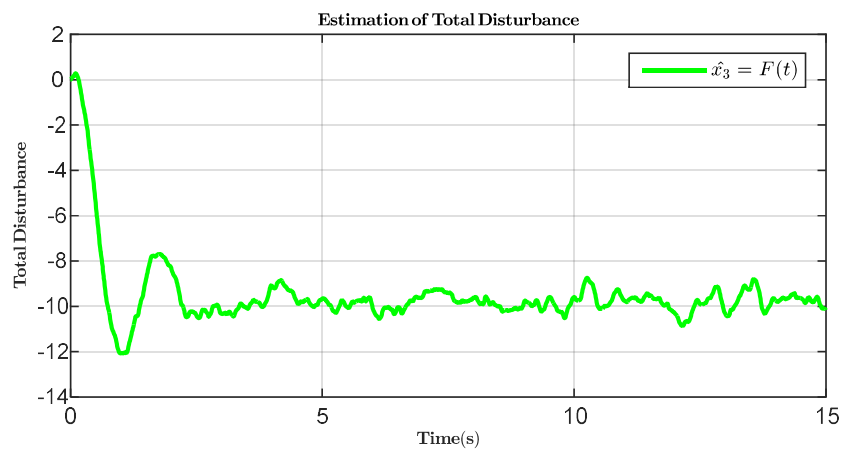


Figure 6.55 Estimation of Total Disturbance during Landing

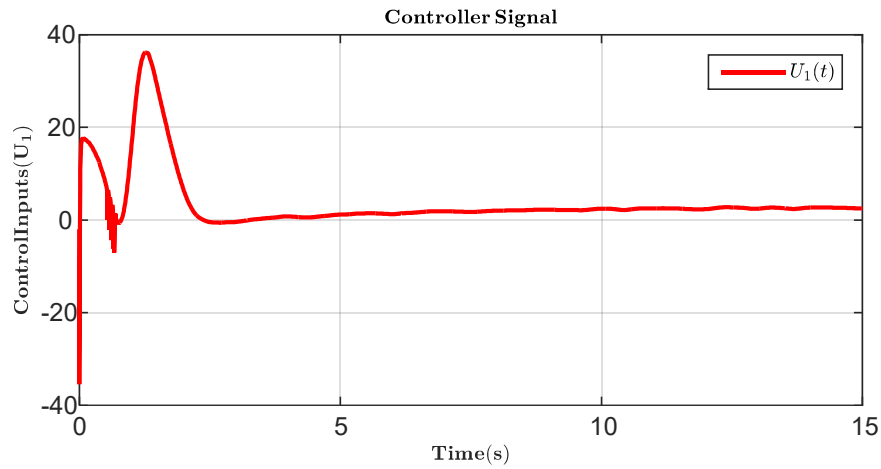


Figure 6.56 U_1 Control Input during Landing

In Figure 6.50, the effect of an integrator on noisy signal can be understood. The noise is attenuated however drift is developed in the signal. Figure 6.51 verified the bouncing nature of the quadrotor when it tries to land in the presence of ground effect. Consequently, the controller developed is able to eliminate these effects from the quadrotor dynamics. Figure 6.52 and 6.53 are the system states in the landing phase. It verified that the noise-attenuator modified ADRC controller stabilizes the quadrotor even in the presence of noisy feedback. It's shown that the noise does not amplify dangerously using this technique. Figure 6.54 is the error in ESO estimation of the altitude dynamics. Figure 6.56 shows the control signal for landing the quadrotor. It acts aggressively to compensate for ground effect between times 0 to 2.5s.

6.4.4 Control of 1-DOF Prismatic Arm with Simple PD Controller

This section presents the prismatic arm position state, velocity and control signal using the controller designed in Section 4.5. The end-effector is extended by 10 cm. The prismatic arm model and control parameters are given in Table 6.5.

Table 6.5: Prismatic Arm Model and Control Parameters.

No.	Prismatic Arm Model and Controller Parameters		
	Parameter	Symbol	Value
1	Mass	$m(kg)$	0.2
2	Change in Center of Mass	z_G	0.08
3	Proportional controller constant	k_p	1.3
4	Derivative controller constant	k_D	1.1

The End-Effector position and velocity states are shown in the following figures:

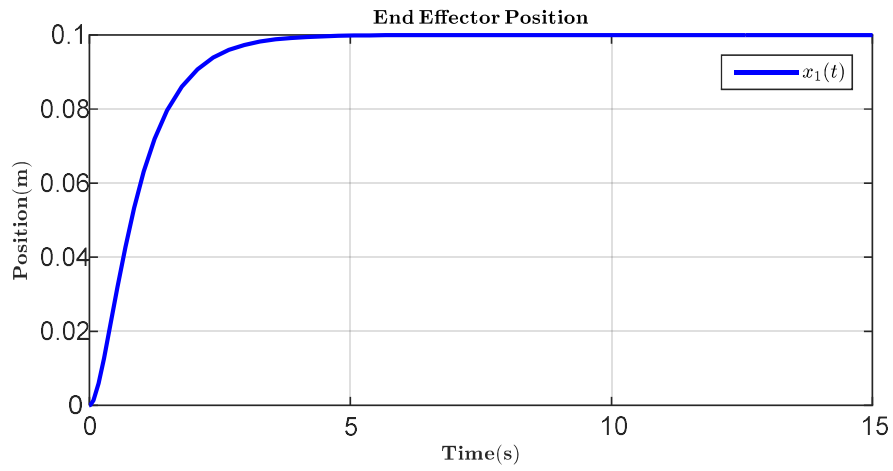


Figure 6.57 Prismatic Arm position dynamics (PD Controller).

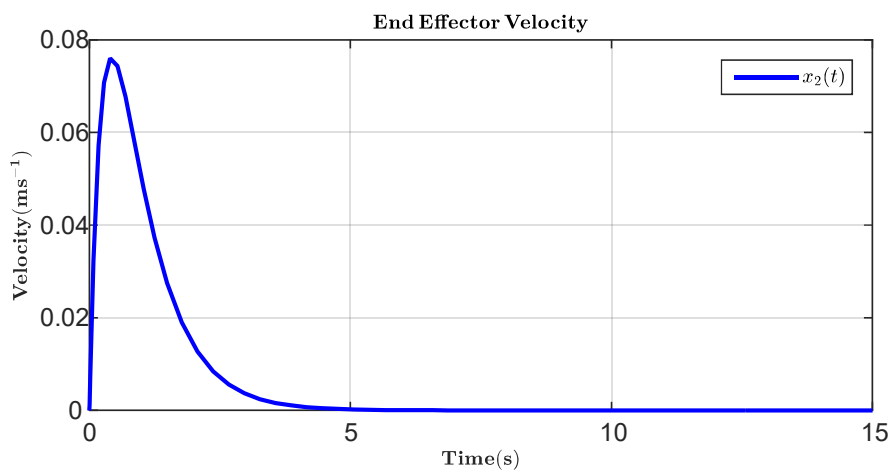


Figure 6.58 Velocity dynamics (PD Controller).

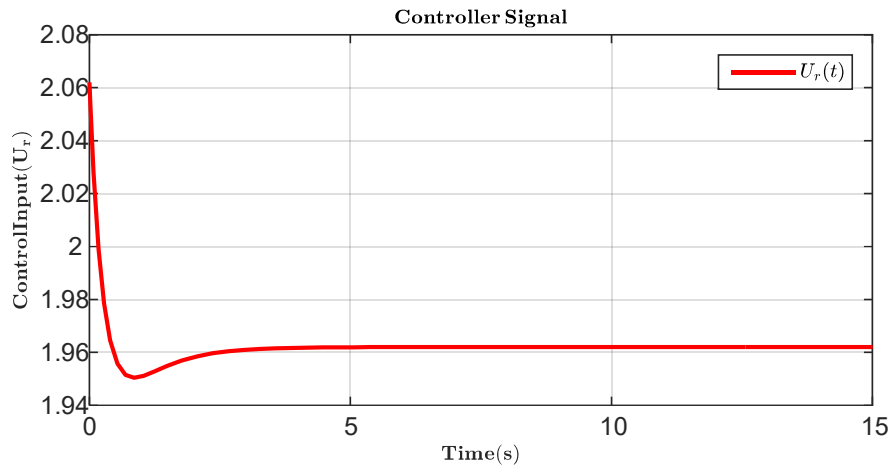


Figure 6.59 Controller Input U_τ (PD Controller).

Figure 6.57 Shows the end-effector position of the 1-DOF prismatic arm using simple PD controller. The hand extends 10cm. Figure 6.58 and 6.59 is the velocity and control signal of the prismatic arm respectively.

6.5 Results: Trajectory Optimization Control

The results in this section are based on the quadrotor model presented in Section 3.4.3 and the controller formulation of Section 4.4.5.

6.5.1 Trajectory Optimization Control

The following plots present the result for the Trajectory optimization control technique proposed in Section 4.4 using Hermite-Simpson Collocation with 50 segment points.

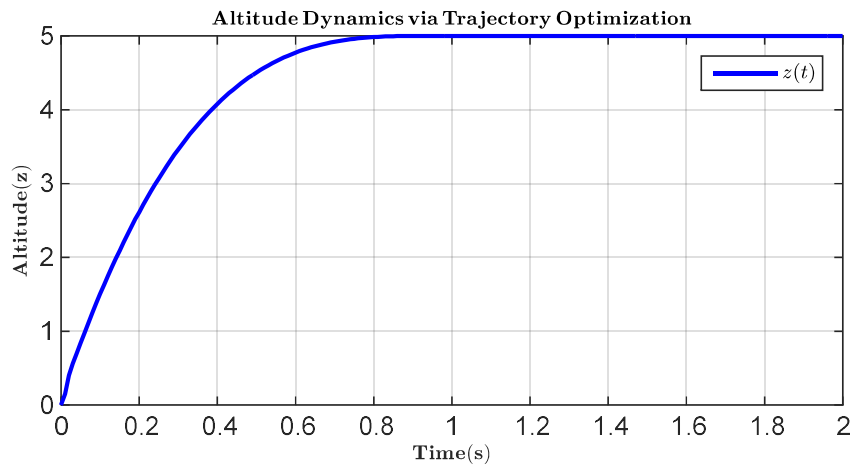


Figure 6.60 Altitude Dynamics (Trajectory Optimization)

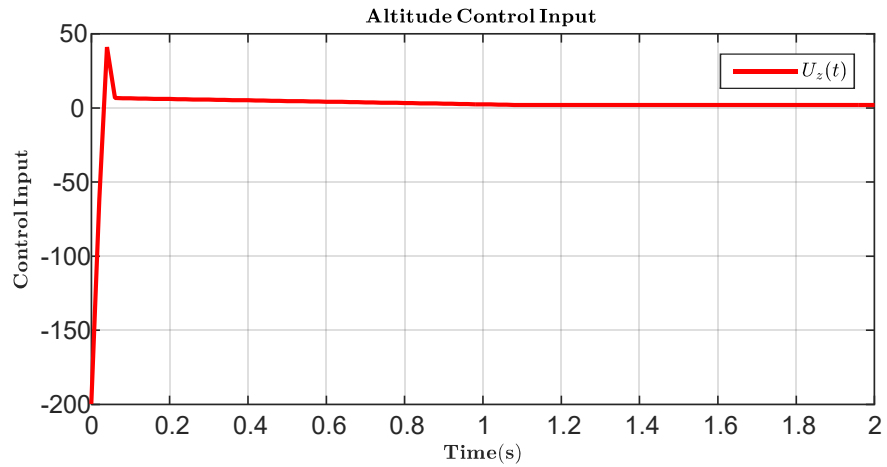


Figure 6.61 Altitude Control Input (Trajectory Optimization)

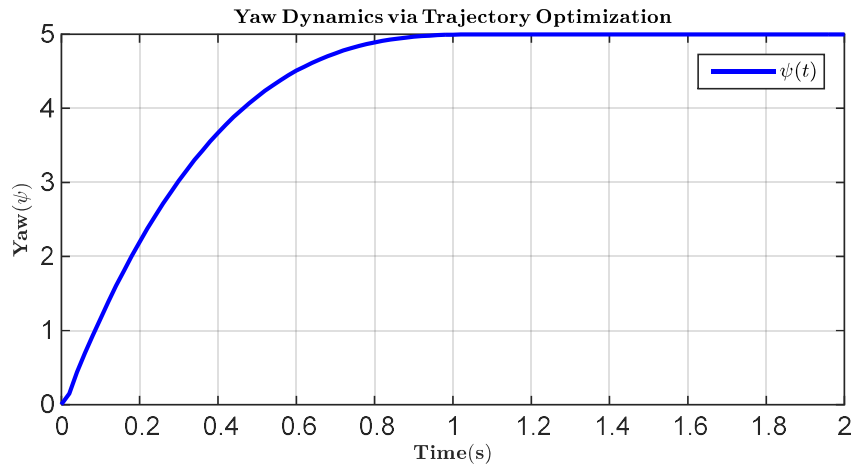


Figure 6.62 Yaw Dynamics (Trajectory Optimization)

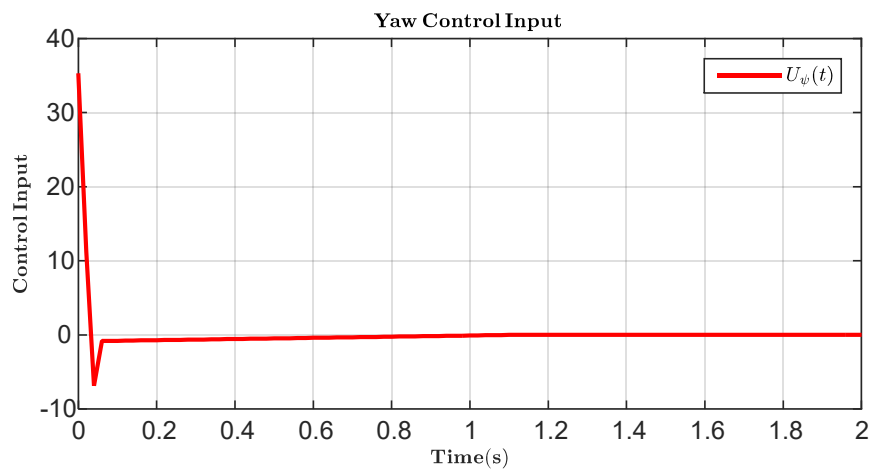


Figure 6.63 Yaw Control Input (Trajectory Optimization)

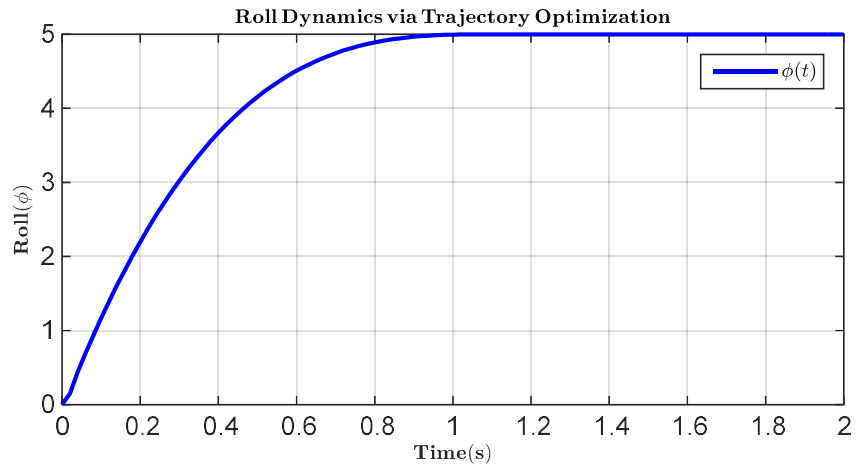


Figure 6.64 Roll Dynamics (Trajectory Optimization)

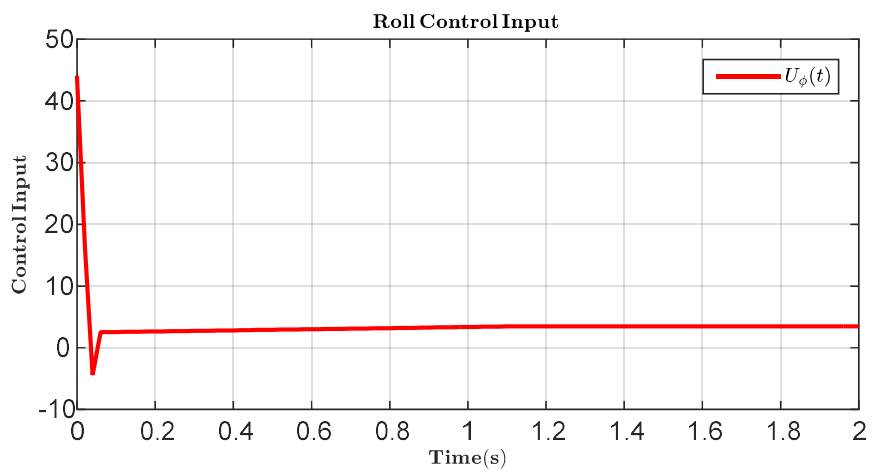


Figure 6.65 Roll Control Input (Trajectory Optimization)

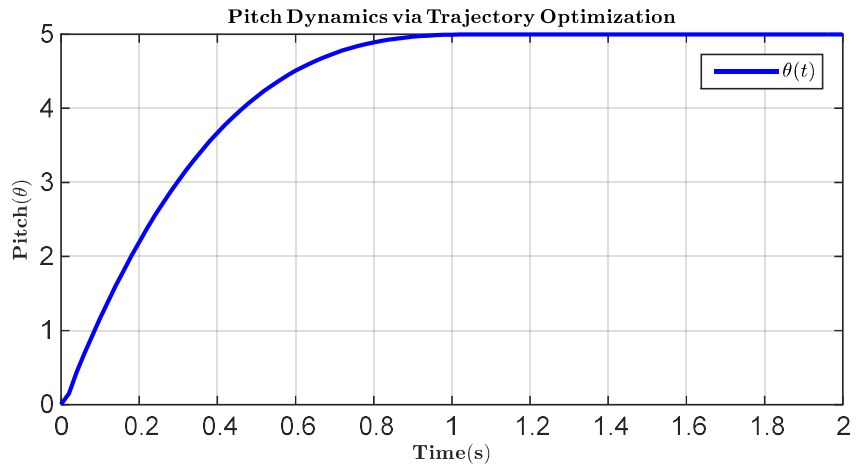


Figure 6.66 Pitch Dynamics (Trajectory Optimization)

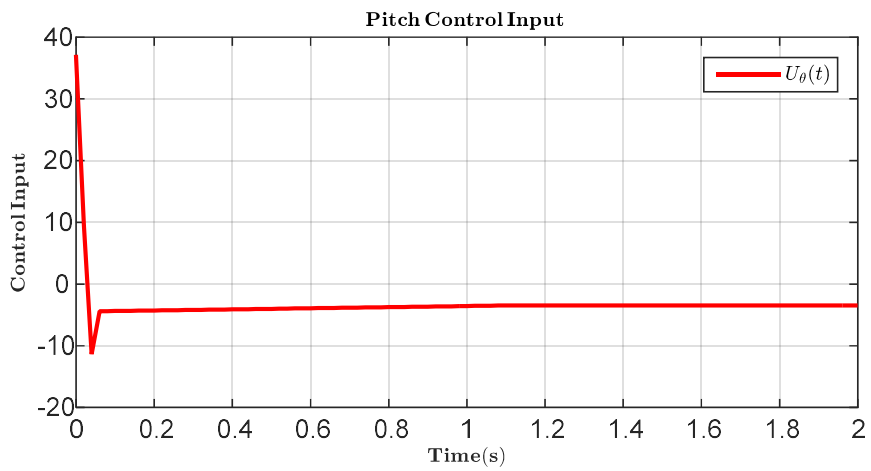


Figure 6.67 Pitch Control Input (Trajectory Optimization)

Figures 6.60 to 6.65 display the results for the trajectory optimization controller using Hermite-Simpson Direct Collocation with 50 segment points. The results presented shows the ability of the technique to achieve optimal performance given system bounds and optimization criteria. The control signal shows the optimal characteristics in terms of minimization of control effort and time.

6.6 Comments on Controller Performance

The performance characteristics of the three controllers used in this work are briefly summarized in this section.

6.6.1 Backstepping Controller

The backstepping controller requires good knowledge of model and parameters. As can be observed from the controller equations in Section 4.2, parameters such as mass and inertia that describe the model need to be known in advance. In the case where accurate information of such parameters is practically unavailable, there are three main approaches. The first is to use an estimator. The second and third approaches involved the use of parameter bounds as found in robust-backstepping control or by defining a tuning law as performed in adaptive-backstepping control. It is also clear that the backstepping controller requires more sensors for full state feedback and the disadvantage of this is the cost implication of having dedicated sensors for each quadrotor state. In terms of disturbance handling, the backstepping controller is weaker than the ADRC controller. Although the quadrotor is capable of damping the effect of external disturbances, it does not fully recover to the desired states as observed in Section 6.3.3. In terms of formulation, the backstepping controller is very straightforward to formulate.

6.6.2 Active Disturbance Rejection Controller (ADRC)

The ADRC controller is very interesting for several reasons. First, it requires very little knowledge of plant model and parameters. The accuracy of model information simply helps the ADRC achieve a more efficient control performance. Basically, the ADRC requires plant input and output information only for operation. Similarly, the extended state observer can be used to obtain unknown plant states instead of using sensors. As a result, the ADRC requires lesser number of sensors. The main advantage of the ADRC in this work is in two fold. Firstly, it overcomes the need for accurate model information. Secondly, it has very good disturbance rejection properties by observing and compensating for the total disturbance in the system via the extended state observer. In summary, the ADRC controller is quick to formulate but requires good hardware to solve the high number of differential equations introduced by the concept of ‘extended state’.

6.6.3 Trajectory Optimization Controller

The trajectory optimization controller provides a means of achieving unique custom control performance by implementing control bounds, state bounds and an objective function. It provides an optimal solution with respect to an objective function. The objective function in this work was to minimize the control effort and response time. The trajectory optimization controller is generated offline. This implies that the solution to the optimal control problem is solved ahead in time like a simulation before the controller commands are sent to the actuators. This is intuitive for avoiding catastrophic failure in terms of the

controller's inability to converge to a solution. In essence, the trajectory optimization controller requires powerful processors and is computationally tasking to implement.

6.7 Robotics Simulation using VREP, OpenCV, Python and Matlab

The Figures 6.68 to 6.79 show different Simulink models and VREP scenes used in the verification of the control structures.

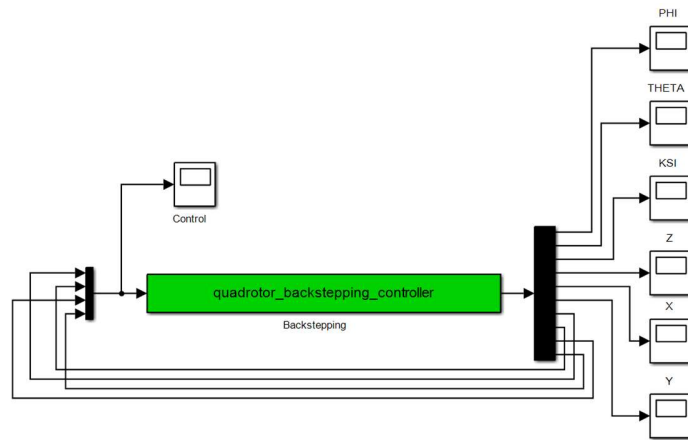


Figure 6.68 Simulink Diagram for Quadrotor Backstepping Control

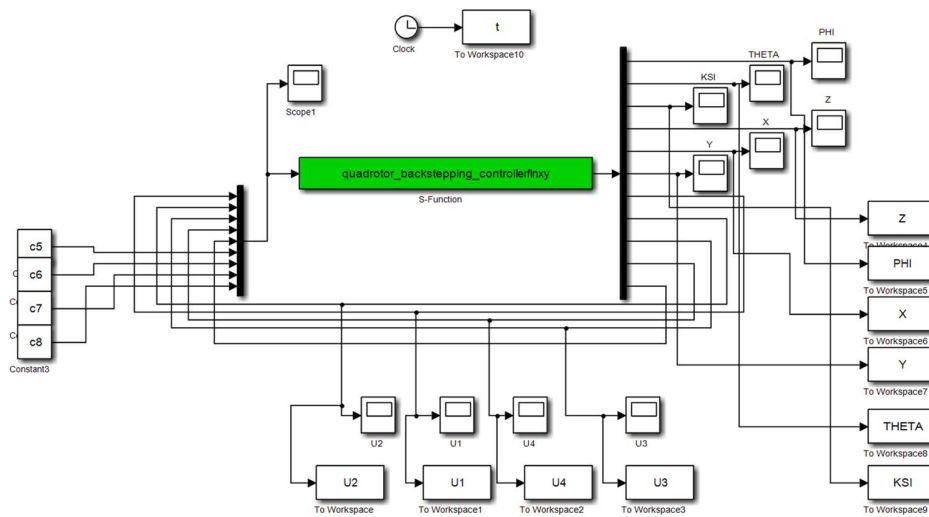


Figure 6.69 Simulink Diagram for Backstepping Control of Quadrotor with Disturbances

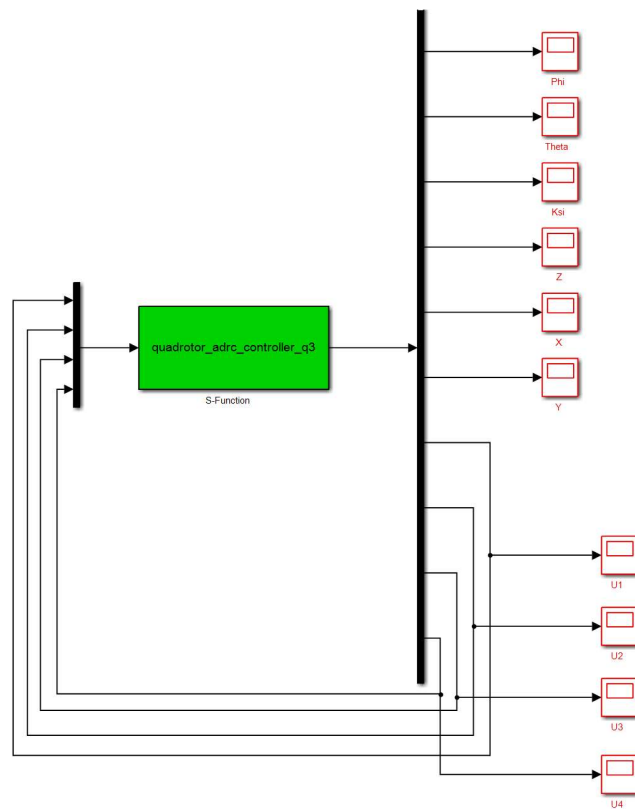


Figure 6.70 Simulink Diagram for ADRC Control of Quadrotor

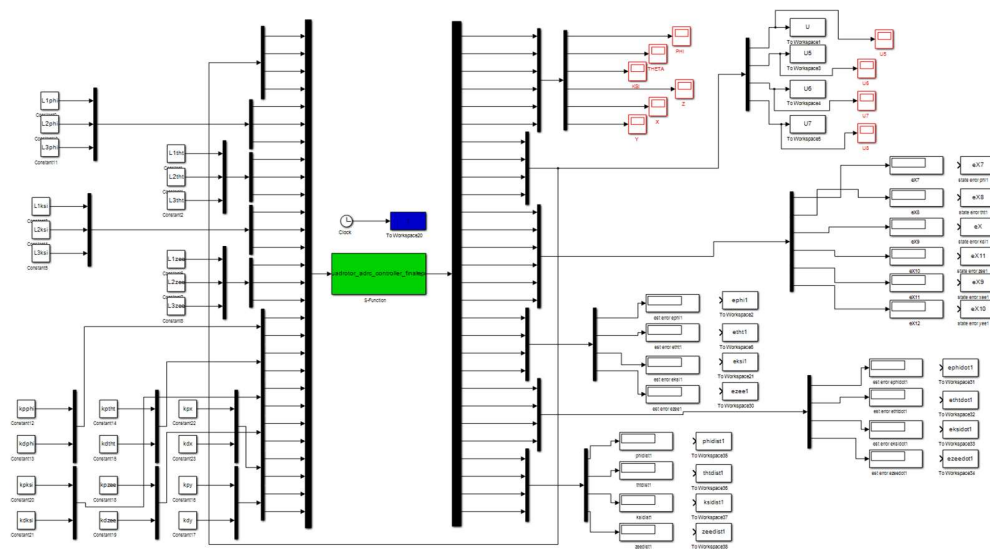


Figure 6.71 Simulink Diagram for Full ADRC Control and Parameter Optimization

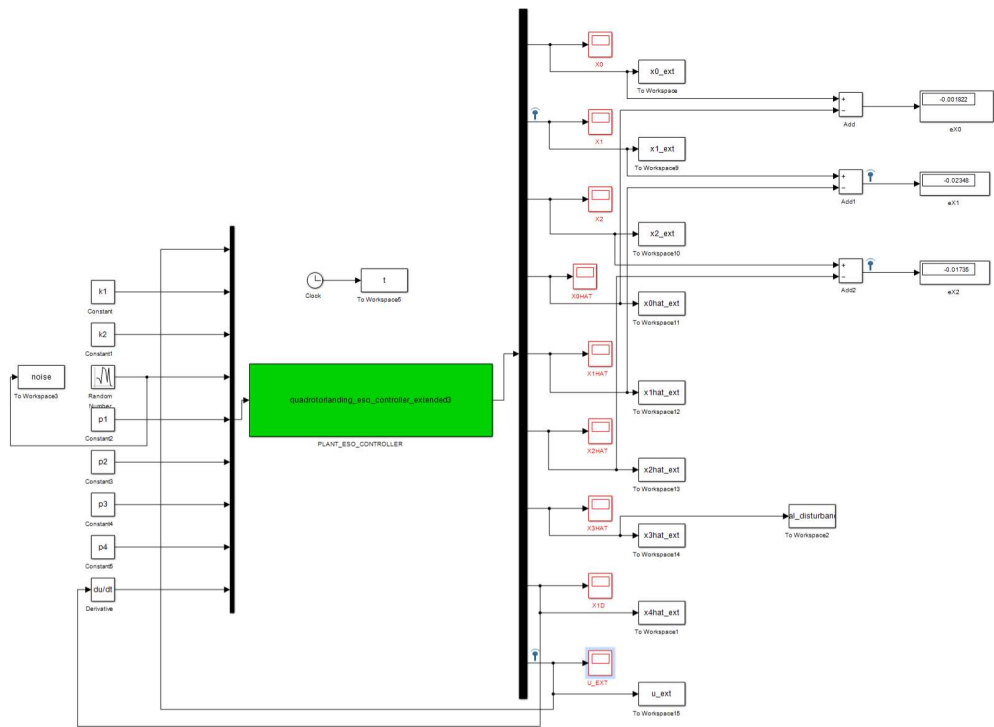


Figure 6.72 Simulink Diagram for Quadrotor Landing with Sensor Noise

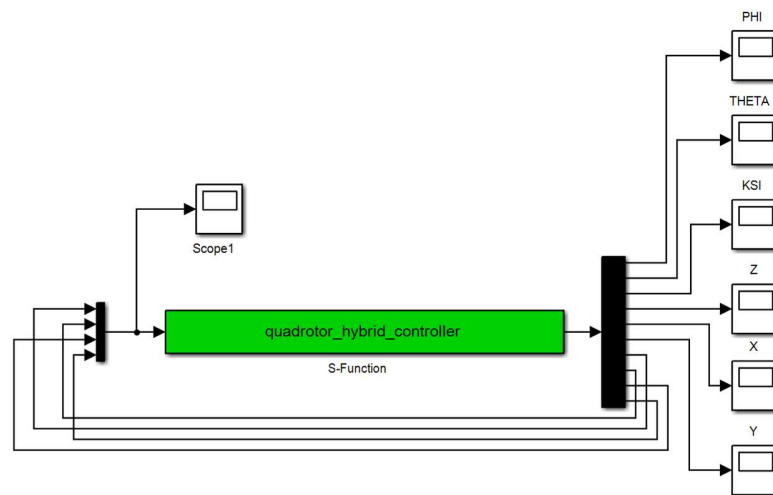


Figure 6.73 Simulink Diagram for Hybrid System

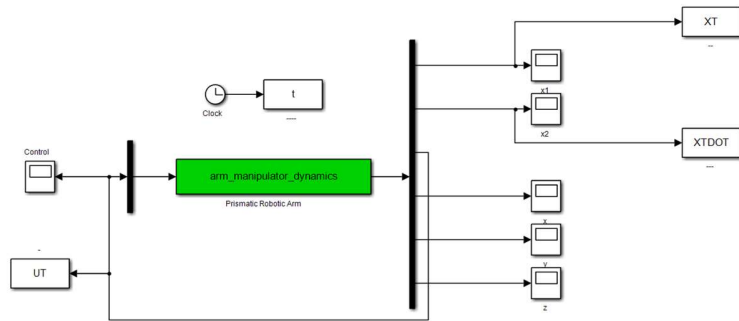


Figure 6.74 Simulink Diagram for Control of Prismatic Arm

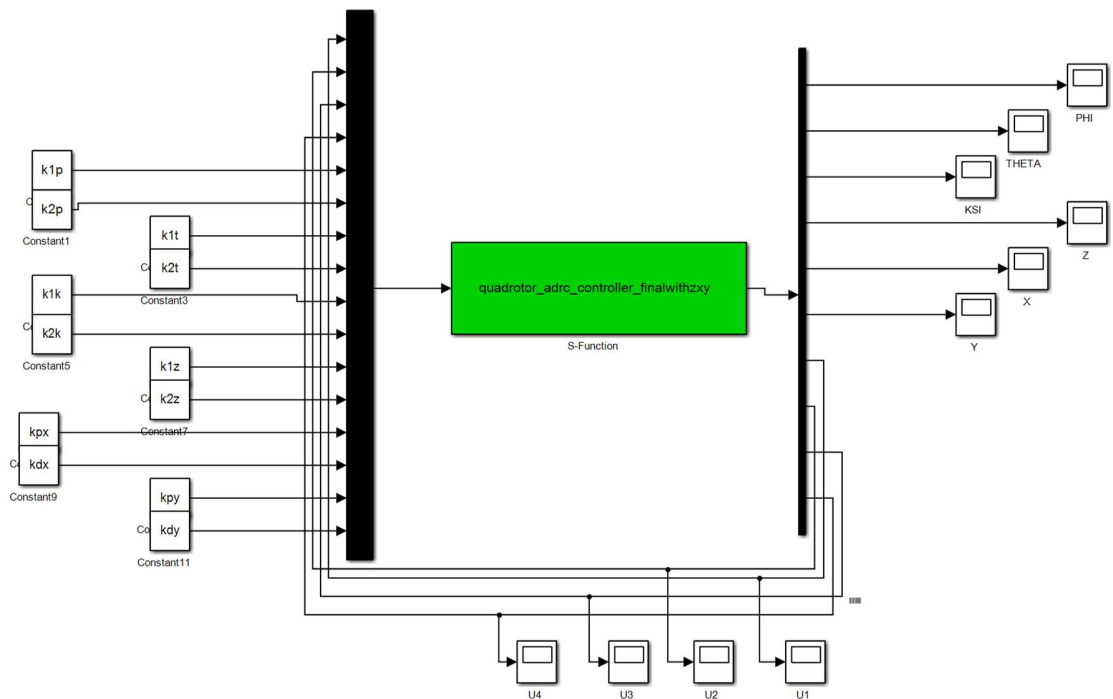


Figure 6.75 Simulink Diagram for ADRC Control and Parameter Optimization

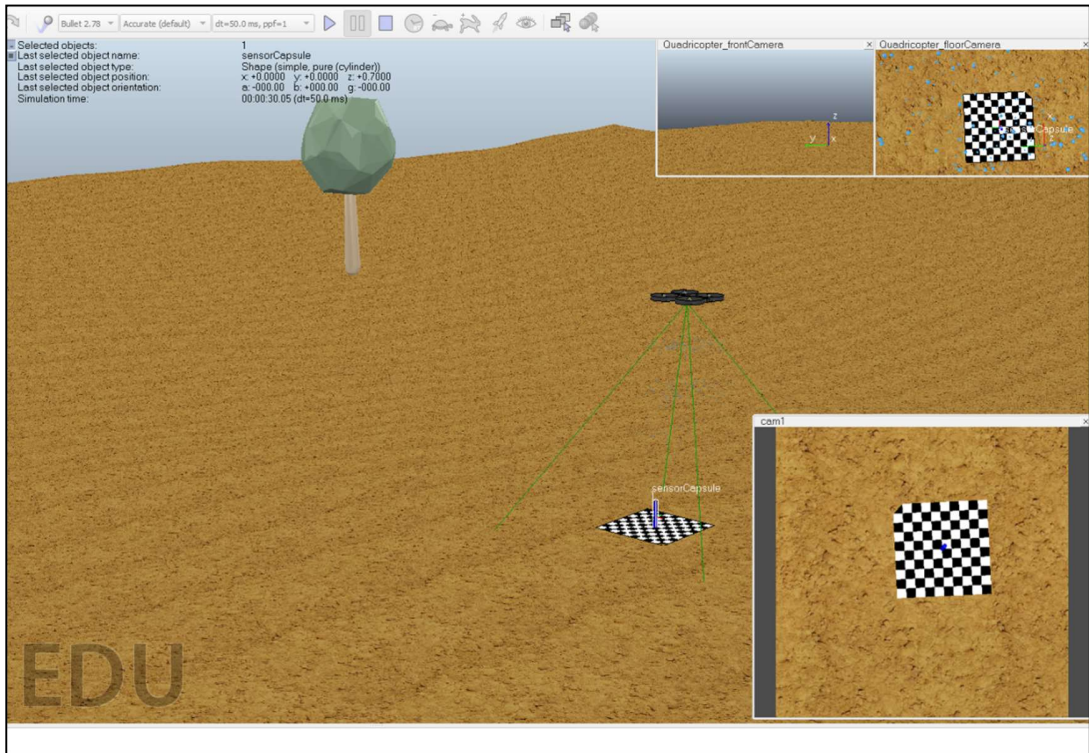


Figure 6.76 VREP Scene of Quadrotor Hovering Above Sensor Capsule

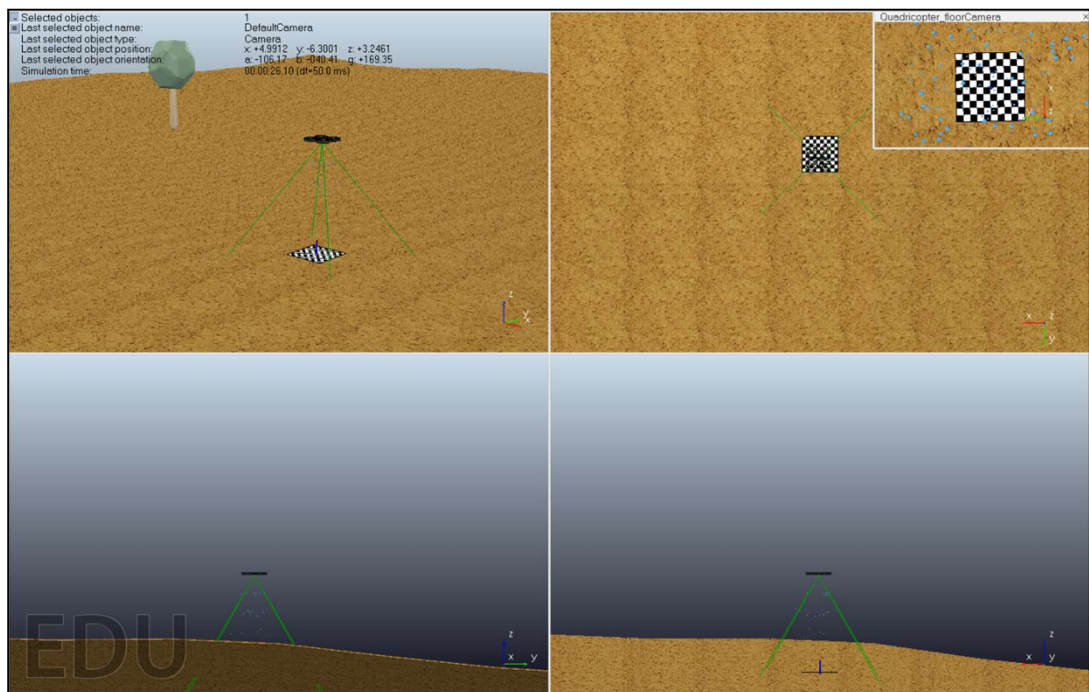


Figure 6.77 VREP Scene Showing Camera Field of View

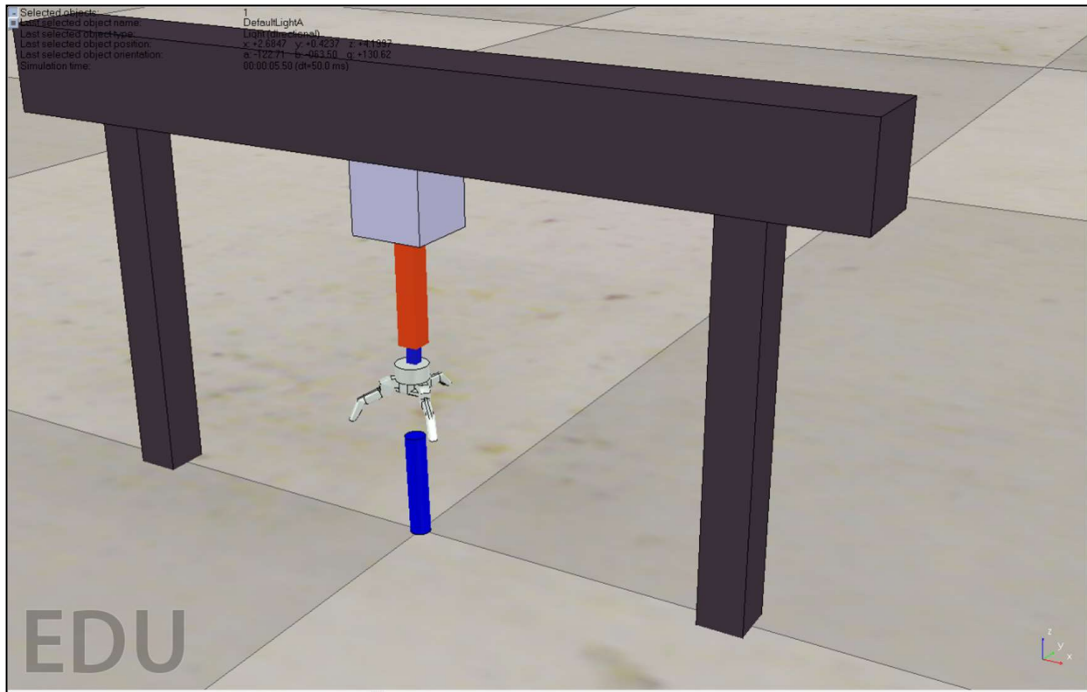


Figure 6.78 VREP Scene Emulating Sensor Capture

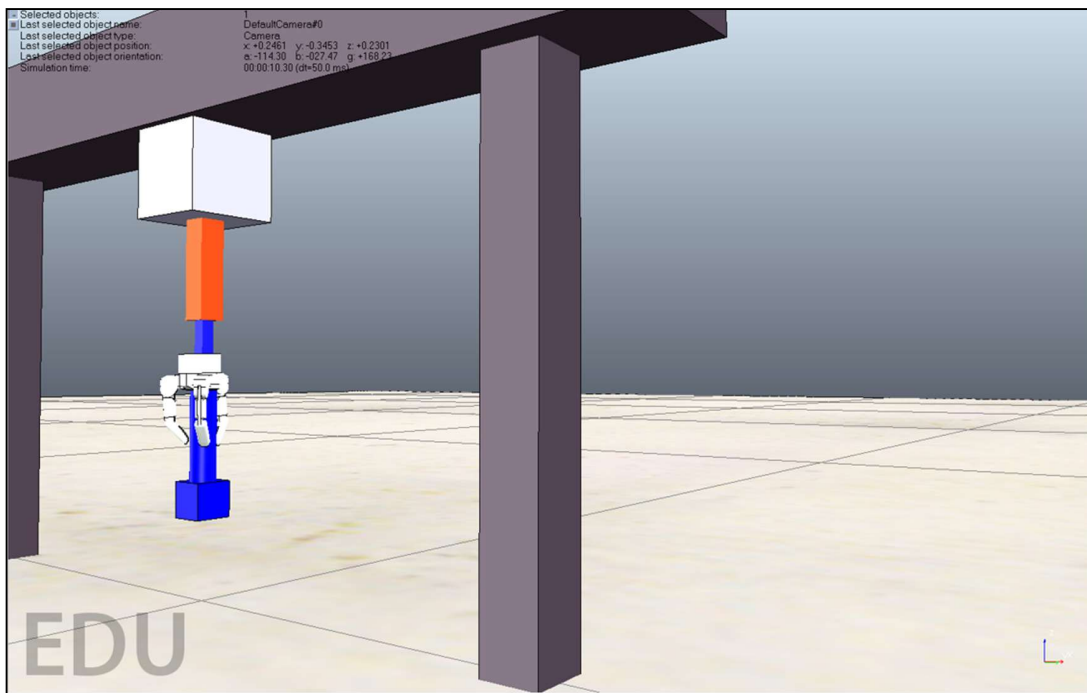


Figure 6.79 VREP Scene Emulating Sensor Pick-Up

CHAPTER 7

SUMMARY AND CONCLUSION

7.1 Summary of Work Done

In this thesis work, an interesting and practical problem was treated. The aim was to develop a control structure for the autonomous deployment of geophone sensors used in reflection seismology to a particular location in the field. This is to be achieved by the use of an unmanned aerial vehicle – the quadrotor. The assumption is an obstacle-free field and the sensor capsule is able to penetrate the ground using its stored potential energy after release from the quadrotor at a particular altitude. Aerodynamic disturbances and dynamic couplings considered in the system: ground effect, wind disturbance, center of mass effect, moment of inertia effect and aerodynamic drag. Three control structures – backstepping, active disturbance rejection and trajectory optimization – were developed based on the equations of motion for the quadrotor-manipulator system. The backstepping controller works well with optimized parameters, however, it requires explicit information about the system parameters. The active disturbance rejection controller on the other hand requires minimal information of the plant dynamics. It rejects disturbances very well and requires lesser number of sensors for operation. The third controller, the trajectory optimization

controller gives time and control-optimal performance characteristics for the quadrotor system. The method used requires a good plant model but is not properly posed for handling disturbances and multi-phase dynamics. Since the deployment and retrieval of the sensor capsule can be modelled as a continuous process with discrete modes of operation. These modes of operation include; take-off, landing and so on. A finite state machine was defined to incorporate all these modes of operation. Furthermore, a vision-assist system was developed to ensure the precision landing of the quadrotor on the sensor capsule for retrieval. Based on the concept of dwell-time, the hybrid structure agrees to switching between controllers and set point references since stability is ensured. This is because there is only one switching instance between the modes of operation; as a result, transient effects disappear. Controller verification was performed in Matlab and the VREP platform was used to virtualize the quadrotor-vision system.

7.2 Summary of Contributions

- Development of a comprehensive model for a quadrotor-manipulator system
- Application of Backstepping control to a quadrotor-manipulator system in the presence of aerodynamic disturbances and dynamic couplings.
- Application of Active disturbance rejection control to a quadrotor-manipulator system with aerodynamic disturbances and dynamic couplings.
- Application of Trajectory optimization control using Hermite-Simpson collocation method to a quadrotor system.
- Application of Gradient Descent Sequential Quadratic Programming optimization to tune controller parameters.

7.3 Future Work

In the future, it would be interesting to explore methodologies that enable the application of trajectory optimization online especially in the presence of obstacles. This could be achieved by obtaining a convex representation of the obstacle and implementing it as a path or state constrain in the trajectory optimization solver. Online application of trajectory optimization is possible if the dynamics of the system under control could be simplified to a simple double integrator. It would also be interesting to build a real physical prototype of an autonomous sensor deployment system and implement the controllers proposed in this work and new control schemes of the future.

7.4 Appreciation

I would like to express my profound gratitude to the Deanship of Graduate Studies for the Scholarship support throughout my graduate program.

REFERENCES

- [1] Teal-Group, “Teal Group,” *Press release on UAV production*, [Online]. Available: <http://tealgroup.com/>, 2015.
- [2] D. Goldberg, M. Corcoran, and R. G. Picard, “Remotely Piloted Aircraft Systems; Journalism; Opportunities and Challenges of Drones in News Gathering, Reuters Institute for the Study of Journalism,” 2013.
- [3] G. Cai, J. Dias, and L. Seneviratne, “A Survey of Small-Scale Unmanned Aerial Vehicles: Recent Advances and Future Development Trends,” *Unmanned Systems*, vol. 2, no. 2, pp. 175–199, 2014.
- [4] B. Maria De Fátima, “Unmanned Aerial Vehicles - An Overview; Report : Inside GNSS,” 2008.
- [5] S. Bouabdallah, “Design and Control of Quadrotors With Application To Autonomous Flying,” *École Polytechnique Federale de Lausanne*, 2007.
- [6] Techinsider, Defensenews, and AviaStar, “UAV Images,” *Google*. [Online]. Available: www.images.google.com.
- [7] B. Samir, B. Marcello, and S. Roland, “Flying Robots,” *Robotics*, vol. 17, pp. 8–10, 2007.
- [8] S. Lupashin, M. Hehn, M. W. Mueller, A. P. Schoellig, M. Sherback, and R. D’Andrea, “A platform for Aerial Robotics Research and Demonstration: The Flying Machine Arena,” *Mechatronics*, vol. 24, no. 1, pp. 41–54, 2014.
- [9] “Geosearches,” *Reflection Seismology*. [Online]. Available: www.geosearches.com.
- [10] “Vibroseis,” *Vibrotesis*. [Online]. Available: hateisworthless.wordpress.com.
- [11] “Remote Piloted Aerial Vehicles: An Anthology,” *Centre for Telecommunications and Information Engineering, Monash University*. [Online]. Available: <http://www.ctie.monash.edu.au/>, 2016.
- [12] N. Aldawoodi, “An Approach to Designing an Unmanned Helicopter Autopilot using Genetic Algorithms and Simulated Annealing,” *University of South Florida*, 2008.
- [13] DraganFly, “Short History of UAVs,” *DraganFly* [Online]. Available: <http://www.draganfly.com/>, 2009.

- [14] R. Maksel, “D.A.S.H. Goes to War,” *Airspace Magazine* [Online]. Available: <http://www.airspacemag.com/>, 2012.
- [15] J. Les Dorr, “FAA Approves First Commercial UAS Flights over Land,” *FAA Press Release* [Online]. Available: https://www.faa.gov/news/press_releases/, 2014.
- [16] T. Stombaugh, Report: “The Use of Unmanned Aircraft Systems in Agriculture.”, 2003 [Online]. Available: <https://www.uky.edu/>.
- [17] P. Alan, “iRobot Co-Founder: 2015 is The Year of the Protecting and Inspecting Drone,” *Dronelife*, 2014. [Online]. Available: <http://dronelife.com/>, 2015.
- [18] “Fire Scout,” *wikimedia*. [Online]. Available: <https://upload.wikimedia.org/>.
- [19] K. N. Kendoul, Farid, Zhenyu, Yu, “Guidance and Nonlinear Control System for Autonomous Flight of Minirotorcraft Unmanned Aerial Vehicles,” *Journal of Field Robotics*, vol. 27, no. 3, pp. 311–334, 2010.
- [20] F. Lewis, D. Vrabie, and V. L. Syrmos, *Optimal Control*, 3rd ed. Wiley, 2012.
- [21] B. M. Chen, *Robust and H-Infinity Control*. Springer, 2000.
- [22] M. Takahashi, G. Schulein, and M. Whalley, “Flight control Law design and development for an autonomous rotorcraft,” in *Annual forum of the American Helicopter Society*, pp. 1652–1671, 2008.
- [23] H. Khalil, *Nonlinear Systems*, 3rd ed. Pearson, 2001.
- [24] G. Cai, B. Chen, X. Dong, and T. H. Lee, “Design and Implementation of a Robust and Nonlinear Flight Control System for an Unmanned Helicopter,” *Journal of Mechatronics*, vol. 21, no. 5, pp. 803–820, 2011.
- [25] H. Kim, D. Shim, and S. Sastry, “Nonlinear Model Predictive Tracking Control for Rotorcraft-based Unmanned Aerial Vehicles,” in *American Control*, pp. 3576–3581, 2002.
- [26] E. Johnson and S. Kannan, “Adaptive Trajectory Control of Autonomous Helicopters,” *Journal of Guidance and Control of Dynamic Systems*, vol. 28, no. 3, pp. 524–538, 2005.
- [27] Q. Li, “A New PID Fuzzy Controller Fuzzy P(I+D),” in *International Conference on Information Management, Innovation Management and Industrial Engineering*, vol. 2, pp. 310–313, 2011.
- [28] E. Abbasi and M. Mahjoob, “Controlling of Quadrotor UAV Using a Fuzzy System for Tuning the PID Gains in Hovering Mode,” *10th International Conference on Advances in Computer Entertainment Technology*, pp. 1–6, 2013.
- [29] K. M. Zemalache and H. Maaref, “Controlling a Drone: Comparison between a based model method and a Fuzzy inference system,” *Applied Soft Computing*, vol. 9, no. 2, pp. 553–562, 2009.

- [30] Lebao Li, Lingling Sun, and Jie Jin, "Survey of advances in control algorithms of quadrotor unmanned aerial vehicle," in *IEEE 16th International Conference on Communication Technology (ICCT)*, pp. 107–111, 2015.
- [31] H. Voos, "Nonlinear Control of a Quadrotor Micro-UAV using," in *IEEE International Conference on Mechatronics*, no. April, pp. 4–9, 2009.
- [32] S. Bouabdallah, R. Siegwart, and G. Caprari, "Design and control of an Indoor Coax Helicopter," in *IEEE International Conference on Intelligent Robots and Systems*, no. January, pp. 2930–2935, 2006.
- [33] E. Altug, "Control of a Quadrotor Helicopter Using Dual Camera Visual Feedback," *The International Journal of Robotics Research*, vol. 24, no. 5, pp. 329–341, 2005.
- [34] H. Seyedehmarzieh and J. Mohammad, "MIMO Sliding Mode and Backstepping Control for a Quadrotor UAV," in *23rd Iranian Conference on Electrical Engineering*, pp. 994–999, 2015.
- [35] F. J. Niroumand and A. Fakharian, "Fuzzy Integral Backstepping Control Approach in Attitude Stabilization of a Quadrotor UAV," in *13th Iranian Conference on Fuzzy Systems*, pp. 1–6, 2013.
- [36] S. Kim, D. Lee, and H. J. Kim, "Image Based Visual Servoing for an Autonomous Quadrotor with Adaptive Backstepping Control (ICCAS 2011)," in *Intl Conference on Control Automation and Systems ICCAS*, no. Iccas, pp. 532–537, 2011.
- [37] H. Liu, M. Liu, X. Wei, Q. J. Song, Y. J. Ge, and F. L. Wang, "Auto Altitude Holding of Quadrotor UAVs with Kalman Filter based Vertical Velocity Estimation," in *World Congress on Intelligent Control and Automation*, pp. 4765–4770, 2014.
- [38] J. Wu, H. Peng, and Q. Chen, "RBF-ARX Model-Based Modeling and Control of Quadrotor," in *International Conference on Control Applications*, pp. 1731–1736, 2010.
- [39] S. L. Waslander, G. M. Hoffmann, and C. J. Tomlin, "Multi-Agent Quadrotor Testbed Control Design : Integral Sliding Mode vs . Reinforcement Learning *," pp. 468–473, 2005.
- [40] N. Mohajerin and S. L. Waslander, "Modular Deep Recurrent Neural Network : Application to Quadrotors," in *International Conference on Systems, Man and Cybernetics*, pp. 1374–1379, 2014.
- [41] I. Ilhan and M. Karakose, "Type-2 Fuzzy Based Quadrotor Control Approach," in *2013 9th Asian Control Conference (ASCC)*, pp. 1–6, 2013.
- [42] M. A. Basri, A. R. Husain, and K. A. Danapalasingam, "Robust Chattering Free Backstepping Sliding Mode Control Strategy for Autonomous Quadrotor Helicopter," *International Journal of Mechanical and Mechatronics Engineering*, vol. 14, no. 3, pp. 36–44, 2014.

- [43] K. Alexis, G. Nikolakopoulos, and A. Tzes, “Experimental Constrained Attitude Control of a Quadrotor subject to Wind Disturbances,” *International Journal of Control, Automation and Systems*, Springer, vol. 12, no. 6, pp. 1289–1302, 2014.
- [44] M. Belkheiri, A. Rabhi, A. El Hajjaji, and C. Pegard, “Different linearization control techniques for a quadrotor system,” in *2nd International Conference on Communications Computing and Control Applications, CCCA*, pp. 1–6, 2012.
- [45] K. Alexis, G. Nikolakopoulos, Y. Koveos, and A. Tzes, “Switching Model Predictive Control for a Quadrotor Helicopter under Severe Environmental Flight Conditions,” pp. 11913–11918, 2011.
- [46] A. A. Mian and W. Daobo, “Nonlinear Flight Control Strategy for an Underactuated Quadrotor Aerial Robot,” in *International Conference on Networking, Sensing and Control, ICNSC*, pp. 938–942, 2008.
- [47] O. Araar, “Quadrotor Control for Trajectory Tracking in Presence of Wind Disturbances,” in *UKACC International Conference in Control*, no. July, pp. 1–6, , 2014.
- [48] D. Abeywardena, Z. Wang, G. Dissanayake, S. L. Waslander, and S. Kodagoda, “Model-aided State Estimation for Quadrotor Micro Air Vehicles amidst Wind Disturbances,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4813–4818, 2014.
- [49] N. Sydney, B. Smyth, and D. A. Paley, “Dynamic control of autonomous Quadrotor flight in an estimated wind field,” *Proceedings of the IEEE Conference on Decision and Control*, pp. 3609–3616, 2013.
- [50] S. Stevanovic, J. Kasac, and J. Stepanic, “Robust tracking control of a quadrotor helicopter without velocity measurement,” *23rd International DAAAM Symposium, Vienna, Austria.*, vol. 23, no. 1, pp. 595–600, 2012.
- [51] J. Gao and Y. Zhuang, “Attitude Tracking Control of a Quadrotor Based on Linear Active Disturbance Rejective Control,” in *IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, pp. 287–292, 2015.
- [52] H. Lee, S. Kim, and H. J. Kim, “Control of an Aerial Manipulator using On-line Parameter Estimator for an Unknown Payload,” in *IEEE International Conference on Automation Science and Engineering (CASE)*, pp. 316–321, 2016.
- [53] G. Arleo, F. Caccavale, G. Muscio, and F. Pierri, “Control of quadrotor aerial vehicles equipped with a robotic arm,” in *21st Mediterranean Conference on Control and Automation*, no. 1, pp. 1174–1180, 2013.
- [54] J. U. Alvarez-Munoz, N. Marchand, J. F. Guerrero-Castellanos, A. E. Lopez-Luna, J. J. Tellez-Guzman, J. Colmenares-Vazquez, S. Durand, J. Dumon, and G. Hasan, “Nonlinear control of a nano-hexacopter carrying a manipulator arm,” in *IEEE International Conference on Intelligent Robots and Systems*, pp. 4016–4021, 2015.

- [55] K. Sreenath, N. Michael, and V. Kumar, “Trajectory generation and control of a quadrotor with a cable-suspended load-A differentially-flat hybrid system,” *Proceedings of the 2013 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4888–4895, 2013.
- [56] M. Orsag, C. Korpela, and P. Oh, “Modeling and Control of MM-UAV : Mobile Manipulating Unmanned Aerial Vehicle,” *Journal of Intelligent & Robotic Systems*, vol. 69, no. (1-4), pp. 227–240, 2013.
- [57] S. Kannan, M. Alma, M. A. Olivares-Mendez, and H. Voos, “Adaptive control of Aerial Manipulation Vehicle,” *IFAC Proceedings Volumes*, vol. 46, no. 30, p. 303–309., 2014.
- [58] S. Dai, T. Lee, and D. S. Bernstein, “Adaptive Control of a Quadrotor UAV Transporting a Cable-Suspended Load with Unknown Mass,” in *53rd IEEE Annual Conference on Decision and Control (CDC)*, pp. 6149–6154, 2014.
- [59] C. De Crousaz, F. Farshidian, M. Neunert, and J. Buchli, “Unified Motion Control for Dynamic Quadrotor Maneuvers Demonstrated on Slung Load and Rotor Failure Tasks,” in *IEEE International Conference on Robotics and Automation (ICRA)*, no. ICRA, pp. 2223–2229, 2015.
- [60] S. Kannan, M. A. Olivares-Mendez, and H. Voos, “Modeling and control of aerial manipulation vehicle with visual sensor,” *IFAC Proceedings Volumes (IFAC-PapersOnline)*, vol. 2, no. PART 1, pp. 303–309, 2013.
- [61] S. Kim, S. Choi, and H. J. Kim, “Aerial manipulation using a quadrotor with a two DOF robotic arm,” *IEEE International Conference on Intelligent Robots and Systems*, pp. 4990–4995, 2013.
- [62] P. E. I. Pounds, A. M. Dollar, and S. Member, “Stability of Helicopters in Compliant Contact Under PD-PID Control,” *IEEE Transactions on Robotics*, vol. 30, no. 6, pp. 1472–1486, 2014.
- [63] P. E. I. Pounds and A. M. Dollar, “Towards Grasping with a Helicopter Platform : Landing Accuracy and Other Challenges,” in *Australasian conference on robotics and Automation, Australian Robotics and Automation Association (ARAA)*., 2010.
- [64] G. Heredia, A. E. Jimenez-Cano, I. Sanchez, D. Llorente, V. Vega, J. Braga, J. A. Acosta, and A. Ollero, “Control of a multirotor outdoor aerial manipulator,” in *IEEE International Conference on Intelligent Robots and Systems*, pp. 3417–3422, 2014.
- [65] Y. Naidoo, R. Stopforth, and G. Bright, “Quad-Rotor Unmanned Aerial Vehicle Helicopter Modelling & Control,” *International Journal of Advanced Robotic Systems*, vol. 8, no. 4, pp. 139–149, 2011.
- [66] T. Luukkonen, Thesis: “Modeling and Control of Quadcopter,” Aalto University, 2011.
- [67] W. Wang and H. Ma, “Control System Design for Multi-Rotor MAV,” *Journal of Theoretical and Applied Mechanics*, vol. 51, no. 4, pp. 1027–1038, 2013.

- [68] R. Mahony, V. Kumar, and P. Corke, "Multirotor Aerial Vehicles: Modeling, Estimation, and Control of Quadrotor," *IEEE Robotics & Automation Magazine*, vol. 19, no. 3, pp. 20–32, 2012.
- [69] M. Z. Mustapa, "Altitude Controller Design for Quadcopter UAV," *Jurnal Teknologi*, vol. 74, no. 1, pp. 181–188, 2015.
- [70] P. Corke, *Robotics, Vision and Control: Fundamental Algorithms in Matlab*, Springer, 2013.
- [71] T. Heba ElKholy, "Dynamic Modeling and Control of a Quadrotor Using Linear and Nonlinear Approaches," American University in Cairo, 2014.
- [72] M. R. Mokhtari and B. Cherki, "A new robust control for minirotorcraft unmanned aerial vehicles," *ISA transactions*, vol. 56, pp. 86–101, 2015.
- [73] M. C. DeSimone, R. Serena, and R. Alessandro, "Influence of Aerodynamics on Quadrotor Dynamics," in *Recent Researches in Mechanical and Transportation Systems Influence*, pp. 111–118, 2015.
- [74] M. Bangura and R. Mahony, "Nonlinear Dynamic Modeling for High Performance Control of a Quadrotor," *Australasian Conference on Robotics and Automation (ACRA 2012)*, pp. 1–10, 2012.
- [75] H. Huang, G. M. Hoffmann, S. L. Waslander, and C. J. Tomlin, "Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 3277–3282, 2009.
- [76] E. Davis and P. E. I. Pounds, "Passive Position Control of a Quadrotor With Ground Effect Interaction," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 539–545, 2016.
- [77] K. Nonaka and H. Sugizaki, "Integral Sliding Mode Altitude Control for a Small Model Helicopter with Ground Effect Compensation," in *IEEE Proceedings of the 2011 American Control Conference*, pp. 202–207, 2011.
- [78] L. Danjun, Z. Yan, S. Zongying, and L. Geng, "Autonomous Landing of Quadrotor based on Ground Effect Modelling," in *34th IEEE Chinese Control Conference*, pp. 5647–5652, 2015.
- [79] I. C. Cheeseman and W. E. Bennet, "The Effect of the Ground on a Helicopter Rotor on Forward Flight," 1957.
- [80] J. M. Seddon and S. Newman., "Basic Helicopter Aerodynamics," in *Basic Helicopter Aerodynamics*, 40th ed., John Wiley & Sons., 2011.
- [81] M. Nahon, I. Sharf, A. Harmat, W. Khan, M. Michini, N. Speal, M. Trentini, T. Tsadok, and T. Wang, "Ground Effect Experiments and Model Validation with Draganflyer X8 Rotorcraft," in *International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 1158–1166, 2014.

- [82] H. Nobahari and A. R. Sharifi, “Continuous ant colony filter applied to online estimation and compensation of ground effect in automatic landing of quadrotor,” in *Engineering Applications of Artificial Intelligence*, vol. 32, pp. 100–111, 2014.
- [83] F. Schiano, J. Alonso-mora, P. Beardsley, R. Siegwart, and B. Siciliano, “Towards Estimation and Correction of Wind Effects on a Quadrotor UAV,” in *International Micro Air Vehicle Conference and Competition, Delft University of Technology*, pp. 134–141, 2014.
- [84] C. Hancer, K. T. Oner, E. Sirimoglu, E. Cetinsoy, and M. Unel, “Robust Position Control of a Tilt-Wing Quadrotor,” in *49th IEEE Conference on Decision and Control (CDC)*, pp. 4908–4913, 2010.
- [85] J. Kasa, S. Stevanovi, and J. Stepani, “Robust Output Tracking Control of a Quadrotor,” *Transactions of FAMENA*, vol. 37, no. 4, pp. 29–42, 2014.
- [86] B. Gross, “MaxSonar operation on a Multi-copter,” 2013. [Online]. Available: <http://www.maxbotix.com/>.
- [87] Clearpathrobotics, “Barret Gripper.” [Online]. Available: <https://www.clearpathrobotics.com/>.
- [88] P. E. I. Pounds, D. R. Bersak, and A. M. Dollar, “Stability of small-scale UAV helicopters and quadrotors with added payload mass under PID control,” *Autonomous Robots*, vol. 33, no. 1–2, pp. 129–142, 2012.
- [89] A. Q. Keemink, M. Fumagalli, S. Stramigioli, and R. Carloni, “Mechanical design of a manipulation system for unmanned aerial vehicles,” in *IEEE International Conference In Robotics and Automation (ICRA)*, pp. 3147–3152, 2012.
- [90] M. Orsag, C. Korpela, M. Pekala, and P. Oh, “Stability control in aerial manipulation,” in *IEEE American Control Conference*, pp. 5581–5586, 2013.
- [91] D. Mellinger, Q. Lindsey, M. Shomin, and V. Kumar, “Design, Modeling, Estimation and Control for Aerial Grasping and Manipulation,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2668–2673, 2011.
- [92] M. Orsag, C. M. Korpela, B. Stjepan, and P. Yu Oh, “Hybrid Adaptive Control of Aerial Manipulator,” *Journal of intelligent & robotic systems*, vol. 73, no. 1–4, pp. 693–707, 2014.
- [93] A. Jimenez-Cano, G. Heredia, M. Bejar, K. Kondak, and A. Ollero, “Modelling and control of an aerial manipulator consisting of an autonomous helicopter equipped with a multi-link robotic arm,” in *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, pp. 1–11, 2015.
- [94] C. Korpela, M. Orsag, and P. Oh, “Towards valve turning using a dual-arm aerial manipulator,” *IEEE International Conference on Intelligent Robots and Systems*, pp. 3411–3416, 2014.

- [95] M. Fumagalli, R. Naldi, A. Macchelli, F. Forte, A. Q. L. Keemink, S. Stramigioli, R. Carloni, and L. Marconi, "Developing an aerial manipulator prototype: Physical interaction with the environment," *IEEE Robotics and Automation Magazine*, vol. 21, no. 3, pp. 41–50, 2014.
- [96] M. Kemper and S. Fatikow, *Impact of center of gravity in quadrotor helicopter controller design*, vol. 39, no. 16. IFAC, 2006.
- [97] I. Palunko and R. Fierro, "Adaptive Control of a Quadrotor with Dynamic Changes in the Center of Gravity," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 2626–2631, 2011.
- [98] G. Peter, Al Hokayem, Eduardo, *Lecture Notes on Nonlinear Systems and Control*. ETHZurich, Switzerland, 2016. [Online]. Available: <http://control.ee.ethz.ch/>
- [99] Johan Dahlgren, "Robust nonlinear control design for a missile using backstepping," Linköping, 2003.
- [100] K. Miroslav, K. Petar V, and K. Ioannis, *Nonlinear and Adaptive Control Design*. John Wiley & Sons, Inc., 1995.
- [101] Z. Gao, "An Alternative Paradigm For Control System Design," in *40th IEEE Conference on Decision and Control, Proceedings*, pp. 4578–4585, 2011.
- [102] A. Radke and Z. Gao, "A Survey of State and Disturbance Observers for Practitioners," in *IEEE American Control Conference.*, no. 3, p. 6, 2006.
- [103] Q. Zheng, "On Active Disturbance Rejection Control: Stability Analysis and Applications in Disturbance Decoupling Control," Cleveland State University, 2009.
- [104] W. Zhou, S. Shao, and Z. Gao, "A Stability Study of the Active Disturbance Rejection Control Problem by a Singular Perturbation Approach," *Applied Mathematical Sciences*, vol. 3, no. 10, pp. 491–508, 2009.
- [105] R. Mandoski, "On Active Disturbance Rejection in Robotic Motion Control," Poznan University of Technology, 2016.
- [106] P. M. W. Plant, L. Sun, D. Li, K. Hu, K. Y. Lee, and F. Pan, "On Tuning and Practical Implementation of Active Disturbance Rejection Controller : A Case Study from a Regenerative Heater in a 1000 MW Power Plant," *Industrial and Engineering Chemistry Research*, vol. 55, pp. 6686–6695, 2016.
- [107] W. Tan, F. Jiayi, and C. FU, "Linear Active Disturbance Rejection Controllers (LADRC) for Boiler-Turbine Units," in *Proceedings of the 32nd Chinese Control Conference, IEEE*, pp. 5339–5344, 2013.
- [108] S. Malladi and N. Yadaiah, "Design and Analysis of Linear Active Disturbance Rejection Controller for AVR System," in *IEEE International Conference on Industrial Instrumentation and Control (ICIC)*, pp. 771–776, 2015.

- [109] Y. Wang, G. Ren, and J. Zhang, “Anti-WindUp schemes for Nonlinear Active Disturbance Rejection Control in Marine Steam Turbine,” *Journal of Marine Science and Technology*, vol. 24, no. 1, pp. 47–53, 2016.
- [110] M. R. Mokhtari, A. C. Braham, and B. Cherki, “Extended State Observer based control for coaxial-rotor UAV,” *ISA Transactions*, vol. 61, pp. 1–14, 2016.
- [111] J. Han, “From PID to Active Disturbance Rejection Control,” *IEEE Transactions on Industrial Electronics*, vol. 56, no. 3, pp. 900–906, 2009.
- [112] R. Madoński and P. Herman, “Survey on methods of increasing the efficiency of extended state disturbance observers,” in *ISA Transactions*, vol. 56, pp. 18–27, 2015.
- [113] H. S. Rodrigues, M. Teresa, T. Monteiro, and D. F. M. Torres, “Optimal Control and Numerical Software: An Overview,” *Systems Theory: Perspectives, Applications and Developments*, Nova Science Publishers, 2014.
- [114] M. Diehl, H. G. Bock, H. Diedam, and P.-B. Wieber, “Fast Direct Multiple Shooting Algorithms for Optimal Robot Control,” *Fast Motions in Biomechanics and Robotics*, 2009.
- [115] M. P. Kelly, “Transcription Methods for Trajectory Optimization A beginners tutorial,” pp. 1–14, 2015. [Online]. Available: <http://www.matthewpeterkelly.com/>
- [116] R. G. Antón, “Commercial Aircraft Trajectory Optimization,” Universidad Carlos III de Madrid, 2014.
- [117] O. von Stryk and R. Bulirsch, “Direct and Indirect methods for trajectory optimization,” *Annals of Operations Research*, vol. 37, no. 1, pp. 357–373, 1992.
- [118] A. V Rao, “A Survey of Numerical Methods for Optimal Control.” in *Advances in Astronautical Sciences*, vol 135, no. 1, pp.497-528, 2009.
- [119] D. A. Benson, G. T. Huntington, T. P. Thorvaldsen, and A. V Rao, “Direct Trajectory Optimization and Costate Estimation via an Orthogonal Collocation Method,” *Journal of Guidance, Control, and Dynamics*, vol. 29, no. 6, pp. 1435–1440, 2006.
- [120] R. Ritz, M. Hehn, S. Lupashin, and R. D. Andrea, “Quadrocopter Performance Benchmarking Using Optimal Control,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5179–5186, 2011.
- [121] L.-C. Lai, C.-C. Yang, and C.-J. Wu, “Time-Optimal Control of a Hovering Quad-Rotor Helicopter,” *Journal of Intelligent and Robotic Systems*, vol. 45, no. 2, pp. 115–135, 2006.
- [122] W. Van Loock, G. Pipeleers, and J. Swevers, “Time-optimal quadrocopter flight,” in *European Control Conference (ECC)*, pp. 1788–1792, 2013.

- [123] M. Hehn, R. Ritz, R. D'Andrea, M. Hehn, R. Ritz, and R. D'Andrea, "Performance benchmarking of quadrotor systems using time-optimal control," *Autonomous Robot*, vol. 33, no. 1–2, pp. 69–88, 2012.
- [124] D. Verscheure, B. Demeulenaere, J. Swevers, J. De Schutter, and M. Diehl, "Practical Time-Optimal Trajectory Planning for Robots: A Convex Optimization Approach," *IEEE Transactions on Automatic Control*, pp. 2318–2327, 2008.
- [125] D. Pardo, L. Möller, M. Neunert, A. W. Winkler, and J. Buchli, "Evaluating direct transcription and nonlinear optimization methods for robot motion planning," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, p. 946–953., 2016.
- [126] R. D. Falck and J. W. Dankanich, "Optimization of Low-Thrust Spiral Trajectories by Collocation," in *Astrodynamics Specialists Conference*, pp. 2012–4423, 2012.
- [127] P. F. Gath, Dissertation: "CAMTOS - A Software Suite Combining Direct and Indirect Trajectory Optimization Methods," 2002.
- [128] M. Geisert and N. Mansard, "Trajectory Generation for Quadrotor Based Systems using Numerical Optimal Control," *International Conference on Robotics and Automation*, pp. 2958–2964, 2016.
- [129] P. K. Mathew, "OptimTraj Matlab Documentation." 2016. [Online]. Available: <http://www.matthempeterkelly.com/>
- [130] O. Elshazly, Z. Zyada, A. Mohamed, and G. Muscato, "Optimized Control of Skid Steering Mobile Robot with Slip Conditions," in *IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, pp. 959–964, 2015.
- [131] J. Cao and B. Cao, "Design of Fractional Order Controllers Based on Particle Swarm Optimization," in *IST IEEE Conference on Industrial Electronics and Applications*, pp. 1–6, 2006.
- [132] Mathworks, "Simulink Design Optimization," *Matlab Documentation*, [Online]. Available: <https://www.mathworks.com/>, 2015.
- [133] J. Malmborg, "Analysis and Design of Hybrid Control Systems," LTH, 1998.
- [134] J. Ding, J. H. Gillula, H. Huang, M. P. Vitus, W. Zhang, and C. J. Tomlin, "Hybrid Systems in Robotics: Toward Reachability-Based Controller Design," *IEEE Robotics & Automation Magazine*, vol. 18, no. 3, pp. 33–43, 2011.
- [135] M. Orsag and S. Bogdan, "Hybrid control of Quadrotor," *17th Mediterranean Conference on Control & Automation*, vol. 1, pp. 1239–1244, 2009.
- [136] Robert J. Bamberger Jr., David P. Watson, David H. Scheidt, and Kevin L. Moore, "Flight Demonstrations of Unmanned Aerial Vehicle Swarming Concepts," *Johns Hopkins APL Technical Digest*, vol. 27, no. 1, pp. 41–55, 2006.
- [137] D. Liberzon and A. Stephen Morse, "Basic Problems in Stability and Design of Switched Systems," *IEEE Control Systems*, vol. 19, no. 5, pp. 59–70, 1999.

- [138] M. Orsag, M. Poropat, and S. Bogdan, “Hybrid fly-by-wire quadrotor controller,” *IEEE International Symposium on Industrial Electronics*, pp. 202–207, 2010.
- [139] S. Mitra and D. Liberzon, “Stability of Hybrid Automata with Average Dwell Time : An Invariant Approach,” in *43rd IEEE Conference In Decision and Control*, pp. 1394–1399, 2004.
- [140] D. Liberzon, *Switching in Systems and Control*. Boston, MA: Springer, 2003.
- [141] “Barret Hand BH8-280,” *Barret Inc. Datasheet*. [Online]. Available: <http://web.barrett.com/support/>.
- [142] S. Hutchinson, G. D. Hager, and P. I. Corke, “A Tutorial on Visual Servo Control,” *IEEE Transactions on Robotics and Automation*, vol. 12, no. 5, pp. 651–670, 1996.
- [143] M. Olivares-Mendez, C. Fu, P. Ludvig, T. Bissyandé, S. Kannan, M. Zurad, A. Annaiyan, H. Voos, and P. Campoy, “Towards an Autonomous Vision-Based Unmanned Aerial System against Wildlife Poachers,” *Sensors*, vol. 15, no. 12, pp. 31362–31391, 2015.
- [144] A. Santamaria-Navarro and J. Andrade-Cetto, “Uncalibrated image-based visual servoing,” in *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 5247–5252, 2013.
- [145] V. Lippiello, J. Cacace, A. Santamaria-Navarro, J. Andrade-Cetto, M. A., Trujillo, Y. R., Esteves, A., Viguria, “Hybrid Visual Servoing with Hierarchical Task Composition for Aerial Manipulation,” *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 259–266, 2016.
- [146] M. Bonak, D. Matko, and S. Blai, “Quadrocopter control using an on-board video system with off-board processing,” *Robotics and Autonomous Systems*, vol. 60, no. 4, pp. 657–667, 2012.
- [147] W. Li, T. Zhang, and K. Kühnlenz, “A vision-guided autonomous quadrotor in an air-ground multi-robot system,” *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 2980–2985, 2011.
- [148] J. Park and Y. Kim, “Landing Site Searching Algorithm of a Quadrotor Using Depth Map of Stereo Vision on Unknown Terrain,” in *AIAA Infotech Aerospace*, no. June, p. 19–21, 2012.
- [149] F. Jurado, G. Palacios, F. Flores, and H. M. Becerra, “Vision-based trajectory tracking system for an emulated quadrotor UAV,” *Asian Journal of Control*, vol. 16, no. 3, pp. 729–741, 2014.
- [150] D. Lee, T. Ryan, and H. J. Kim, “Autonomous landing of a VTOL UAV on a moving platform using image-based visual servoing,” *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 971–976, 2012.
- [151] S. Saripalli, J. F. Montgomery, and G. S. Sukhatme, “Vision-based Autonomous Landing of an Unmanned Aerial Vehicle,” in *IEEE International Conference Proceedings In Robotics and Automation, ICRA '02.*, pp. 2799–2804, 2002.

- [152] J. Hermansson, A. Gising, M. Skoglund, and T. B. Schön, “Autonomous Landing of an Unmanned Aerial Vehicle,” Technical report from Automatic Control at Linköpings universitet, 2010. [Online]. Available: <http://www.control.isy.liu.se>.
- [153] S. Lange, S. Niko, and P. Protzel, “A Vision Based Onboard Approach for Landing and Position Control of an Autonomous Multirotor UAV in GPS-Denied Environments,” in *International Conference on Advanced Robotics, ICAR , IEEE*, pp. 1–6, 2009.
- [154] R. Barták, A. Hraško, and D. Obdržálek, “On Autonomous Landing of AR.Drone: Hands-on Experience,” in *27th International FLAIRS Conference*, pp. 400-405, 2014.
- [155] K. E. Wenzel, A. Masselli, and A. Zell, “Automatic take off, tracking and landing of a miniature UAV on a moving carrier vehicle,” *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 61, no. 1–4, pp. 221–238, 2011.
- [156] B. Fisher, “Imaging Geometry,” *Image Processing Lecture Notes*. [Online]. Available: <http://homepages.inf.ed.ac.uk/rbf/>.
- [157] Mathworks, “Matlab Camera Calibration,” *Matlab Documentation*, 2016. [Online]. Available: <https://www.mathworks.com/>.

Vitae

- Name: Abdulmajeed Muhammad Kabir
- Nationality: Nigerian
- Date of Birth: December 2nd, 1992
- Email: kbmajeed1@gmail.com
- Permanent Address: Kwara, Nigeria