

https://dspace.org/

Nonlinear intelligent fault-tolerant control of quadrotor-based aerial manipulators for precision agriculture applications

Kuchwa-Dube, Chioniso

2024

https://hdl.handle.net/10539/37385

Downloaded from a DSpace repository. https://dspace.org/



Nonlinear Intelligent Fault-Tolerant Control of Quadrotor-Based Aerial Manipulators for Precision Agriculture Applications

Chioniso Kuchwa-Dube

School of Mechanical, Industrial and Aeronautical Engineering
University of the Witwatersrand
Johannesburg, South Africa.

Supervisor:
Prof Jimoh O. Pedro

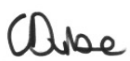
A **thesis** submitted to the Faculty of Engineering and the Built Environment, University of the Witwatersrand, in fulfilment of the requirements for the degree of Doctor of Philosophy in Engineering.

Date: January 10, 2023

Declaration

I declare that this thesis is my own, unaided work, except where otherwise acknowledged. It is being submitted for the degree of Doctor of Philosophy in the University of the Witwatersrand, Johannesburg. It has not been submitted before for any degree or examination at any other university.

Signed on this 10 day of January 20 23



Chioniso Kuchwa-Dube (1569006)

Acknowledgements

Thank you to my PhD supervisor, Prof Jimoh O. Pedro, for his invaluable guidance, feedback and support throughout my PhD studies. Thank you to my colleagues within the School of Mechanical, Industrial and Aeronautical Engineering, who have provided assistance in differing ways. My thanks to my parents, siblings, and friends for their support and encouragement. Finally, thank you to my dear husband and two wonderful children for their unwavering love and support.

Funding Acknowledgement

The work in this thesis is based on the research supported in part by the National Research Foundation of South Africa (Grant Numbers: 107025 and 120615).

Abstract

An aerial manipulator, that is an aerial robot such as a quadrotor, endowed with a robotic manipulator, can extend the application of aerial robots by allowing the aerial robot to physically interact with objects. For example, in precision agriculture, autonomous, accurate and fault-tolerant control (FTC) of the aerial manipulator is important to allow the aerial manipulator to collect viable plant samples without causing damage to the plant. Quadrotors are, however, challenging to control as they are underactuated, and have coupled, nonlinear dynamics. The control of quadrotor-based aerial manipulators is further complicated due to the coupling between the motion of the manipulator and that of the quadrotor. In this study, an intelligent, nonlinear FTC for quadrotor-based aerial manipulators is developed. Both active and passive sliding mode FTCs are investigated. The passive FTCs rely on the robust properties of the sliding mode controllers (SMCs) to mitigate rotor faults. For the active FTCs, different static neural network structures are investigated for the fault detection and diagnosis (FDD) unit used to detect and measure rotor faults. Several different SMC formulations are investigated in this study, including; continuous and discontinuous SMCs, integral SMCs, adaptive SMCs, and super-twisting SMCs. SMCs typically suffer from the undesirable chattering phenomena, which can be observed as high-frequency oscillations of the controlled system. A novel chattering index is therefore developed to measure the chattering levels of SMCs. When tuning the controllers' gains, the objectives are to minimise the trajectory tracking error, control input, and chattering. Manual tuning of controller gains, taking into account multiple conflicting objectives, is time consuming and does not result in optimal gains being selected. In this study, two intelligent multi-objective optimisation (MOO) algorithms, the multi-objective particle swarm optimisation (MOPSO) algorithm and a multi-objective genetic algorithm (MOGA) are compared for the SMC gains tuning. The novel chattering index is used as the chattering minimisation objective, together with trajectory tracking error and control input minimisation objectives, when tuning the SMC gains using the MOPSO controller gains-optimisation method. Simulations are conducted for the FTC of a large aerial manipulator with a two degree-of-freedom (DoF) manipulator. The simulation results show that the aerial manipulator successfully performs farm coverage for remote sensing while collecting plant samples, in the presence of a rotor fault. Laboratory experiments of a miniature aerial manipulator, including manipulator motion in the presence of a rotor fault, are conducted. The miniature aerial manipulator used is a Crazyflie quadrotor with an optical flow deck for position feedback and a one-DoF manipulator actuated by a servo motor. The simulations and experiments show the effectiveness of the passive and active sliding mode FTCs, the neural network-based FDD units, the intelligent MOO controller gains-tuning method, and the developed chattering index.

Contents

1 Introduction **1**

1.1 Aerial Manipulators for Precision Agriculture 1

1.2 Control of Aerial Manipulators 2

1.2.1 Linear controllers 3

1.2.2 Nonlinear controllers 4

1.3 Fault-Tolerant Control 6

1.3.1 Fault-tolerant control of multi-rotor-based aerial manipulators . . . 7

1.3.2 Passive and active fault-tolerant control of quadrotors 8

1.3.3 Sliding mode fault-tolerant control of quadrotors 10

1.4 Objectives and Contribution of Thesis 11

1.4.1 Research objectives and contribution 11

1.4.2 Outline of thesis 12

1.4.3 Publications arising from this research work 13

2 Aerial Manipulator Modelling and System Identification **14**

2.1 Newton-Euler Formulation for Aerial Manipulator 14

2.1.1 System reference frames and transformations 16

2.1.2 Dynamic modelling - Newton-Euler formulation 18

2.1.3 Aerial manipulator moments of inertia 19

2.2 Quadrotor Forces and Moments 20

2.2.1 Quadrotor forces 20

2.2.2 Quadrotor moments 22

2.2.3 Rotor modelling 23

2.3 Manipulator Forward Kinematics Modelling 24

2.3.1 Direct kinematics modelling . . . 25
2.3.2 Differential kinematic modelling . . . 26
2.4 Manipulator Dynamics - Recursive Newton-Euler Formulation . . . 27
2.4.1 Forward recursion . . . 27
2.4.2 Backward recursion . . . 28
2.4.3 Initial and terminal conditions . . . 28
2.5 Aerial Manipulator System Equations of Motion . . . 30
2.6 System Identification . . . 31
2.6.1 Rotor parameter identification - rotor speed to PWM . . . 32
2.6.2 Rotor parameter identification - rotor time constants . . . 34
2.6.3 System identification from flight data . . . 35
2.6.4 System Parameters . . . 37
2.7 Summary . . . 39

3 Aerial Manipulator Fault-Tolerant Control 41

3.1 Related Work . . . 41
3.1.1 Sliding mode fault-tolerant control of quadrotors . . . 41
3.1.2 Closed-loop inverse kinematics control for manipulators . . . 43
3.2 Fault-Tolerant Control Structures . . . 43
3.2.1 Passive fault-tolerant control . . . 43
3.2.2 Active fault-tolerant control . . . 44
3.2.3 Fault detection and diagnosis using neural networks . . . 45
3.3 Quadrotor Trajectory Tracking Control . . . 48
3.3.1 Trajectory generation . . . 48
3.3.2 Trajectory tracking . . . 49
3.4 Sliding Mode Control Design . . . 50
3.4.1 Classical first-order sliding surface . . . 51
3.4.2 First-order corrective control . . . 51

3.4.3 Pseudo/quasi sliding mode control 52
3.4.4 Classical first-order sliding mode control law 52
3.4.5 Chattering simulations and experiments 53
3.4.6 Integral first-order sliding mode control 56
3.5 Adaptive Sliding Mode Control 58
3.5.1 Adaptive control gains - Plestan method 59
3.5.2 Adaptive control gains - Roy method 60
3.6 Super-Twisting Sliding Mode Control 61
3.6.1 Classical super-twisting sliding mode control 62
3.7 Adaptive Super-Twisting Sliding Mode Control 63
3.7.1 Adaptive control gains - Shtessel method 63
3.8 Manipulator Closed-Loop Inverse Kinematics Control 65
3.8.1 Weighted damped least-squares pseudo-inverse Jacobian 66
3.8.2 Joint limit weighing matrix 66
3.9 Summary 67

4 Sliding Mode Control Chattering Performance Indices 69

4.1 Related Work 69
4.2 Error and Control Input Performance Indices 69
4.2.1 Error indices 70
4.2.2 Control input indices 71
4.3 Chattering Performance Indices 71
4.3.1 Indices based on the fast Fourier transform 71
4.3.2 Indices based on short-time Fourier transform 73
4.4 Control Input Oscillations Performance Indices 74
4.5 Quadrotor Control Simulations 75
4.5.1 Trajectory tracking error results 76
4.5.2 Control input results 77

4.5.3 Fast Fourier transform chattering indices results 79
4.5.4 Short-time Fourier transform chattering indices results 81
4.5.5 Trajectory tracking error indices results 84
4.5.6 Control input indices results 87
4.6 Summary 88

5 Multi-Objective Optimisation for Controller Gains Tuning 89

5.1 Related Work 89
5.2 Multi-Objective Particle Swarm Optimisation 91
5.3 Multi-Objective Genetic Algorithm 92
5.4 Quadrotor Controller Gains Tuning Simulations 93
5.5 Multi-Objective Optimisation Controller Gains Tuning Results 97
5.6 Summary 101

6 Aerial Manipulator Farm Coverage and Plant Sample Collection FTC Simulations 103

6.1 Farm Coverage Path Planning 103
6.2 Controller Gains Tuning Simulations 105
6.3 Passive Fault-Tolerant Control Results 109
6.3.1 First-order sliding mode control results 109
6.3.2 Adaptive sliding mode control results 109
6.3.3 Super-twisting sliding mode control results 113
6.4 Active Fault-Tolerant Control Gains Tuning Simulation Results 117
6.4.1 Active first-order sliding mode control results 117
6.4.2 Active adaptive first-order sliding mode control results 117
6.4.3 Active super-twisting sliding mode control results 121
6.5 Active versus Passive Fault-Tolerant Control 125
6.6 Farm Coverage and Plant Sample Collection Simulations 128
6.7 Summary 133

7 Fault-Tolerant Control Experiments **134**

7.1 Crazyflie Aerial Manipulator FTC Structures . . . 134

7.2 Controller Gains Tuning . . . 137

7.2.1 Controller gains selection . . . 138

7.3 Quadrotor Fault-Tolerant Control Experiments . . . 141

7.3.1 Discontinuous versus continuous passive fault-tolerant control experiments . . . 142

7.3.2 Quadrotor passive fault-tolerant control experiments . . . 145

7.3.3 Quadrotor active fault-tolerant control experiments . . . 146

7.3.4 Quadrotor neural network-based FDD . . . 148

7.3.5 Quadrotor active fault-tolerant control experiments with neural network-based FDD . . . 151

7.3.6 Quadrotor passive and active fault-tolerant control comparison . . . 153

7.4 Aerial Manipulator Fault-Tolerant Control Experiments . . . 156

7.4.1 Passive fault-tolerant control experiments . . . 158

7.4.2 Active fault-tolerant control experiments . . . 159

7.4.3 Neural network-based FDD . . . 160

7.4.4 Active fault-tolerant control experiments with neural network-based FDD . . . 161

7.4.5 Passive and active fault-tolerant control comparison . . . 161

7.5 Summary . . . 165

8 Conclusions and Recommendations for Future Work **166**

8.1 Conclusions . . . 166

8.2 Recommendations for Future Work . . . 170

List of Figures

2.1 Crazyflie quadrotor reference frames. 15

2.2 Aerial manipulator structure. 15

2.3 Crazyflie aerial manipulator structure. 16

2.4 Crazyflie aerial manipulator. 31

2.5 Rotor speed measurement. 33

2.6 Rotor step response. 33

2.7 Rotor speed to PWM duty cycle estimate. 34

2.8 Rotor step response. 35

2.9 Crazyflie quadrotor parameter identification. 37

2.10 Crazyflie quadrotor-based aerial manipulator parameter identification. . . 38

3.1 Passive fault-tolerant control scheme 44

3.2 Active fault-tolerant control scheme 45

3.3 Two neural network fault detection and diagnosis unit. 46

3.4 One neural network fault detection and diagnosis unit. 46

3.5 Four neural network fault detection and diagnosis unit. 47

3.6 Feed-forward neural network 47

3.7 Cascade-forward neural network 48

3.8 Discontinuous versus continuous functions. 53

3.9 Pitch-angle step change response. 55

3.10 Phase diagram for a pitch-angle step change. 55

3.11 Pitch-angle chattering experiments - error. 57

3.12 Pitch-angle chattering experiments - control input. 57

4.1 Pitch angle simulation. 76

4.2 Trajectory tracking error results. 77
4.3 Desired control inputs. 78
4.4 Actual control inputs. 78
4.5 Sliding surface results. 79
4.6 FFT results. 80
4.7 FFT chattering indices results. 81
4.8 Hann window size and overlap results. 82
4.9 Hamming window size and overlap results. 83
4.10 Bartlett window size and overlap results. 83
4.11 STFT results (Hann window size: 45, overlap: 90%). 85
4.12 STFT mean results (Hann window size: 45, overlap: 90%). 86
4.13 STFT chattering indices results (Hann window size: 45, overlap: 90%). . . 86
4.14 Trajectory tracking error indices. 86
4.15 Control input indices results. 87

5.1 SMC and MOO controller gains tuning structure for a quadrotor. 94
5.2 SMC and MOO controller gains tuning structure for an aerial manipulator. 94
5.3 Desired path. 95
5.4 Desired trajectories. 95
5.5 MOPSO and NSGA-II results. 98
5.6 Portion of MOPSO and NSGA-II Pareto fronts - X. 98
5.7 Portion of MOPSO and NSGA-II Pareto fronts - Y. 99
5.8 Portion of MOPSO and NSGA-II Pareto fronts - Z. 99
5.9 Portion of MOPSO and NSGA-II Pareto Fronts - Roll. 100
5.10 Portion of MOPSO and NSGA-II Pareto fronts - Pitch. 100
5.11 Portion of MOPSO and NSGA-II Pareto fronts - Yaw. 101
5.12 MOPSO and NSGA-II tuned gains - Z. 102

6.1 Field coverage path - spiral coverage. 104

6.2 Field coverage path - back-and-forth coverage with return path. 104

6.3 Desired path. 105

6.4 Desired trajectories. 106

6.5 Manipulator joint angles. 106

6.6 Manipulator states for a quadrotor-based aerial manipulator with a two-DoF manipulator. 106

6.7 Rotor remaining effectiveness. 107

6.8 MOPSO Pareto fronts - CSMC and ISMC. 110

6.9 MOPSO Pareto fronts - ISMC, ISMC_h and ISMC_b. 111

6.10 MOPSO Pareto fronts - ISMC, AISMC_r and AISMC_p. 112

6.11 MOPSO Pareto fronts - ISMC_h , AISMC_r and AISMC_rh. 114

6.12 MOPSO Pareto fronts - ISMC_h, AISMC_p and AISMC_ph. 115

6.13 MOPSO Pareto fronts - CSTSMC and ACSTSMC_t. 116

6.14 MOPSO Pareto fronts - $\text{CSMC}^{\text{AFTC}}$ and $\text{ISMC}^{\text{AFTC}}$. 118

6.15 MOPSO Pareto fronts - $\text{ISMC}^{\text{AFTC}}$, $\text{ISMC}_\text{h}^{\text{AFTC}}$ and $\text{ISMC}_\text{b}^{\text{AFTC}}$. 119

6.16 MOPSO Pareto fronts - $\text{ISMC}^{\text{AFTC}}$, $\text{AISMC}_\text{r}^{\text{AFTC}}$ and $\text{AISMC}_\text{p}^{\text{AFTC}}$. 120

6.17 MOPSO Pareto fronts - $\text{ISMC}_\text{h}^{\text{AFTC}}$, $\text{AISMC}_\text{r}^{\text{AFTC}}$ and $\text{AISMC}_\text{rh}^{\text{AFTC}}$. 122

6.18 MOPSO Pareto fronts - $\text{ISMC}_\text{h}^{\text{AFTC}}$, $\text{AISMC}_\text{p}^{\text{AFTC}}$ and $\text{AISMC}_\text{ph}^{\text{AFTC}}$. 123

6.19 MOPSO Pareto fronts - $\text{ISMC}^{\text{AFTC}}$, $\text{CSTSMC}^{\text{AFTC}}$ and $\text{ACSTSMC}_\text{t}^{\text{AFTC}}$. . 124

6.20 MOPSO Pareto fronts - AISMC_rh and $\text{AISMC}_\text{rh}^{\text{AFTC}}$. 126

6.21 MOPSO Pareto fronts - AISMC_ph and $\text{AISMC}_\text{ph}^{\text{AFTC}}$. 127

6.22 Aerial manipulator farm coverage and plant sample collection: back-and-forth coverage with return path, showing plant sample locations. 129

6.23 Aerial manipulator farm coverage and plant sample collection: rotor remaining effectiveness. 129

6.24 Aerial manipulator farm coverage and plant sample collection: manipulator joint angles. 130

6.25 Aerial manipulator farm coverage path following results. 130

6.26 Aerial manipulator farm coverage and plant sample collection: quadrotor trajectory tracking errors. 131

6.27 Aerial manipulator farm coverage and plant sample collection: quadrotor control inputs. 132

7.1 Crazyflie quadrotor on-board controllers: trajectory tracking. 135

7.2 Crazyflie quadrotor-based aerial manipulator passive fault-tolerant control scheme 136

7.3 Crazyflie quadrotor-based aerial manipulator active fault-tolerant control scheme 137

7.4 MOPSO Pareto fronts - ISMC and $ISMC_h$ of the quadrotor. 139

7.5 ISMC simulation: trajectory tracking errors for the quadrotor. 140

7.6 ISMC simulation: control inputs for the quadrotor. 140

7.7 Crazyflie quadrotor experiment: hovering with a rotor fault. 142

7.8 ISMC and $ISMC_h$ passive FTC trajectory tracking errors for the quadrotor. 144

7.9 ISMC and $ISMC_h$ passive FTC control inputs for the quadrotor. 144

7.10 $AISMC_p$ and $AISMC_{ph}$ passive FTC trajectory tracking errors for the quadrotor. 145

7.11 $AISMC_p$ and $AISMC_{ph}$ passive FTC control inputs for the quadrotor. . . 145

7.12 $AISMC_r$ and $AISMC_{rh}$ FTC trajectory tracking errors for the quadrotor. . 146

7.13 $AISMC_r$ and $AISMC_{rh}$ FTC control inputs for the quadrotor. 146

7.14 Passive FTC trajectory tracking errors for the quadrotor. 147

7.15 Passive FTC control inputs for the quadrotor. 147

7.16 Active FTC trajectory tracking errors for the quadrotor. 147

7.17 Active FTC control inputs for the quadrotor. 148

7.18 Training results for quadrotor system identification feed-forward neural networks 149

7.19 Training results for quadrotor system identification cascade-forward neural network. 149

7.20 Training results for quadrotor FDD neural networks. 150

7.21 Training results for quadrotor FDD units comprised of a single neural network. . . . 150

7.22 Training results for quadrotor FDD units comprising four neural networks. 151

7.23 Training results for a quadrotor single rotor FDD unit comprised of one neural network. . . . 152

7.24 Active FTC with neural network-based FDD: trajectory tracking errors for the quadrotor. . . . 152

7.25 Active FTC with neural network-based FDD: control inputs for the quadrotor. . . . 153

7.26 Neural network-based FDD outputs for the quadrotor. . . . 153

7.27 Mellinger passive and active FTC for the quadrotor. . . . 154

7.28 ISMC passive and active FTC trajectory tracking errors for the quadrotor. 155

7.29 AISMC Plestan adaptation passive and active FTC trajectory tracking errors for the quadrotor. . . . 155

7.30 AISMC Roy adaptation passive and active FTC trajectory tracking errors for the quadrotor. . . . 156

7.31 Manipulator positions for the Crazyflie quadrotor-based aerial manipulator. 158

7.32 Crazyflie quadrotor-based aerial manipulator experiment. . . . 158

7.33 Passive FTC trajectory tracking errors for the aerial manipulator. . . . 159

7.34 Passive FTC control inputs for the aerial manipulator. . . . 159

7.35 Active FTC trajectory tracking errors for the aerial manipulator. . . . 160

7.36 Active FTC control inputs for the aerial manipulator. . . . 160

7.37 Training results for an aerial manipulator single rotor FDD unit comprised of one neural network 161

7.38 Active FTC with neural network-based FDD: trajectory tracking errors for the aerial manipulator. . . . 162

7.39 Active FTC with neural network-based FDD: control inputs for the aerial manipulator. . . . 162

7.40 Neural network-based FDD results for the aerial manipulator. . . . 162

7.41 Mellinger passive and active FTC trajectory tracking errors for the aerial manipulator. . . . 163

7.42 AISMC Plestan passive and active FTC trajectory tracking errors for the aerial manipulator. 164

7.43 AISMC Roy passive and active FTC trajectory tracking errors for the aerial manipulator. 165

List of Tables

1.1 Quadrotor and multi-rotor fault-tolerant control methods. 8

1.2 Quadrotor fault detection and diagnosis methods. 9

2.1 Manipulator Denavit-Hartenberg parameters 26

2.2 Crazyflie aerial manipulator parameters. 39

2.3 Crazyflie quadrotor parameters. 39

2.4 Larger aerial manipulator parameters. 40

3.1 Quadrotor pitch control simulation gains. 54

3.2 Quadrotor altitude and attitude control gains. 54

3.3 Quadrotor controller gains - chattering comparison. 56

4.1 Quadrotor control simulation gains. 76

5.1 MOPSO parameters. 96

5.2 NSGA-II parameters. 96

5.3 Controller gains limits for MOO gains tuning. 96

5.4 MOO controller gains tuning results. 97

6.1 MOPSO parameters - aerial manipulator gains tuning. 107

6.2 Controller gains tuning maximum limits - first-order SMCs. 108

6.3 Controller gains tuning maximum limits - STSMCs. 108

6.4 Controller gains for farm coverage and sample collection simulations. . . . 128

7.1 Quadrotor simulation MOPSO controller gains. 139

7.2 Crazyflie FTC controller gains. 143

7.3 Crazyflie quadrotor-based aerial manipulator controller gains. 157

Abbreviations

ACSTSMC	adaptive classical super-twisting sliding mode control.
AISMC	adaptive integral sliding mode control.
ASMC	adaptive sliding mode control.
ASTSMC	adaptive super-twisting sliding mode control.
CSMC	classical sliding mode control.
CSTSMC	classical super-twisting sliding mode control.
DC	direct current.
DoF	degree-of-freedom.
ESC	electronic speed controller.
FDD	fault detection and diagnosis.
FFT	fast Fourier transform.
FTC	fault-tolerant control.
IACI	integral absolute control input.
IACO	integral absolute control input oscillation.
IAE	integral absolute error.
IAFA	integral absolute control input frequency amplitude.
IAP	integral absolute chattering frequency power spectrum.
IFACO	integral frequency-weighted absolute control input oscillation.
IFAFA	integral frequency-weighted absolute control input frequency amplitude.
IFAP	integral frequency-weighted absolute chattering frequency power spectrum.
IFSCO	integral frequency-weighted squared control input oscillation.
IFSFA	integral frequency-weighted squared control input frequency amplitude.
IFSP	integral frequency-weighted squared chattering frequency power spectrum.
INDI	incremental nonlinear dynamic inversion.

ISCI	integral squared control input.
ISCO	integral squared control input oscillation.
ISE	integral squared error.
ISFA	integral squared control input frequency amplitude.
ISMC	integral sliding mode control.
ISP	integral squared chattering frequency power spectrum.
ITACI	integral time-weighted absolute control input.
ITAE	integral time-weighted absolute error.
ITSCI	integral time-weighted squared control input.
ITSE	integral time-weighted squared error.
MODE	multi-objective differential evolution.
MOGA	multi-objective genetic algorithm.
MOO	multi-objective optimisation.
MOPSO	multi-objective particle swarm optimisation.
NSGA-II	non-dominated sorting genetic algorithm.
PAES	Pareto archived evolution strategy.
PD	proportional-derivative.
PI	proportional-integral.
PID	proportional-integral-derivative.
PWM	pulse width modulation.
RPM	revolutions per minute.
SMC	sliding mode control.
STFT	short-time Fourier transform.
STSMC	super-twisting sliding mode control.
VTOL	vertical take-off and landing.

List of Symbols

${}^{i}\boldsymbol{\mu}_{i}$	Moment exerted by link $i-1$ on link i with respect to frame 0_{i-1} expressed in frame i.
Δt	Time interval.
${}^{E}\boldsymbol{\Gamma}$	Position of the body-fixed frame with respect to the inertial frame.
${}^{E}\boldsymbol{\Theta}$	Angular position of the body-fixed frame with respect to the inertial frame.
α	Sliding mode adaptive gain.
α_{i}	Link twist i.
β	Sliding mode adaptive gain.
η_{α}	Adaptive sliding mode parameter.
η_{β}	Adaptive sliding mode parameter.
η_{D}	Adaptive sliding mode parameter.
λ	Damping constant.
μ	Adaptive sliding mode parameter.
∇	Gradient.
$\boldsymbol{\nu}$	Generalised velocity vector in the body-fixed frame.
${}^{B}\dot{\boldsymbol{\omega}}$	Angular acceleration in the body-fixed frame.
${}^{B}\boldsymbol{\omega}$	Angular velocity in the body-fixed frame.
${}^{i}\dot{\boldsymbol{\omega}}_{i}$	Angular acceleration of link i expressed in frame i.
${}^{i}\boldsymbol{\omega}_{i}$	Angular velocity of link i expressed in frame i.
$\boldsymbol{\omega}$	Link angular velocity.
ω_{ri}	Angular velocity of rotor i.
ϕ	Roll angle (rotation about x-axis).
ψ	Yaw angle (rotation about z-axis.
ρ	Adaptive sliding mode parameter.
σ	Sliding surface.
τ_{ci}	Counter torque of rotor i.
τ_{θ}	Pitching moment.
τ_{ϕ}	Rolling moment.
τ_{ψ}	Yawing moment.
${}^{B}\boldsymbol{\tau}_{G}$	Gyroscopic torques in the body-fixed frame.
$\boldsymbol{\tau}_{i}$	Torque of link i.
${}^{B}\boldsymbol{\tau}_{T}$	Thrust torques in the body-fixed frame.
${}^{B}\boldsymbol{\tau}$	Torques in the body-fixed frame.

θ_i	Joint angle i.
θ	Pitch angle (rotation about y-axis).
$\mathbf{0_n}$	$n \times n$ zero matrix.
$\mathbf{1_n}$	$n \times n$ identity matrix.
${}^{i-1}\boldsymbol{A}_i$	Homogeneous transformation matrix.
a_i	Link length i.
$\boldsymbol{C}_i$	Vector between centre of mass position of link i and system centre of mass.
$\boldsymbol{C}_{Mi}$	Centre of mass position of link i.
$\boldsymbol{C}_Q$	Vector between centre of mass position of quadrotor and system centre of mass.
$\boldsymbol{C}_{MQ}$	Centre of mass position of quadrotor.
$\boldsymbol{C}_M$	Centre of mass position of aerial manipulator.
c_i	Sliding mode parameter.
c_Q	Constant of proportionality.
d_i	Joint offset i.
e	Control error.
${}^B\boldsymbol{F}$	Total quadrotor forces acting on the centre of mass expressed in the body-fixed frame.
${}^i\boldsymbol{f}_i$	Force exerted by link $i-1$ on link i expressed in frame i.
g	Acceleration due to gravity.
$\boldsymbol{H}$	Joint limit function.
$\boldsymbol{I}_{cm_i}$	Inertia tensor of link i in the base frame.
$\boldsymbol{I}_i$	Inertia tensor of link i in the base frame.
${}^i\boldsymbol{I}_{cm_i}$	Inertia tensor of link i.
${}^i\boldsymbol{I}_i$	Inertia tensor of link i.
I_r	Moment of inertia of the rotors.
I_{xx}	Moment of inertia about the x-axis.
I_{yy}	Moment of inertia about the y-axis.
I_{zz}	Moment of inertia about the z-axis.
$\boldsymbol{I}_Q$	Moment of inertia of quadrotor.
$\boldsymbol{I}_S$	Moment of inertia of aerial manipulator.
$\boldsymbol{J}$	Jacobian.

k	Rotor thrust factor.
k_i	Sliding mode parameter.
l	The length of the propeller arm.
m_i	Mass of link i.
m_Q	Mass of quadrotor.
m	Mass.
O_i	Origin of reference frame number i.
$\boldsymbol{\xi}$	Generalised position in the inertial frame.
${}^{i}\ddot{\boldsymbol{p}}_{ci}$	Linear acceleration of centre of mass of link i relative to coordinate frame O_i.
${}^{i}\ddot{\boldsymbol{p}}_{i}$	Linear acceleration of the origin of coordinate frame O_i relative to coordinate frame O_i.
p	Angular rate in the body-fixed frame.
$\boldsymbol{p}$	Link linear position.
${}^{i-1}\boldsymbol{p}_{i}$	The position of the origin of coordinate frame O_i relative to coordinate frame O_{i-1}.
q	Angular rate in the body-fixed frame.
q_{iM}	Maximum joint limit.
q_{im}	Minimum joint limit.
q_i	Joint variable.
$\boldsymbol{q}$	Vector of manipulator joint variables.
${}^{i-1}\boldsymbol{R}_{i}$	Rotation matrix transforming vectors from frame O_i to frame O_{i-1}.
$\boldsymbol{R}_x(\phi)$	Rotation matrix: rotation about x-axis.
$\boldsymbol{R}_y(\theta)$	Rotation matrix: rotation about y-axis.
$\boldsymbol{R}_z(\psi)$	Rotation matrix: rotation about z-axis.
${}^{E}\boldsymbol{R}_{B}$	Rotation matrix: body-fixed frame to inertial frame.
r	Angular rate in the body-fixed frame.
r_i	Adaptive sliding mode parameter.
${}^{i}\boldsymbol{r}_{i,ci}$	Vector from coordinate frame O_i to centre of mass of link i expressed in coordinate frame O_i.
${}^{i}\boldsymbol{r}_{i-1,i}$	Vector from coordinate frame O_{i-1} to coordinate frame O_i expressed in coordinate frame O_i.

$\boldsymbol{S}$	Skew symmetric operator.
sgn	Signum function.
s	Sliding surface.
${}^{E}\boldsymbol{T}_{B}$	Transformation matrix: body-fixed frame to inertial frame.
T_c	Rotor time constant.
t_k	Time instant.
$\boldsymbol{U}$	Inputs vector.
u	Linear rate in the body-fixed frame, along the x-axis.
${}^{B}\boldsymbol{V}$	Linear velocity in the body-fixed frame.
v	Linear rate in the body-fixed frame, along the y-axis.
$\boldsymbol{v}$	Link linear and angular velocity vector.
$\boldsymbol{W}_{JL}$	Joint limit weighing matrix.
$\boldsymbol{W}$	Positive definitive matrix.
w	Linear rate in the body-fixed frame, along the z-axis.
X	Position coordinate in the inertial frame, along the x-axis.
$\boldsymbol{x}_i$	x axis of frame i.
$\mathbf{x}$	State vector.
Y	Position coordinate in the inertial frame, along the y-axis.
$\boldsymbol{y}_i$	y axis of frame i.
Z	Position coordinate in the inertial frame, along the z-axis.
$\boldsymbol{z}_i$	z axis of frame i.

1 Introduction

Aerial robots are typically used for tasks such as surveillance, mapping, precision agriculture remote sensing, forest fire detection, image acquisition, monitoring of inaccessible sites, and search and rescue operations [1]. These tasks are performed without direct physical contact with the environment or the object of interest. However, several benefits can be derived from an aerial robot with the ability to physically interact with the environment. Aerial manipulators consist of an aerial robot, such as a quadrotor or unmanned helicopter, with an added robotic manipulator. The increased capabilities of aerial manipulators allow them to be used for a wide range of applications [2–5] including industrial valve turning [6, 7], tree canopy sampling [8], opening and closing of drawers [9], contact inspection of structures [10–15], and collection and transportation of objects [16–21]. Despite their capabilities, the use of aerial manipulators for precision agriculture is, however, yet to applied. In addition to remote sensing for monitoring of crop health, the aerial manipulator would be able to collect plant samples for further analysis - an important aspect of precision agriculture. When using such a system for precision agriculture, it is important to avoid damage to the crops or livestock. In the event that a fault occurs in the aerial manipulator system, safe operation of the system must still be ensured. This thesis therefore focuses on the fault-tolerant control (FTC) of aerial manipulators for the precision agriculture tasks of remote sensing and plant sample collection.

1.1 Aerial Manipulators for Precision Agriculture

Precision agriculture involves the use of technology to manage farming operations and manage variability within a farm to grow more produce, using fewer resources, while reducing environmental impact. Crop growth and farming productivity are influenced by a number of factors including environmental factors such as soil, weather, and water. Factors such as soil composition and weed and pest prevalence are subject to high variations within a farm. As plant growth can be linked to these variations, precision agriculture aims to increase productivity and profit of the farming operations through the use of technology to manage such variability within the farm. Within the precision agriculture domain, aerial robots such as quadrotors have typically been used for remote sensing of crops and livestock [22–28]. The aerial robot is loaded with sensors such as multispectral cameras which are used to collect data to evaluate soil and crop health. This information allows the farmers to plan their land use and intervention measures. Seed, water, fertiliser or herbicides can then be applied according to the variations in plant growth, soil nutrients and weed growth.

While the use of aerial robots for remote sensing provides valuable information for precision agriculture, it does not provide highly detailed information. Direct measurement of plant properties is an important aspect of precision agriculture [29–35]. Plant samples are required to determine the actual pests or diseases that may be attacking the plants. Endowing an aerial robot with a robotic manipulator to form an aerial manipulator would allow the system to collect plant samples in addition to remote sensing of the farm.

For precision agriculture, the aerial manipulator would be required to perform the remote sensing task, that is a complete coverage of the desired farm area. If any areas of interest are detected, such as areas with poor plant growth, the aerial manipulator would collect plant samples from those particular areas. Thus, there are two main types of action that the aerial manipulator would perform - trajectory following of the coverage path and hovering with the manipulator in motion in order to collect a sample.

Aerial robots can be classified as fixed-wing aerial robots and rotary-wing aerial robots, based on the mechanism used to produce flight. Fixed-wing aerial robots tend to have higher payloads and longer flight durations, however they require some form of runway to lift off and are not flexible in terms of manoeuvres. Rotary-wing aerial robots such as helicopters and quadrotors are capable of vertical take-off and landing (VTOL), hovering, and low-speed flight. They have high manoeuvrability and also tend to be cheaper than fixed-wing aerial robots. They, however, have shorter flight times and lower payloads. Helicopters are more complex mechanically than quadrotors. This study focuses on the use of quadrotor aerial robots for aerial manipulation. There are several challenges in the control of quadrotors. Quadrotors are underactuated systems, and have fast, complex, coupled dynamics, which impact on the accuracy of control of the system. Adding a manipulator to a quadrotor further complicates the dynamics and therefore the control of the entire aerial manipulator system.

1.2 Control of Aerial Manipulators

While much work has been done on the control of aerial robots in general and quadrotors [1] in particular, control of aerial manipulators is still in its infancy. As quadrotors (and similar multirotors) are underactuated systems, the X and Y positions are not directly controllable, thus a cascaded control approach is typically used for the quadrotor subsystem in which the outputs of the X and Y position controllers are used as inputs into the quadrotor roll and pitch angle controllers [18, 36–38]. A variety of control methods have been applied for the control of quadrotor-based aerial manipulators. Controllers can be classified as linear controllers or nonlinear controllers. For the aerial manipulator control, different control structures can be used. One approach is a decoupled control approach in which the controllers of the quadrotor/multirotor subsystem and the manipulator subsystem are developed independently and the manipulator motion is treated as a disturbance to the quadrotor / multirotor subsystem [18, 36, 39]. Another

approach is also a decoupled control approach, however, in this case, the manipulator motion forms an input to the quadrotor/multirotor subsystem controller and is not treated as a disturbance [37, 38, 40–43]. The final approach to controlling an aerial manipulator is a coupled control approach in which the aerial manipulator is controlled as a single system [9, 19, 44, 45].

1.2.1 Linear controllers

Linear controllers can be successfully applied to nonlinear systems such as quadrotors by linearising the system about a specified operating point, for example hovering. The trajectory and operating conditions are then constrained to be relatively simple. Proportional-integral-derivative (PID) control is a widely used linear control technique that has been applied to quadrotor-based aerial manipulators. Korpela et al., [39] used a decoupled control approach in which PID control was used for the attitude (i.e. the roll, pitch and yaw angle) control of the quadrotor subsystem, with the manipulator motion treated as a disturbance. Manual control by an operator was used for the manipulator subsystem. They performed experiments in which a physical quadrotor-based aerial manipulator had to hover while the manipulator subsystem performed manoeuvres such as grasping of a long cylindrical rod and moving the rod up and down and grasping and dropping a foam block. Instabilities/oscillations in the aerial manipulator were experienced during these manipulation experiments. Ghadiok et al., [18] also used a decoupled control approach for the their aerial manipulator. They used a cascaded control for the quadrotor subsystem with the manipulator motion treated as a disturbance. PID control was used for the altitude and the X and Y positions of the quadrotor subsystem and proportional-derivative (PD) control was used for the quadrotor attitude control [18]. For the gripping task, the aerial manipulator first detected the object, decreased altitude to grip the object, then returned to hover. They found that their controller was stable when the quadrotor was near the hover location. The controller was unstable in the presence of disturbances and for desired locations that were far away.

To improve the performance of PID controllers, adaptive PID controllers have also been used. Orsag et al., [40] developed a combined gain scheduling adaptation control and Lyapunov-based model reference adaptive control used together with PID control for the attitude control of the quadrotor subsystem. The gain scheduling adaptation control of the quadrotor subsystem made use of the manipulator joint angles to ensure stability of the quadrotor subsystem during a manipulation task. In their simulation, the quadrotor-based aerial manipulator took off with the manipulator stored and then hovered. The manipulator subsystem then fully extended while the quadrotor subsystem was in hover. They found that the controller was able to stabilise the system. Kannan et al., [36] developed a decoupled controller for a quadrotor-based aerial manipulator, with the motion of the manipulator treated as a disturbance when designing the quadrotor subsystem controller. PD control was used for the attitude control of the quadrotor subsystem and adaptive PID control was used for the altitude and the X and Y position controllers. A

type of PID controller was used for the manipulator subsystem. They simulated object manipulation with the quadrotor subsystem in hover, and quadrotor trajectory tracking with the manipulator subsystem in a fixed position.

1.2.2 Nonlinear controllers

Nonlinear controllers that have been used for quadrotor-based aerial manipulators include backstepping controllers, intelligent control techniques, and sliding mode control techniques. Heredia et al., [41] developed a variable parameter backstepping-based controller for the attitude control of the multirotor subsystem of a multirotor-based aerial manipulator. The multirotor subsystem controller made use of the full dynamic model of the aerial manipulator. The manipulator subsystem made use of a position-based Cartesian impedance controller. They performed outdoor experiments with the quadrotor subsystem hovering while the manipulator subsystem performed large movements at high speed. They found that disturbances such as wind gusts introduced significant perturbations in the attitude control. They compared their controller to standard PID and found that their controller had better performance in terms of decreased pitch angle oscillations. They also performed grasping experiments with inverse kinematics used to obtain the desired servo motor commands for the manipulator. They found that for fast changes in the reference trajectory, the manipulator lagged behind the reference trajectory. Di Lucia et al., [37] developed a quaternion-based backstepping controller for the attitude control of the hexarotor subsystem of a hexarotor-based aerial manipulator and PID control for the position control. The manipulator joint angles were included as inputs for the hexarotor subsystem controller. Computed torque control was used for the manipulator joint angle control. They performed simulations in which the manipulator subsystem was made to move while the hexarotor subsystem was in hover. For the second simulation, the manipulator grabbed an object from a known position. They found that their controller performed better than a PD controller, reducing the oscillations caused by the moving manipulator. Mebarki and Lippiello, [44] developed an integral backstepping controller for a quadrotor-based aerial manipulator system. They simulated the system and found that the controller compensated well for perturbations due to adding a mass to the manipulator end-effector. Yang and Lee, [46] developed a backstepping-like end-effector trajectory tracking controller for a quadrotor-based aerial manipulator with a two-degree-of-freedom (DoF) manipulator. The controller allowed for assigning of different roles to the centre of mass dynamics and internal rotational dynamics for the system. They simulated trajectory tracking of the manipulator end-effector.

Kim et al., [45] developed an adaptive sliding mode control (SMC) for a quadrotor-based aerial manipulator system. They performed experiments in picking up an object and delivering the object during autonomous flight and placing the object at a new location on a shelf. They found a small error in the desired position during the pick up phase and release phase. Jiao et al., [47] used a sigmoid generalized super-twisting sliding mode control (STSMC) with a fuzzy adaptive sigmoid generalized super-twisting extended state

observer for disturbance estimation for attitude control of the quadrotor subsystem of a quadrotor-based aerial manipulator. The aerial manipulator had a two-DoF manipulator. They performed simulations in Gazebo for hovering control of the aerial manipulator and compared their controller to a PID controller, a backstepping controller and a generalised STSMC. They found that the both the STSMCs were robust against the wind disturbances, however, the generalised STSMC experienced more chattering than the sigmoid generalized STSMC. Jiao et al., [47] also performed hovering experiments with a physical aerial manipulator, using a fan to create a wind disturbance. They found that the sigmoid generalized STSMC with a fuzzy adaptive sigmoid generalized super-twisting extended state observer was able reject the disturbances while reducing chattering. It should be noted that the simulations and experiments where conducted without any manipulator motion.

Khalifa et al., [38,42,43] developed a number of controllers for aerial manipulators. They first developed a feedback linearisation-based controller to track desired trajectories of a quadrotor-based aerial manipulator [38]. The manipulator states were included in the quadrotor subsystem control. They then developed two intelligent controllers for the quadrotor-based aerial manipulator; a direct fuzzy logic controller and fuzzy model reference learning control [42]. They simulated the controllers and compared them against the feedback linearisation controller. The system had to track a reference trajectory, pick up a load and place it at a particular time instant. They found that the fuzzy model reference learning control produced the best results. They then developed a robust internal-loop compensator controller [43]. They compared this against the fuzzy model reference learning control. They found that the robust internal-loop compensator controller was better at disturbance rejection and had smoother trajectories. It was also simpler and less computationally expensive.

Garimella and Kobilarov, [19] developed a model predictive controller for a quadrotor-based aerial manipulator. They performed experiments where an aerial manipulator flew towards an object and picked the object up from a shelf. Their system, however, was not a real-time system due to computational limitations. Jafarinasab et al., [48] developed a virtual decomposition-based adaptive motion controller for an octorotor-based aerial manipulator. They simulated the controller for trajectory tracking of both the aerial robot subsystem and the manipulator subsystem. They found the controller's performance to be better than that of PID control. Kim et al., [9], developed a controller for hexarotor-based aerial manipulator that allowed for end-effector torque control for opening and closing a drawer and made use of PD control for controlling the velocity of the drawer. They performed experiments in which a hexarotor-based aerial manipulator had to open and close a drawer with no prior information about the drawer. They found that the speed of the drawer and force applied to the drawer were well regulated.

Poorly-chosen gains can affect the performance of controllers. Usually, when tuning controller gains, there is a trade-off between obtaining good tracking performance while minimising control input, and for SMCs, minimising the undesirable chattering phenomena. Manual tuning of controller gains is time consuming and does not guarantee that optimal

gains are found. In most cases, there are conflicting performance requirements for the controller, making manual tuning of gains even more difficult. Multi-objective metaheuristic computational intelligence algorithms such as multi-objective particle swarm optimisation (MOPSO) and multi-objective genetic algorithm (MOGA) [49–52] can be employed to obtain optimal gains in such cases. In this thesis, these two intelligent multi-objective optimisation (MOO) methods are used and compared for tuning of the aerial manipulator controller gains.

1.3 Fault-Tolerant Control

When using an aerial manipulator for precision agriculture tasks, it is important that the system does not cause any damage to plants or livestock. A fault in a system occurs when there is a deviation of one or more characteristics or parameters of the system from its acceptable condition. A fault can result in reduced effectiveness of the controlled system, and can also lead to its overall failure. When a system fails, it is no longer able to perform its required operation. Faults and failures in the aerial manipulator system can lead to:

- The aerial manipulator crashing thereby damaging itself and crops or livestock in the farm
- Poor tracking performance, thus leading to insufficient coverage of the entire farm area.
- Failure to collect the desired plant sample.
- Damage to the plant due to collecting the wrong part of the plant.

FTC of the quadrotor-based aerial manipulator is therefore important for precision agriculture applications. Faults can be classified as [53]:

Actuator faults: These are faults in the system's actuators such that there is partial or complete loss of effectiveness of the actuator.

Sensor faults: These are faults in the system's sensors such that incorrect and/or no readings are obtained from the sensors.

Structural faults: These are faults in the system's structural components such that there are changes in the physical parameters of the system, for example the system mass and moments of inertia.

A quadrotor's rotor is made up of a direct current (DC) motor and the propeller that is attached to the motor. Rotor loss of effectiveness is a common actuator fault experienced by quadrotors [53]. The design of a quadrotor is such that loss of effectiveness of a rotor

results in degraded performance of the entire system. The added manipulator has the potential of increasing the severity of the performance degradation of a quadrotor-based aerial manipulator. This thesis will focus on quadrotor actuator faults, specifically partial loss of effectiveness of the rotors.

FTC methods enable a system to maintain acceptable performance in the presence of one or more faults or failures in the system. FTC systems can be classified as passive FTCs and active FTCs. Passive FTCs [53, 54] are designed to be robust and are capable of dealing with a number of predetermined faults. Passive FTCs are generally less computationally expensive; however, they are not able to deal effectively with faults outside of the range of the predetermined faults. Active FTCs [53, 54] are based on reconfiguring control actions or selecting pre-designed controllers based on detected faults. These controllers make use of a fault detection and diagnosis (FDD) scheme to detect and identify the faults. The detected faults are passed on to the reconfiguration mechanism that reconfigures the controllers in order to deal with the faults and maintain acceptable system performance. Active FTC methods are able to deal with a wider range of faults than passive FTC methods.

An FDD scheme provides information on the fault that has occurred in the controlled system. To be effective, the FDD scheme must detect faults quickly while being robust against disturbances, model uncertainties, and different operating conditions [54]. FDD typically involves [54]:

Fault detection: This determines whether a fault has occurred and the time of occurrence of the fault.

Fault isolation: This determines the type of fault experienced and the location of the fault.

Fault identification: This provides an indication of the magnitude of the fault.

1.3.1 Fault-tolerant control of multi-rotor-based aerial manipulators

The FTC of aerial manipulators has received limited attention. Ding et al., [55] designed a cascaded FTC for position control of a quadrotor-based aerial manipulator with rotor faults. Each subsystem of the cascaded controller had four parts: a backstepping technique combined with prescribed performance bounds was used for trajectory tracking, an adaptive nonsingular terminal sliding mode was used for handling of actuator faults, and an extended state observer was to compensate for the system's coupling dynamics and for external disturbances. They performed trajectory tracking simulations using Simulink, with rotor faults and with the two-DoF manipulator in motion. They found that their controller had reduced trajectory tracking errors in the presence of of an actuator fault when compared to a backstepping controller, a modal observer and a disturbance

observer-based controller, and a combined sliding mode and backstepping controller. Pose et al., [56] made use of control allocation and changes in the one-DoF manipulator position for the FTC of a hexarotor-based aerial manipulator. The system's battery was attached to the manipulator, hence changes in the manipulator position would result in significant changes in the system's center of mass. The change in center of mass was used to mitigate a fault in a single rotor. Pose et al., [56] used a bank of observers for the system's FDD unit. They performed experiments with a rotor fault for different manoeuvres, i.e. hovering, roll, pitch, and yaw manoeuvres and found that the system was able to recover from the rotor fault.

1.3.2 Passive and active fault-tolerant control of quadrotors

Much work has been done on the passive and active FTC of quadrotors and other multi-rotors, as detailed in the bibliographic reviews by Zhang et al., [53] and Zhang and Jiang [54]. Control methods that have been used for the FTC of quadrotors and multi-rotors, which can be investigated for the FTC of aerial manipulators are listed in Table 1.1.

Table 1.1: Quadrotor and multi-rotor fault-tolerant control methods.

FTC Method	**References**
Sliding mode control	[57–75]
Backstepping control	[76–80]
Feedback linearisation	[81–83]
PID control	[84–86]
Model Reference Control	[84, 87]
Model Predictive Control	[88, 89]

Active FTC makes use of a FDD unit. FDD can be classified as model-based or model-free (also known as data-based). Model-based FDD makes use of a mathematical model of the system or of dependencies between measurable signals [90]. Residuals which give a measure of the difference between normal and faulty operations are then generated and are used to determine the occurrence of faults. The disadvantage of model-based methods is that mathematical models often do not fully capture all aspects of complex dynamic systems. Model-free methods on the other hand make use of system's data and computational intelligence techniques such as neural networks or fuzzy logic to detect and diagnose faults [90]. For the FDD of quadrotor and multi-rotor faults, the methods listed in Table 1.2 have been used. In this thesis, neural networks, which are an artificial intelligence method, are used for the FDD of the quadrotor-based aerial manipulator rotor faults.

Passive FTCs for quadrotors, based on backstepping control were developed by Benrezki et al., [76], Zhang et al., [77] and Khebbache [78]. Benrezki et al., [76] performed numerical

Table 1.2: Quadrotor fault detection and diagnosis methods.

FDD Method	**References**
Observer-based	[60, 61, 64, 75, 91–95]
Unscented Kalman Filter	[96–99]
Neural Network	[69]

simulations to determine the tracking performance of their controller. Zhang et al., [77] simulated their system under single multiple actuator partial faults. Khebbache [78] also performed simulations to determine the performance of the controller under trajectory tracking.

A feedback linearisation approach was used by Ghandour et al., [81], Zhou et al., [82], and Freddi et al., [83] for quadrotor FTC. Zhou et al., [82] used a feedback linearisation technique for the inner loop attitude control and a PID outer loop for trajectory tracking. Ghandour et al., [81] assumed the availability of a FDD scheme. Their controller had a double control loop structure with an inner controller for the fast dynamics and outer controller for the slow dynamics. Freddi et al., [83] developed a double control loop architecture for the complete failure of one of the quadrotor's rotors. They performed position (X, Y and Z) trajectory tracking control simulations and roll and pitch control simulations in the presence of rotor's complete failure. Ghandour et al., [81] Zhou et al., [82] and Freddi et al., [83] all performed simulations to test their controllers.

Sadeghzadeh and Mehta [84] developed gain-scheduled PID FTC for quadrotors. They performed trajectory tracking experiments and assumed the availability of a FDD scheme. Tuning of the controller gains proved to be time consuming. Sadeghzadeh and Mehta [84] also developed a model reference adaptive controller for quadrotors. They performed experiments in the presence of actuator faults for trajectory tracking and compared the model reference adaptive control to their gain-scheduled PID FTC method. They assumed the availability of a FDD scheme. They found that the model reference adaptive controller was more robust to faults and noise. Amoozgar et al., [85] developed a fuzzy gain-scheduled PID control for a quadrotor. They used an adaptive PID controller to eliminate the need for FDD. The controller gains were tuned using a fuzzy inference scheme. They performed hover experiments in the presence of actuator faults to test their controller.

Yu et al., [88] developed a model predictive-based quadrotor FTC with a state-augmented Kalman filter FDD scheme. They simulated the controller in the presence of actuator faults. Izadi et al., [89] developed a data-driven model predictive-based quadrotor FTC. They performed simulations of quadrotor hover control in the presence of motor faults. In [98], they went on to formulate an active FTC system using an integrated moving horizon estimation and unscented Kalman filter-based fault detection with model predictive control. They applied the controller to the hover control of a quadrotor with actuator faults.

1.3.3 Sliding mode fault-tolerant control of quadrotors

Due to their robust properties, SMC techniques, in particular, have been applied to both the passive and active FTC of quadrotors. Wang et al., [71] used an incremental nonsingular terminal SMC for the passive FTC of a quadrotor. They simulated the system and found that the controller was effective for mitigating successive actuator faults. Barghandan et al., [74] developed an adaptive fuzzy SMC for the passive FTC of a quadrotor. Simulation results showed the effectiveness of their controller for a quadrotor with partial loss of effectiveness faults.

Li et al., [63] designed both a passive and active FTC based on sliding mode control for a quadrotor. They tested the controllers experimentally under propeller damage and actuator fault for trajectory tracking. They found that the active controller was better able to handle disturbances and had better tracking performance compared to the passive controller. They had, however, assumed the availability of a FDD scheme for the active controller.

Merheb et al., developed both active [58, 60, 61] and passive [57, 59–61] sliding mode FTCs. They then developed an integrated FTC [62] that used both the passive and active controllers along with a sliding mode observer which detected and estimated actuator faults. They performed simulations to test the performance of the integrated controller in the presence of multiple actuator faults.

Wang et al., [71] developed an adaptive sliding mode control with adaptive boundary layer for quadrotor FTC. They found the controller to be effective for partial rotor loss effectiveness FTC, while reducing chattering. In [69], they used an SMC with a neural network-based FDD. They performed experiments for rotor partial loss of effectiveness and found that the neural networks performed better than a model-based two-stage extended Kalman filter for FDD.

Chen et al., [75] developed an FTC with unknown model parameters, making use of a sliding mode fault observer. They used an online adaptive estimation method to learn unknown parameters. They simulated the system for multiple rotor partial loss of effectiveness faults. Mallavalli and Fekih [65] used a backstepping integral nonsingular fast terminal sliding mode controller with a fixed-time sliding mode observer for active FTC of a quadrotor. The observer was used to detect the occurrence of faults, but not the fault magnitude. They performed simulations to assess the controller for rotor partial loss of effectiveness.

Gong et al., [72] used an integral-type SMC with an observer for the FTC of a quadrotor. They simulated the system for rotor partial loss of effectiveness. The controllers, however, require full state measurements of the quadrotor, which are typically not all available. Sharifi et al., [64] also developed a sliding mode approach to control a quadrotor in the presence of external disturbance and actuator partial loss of effectiveness faults. For the fault detection, a Luenberger observer was used to estimate the output of each rotor which

were then compared to the desired rotor outputs. They simulated the system to test its effectiveness for the partial loss of effectiveness of one rotor.

Mohammadi and Ramezani [66] used analytic redundancy relations to determine whether rotor loss of effectiveness faults have occurred. They simulated the quadrotor system and found that their method performed better than observer-based fault detection methods.

Due to the robust properties of SMCs, variations of SMCs will be investigated in this thesis for the FTC of aerial manipulators. Even though SMCs have been used for FTC of quadrotors, the addition of a manipulator further complicates the system dynamics and hence the control of the system. The suitability of various SMCs when applied to aerial manipulators therefore needs to be investigated. Several different SMC formulations are investigated in this thesis, including continuous and discontinuous SMCs, integral SMCs, adaptive SMCs and super-twisting SMCs.

One of the drawbacks of SMC is the occurrence chattering, that is high-frequency switching action which can be observed as high-frequency oscillations of the control input. When designing SMCs, it is therefore useful to have a means of estimating the chattering that the controller could experience. Existing methods of chattering analysis such as the describing function method [100–106] have limitations in that they can only be applied to linear systems or systems linearised about a certain operation point, and that can be modelled as having low-pass filter characteristics. A quadrotor-based aerial manipulator, however has complex, coupled, nonlinear dynamics. A performance criterion that measures chattering for such a system for the purposes of gain tuning is therefore developed in this thesis.

1.4 Objectives and Contribution of Thesis

There are several areas of interest that can be explored in the development of an FTC for an aerial manipulator suffering rotor faults. This study aims to develop an intelligent nonlinear FTC for quadrotor-based aerial manipulators. The targeted application for this study is farm coverage and plant sample collection for precision agriculture.

1.4.1 Research objectives and contribution

The specific objectives of this study are to:

1. Develop a model of the quadrotor-based aerial manipulator system.
2. Develop a sliding mode FTC to accurately control the motion of the quadrotor-based aerial manipulator in the presence of rotor faults.

3. Develop an intelligent FDD scheme for quadrotor-based aerial manipulator partial loss of rotor effectiveness.
4. Develop a sliding mode chattering performance criterion for MOO controller gains tuning.
5. Evaluate the effectiveness of the FTC of the quadrotor-based aerial manipulation system.

The main contributions of this thesis are the:

1. Application of aerial manipulators to precision agriculture.
2. Development of a novel SMC chattering performance index.
3. Development and application of intelligent FTC for aerial manipulators.
4. Application of intelligent MOO to the controller gains tuning of aerial manipulators.
5. Experimental validation of the aerial manipulator FTCs and of the intelligent MOO controller gains-tuning method.

1.4.2 Outline of thesis

The rest of this thesis is organised as follows; in Chapter 2, modelling of the quadrotor-based aerial manipulator using the Newton-Euler formulation for the quadrotor subsystem and the Recursive Newton-Euler formulation for the manipulator subsystem are presented, together with the modelling of the quadrotor rotor loss of effectiveness. System identification experiments are conducted to obtain the physical parameters used in the aerial manipulator model. Chapter 3 then presents different SMC formulations and illustrates the chattering phenomena and some methods used to mitigate chattering. The control structures for passive and active FTC, as well as different neural network structures for the FDD unit are also presented. The trajectory generation method used and the trajectory tracking method, given that the quadrotor is an underactuated system, is also presented. Novel SMC chattering performance indices are then developed and compared in Chapter 4. Two intelligent MOO formulations (MOPSO and MOGA) used for controller gains tuning are presented and compared in Chapter 5. Controller gains-tuning simulations are conducted to show the effectiveness of the methods. In Chapter 6, aerial manipulator farm coverage and plant sample collection simulations, in the presence of a rotor fault are conducted. For the simulations, the controller gains for different formulations of SMCs for the passive and active FTC of quadrotor-based aerial manipulators are optimised using the MOPSO formulation and the performance of the optimised controllers is compared. Experimental validation of the MOPSO controller gains-tuning method, and the passive and active FTC of a physical quadrotor-based aerial manipulator is then presented in Chapter 7. Finally, the conclusions and recommendations for future work are presented in Chapter 8

1.4.3 Publications arising from this research work

Aspects of this thesis have been published in the following journal articles and conference proceedings:

Journal Articles

1. C. Kuchwa-Dube and J. O. Pedro, "Chattering Performance Criteria for Multi-Objective Optimisation Gain Tuning of Sliding Mode Controllers", *Control Engineering Practice*, volume 127, article number 105284, 2022

2. C. Kuchwa-Dube and J. O. Pedro, "Multi-Objective Gain Tuning of Sliding Mode Controllers - MOPSO versus NSGA-II", *Applied Soft Computing*, [Submitted. Reference number: ASOC-D-21-04340]

Conference Papers

1. C. Kuchwa-Dube and J. O. Pedro, "Quadrotor-Based Aerial Manipulator Altitude and Attitude Tracking using Adaptive Super-Twisting Sliding Mode Control, in *the International Conference on Unmanned Aircraft Systems (ICUAS)*, (Atlanta, GA), pp. 144-151, 11-14 Jun 2019.

2. C. Kuchwa-Dube and J. O. Pedro, "Altitude and Attitude Tracking of a Quadrotor-Based Aerial Manipulator using Super Twisting Sliding Mode Control", in *the 6th International Conference on Control, Mechatronics and Automation*, (Tokyo), pp. 65-69, 12-14 Oct 2018.

3. C. Dube and J. O. Pedro, "Modelling and Closed-Loop System Identification of a Quadrotor-Based Aerial Manipulator", *Journal of Physics Conference Series* 1016 012007, pp. 1-9, 2018.

2 Aerial Manipulator Modelling and System Identification

Kinematic and dynamic models are required for simulation of the aerial manipulator system and for formulation of the system's control laws. This chapter details the kinematic and dynamic modelling of quadrotor-based aerial manipulators. First, the Newton-Euler formulation for the quadrotor-based aerial manipulator is presented. The kinematics and dynamics of the quadrotor subsystem and the manipulator subsystem are then separately formulated from first principles. The dynamics of a quadrotor can be modelled from first principles using the Newton-Euler or Euler-Lagrange formulations. Due to its computational efficiency [107], the Newton-Euler formulation is used in this chapter. The integrated model of the quadrotor-based aerial manipulator is then presented. To obtain the quadrotor and aerial manipulator parameters required for the system models, system identification experiments are conducted.

2.1 Newton-Euler Formulation for Aerial Manipulator

Two quadrotor-based aerial manipulators are used in this thesis. One is a simulated aerial manipulator that represents the configuration that would be used for plant sample collection. The second is a physical quarotor-based aerial manipulator that is used for physical experiments. The second quadrotor, without a manipulator, is also used for some of the validation experiments. Figure 2.2 shows an aerial manipulator with a two-DoF serial arm and a gripper, which is a suitable configuration to be used for plant sample collection as the two-DoFs allows for positioning of the gripper to collect the plant sample while the aerial manipulator is hovering above the plant. The quadrotor-based aerial manipulator used for the physical experiments is based on the miniature Crazyflie quadrotor, which has reference frames as shown in Figure 2.1. The modelling in this chapter therefore makes use of the Crazyflie reference frames. The Crazyflie quadrotor used for the experiments in this thesis has battery, weight and computational limitations which mean that only a one-DoF arm is used for the experiments as illustrated in Figure 2.3. The one-DoF arm is also mounted on top of the quadrotor as the optical flow deck used for position feedback of the Crazyflie must be mounted on the bottom of the quadrotor.

There are two reference frames for the quadrotor component of the aerial manipulator, the Earth inertial frame (E-frame) represented by $({}^{E}X, {}^{E}Y, {}^{E}Z)$ and the quadrotor body-fixed frame (B-frame) represented by $({}^{B}x, {}^{B}y, {}^{B}z)$. The inertial and body-fixed frames are

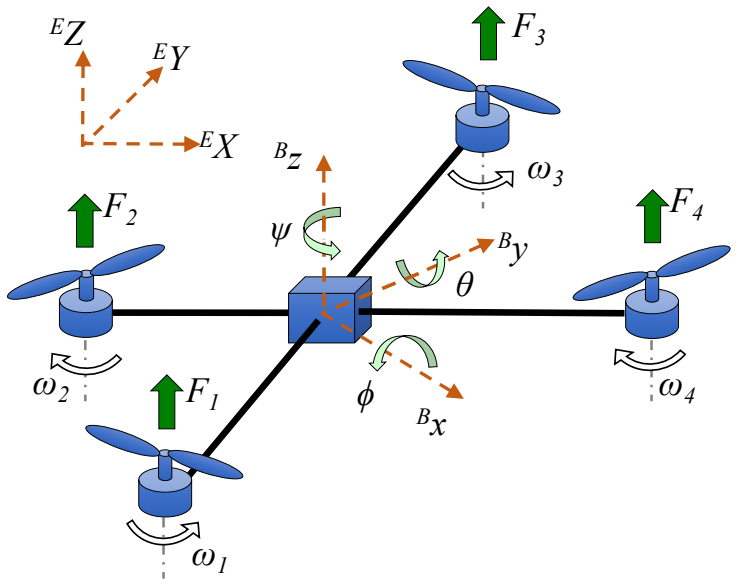


Figure 2.1: Crazyflie quadrotor reference frames.

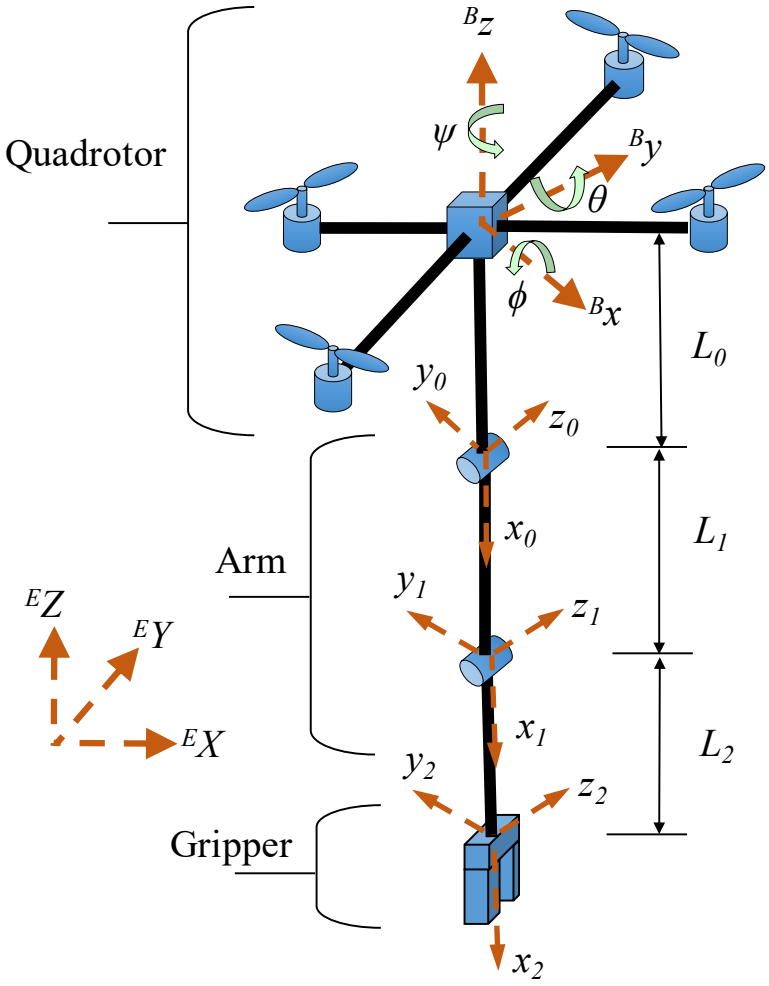


Figure 2.2: Aerial manipulator structure.

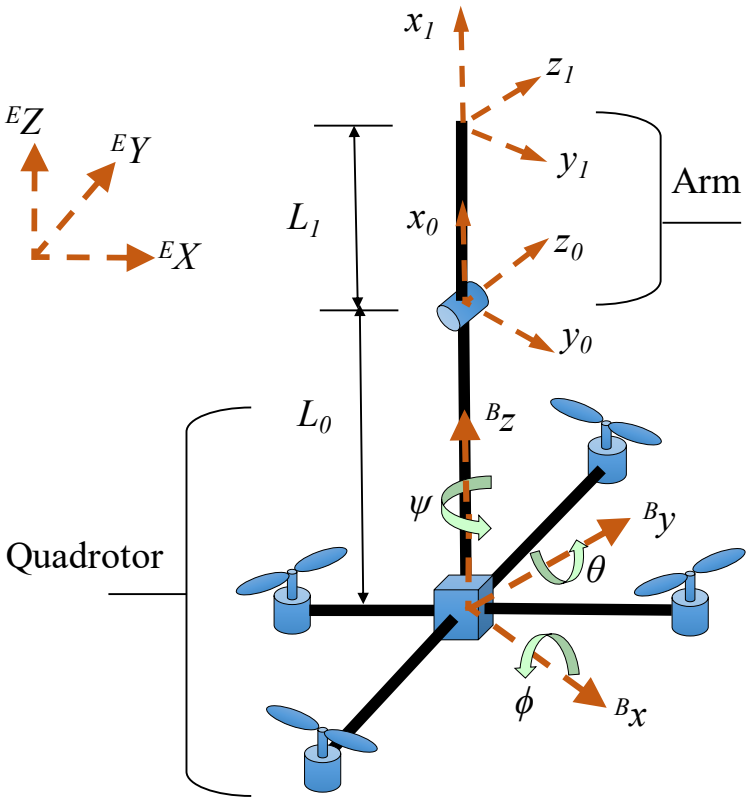


Figure 2.3: Crazyflie aerial manipulator structure.

related by three successive rotations about the z, y and x axes. These are given by the Euler(z-y-x), also named Tait-Bryan, angles; roll ϕ is the rotation about the x-axis, pitch θ is the rotation about the y-axis, and yaw ψ is the rotation about the z-axis.

2.1.1 System reference frames and transformations

The position of the aerial manipulator system in the inertial frame is:

$$^E\mathbf{\Gamma} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{2.1}$$

And the angular position with respect to the inertial frame is:

$$^E\mathbf{\Theta} = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} \tag{2.2}$$

The combined linear position and angular position of the system with respect to the inertial frame is thus:

$$\boldsymbol{\xi} = \begin{bmatrix} ^E\mathbf{\Gamma} \\ ^E\mathbf{\Theta} \end{bmatrix} \tag{2.3}$$

The linear velocity of the system in the body-fixed frame is:

$$^{B}\boldsymbol{V} = \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{2.4}$$

And the angular velocity in the body-fixed frame is:

$$^{B}\boldsymbol{\omega} = \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{2.5}$$

The combined linear velocity and angular velocity of the system in the body-fixed frame is thus:

$$\boldsymbol{\nu} = \begin{bmatrix} ^{B}\boldsymbol{V} \\ ^{B}\boldsymbol{\omega} \end{bmatrix} \tag{2.6}$$

The rotation from the body-fixed frame to the inertial frame is given by the rotation matrix $^{E}\boldsymbol{R}_{B}$:

$$^{E}\boldsymbol{R}_{B} = \boldsymbol{R}_{z}(\psi)\boldsymbol{R}_{y}(\theta)\boldsymbol{R}_{x}(\phi) \tag{2.7}$$

where the rotation matrices about the x y and z axes are:

$$\boldsymbol{R}_{x}(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \tag{2.8}$$

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

$$\boldsymbol{R}_{z}(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.10}$$

This evaluates to:

$$^{E}\boldsymbol{R}_{B} = \begin{bmatrix} \cos\psi\cos\theta & -\cos\psi\sin\theta\sin\phi - \sin\psi\cos\phi & -\cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi \\ \sin\psi\cos\theta & -\sin\psi\sin\theta\sin\phi + \cos\psi\cos\phi & -\sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi \\ \sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix} \tag{2.11}$$

The linear velocities of the body-fixed in the inertial frame are therefore related to the velocities in the body-fixed frame as follows:

$$ {}^{E}\dot{\boldsymbol{\Gamma}} = {}^{E}\boldsymbol{R}_{B}\ {}^{B}\boldsymbol{V} \tag{2.12} $$

To transform the time variation of the Euler angles to angular velocities in the body-fixed frame [108]:

$$ \begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + \boldsymbol{R}_{x}(\phi)^{-1} \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + \boldsymbol{R}_{x}(\phi)^{-1}\boldsymbol{R}_{y}(\theta)^{-1} \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} \tag{2.13} $$

Thus:

$$ {}^{E}\boldsymbol{T}_{B}^{-1} = \begin{bmatrix} 1 & 0 & \sin\theta \\ 0 & \cos\phi & \sin\phi\cos\theta \\ 0 & -\sin\phi & \cos\phi\cos\theta \end{bmatrix} \tag{2.14} $$

and:

$$ {}^{B}\boldsymbol{\omega} = {}^{E}\boldsymbol{T}_{B}^{-1}\ {}^{E}\dot{\boldsymbol{\Theta}} \tag{2.15} $$

To transform from the angular velocities in the body-fixed frame to the time variation of the Euler angles:

$$ {}^{E}\dot{\boldsymbol{\Theta}} = {}^{E}\boldsymbol{T}_{B}\ {}^{B}\boldsymbol{\omega} \tag{2.16} $$

where:

$$ {}^{E}\boldsymbol{T}_{B} = \begin{bmatrix} 1 & -\sin\phi\tan\theta & -\cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \tag{2.17} $$

2.1.2 Dynamic modelling - Newton-Euler formulation

Combining the quadrotor and the manipulator to form an aerial manipulator as shown in Figure 2.2 and Figure 2.3 means that the manipulator has a floating base, that is the quadrotor, instead of a fixed base. The forces and moments of the manipulator also have an effect on the quadrotor. Because the centre of mass position of the entire aerial manipulator system changes with the change in manipulator joint angles, the moments of inertia of the entire system also change. For the combined aerial manipulator system, the Newton-Euler formulation for the dynamics can be written as [107]:

$$ \begin{bmatrix} {}^{B}\boldsymbol{F} \\ {}^{B}\boldsymbol{\tau} \end{bmatrix} + \begin{bmatrix} {}^{B}\boldsymbol{f}_{1} \\ {}^{B}\boldsymbol{\mu}_{1} \end{bmatrix} + \begin{bmatrix} {}^{B}\boldsymbol{F}_{D} \\ {}^{B}\boldsymbol{\tau}_{D} \end{bmatrix} = \begin{bmatrix} m\mathbf{1}_{3} & \mathbf{0}_{3} \\ \mathbf{0}_{3} & \boldsymbol{I}_{S} \end{bmatrix} \begin{bmatrix} {}^{B}\dot{\boldsymbol{V}} \\ {}^{B}\dot{\boldsymbol{\omega}} \end{bmatrix} + \begin{bmatrix} {}^{B}\boldsymbol{\omega} \times m\ {}^{B}\boldsymbol{V} \\ {}^{B}\boldsymbol{\omega} \times \boldsymbol{I}_{S}\ {}^{B}\boldsymbol{\omega} \end{bmatrix} \tag{2.18} $$

where:
${}^{B}\boldsymbol{F}$ is the force due to the quadrotor subsystem,
${}^{B}\boldsymbol{\tau}$ is the torque due to the quadrotor subsystem,
${}^{B}\boldsymbol{f}_1$ is the force due to the manipulator subsystem,
${}^{B}\boldsymbol{\mu}_1$ is the torque due to the manipulator subsystem,
${}^{B}\boldsymbol{F}_D$ is the disturbance force,
${}^{B}\boldsymbol{\tau}_D$ is the disturbance torque,
m is the mass of the system,
$\boldsymbol{I}_S$ is the inertia tensor of the system,
$\mathbf{1}_3$ is a 3×3 identity matrix, and
$\mathbf{0}_3$ is a 3×3 zero matrix.

2.1.3 Aerial manipulator moments of inertia

The centre of mass $\boldsymbol{C}_M$ and moments of inertia $\boldsymbol{I}_S$ of the aerial manipulator are time varying, dependant on the motion of the manipulator. The centre of mass $\boldsymbol{C}_M$ of the aerial manipulator system in the body-fixed frame at a given time instant is:

$$\boldsymbol{C}_M = \frac{\boldsymbol{C}_{MQ} m_Q + \sum_{i=1}^{n} \boldsymbol{C}_{Mi} m_i}{m_Q + \sum_{i=1}^{n} m_i} \tag{2.19}$$

where:
$\boldsymbol{C}_{MQ}$ is the centre of mass of the quadrotor expressed in the body-fixed frame;
$\boldsymbol{C}_{Mi}$ is the centre of mass of link i expressed in the body-fixed frame.

The centre of mass of link i is:

$$\boldsymbol{C}_{Mi} = {}^{B}\boldsymbol{R}_i \, {}^{i}\boldsymbol{r}_{i,ci} + {}^{B}\boldsymbol{p}_0 \tag{2.20}$$

where:

$${}^{B}\boldsymbol{R}_i = {}^{B}\boldsymbol{R}_0 \, {}^{0}\boldsymbol{R}_i \tag{2.21}$$

Transforming the moments of inertia for each link to the system body-fixed frame:

$$\boldsymbol{I}_{cm_i} = {}^{B}\boldsymbol{R}_i \, {}^{i}\boldsymbol{I}_{cm_i} \, {}^{B}\boldsymbol{R}_i^T \tag{2.22}$$

Expressing the link moments of inertia about the aerial manipulator centre of mass instead of the centre of mass of the link:

$$\boldsymbol{I}_i = \boldsymbol{I}_{cm_i} + m_i \boldsymbol{S}\left(\boldsymbol{C}_i\right) \boldsymbol{S}\left(\boldsymbol{C}_i\right)^T \tag{2.23}$$

where:

$$\boldsymbol{C}_i = \boldsymbol{C}_{Mi} - \boldsymbol{C}_M, \tag{2.24}$$

and the skew-symmetric matrix of $\boldsymbol{C}_i$ is:

$$\boldsymbol{S}\left(\boldsymbol{C}_i\right) = \begin{bmatrix} 0 & -z_{c_i} & y_{c_i} \\ z_{c_i} & 0 & -x_{c_i} \\ -y_{c_i} & x_{c_i} & 0 \end{bmatrix}. \tag{2.25}$$

Due to its symmetry, the tensor of inertia of the quadrotor is:

$$\boldsymbol{I}_Q = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \tag{2.26}$$

The moments of inertia $\boldsymbol{I}_S$ of the aerial manipulator at a given time instant are thus:

$$\boldsymbol{I}_S = \sum_{i=1}^{n} \boldsymbol{I}_i + \boldsymbol{I}_Q + m_Q \boldsymbol{S}\left(\boldsymbol{C}_Q\right) \boldsymbol{S}\left(\boldsymbol{C}_Q\right)^T \tag{2.27}$$

2.2 Quadrotor Forces and Moments

The quadrotor subsystem forces, ${}^B\boldsymbol{F}$, and moments ${}^B\boldsymbol{\tau}$ are obtained by considering the sum of different forces and moments acting on the quadrotor. The following simplifying assumptions are made for the quadrotor subsystem:

- The centre of mass of the quadrotor coincides with the origin of the quadrotor body-fixed frame.
- The quadrotor body-fixed frame axes coincide with the body principal axes of inertia.
- The quadrotor is a rigid body.
- The propellers are rigid bodies.
- The quadrotor thrust and drag forces are proportional to the square of the rotor speed.

2.2.1 Quadrotor forces

Neglecting drag force, there are two forces acting on the quadrotor, gravitational force ${}^B\boldsymbol{F}_G$ and thrust force ${}^B\boldsymbol{F}_T$. The total force acting on the quadrotor is therefore:

$${}^B\boldsymbol{F} = {}^B\boldsymbol{F}_T + {}^B\boldsymbol{F}_G \tag{2.28}$$

Gravitational Force

The gravitational force experienced by the quadrotor in the inertial frame is:

$$^{E}\boldsymbol{F}_{G} = \begin{bmatrix} 0 \\ 0 \\ -m_{Q}g \end{bmatrix} \tag{2.29}$$

In the body-fixed frame the gravitational force experienced by the quadrotor is:

$$^{B}\boldsymbol{F}_{G} = {}^{E}\boldsymbol{R}_{B}^{-1}\ {}^{E}\boldsymbol{F}_{G} \tag{2.30}$$

Thus:

$$^{B}\boldsymbol{F}_{G} = -m_{Q}g \begin{bmatrix} \sin\theta \\ \sin\phi\cos\theta \\ \cos\phi\cos\theta \end{bmatrix} \tag{2.31}$$

Thrust Force

The force produced by the ith rotor is:

$$F_{i} = k\omega_{ri}^{2} \tag{2.32}$$

where:
ω_{ri} is the angular velocity of rotor, i
k is the thrust factor.

The total thrust force experienced by the quadrotor in the body-fixed frame is therefore:

$$^{B}\boldsymbol{F}_{T} = k \begin{bmatrix} 0 \\ 0 \\ \omega_{r1}^{2} + \omega_{r2}^{2} + \omega_{r3}^{2} + \omega_{r4}^{2} \end{bmatrix} \tag{2.33}$$

The thrust force in the inertial frame is:

$$^{E}\boldsymbol{F}_{T} = {}^{E}\boldsymbol{R}_{B}\ {}^{B}\boldsymbol{F}_{T} \tag{2.34}$$

Let

$$U_{1} = k\left(\omega_{r1}^{2} + \omega_{r2}^{2} + \omega_{r3}^{2} + \omega_{r4}^{2}\right) \tag{2.35}$$

Thus:

$$^{E}\boldsymbol{F}_{T} = U_{1} \begin{bmatrix} -\cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi \\ -\sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi \\ \cos\theta\cos\phi \end{bmatrix} \tag{2.36}$$

2.2.2 Quadrotor moments

The gravitational force does not have any moments as the centre of gravity coincides with the centre of mass. There are therefore gyroscopic moments due to inertia of the rotors ${}^{B}\boldsymbol{\tau}_{G}$ and thrust moments ${}^{B}\boldsymbol{\tau}_{T}$ acting on the quadrotor. The total moment acting on the quadrotor is therefore:

$$ {}^{B}\boldsymbol{\tau} = {}^{B}\boldsymbol{\tau}_{T} + {}^{B}\boldsymbol{\tau}_{G} \tag{2.37} $$

Thrust Moments

The thrust moment in the body-fixed frame is:

$$ {}^{B}\boldsymbol{\tau}_{T} = \begin{bmatrix} \tau_{\phi} \\ \tau_{\theta} \\ \tau_{\psi} \end{bmatrix} \tag{2.38} $$

The rolling moment for the cross configuration is given by:

$$ \tau_{\phi} = l_{f}\left(-F_{1} - F_{2} + F_{3} + F_{4}\right) \tag{2.39} $$

The pitching moment for the cross configuration is given by:

$$ \tau_{\theta} = l_{s}\left(F_{1} - F_{2} - F_{3} + F_{4}\right) \tag{2.40} $$

where:
l_f is the length from the propeller centre to the quadrotor centre along the y-axis, and
l_s is the length from the propeller centre to the quadrotor centre along the x-axis.

The yawing moment is given by:

$$ \tau_{\psi} = -\tau_{c1} + \tau_{c2} - \tau_{c3} + \tau_{c4} \tag{2.41} $$

where the counter torque produced by the ith rotor is:

$$ \tau_{ci} = c_{Q}\omega_{ri}^{2} \tag{2.42} $$

where c_Q is a constant of proportionality.

Thus for the cross configuration:

$$ {}^{B}\boldsymbol{\tau}_{T} = \begin{bmatrix} l_{f}k\left(-\omega_{r1}^{2} - \omega_{r2}^{2} + \omega_{r3}^{2} + \omega_{r4}^{2}\right) \\ l_{s}k\left(\omega_{r1}^{2} - \omega_{r2}^{2} - \omega_{r3}^{2} + \omega_{r4}^{2}\right) \\ c_{Q}\left(-\omega_{r1}^{2} + \omega_{r2}^{2} - \omega_{r3}^{2} + \omega_{r4}^{2}\right) \end{bmatrix} \tag{2.43} $$

Gyroscopic Moment

The gyroscopic moment due to the rotating rotors is:

$$ {}^{B}\boldsymbol{\tau}_{G} = I_r \begin{bmatrix} q\left(-\omega_{r1} + \omega_{r2} - \omega_{r3} + \omega_{r4}\right) \\ p\left(\omega_{r1} - \omega_{r2} + \omega_{r3} - \omega_{r4}\right) \\ 0 \end{bmatrix} \tag{2.44} $$

where I_r is the rotor moment of inertia. Let:

$$ \bar{\omega_r} = -\omega_{r1} + \omega_{r2} - \omega_{r3} + \omega_{r4} \tag{2.45} $$

Thus:

$$ {}^{B}\boldsymbol{\tau}_{G} = I_r \begin{bmatrix} q\bar{\omega_r} \\ -p\bar{\omega_r} \\ 0 \end{bmatrix} \tag{2.46} $$

2.2.3 Rotor modelling

The control inputs for the cross configuration are:

$$ \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} k & k & k & k \\ -kl_f & -kl_f & kl_f & kl_f \\ kl_s & -kl_s & -kl_s & kl_s \\ -c_Q & +c_Q & -c_Q & +c_Q \end{bmatrix} \begin{bmatrix} \omega_{r1}^2 \\ \omega_{r2}^2 \\ \omega_{r3}^2 \\ \omega_{r4}^2 \end{bmatrix} \tag{2.47} $$

In the faulty case, the control inputs can thus be written as [61]:

$$ \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} k & k & k & k \\ -kl_f & -kl_f & kl_f & kl_f \\ kl_s & -kl_s & -kl_s & kl_s \\ -c_Q & c_Q & -c_Q & c_Q \end{bmatrix} \begin{bmatrix} \left(f_{e1}\ \omega_{r1}\right)^2 \\ \left(f_{e2}\ \omega_{r2}\right)^2 \\ \left(f_{e3}\ \omega_{r3}\right)^2 \\ \left(f_{e4}\ \omega_{r4}\right)^2 \end{bmatrix} \tag{2.48} $$

where $0 < f_{ei} \leq 1$ is the remaining effectiveness of rotor i.

2.3 Manipulator Forward Kinematics Modelling

The forward kinematics modelling describes the pose and velocities of the manipulator links given the joint angles of the manipulator. With direct kinematics, the pose of the manipulator is found given the joint angles whereas with differential kinematics the motion of the whole manipulator is described given the joint velocities. For the one-DoF manipulator mounted on the Crazyflie quadrotor, the arm position is simply changed by changing the joint angle of the actuator. The forward kinematics for a multi-DoF manipulator are, however, presented for completeness and for modelling of the two-DoF manipulator. To model the direct kinematics of a multi-DoF manipulator with n joints, a reference frame O_i, $(i = 0, 1, \ldots, n)$, is attached to each joint of the manipulator and to the end effector using the Denavit-Hartenberg convention [109, 110]:

- The $\boldsymbol{z}_i$ axis of frame O_i is located at the axis of joint $i + 1$,
- The $\boldsymbol{x}_i$ axis of frame O_i is located perpendicular to $\boldsymbol{z}_{i-1}$ and $\boldsymbol{z}_i$,
- The $\boldsymbol{x}_0$ axis is arbitrarily chosen,
- The $\boldsymbol{y}_i$ axis is found using the right hand rule.

Four parameters are then used to locate one reference frame relative to another [109,110]:

- The link length a_i, is the distance between the $\boldsymbol{z}_{i-1}$ and $\boldsymbol{z}_i$ axes along the $\boldsymbol{x}_i$ axis,
- The joint offset d_i, is the $\boldsymbol{x}_i$ coordinate along the $\boldsymbol{z}_{i-1}$ axis,
- The link twist α_i, is the angle from the $\boldsymbol{z}_{i-1}$ axis to the $\boldsymbol{z}_i$ axis about the $\boldsymbol{x}_i$ axis, where the positive rotation is counter-clockwise,
- The joint angle θ_i, is the angle from the $\boldsymbol{x}_{i-1}$ axis to the $\boldsymbol{x}_i$ axis about the $\boldsymbol{z}_{i-1}$ axis, where the positive rotation is counter-clockwise.

Of the four parameters, two parameters a_i and α_i are constant and are dependant on the geometry of the connection between consecutive joints [109–111]. For revolute joints the θ_i parameter is a variable, while for translational joints, the d_i parameter is a variable [109–111].

2.3.1 Direct kinematics modelling

The position of the origin of coordinate frame O_i relative to coordinate frame O_{i-1} can be denoted by vector ${}^{i-1}\boldsymbol{p}_i$. The orientation of frame O_i relative to frame O_{i-1} can be denoted by the rotation matrix ${}^{i-1}\boldsymbol{R}_i$. The homogeneous transformation matrix ${}^{i-1}\boldsymbol{A}_i$ combines the position vectors and rotation matrices [110, 111]:

$$ {}^{i-1}\boldsymbol{A}_i = \begin{bmatrix} {}^{i-1}\boldsymbol{R}_i & {}^{i-1}\boldsymbol{p}_i \\ \boldsymbol{0} & 1 \end{bmatrix} \tag{2.49} $$

Its inverse ${}^{i-1}\boldsymbol{A}_i^{-1}$ transforms vectors from coordinate frame O_{i-1} to coordinate frame O_i.

$$ {}^{i}\boldsymbol{A}_{i-1} = {}^{i-1}\boldsymbol{A}_i^{-1} = \begin{bmatrix} {}^{i-1}\boldsymbol{R}_i^T & -{}^{i-1}\boldsymbol{R}_i^T \, {}^{i-1}\boldsymbol{p}_i \\ \boldsymbol{0} & 1 \end{bmatrix} \tag{2.50} $$

Using the Denavit-Hartenberg parameters, the transformation matrix ${}^{i-1}\boldsymbol{A}_i$ of frame O_i to frame O_{i-1} is then [110]:

$$ {}^{i-1}\boldsymbol{A}_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i \cos\alpha_i & \sin\theta_i \sin\alpha_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\theta_i \cos\alpha_i & -\cos\theta_i \sin\alpha_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.51} $$

The homogeneous transformation matrix, ${}^{0}\boldsymbol{T}_i$ gives the position and orientation of link i with respect to the base frame and transforms vectors from coordinate frame O_i to coordinate frame O_0 [109,110]. The transformation matrix of frame n to the base is [110]:

$$ {}^{0}\boldsymbol{T}_n(\boldsymbol{q}) = \; {}^{0}\boldsymbol{A}_1(q_1) \;\; {}^{1}\boldsymbol{A}_2(q_2) \;\; \ldots \;\; {}^{n-1}\boldsymbol{A}_n(q_n) \tag{2.52} $$

where $\boldsymbol{q}$ is the $n \times 1$ vector of joint variables.

$$ \begin{aligned} \boldsymbol{q} &= [\, q_1 \;\; q_2 \;\; \ldots \;\; q_n \,]^T \\ &= [\, \theta_1 \;\; \theta_2 \;\; \ldots \;\; \theta_n \,]^T \end{aligned} \tag{2.53} $$

Figure 2.2 shows the manipulator structure for a manipulator with a two-DoF serial arm and a gripper. Table 2.1 shows the manipulator Denavit-Hartenberg parameters for the two-DoF arm.

Table 2.1: Manipulator Denavit-Hartenberg parameters

Frame	a_i	d_i	α_i (degrees)	θ_i (degrees)
1	L_1	0	0	θ_1
2	L_2	0	0	θ_2

2.3.2 Differential kinematic modelling

Differential kinematics deals with the relationship between the manipulator's joint velocities and the corresponding link's linear and angular velocities. The relationship between the manipulator joint velocities $\dot{\boldsymbol{q}}$ and the manipulator link linear velocities $\dot{\boldsymbol{p}}$ and angular velocity $\boldsymbol{\omega}$ is given by the manipulator's Jacobian matrix $\boldsymbol{J}$ [110]:

$$\boldsymbol{v} = \begin{bmatrix} \dot{\boldsymbol{p}} \\ \boldsymbol{\omega} \end{bmatrix} = \boldsymbol{J}\dot{\boldsymbol{q}} \tag{2.54}$$

For a serial manipulator with rotational joints, the Jacobian is given by [110]:

$$\boldsymbol{J} = \begin{bmatrix} \boldsymbol{J}_{P1} & \dots & \boldsymbol{J}_{Pn} \\ \boldsymbol{J}_{O1} & \dots & \boldsymbol{J}_{On} \end{bmatrix} = \begin{bmatrix} \boldsymbol{J}_P \\ \boldsymbol{J}_O \end{bmatrix} \tag{2.55}$$

where [110]:

$$\begin{bmatrix} \boldsymbol{J}_{Pi} \\ \boldsymbol{J}_{Oi} \end{bmatrix} = \begin{bmatrix} \boldsymbol{z}_{i-1} \times (\boldsymbol{p} - \boldsymbol{p}_{i-1}) \\ \boldsymbol{z}_{i-1} \end{bmatrix} \tag{2.56}$$

where:
$\boldsymbol{J}_P$ is the $3 \times n$ matrix which gives the contribution of the manipulator joints' angular velocities $\dot{\boldsymbol{q}}$ to the manipulator's linear velocities $\dot{\boldsymbol{p}}$;
$\boldsymbol{J}_O$ is the $3 \times n$ matrix which gives the contribution of the manipulator joints' angular velocities $\dot{\boldsymbol{q}}$ to the manipulator's angular velocities $\boldsymbol{\omega}$;
$\boldsymbol{z}_{i-1}$ is the unit vector of $\boldsymbol{z}$ axis of frame O_{i-1} and is obtained from the third column of the rotation matrix ${}^0\boldsymbol{R}_{i-1}$.

$\boldsymbol{z}_0$ is given as:

$$\boldsymbol{z}_0 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{2.57}$$

$\boldsymbol{p}_{i-1}$ is the position vector of the origin of frame O_{i-1} and is obtained from the first three elements of the fourth column of the transformation matrix ${}^0\boldsymbol{T}_{i-1}$.

$\boldsymbol{p}_0$ is given by:

$$\boldsymbol{p}_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{2.58}$$

2.4 Manipulator Dynamics - Recursive Newton-Euler Formulation

The recursive Newton-Euler formulation is used in this thesis to describe the manipulator dynamics [110]. First, there is a forward recursion which results in the manipulator link accelerations. The backwards recursion then gives the forces and moments exerted by the links.

2.4.1 Forward recursion

For a revolute joint, the angular velocity ${}^i\boldsymbol{\omega}_i$ of link i expressed in frame i is [110]:

$${}^i\boldsymbol{\omega}_i = {}^{i-1}\boldsymbol{R}_i^T \left({}^{i-1}\boldsymbol{\omega}_{i-1} + \dot{\theta}_i \boldsymbol{z}_0 \right) \tag{2.59}$$

The angular acceleration ${}^i\dot{\boldsymbol{\omega}}_i$ of link i expressed in frame i is then [110]:

$${}^i\dot{\boldsymbol{\omega}}_i = {}^{i-1}\boldsymbol{R}_i^T \left({}^{i-1}\dot{\boldsymbol{\omega}}_{i-1} + \ddot{\theta}_i \boldsymbol{z}_0 + \dot{\theta}_i \, {}^{i-1}\boldsymbol{\omega}_{i-1} \times \boldsymbol{z}_0 \right) \tag{2.60}$$

The linear acceleration ${}^i\ddot{\boldsymbol{p}}_i$ of link i is the linear acceleration of the origin of coordinate frame O_i relative to coordinate frame O_i and is given as [110]:

$${}^i\ddot{\boldsymbol{p}}_i = {}^{i-1}\boldsymbol{R}_i^T \, {}^{i-1}\ddot{\boldsymbol{p}}_{i-1} + {}^i\dot{\boldsymbol{\omega}}_i \times {}^i\boldsymbol{r}_{i-1,i} + {}^i\boldsymbol{\omega}_i \times \left({}^i\boldsymbol{\omega}_i \times {}^i\boldsymbol{r}_{i-1,i} \right) \tag{2.61}$$

where:
${}^i\boldsymbol{r}_{i-1,i}$ is the vector from coordinate frame O_{i-1} to coordinate frame O_i expressed in coordinate frame O_i.

The linear acceleration ${}^i\ddot{\boldsymbol{p}}_{ci}$ of the centre of mass of link i relative to coordinate frame O_i is [110]:

$${}^i\ddot{\boldsymbol{p}}_{ci} = {}^i\ddot{\boldsymbol{p}}_i + {}^i\dot{\boldsymbol{\omega}}_i \times {}^i\boldsymbol{r}_{i,ci} + {}^i\boldsymbol{\omega}_i \times \left({}^i\boldsymbol{\omega}_i \times {}^i\boldsymbol{r}_{i,ci} \right) \tag{2.62}$$

where:
${}^i\boldsymbol{r}_{i,ci}$ is the vector from coordinate frame O_i to centre of mass of link i expressed in coordinate frame O_i.

2.4.2 Backward recursion

The force ${}^{i}\boldsymbol{f}_{i}$ exerted by link $i-1$ on link i expressed in frame i is [110]:

$$ {}^{i}\boldsymbol{f}_{i} = {}^{i}\boldsymbol{R}_{i+1}\ {}^{i+1}\boldsymbol{f}_{i+1} + m_i\ {}^{i}\ddot{\boldsymbol{p}}_{ci} \tag{2.63} $$

where m_i is the mass of link i.

The moment ${}^{i}\boldsymbol{\mu}_{i}$ exerted by link $i-1$ on link i with respect to frame 0_{i-1} expressed in frame i is [110]:

$$ {}^{i}\boldsymbol{\mu}_{i} = -{}^{i}\boldsymbol{f}_{i}\times\left({}^{i}\boldsymbol{r}_{i-1,i} + {}^{i}\boldsymbol{r}_{i,ci}\right) + {}^{i}\boldsymbol{R}_{i+1}\ {}^{i+1}\boldsymbol{\mu}_{i+1} + {}^{i}\boldsymbol{R}_{i+1}\ {}^{i+1}\boldsymbol{f}_{i+1}\times{}^{i}\boldsymbol{r}_{i,ci} + {}^{i}\boldsymbol{I}_{i}\ {}^{i}\dot{\boldsymbol{\omega}}_{i} + {}^{i}\boldsymbol{\omega}_{i}\times\left({}^{i}\boldsymbol{I}_{i}\ {}^{i}\boldsymbol{\omega}_{i}\right) \tag{2.64} $$

where:
${}^{i}\boldsymbol{I}_{i}$ is the inertia tensor of link i expressed in frame i located at the centre of mass of link i.

The torque $\boldsymbol{\tau}_{i}$ of link i is [110]:

$$ \boldsymbol{\tau}_{i} = {}^{i}\boldsymbol{\mu}_{i}^{T}\ {}^{i-1}\boldsymbol{R}_{i}^{T}\ \boldsymbol{z}_{0} \tag{2.65} $$

Modelling each link of the manipulator as a solid cylinder, the moments of inertia are:

$$ I_{xi} = I_{zi} = \frac{m_i}{12}\left(3r_{li}^{2} + L_{li}^{2}\right) \tag{2.66} $$

and:

$$ I_{yi} = \frac{m_i r_{li}^{2}}{2} \tag{2.67} $$

where L_{li} is the length of the cylinder and r_{li} is the radius of the cylinder.

2.4.3 Initial and terminal conditions

The position of the origin of the manipulator base frame expressed in the quadrotor body-fixed frame for the aerial manipulator with the two-DoF manipulator in Figure 2.2 is:

$$ {}^{B}\boldsymbol{p}_{0} = \begin{bmatrix} 0 \\ 0 \\ -L_0 \end{bmatrix} \tag{2.68} $$

For the one-DoF manipulator of the Crazyflie-based aerial manipulator in Figure 2.3, since the manipulator is mounted on top of the quadrotor, the position of the origin of the manipulator base frame expressed in the quadrotor body-fixed frame is:

$$^{B}\boldsymbol{p}_0 = \begin{bmatrix} 0 \\ 0 \\ L_0 \end{bmatrix} \tag{2.69}$$

For a manipulator with a floating base, the initial velocities and accelerations are those of the quadrotor. The initial linear velocity $^{0}\dot{\boldsymbol{p}}_0$ and initial linear acceleration $^{0}\ddot{\boldsymbol{p}}_0$, expressed in the manipulator base frame are:

$$^{0}\dot{\boldsymbol{p}}_0 = {}^{B}\boldsymbol{R}_0^T \; {}^{E}\boldsymbol{R}_B^T \; {}^{E}\dot{\boldsymbol{\Gamma}} \tag{2.70}$$

$$^{0}\ddot{\boldsymbol{p}}_0 = {}^{B}\boldsymbol{R}_0^T \; {}^{E}\boldsymbol{R}_B^T \left({}^{E}\ddot{\boldsymbol{\Gamma}} - \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} \right) \tag{2.71}$$

For the manipulator in Figure 2.2 the rotation matrix is:

$$^{B}\boldsymbol{R}_0 = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 1 \\ -1 & 0 & 0 \end{bmatrix} \tag{2.72}$$

For the Crazyflie-based aerial manipulator in Figure 2.3 the rotation matrix is:

$$^{B}\boldsymbol{R}_0 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \tag{2.73}$$

The initial angular velocity $^{0}\boldsymbol{\omega}_0$ and initial angular acceleration $^{0}\dot{\boldsymbol{\omega}}_0$,expressed in the manipulator base frame are:

$$^{0}\boldsymbol{\omega}_0 = {}^{B}\boldsymbol{R}_0^T \; {}^{B}\boldsymbol{\omega} \tag{2.74}$$

$$^{0}\dot{\boldsymbol{\omega}}_0 = {}^{B}\boldsymbol{R}_0^T \; {}^{B}\dot{\boldsymbol{\omega}} \tag{2.75}$$

And the terminal conditions for the forces $^{n+1}\boldsymbol{f}_{n+1}$ and moments $^{n+1}\boldsymbol{\mu}_{n+1}$ when position control alone is considered with no external load are [110]:

$$^{n+1}\boldsymbol{f}_{n+1} = \mathbf{0} \tag{2.76}$$

$$^{n+1}\boldsymbol{\mu}_{n+1} = \mathbf{0} \tag{2.77}$$

where:

$$^{B}\boldsymbol{f}_1 = {}^{B}\boldsymbol{R}_1 \; {}^{1}\boldsymbol{f}_1 \tag{2.78}$$

and:

$$^{B}\boldsymbol{\mu}_1 = {}^{B}\boldsymbol{R}_1 \; {}^{1}\boldsymbol{\mu}_1 \tag{2.79}$$

are the forces $^{B}\boldsymbol{f}_1$, and torques $^{B}\boldsymbol{\mu}_1$ exerted by the manipulator on the quadrotor.

2.5 Aerial Manipulator System Equations of Motion

From equation (2.18):

$$^{B}\boldsymbol{F} + {}^{B}\boldsymbol{f}_1 + {}^{B}\boldsymbol{F}_D = m\mathbf{1}_3 \; {}^{B}\dot{\boldsymbol{V}} + {}^{B}\boldsymbol{\omega} \times m \; {}^{B}\boldsymbol{V} \tag{2.80}$$

Thus:

$$^{B}\dot{\boldsymbol{V}} = \frac{1}{m}\left({}^{B}\boldsymbol{F} + {}^{B}\boldsymbol{f}_1 + {}^{B}\boldsymbol{F}_D - {}^{B}\boldsymbol{\omega} \times m \; {}^{B}\boldsymbol{V}\right) \tag{2.81}$$

From equation (2.81):

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = \frac{1}{m}\left(-m_Q g \begin{bmatrix} \sin\theta \\ \sin\phi\cos\theta \\ \cos\phi\cos\theta \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ U_1 \end{bmatrix} - m \begin{bmatrix} qw - rv \\ ru - pw \\ pv - qu \end{bmatrix} + {}^{B}\boldsymbol{f}_1 + {}^{B}\boldsymbol{F}_D\right) \tag{2.82}$$

From (2.18)

$$^{B}\boldsymbol{\tau} + {}^{B}\boldsymbol{\mu}_1 + {}^{B}\boldsymbol{\tau}_D = \boldsymbol{I}_S \; {}^{B}\dot{\boldsymbol{\omega}} + {}^{B}\boldsymbol{\omega} \times \boldsymbol{I}_S \; {}^{B}\boldsymbol{\omega} \tag{2.83}$$

Thus:

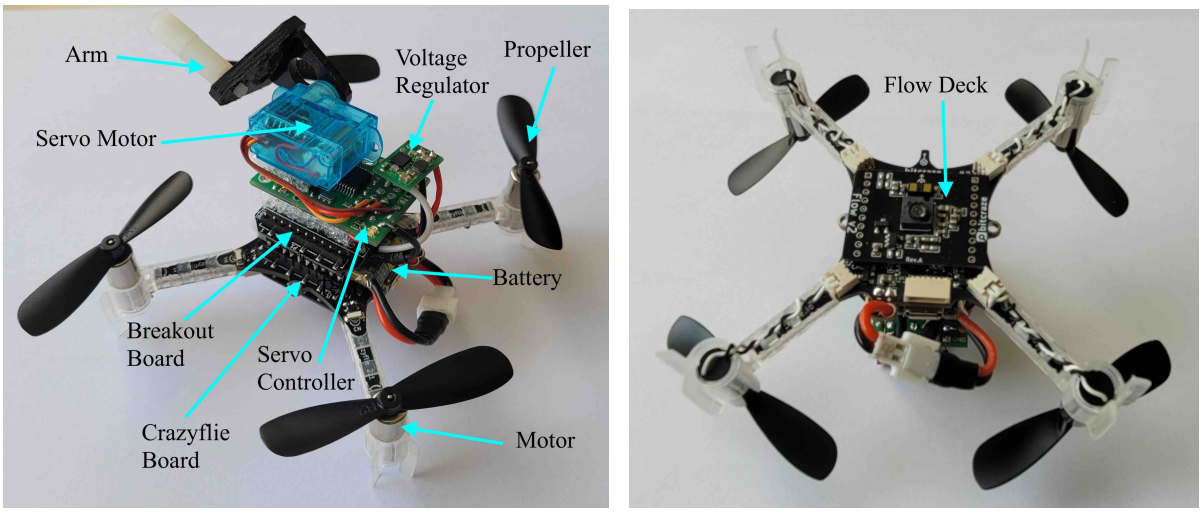
$$^{B}\dot{\boldsymbol{\omega}} = \boldsymbol{I}_S^{-1}\left({}^{B}\boldsymbol{\tau} + {}^{B}\boldsymbol{\mu}_1 + {}^{B}\boldsymbol{\tau}_D - {}^{B}\boldsymbol{\omega} \times \boldsymbol{I}_S \; {}^{B}\boldsymbol{\omega}\right) \tag{2.84}$$

and:

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \boldsymbol{I}_S^{-1}\left(\begin{bmatrix} U_2 \\ U_3 \\ U_4 \end{bmatrix} + I_r \begin{bmatrix} q\bar{\omega_r} \\ -p\bar{\omega_r} \\ 0 \end{bmatrix} - {}^{B}\boldsymbol{\omega} \times \boldsymbol{I}_S \; {}^{B}\boldsymbol{\omega} + {}^{B}\boldsymbol{\mu}_1 + {}^{B}\boldsymbol{\tau}_D\right) \tag{2.85}$$

2.6 System Identification

The experimental platform used in this thesis is a Crazyflie quadrotor-based aerial manipulator shown in Figure 2.4. The aerial manipulator is made up of the Crazyflie quadrotor subsystem and the one-DoF manipulator subsystem. Quadrotors with a very small frame size, such as the Crazyflie quadrotor make use of coreless DC motors, unlike larger-sized quadrotors which make use of brushless DC motors. An optical flow sensor system is used for position feedback of the quadrotor. Since the optical flow sensor deck is mounted on the bottom of the quadrotor, the manipulator is mounted on top of the quadrotor. The Crazyflie breakout deck is used to send a signal to the manipulator's servo controller in order to extend and retract the manipulator. The manipulator is powered by the Crazyflie battery via the breakout board and a voltage regulator. Radio communication is used to communicate between the quadrotor and a computer.



(a) Top (b) Bottom

Figure 2.4: Crazyflie aerial manipulator.

Crazylie quadrotor components and radio communication:

- 1 x Crazyflie 2.1 control board,
- 4 x 7mm DC coreless motor,
- 1 x Flow deck v2,
- 4 x 7mm motor mounts,
- 4 x Propellers,
- 2 x Long male deck connectors,
- 1 x LiPo battery - 1 cell, 3.7V, 250mAh,
- 1 x Crazyflie radio.

Manipulator components:

- 1 x Crazyflie Breakout deck,
- 1 x 5V Step-Up Voltage Regulator,
- 1 x Micro Maestro Servo Controller,
- 1 x CYS S0307 3.9g Analog Servo Motor,
- 1 x Robot arm.

2.6.1 Rotor parameter identification - rotor speed to PWM

The Crazyflie quadrotor's rotors are controlled by sending pulse width modulation (PWM) duty cycle signals to the rotors rather than sending the desired rotor speeds. The Crazyflie has a 16 bit PWM duty cycle. To obtain the relationship between the PWM duty cycle values and the rotor speeds in revolutions per minute (RPM) of the quadrotor, an infrared sensor switch module with 3.3 V to 5 V range and working range of 2 cm to 4 cm connected to an Arduino-based microcontroller is used as shown in Figure 2.5. A bright lamp is used to keep the light levels consistent during the experiments. The quadrotor is placed on a mount that has weights attached to it. PWM duty cycle step inputs are applied to the rotor and the output logged for 10 seconds. The time in milliseconds and the output from the sensor is logged via the Arduino serial monitor and the rotor speed in RPM is calculated. The experiment is repeated a number of times for each PWM duty cycle value.

Figure 2.6 shows an example of the output obtained for a PWM duty cycle value step change of 20000 to 40000. There are transient periods when the rotor speed increases from its initial value to the final value, and when the speed decreases to the initial value. The average rotor speed obtained, excluding the transient periods, is used as the rotor speed for the given PWM duty cycle step input. Figure 2.7 shows the output of the rotor speed experiments for the four rotors. As can be seen from the figure, there is a nonlinear relationship between the PWM duty cycle values and the rotor speeds. The relationship between the PWM duty cycle values and the rotor speeds in RPM is then given by:

$$PWM_{ri} = 0.0001164 \times \omega_{ri}^2 - 0.3556 \times \omega_{ri} + 2188 \tag{2.86}$$

With larger-sized quadrotors that make use of brushless DC motors, electronic speed controllers (ESCs) are used to control the brushless DC motors. The ESCs typically give a linear relationship between the PWM duty cycle values and rotor speed values in RPM. Thus, the relationship for the larger quadrotor is estimated as:

$$PWM_{ri} = 1.35 \times \omega_{ri} + 215.9 \tag{2.87}$$

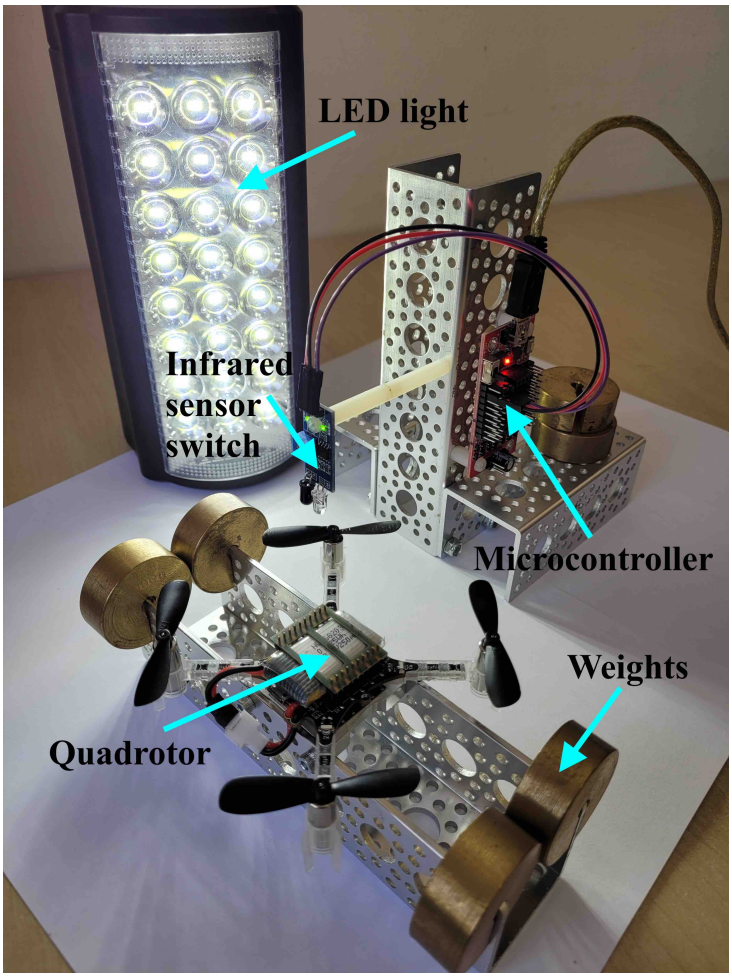


Figure 2.5: Rotor speed measurement.

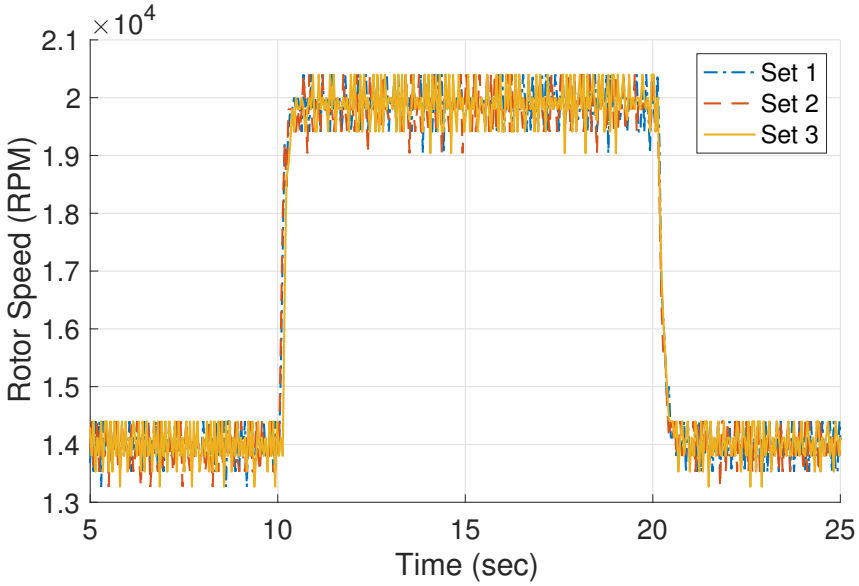


Figure 2.6: Rotor step response.

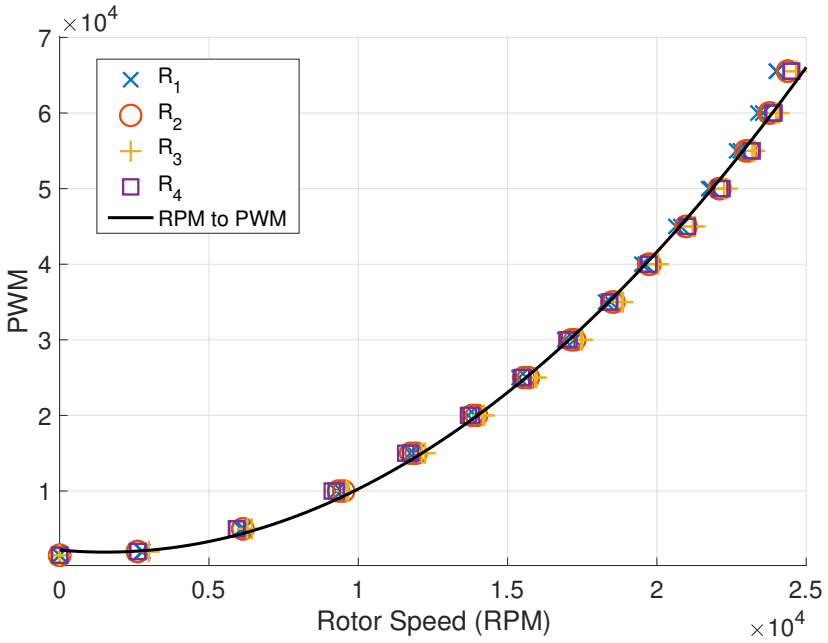


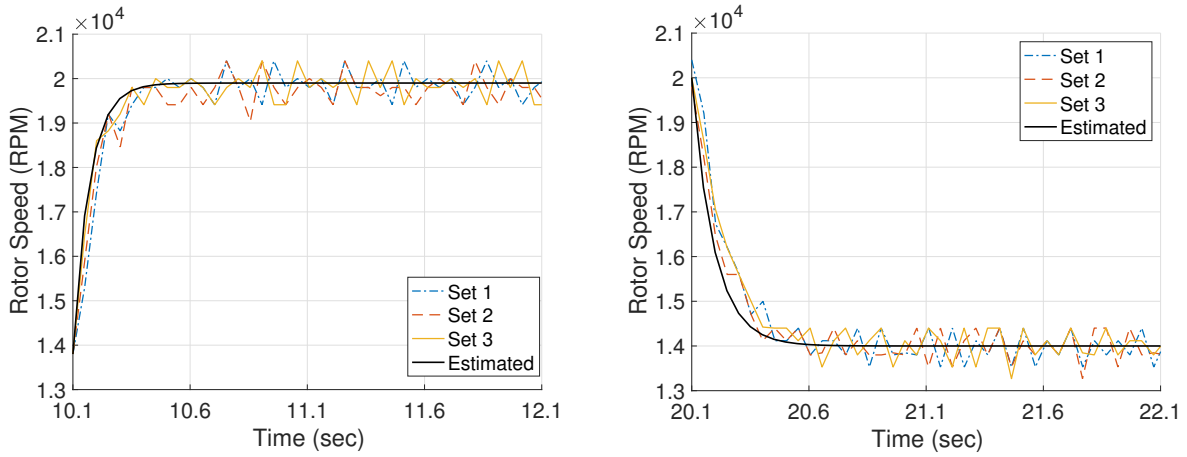
Figure 2.7: Rotor speed to PWM duty cycle estimate. The PWM values are the duty cycle values for a 16 bit PWM register.

2.6.2 Rotor parameter identification - rotor time constants

The time constants for the rotor dynamics are obtained using the same data from the step response experiments. As can be seen from Figure 2.6, the rotors have different responses based on whether the rotor speed is increasing or decreasing. Figures 2.8a and 2.8b show more detailed views of the increasing and decreasing transient periods for the step response. Two time constants, one for an increase in rotor speed and another for a decrease in rotor speed, are therefore used and their values are given in Tables 2.2 and 2.3. A first-order dynamic model is used to model the rotor dynamics. Figure 2.8a shows the estimated rotor response for an increase in rotor speed and Figure 2.8b shows the estimated response for a decrease in rotor speed using the first-order dynamic model. The model is:

$$\Delta\omega_{ri} = \left(\omega_{ri}^{des} - \omega_{ri}\right) \times \left(1 - e^{\left(\frac{-\Delta t}{T_c}\right)}\right) \tag{2.88}$$

where:
T_c is the time constant,
Δt is the change in time,
$\Delta\omega_{ri}$ is the change in rotor speed,
ω_{ri} is the rotor speed, and
ω_{ri}^{des} is the desired rotor speed.



(a) Increasing speed.

(b) Decreasing speed.

Figure 2.8: Rotor step response.

2.6.3 System identification from flight data

To obtain the thrust factor, drag factor and moments of inertia of the Crazyflie quadrotor and of the Crazyflie quadrotor-based aerial manipulator, flight data from the quadrotor and the aerial manipulator under altitude and attitude control (that is the Z position, and the roll, pitch and yaw angle control) are used. The Crazyflie firmware allows for position control (that is Z, X and Y control) and yaw angle control. The quadrotor is an underactuated system thus the X and Y DoFs are not directly controllable and are used to create virtual control inputs to control the roll and pitch DoFs as described later in Chapter 3. The firmware is therefore modified to allow for altitude and attitude control of the quadrotor. This modification allows for all of the controllable DoFs (the Z, roll, pitch, and yaw DoFs) of the quadrotor to be directly controlled. For the aerial manipulator, the manipulator is at the rest position throughout the flight. Using the quadrotor's modified firmware, the controllers output the desired control inputs, which lead to the desired PWM duty cycle values and desired rotor speeds using equations (2.47) and (2.86). By using equation (2.88), the actual rotor speeds are estimated with $\omega_{ri} = \omega_{ri}^{act}$.

Simplified dynamic equations of the quadrotor and the quadrotor-based aerial manipulator, obtained from equations (2.82) and (2.85), are used to estimate the quadrotor and aerial manipulator system parameters. Converting into the inertial frame, the inertial frame centrifugal force are cancelled out. For the aerial manipulator, the manipulator motion terms are excluded since the manipulator remains in its rest position and no manipulator motion is applied. For both the quadrotor and the aerial manipulator, disturbance terms are also excluded since disturbances are not applied during the test, thus:

$$ {}^{E}\boldsymbol{F}_T + {}^{E}\boldsymbol{F}_G = m\mathbf{1}_3 {}^{E}\ddot{\boldsymbol{\Gamma}} \tag{2.89} $$

which leads to:

$$\begin{aligned}
\ddot{X} &= (-\cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi)\,\frac{U_1}{m} \\
\ddot{Y} &= (-\sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi)\,\frac{U_1}{m} \\
\ddot{Z} &= -g + (\cos\theta\cos\phi)\,\frac{U_1}{m}
\end{aligned} \tag{2.90}$$

Assuming that angular velocity in the body-fixed frame and the rate of change of the Euler angles are equal if the change in pitch and roll angles are small, using the small-angle approximation, the matrix that transforms the angular velocities in the body-fixed frame to the time variation of the Euler angles, ${}^{E}\boldsymbol{T}_{B}$, is approximately equal to $\mathbf{1}_3$. The terms that include the rotor inertia I_r are considered as negligible in the simplified dynamic equations thus:

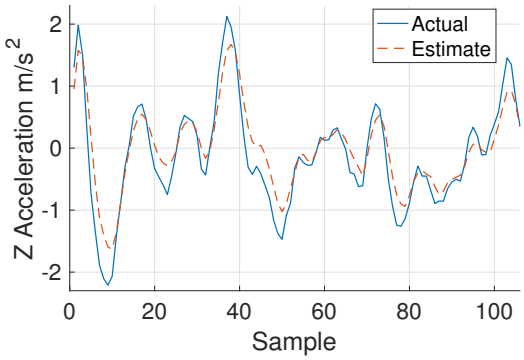
$$\begin{aligned}
\ddot{\phi} &= \frac{U_2}{I_{xx}} + \frac{I_{yy} - I_{zz}}{I_{xx}}\dot{\theta}\dot{\psi} \\
\ddot{\theta} &= \frac{U_3}{I_{yy}} + \frac{I_{zz} - I_{xx}}{I_{yy}}\dot{\phi}\dot{\psi} \\
\ddot{\psi} &= \frac{U_4}{I_{zz}} + \frac{I_{xx} - I_{yy}}{I_{zz}}\dot{\theta}\dot{\phi}
\end{aligned} \tag{2.91}$$

Using equations (2.90) and (2.91), and equation (2.47), with $\omega_{ri} = \omega_{ri}^{act}$ the relevant equations for the system identification become:

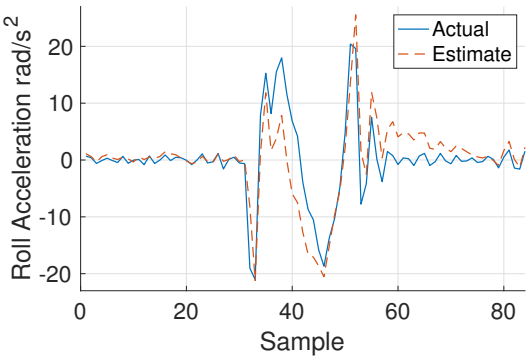
$$\begin{aligned}
\ddot{Z} &= \frac{k\left(\omega_{r1}^2 + \omega_{r2}^2 + \omega_{r3}^2 + \omega_{r4}^2\right)}{m_Q}\left(\cos\theta\cos\phi\right) - g \\
\ddot{\phi} &= \frac{kl_f\left(-\omega_{r1}^2 - \omega_{r2}^2 + \omega_{r3}^2 + \omega_{r4}^2\right)}{I_{xx}} + \frac{I_{yy} - I_{zz}}{I_{xx}}\dot{\theta}\dot{\psi} \\
\ddot{\theta} &= \frac{kl_s\left(\omega_{r1}^2 - \omega_{r2}^2 - \omega_{r3}^2 + \omega_{r4}^2\right)}{I_{yy}} + \frac{I_{zz} - I_{xx}}{I_{yy}}\dot{\phi}\dot{\psi} \\
\ddot{\psi} &= \frac{c_Q\left(-\omega_{r1}^2 + \omega_{r2}^2 - \omega_{r3}^2 + \omega_{r4}^2\right)}{I_{zz}} + \frac{I_{xx} - I_{yy}}{I_{zz}}\dot{\theta}\dot{\phi}
\end{aligned} \tag{2.92}$$

First, the thrust factor k is obtained by using equation (2.92) for the Z acceleration. The Z acceleration, the estimated actual rotor speeds, and the roll and pitch angles obtained from the experiments are used and the thrust factor is estimated using the

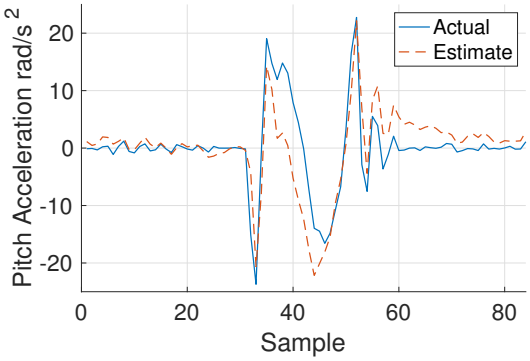
Matlab *nlinfit* and *lsqcurvefit* functions. For the moments of inertia I_{xx}, I_{yy}, and I_{zz} estimations, the Matlab *nlinfit* and *lsqcurvefit* functions do not produce good results. An initial estimate for the inertias is therefore first obtained by using weights and dimensions of different components of the quadrotor and using the parallel axis theorem to calculate the quadrotor inertias. The thrust factor that is obtained is used in equation (2.92) for the roll and pitch acceleration, along with the data from the experiments to estimate the moments of inertia I_{xx} and I_{yy}. The moments of inertia estimates are manually adjusted until a good fit is obtained. Finally, the I_{zz} moment of inertia and the parameter c_Q are estimated using the yaw acceleration equation from equation (2.92). The results from the curve fitting estimations for the Crazyflie quadrotor are shown in Figures 2.9a, 2.9b, 2.9c and 2.9d. The results from the curve fitting estimations for the Crazyflie quadrotor-based aerial manipulator are shown in Figures 2.10a, 2.10b, 2.10c and 2.10d. To obtain the propeller inertia I_r, the propeller is approximated as a rectangular plate with uniform density and the mass moment of inertia is calculated.



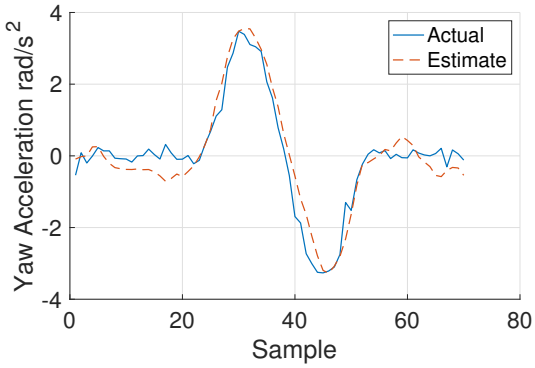
(a) Thrust factor identification data.



(b) I_{xx} identification data.



(c) I_{yy} identification data.

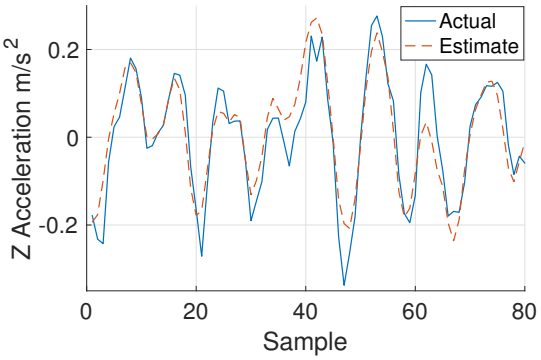


(d) I_{zz} identification data.

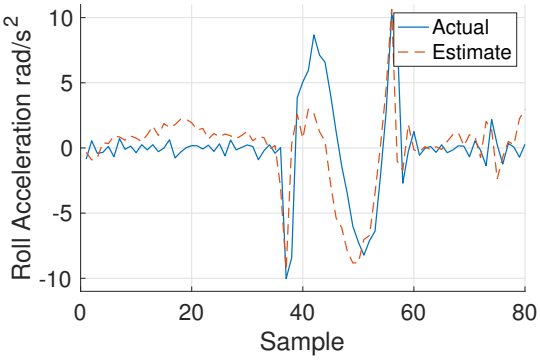
Figure 2.9: Crazyflie quadrotor parameter identification.

2.6.4 System Parameters

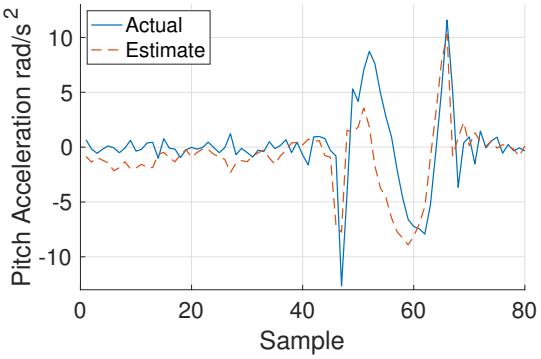
The parameters for the Crazyflie quadrotor-based aerial manipulator, obtained from the system identification experiments and from direct measurements of the quadrotor and the manipulator, are given in Table 2.2, and those for the Crazyflie quadrotor are given in



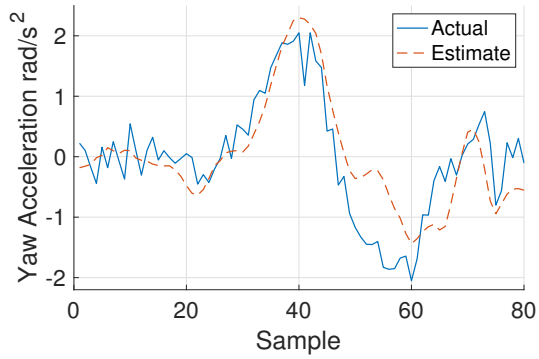
(a) Thrust factor identification data.



(b) I_{xx} identification data.



(c) I_{yy} identification data.



(d) I_{zz} identification data.

Figure 2.10: Crazyflie quadrotor-based aerial manipulator parameter identification.

Table 2.3. The parameters obtained from the system identification experiments for the Crazyflie quadrotor are well aligned with the literature [112–114]. For the moments of inertia, there are some differences from the literature since different Crazyflie decks and components have been added to the quadrotor used for experiments in this thesis. The Crazyflie quadrotor and the Crazyflie quadrotor-based aerial manipulator models and parameters will be used for the development of the SMCs, and for controller simulations and experiments in Chapter 3. In Chapter 4, the Crazyflie model and parameters will be used for SMC chattering simulations as part of the development of the chattering performance indices, and in Chapter 5, they will be used for MOO controller gains-tuning simulations. Both the Crazyflie quadrotor and the Crazyflie quadrotor-based aerial manipulator models and parameters will be used for controller gains tuning and fault-tolerant SMC simulations and experiments in Chapter 7. The parameters for the simulated, larger quadrotor-based aerial manipulator are given in Table 2.4. The larger quadrotor-based aerial manipulator model and parameters will be used in Chapter 6 for controller gains tuning and comparison of different fault-tolerant SMCs for simulated farm coverage and plant sample collection.

Table 2.2: Crazyflie aerial manipulator parameters.

Parameter	**Value**	**Units**
I_{xx}	3.6×10^{-5}	kg m^2
I_{yy}	3.7×10^{-5}	kg m^2
I_{zz}	4.8×10^{-4}	kg m^2
I_r	5.1×10^{-8}	kg m^2
m_Q	0.04173	kg
m_1	1.43×10^{-3}	kg
l_f	0.032	m
l_s	0.032	m
L_{l1}	0.022	m
c_Q	1.55×10^{-12}	Nm/RPM^2
k	1.98×10^{-10}	N/RPM^2
T_c (increasing)	0.070	sec
T_c (decreasing)	0.095	sec

Table 2.3: Crazyflie quadrotor parameters.

Parameter	**Value**	**Units**
I_{xx}	1.9×10^{-5}	kg m^2
I_{yy}	2.0×10^{-5}	kg m^2
I_{zz}	3.5×10^{-4}	kg m^2
I_r	5.1×10^{-8}	kg m^2
m_Q	0.03144	kg
l_f	0.032	m
l_s	0.032	m
c_Q	1.55×10^{-12}	Nm/RPM^2
k	1.96×10^{-10}	N/RPM^2
T_c (increasing)	0.070	sec
T_c (decreasing)	0.095	sec

2.7 Summary

The kinematic and dynamic modelling of a quadrotor and two quadrotor-based aerial manipulators were presented in this chapter. For the aerial manipulators, the quadrotor subsystem dynamics were modelled using the Newton-Euler formulation and the manipulator subsystem dynamics were modelled using the recursive Newton-Euler formulation. The two models were combined by treating the quadrotor as the floating base for the manipulator. Loss of effectiveness parameters for the quadrotor rotors due to rotor faults

Table 2.4: Larger aerial manipulator parameters.

Parameter	**Value**	**Units**
I_{xx}	1.0101×10^{-2}	kg m^2
I_{yy}	1.0107×10^{-2}	kg m^2
I_{zz}	8.2800×10^{-3}	kg m^2
I_r	6.010×10^{-7}	kg m^2
m_Q	0.727	kg
m_1	0.03	kg
m_2	0.06	kg
l_f	0.085	m
l_s	0.068	m
L_{l1}	0.06	m
L_{l2}	0.12	m
c_Q	9.1004×10^{-11}	Nm/RPM^2
k	1.797×10^{-9}	N/RPM^2
T_c (increasing)	0.010	sec
T_c (decreasing)	0.012	sec

were incorporated into the quadrotor model. System identification experiments were conducted to obtain various model parameters of the Crazyflie quadrotor and the Crazyflie quadrotor-based aerial manipulator, such as the moments of inertia, rotor thrust factor, and the rotor time constants. The quadrotor and quadrotor-based aerial manipulator models and parameters obtained will be used for controller development, chattering analysis, controller gains tuning, and FTC simulations and experiments in the remaining chapters of this thesis.

3 Aerial Manipulator Fault-Tolerant Control

Variations of SMC techniques have successfully been used for the FTC of quadrotors. However, the use of these techniques is limited for the FTC of quadrotor-based aerial manipulators. The objective of this chapter is to present and investigate the various types of SMC techniques and how they can be applied in FTC of quadrotor-based aerial manipulators. The key objective is to develop, adapt, or use techniques that will achieve stable control of the aerial manipulator given rotor faults of the quadrotor subsystem. As such, FTC is only investigated for the quadrotor subsystem and not for the manipulator subsystem.

This chapter presents different SMC formulations used for the passive and active FTC of quadrotor-based aerial manipulators. Passive sliding mode FTC makes use of the robust properties of SMCs to mitigate the effects of faulty rotors. Active FTC, on the other hand, uses an estimate of the fault occurrence obtained from an FDD unit and uses the estimate within the controller. Different structures of neural network FDD units are presented. Variations of SMCs investigated for the quadrotor subsystem include the use of discontinuous versus continuous SMCs, classical versus integral sliding surfaces, gain adaptation using different adaptation laws, and first-order versus second-order SMC. An overview of SMC design is given and the formulation of the SMCs is then presented. Chattering, which is an undesirable phenomena that SMCs are prone to, is illustrated, along with some methods and SMCs commonly used to reduce chattering. For the quadrotor-based aerial manipulator, a decoupled control approach is used for the quadrotor and the manipulator subsystems. For the manipulator subsystem control, closed-loop inverse kinematics control is therefore presented.

3.1 Related Work

As noted in the section above, a decoupled approach is used to control the quadrotor subsystem and the manipulator subsystem of the aerial manipulator. Fault-tolerant SMCs that have been used for quadrotors are investigated. The use of closed-loop inverse kinematics control for manipulators is also investigated.

3.1.1 Sliding mode fault-tolerant control of quadrotors

SMC techniques have been investigated for the control of quadrotor-based aerial manipulators [45] and for the FTC of quadrotors and multi-rotors, however, they are yet to be

investigated for the FTC of quadrotor-based aerial manipulators. Classical sliding mode control (CSMC) has been used by Sharifi et al., [64], Mohammadi and Ramezan [66], Barghandan et al., [74] for quadrotor FTC and by Zeghlache et al., [73] for an octorotor. Integral sliding mode control (ISMC) can provide better tracking performance than CSMC and has been used by Merheb et al., [115], Li et al., [63], Chen et al., [75] for quadrotor FTC.

Second-order SMCs such as STSMC are formulated to the chattering phenomena that SMCs are prone to. Jayakrishnan [116], Derafa et al., [117], and Bouchoucha et al., [118] each developed a STSMC for attitude stabilisation of a quadrotor. STSMC is, however, yet to be applied to the FTC of quadrotors or quadrotor-based aerial manipulators.

Adaptive sliding mode control (ASMC) can also help to reduce chattering and help to deal with system uncertainties by adapting the controller gains. Several studies have also attempted the use of adaptive super-twisting sliding mode control (ASTSMC) for the control of quadrotors. Rajappa et al., [119], Akbar and Uchiyama, [120], and Castañeda and Gordillo, [121] all used ASTSMCs to control a quadrotor. The adaptive law was based on that developed by Shtessel et al., [122]. Roy et al., [123] proposed an alternative gain adaptation law, which is, however yet to be applied to the FTC of quadrotors or quadrotor-based aerial manipulators.

The different SMC formulations all have various advantages which need to be investigated with respect to both the passive and active FTC of aerial manipulators. Classical and integral SMCs, adaptive SMCs, first-order SMCs and STSMC will therefore be presented in this chapter for the FTC of quadrotor-based aerial manipulators.

Active FTC makes use of a FDD unit, the results of which are used to adjust the control inputs, based on the detected fault. Neural networks, which are an artificial intelligence method, have the ability to approximate complex functions and to deal with parameter uncertainties [90]. Neural networks have been used for the FDD of satellites [124], industrial processes [125, 126], aircraft [127] and gas turbine engines [128, 129], however they have not yet found wide use for the rotor loss of effectiveness FDD of quadrotors or quadrotor-based aerial manipulators. Wang et al., [69] used recurrent neural networks to estimate the actuator loss of effectiveness of a quadrotor. Locally recurrent neural networks are a type of network structure that is able to store and make use of historical data which makes them useful for FDD [90]. A single neural network to detect faults for all of the rotors would be more complex and would likely struggle to detect faults in the case where more than one rotor experiences a fault [69], hence Wang et al., [69] make use of a parallel bank of four locally recurrent neural networks (one for each actuator), each with two hidden layers of size ten (neurons) for their FDD scheme. The neural networks make use of system control inputs and the quadrotor trajectory tracking errors as inputs and output the rotor fault magnitude. The neural networks are trained in Matlab using the Bayesian regularisation technique. In other neural network-based FDD schemes, the neural network is used to approximate the system model and the outputs of this are compared to the actual outputs of the system [90, 125, 128, 129] and a residual is computed. The residual is then used to determine whether or not a fault has occurred.

In this chapter, three different neural network FDD structures for the active FTC of aerial manipulators are therefore investigated. The first structure makes use of two neural networks, one neural network for system identification and a second neural network that makes use of the system identification outputs and the actual system outputs to detect faults. The second structure makes use of one neural network to identify the faults for all four rotors. The third structure makes use of four neural networks, i.e, one neural network is used for the fault detection of a single rotor.

3.1.2 Closed-loop inverse kinematics control for manipulators

Closed-loop inverse kinematics has been widely used for the control of robotic manipulators with a fixed base [110,130–133]. The method involves the use of the inverse kinematics of the manipulator to obtain desired joint angles that allow the manipulator to track a specified Cartesian trajectory. Closed-loop inverse kinematics has also been used for systems with a moving base, for instance for humanoid robot motion control [134–137]. The method has also found use for the control of the manipulator subsystem of aerial manipulators, in which case the manipulator has a floating base. Sánchez et al., [138] used closed-loop inverse kinematics with added integral actions for the manipulator control of a seven-DoF aerial manipulator. They found that, for simulations of the system, adding the integral action resulted in smoother joint trajectories and better trajectory tracking for the manipulator. However, for the physical experiments, they found that the results with the added integral action were not as effective as expected. Due to its wide use for the control of robotic manipulators and its applicability to manipulators with a floating base, closed-loop inverse kinematics control is used for the control of the manipulator subsystem of the aerial manipulator.

3.2 Fault-Tolerant Control Structures

When designing the aerial manipulator controller, the quadrotor subsystem and manipulator subsystem controllers are decoupled and the controllers for the two subsystems are developed independently. The motion of the manipulator is treated as a disturbance when designing the quadrotor controller. The quadrotor makes use of SMCs and the manipulator makes use of closed-loop inverse kinematics control. As was presented in Chapter 2, the quadrotor is treated as a floating base for the manipulator. The overall control structures for the aerial manipulator with passive and active FTC structures for the quadrotor subsystem are presented, along with the different neural network FDD structures.

3.2.1 Passive fault-tolerant control

In order for the aerial manipulator to perform remote sensing, covering the entire farm while performing plant sample collection, position and yaw angle trajectory tracking con-

trol of the quadrotor-based aerial manipulator system is required, i.e., tracking of the X, Y, and Z positions and the yaw angle. The quadrotor is an underactuated system with the X and Y positions not controlled directly. A cascading controller is therefore used for position tracking of the quadrotor subsystem. The desired X, Y, Z and ψ trajectories are specified and the desired ϕ and θ needed to achieve the desired X and Y trajectories are then computed using equations (3.6) and (3.7). The controller computes the desired inputs U_1, U_2, U_3, and U_4. These are converted by the 'Control Mixer' unit into the desired PWM signals for each rotor of the quadrotor using equations (2.47) and (2.86). The overall trajectory tracking control scheme is shown in Figure 3.1. This structure is used for the passive FTC control of the quadrotor subsystem of the aerial manipulator since passive FTC relies on the robustness of the SMC.

The manipulator subsystem of the aerial manipulator makes use of closed-loop inverse kinematics control. The desired end-effector trajectory is sent to the manipulator controller and the desired manipulator joint angles to achieve the end-effector trajectory tracking are computed. Depending on whether angular position feedback can be obtained for the manipulator's joint angles, these are incorporated into the manipulator's controller. Since the quadrotor subsystem serves as the floating base of the manipulator subsystem, the quadrotor's state information is also used by the manipulator controller.

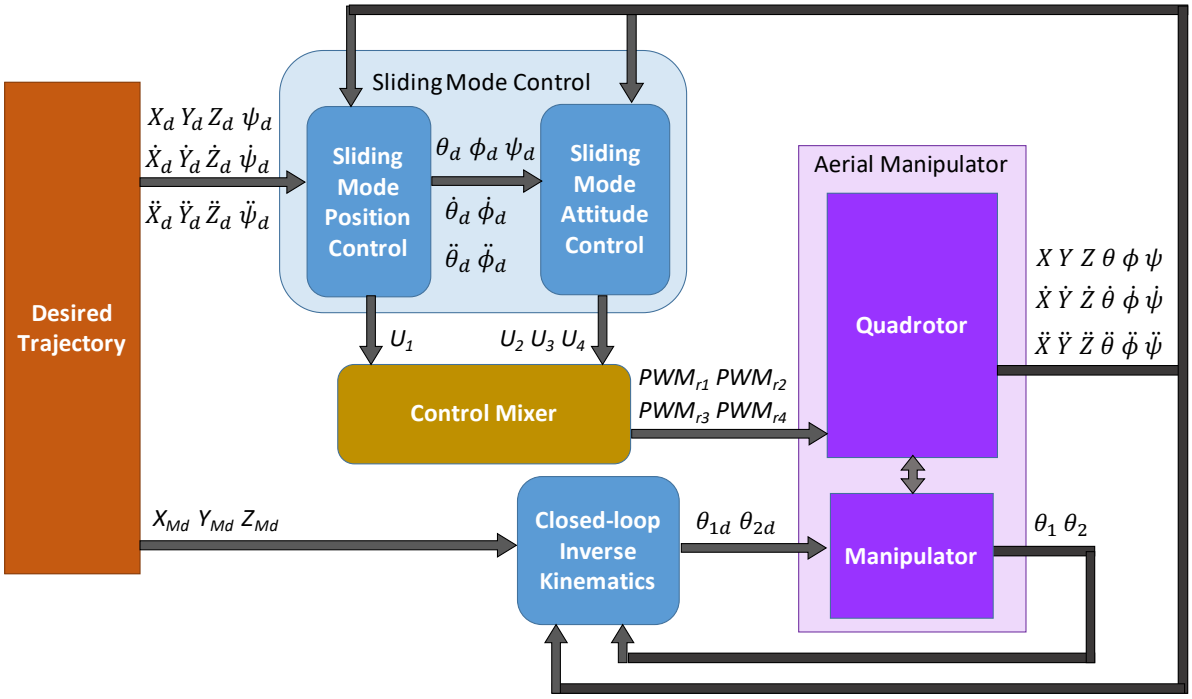


Figure 3.1: Passive fault-tolerant control scheme

3.2.2 Active fault-tolerant control

The active FTC has a similar structure to the passive FTC, with the addition of a neural network-based FDD unit as shown in Figure 3.2. The FDD unit takes in the quadrotor

desired control inputs, quadrotor state, and the manipulator state, and uses these to determine the fault occurrence and fault magnitude for the quadrotor rotors. This information is then used to adjust the quadrotor control inputs which are then sent to the 'Control Mixer' unit to compute the desired PWM signals for each rotor of the quadrotor.

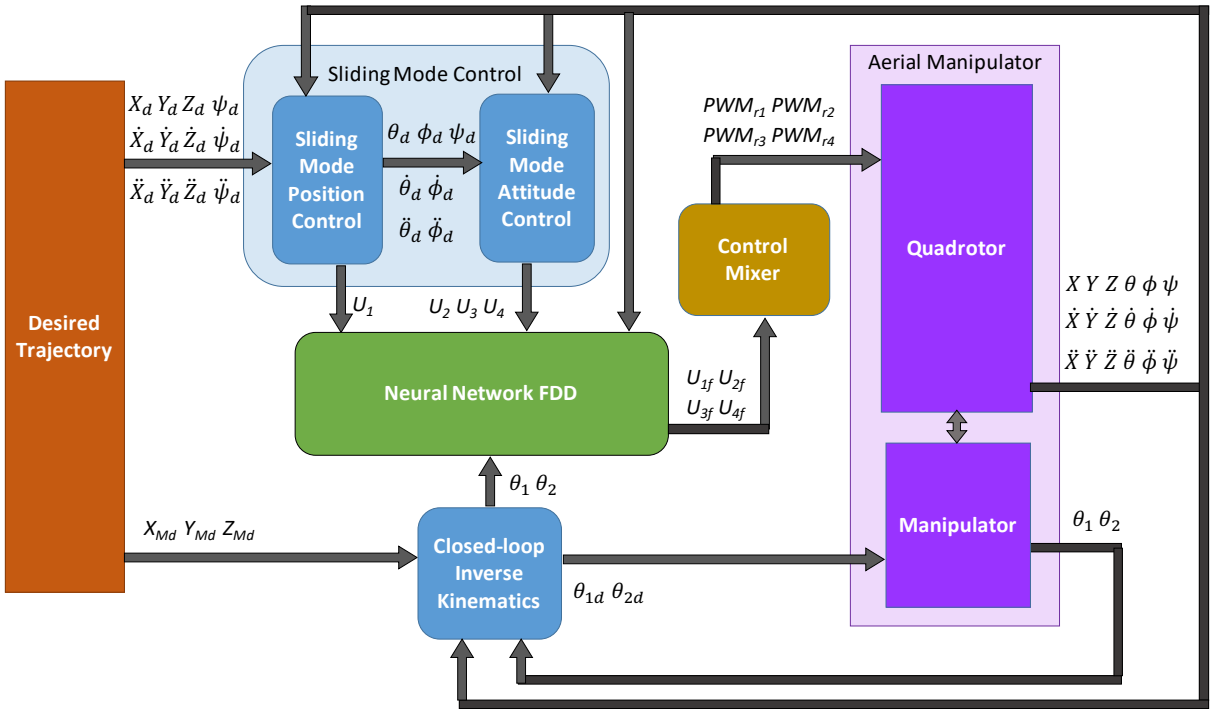


Figure 3.2: Active fault-tolerant control scheme

3.2.3 Fault detection and diagnosis using neural networks

Three different configurations of neural network-based FDD units are considered for the FDD unit as found from the literature. These include a two neural network structure, one neural network to estimate the system accelerations, and the second neural network which uses the actual versus the estimated accelerations to estimate the rotor remaining effectiveness as shown in Figure 3.3. The second FDD structure, shown in Figure 3.4, is one that performs the entire FDD for all four rotors using a single neural network. The third FDD structure has a bank of four neural networks, one for the FDD of each rotor as illustrated in Figure 3.5.

The inputs to the networks are the roll, pitch, and yaw DoF angles, angular velocities and angular accelerations, the Z DoF acceleration, the control inputs from the previous time step of the SMC and the arm state. The final outputs of the neural networks are the estimated remaining effectiveness of each of the quadrotor rotors. An exponential moving average filter is used to smooth the neural network outputs. For simplicity, the neural network blocks shown in Figures 3.3 to 3.5 also include the output filtering. The filtered rotor remaining effectiveness information is then used by the FDD mixer of the FDD unit to calculate new control inputs for the system using equation (3.1). The desired rotor

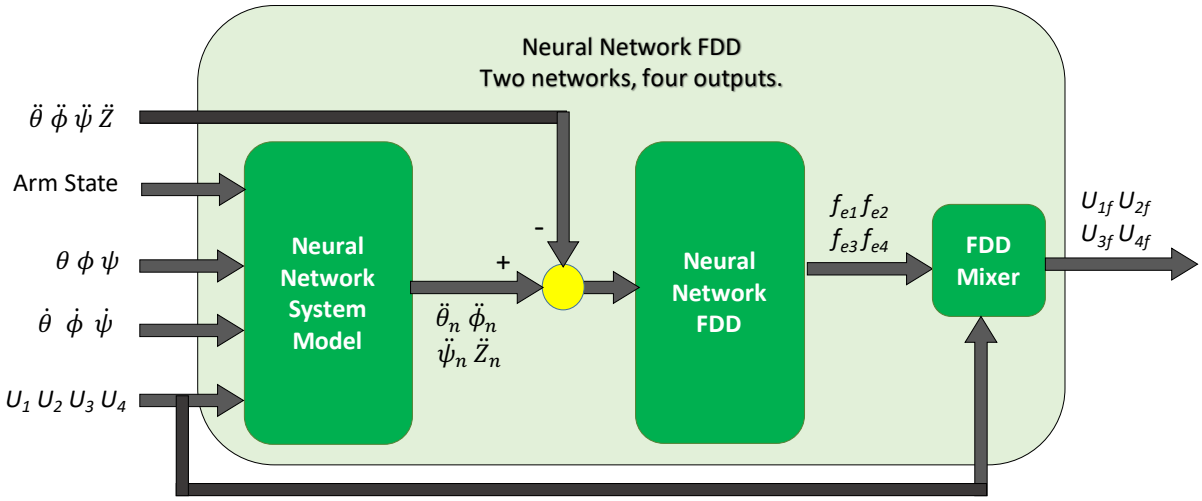


Figure 3.3: Two neural network fault detection and diagnosis unit.

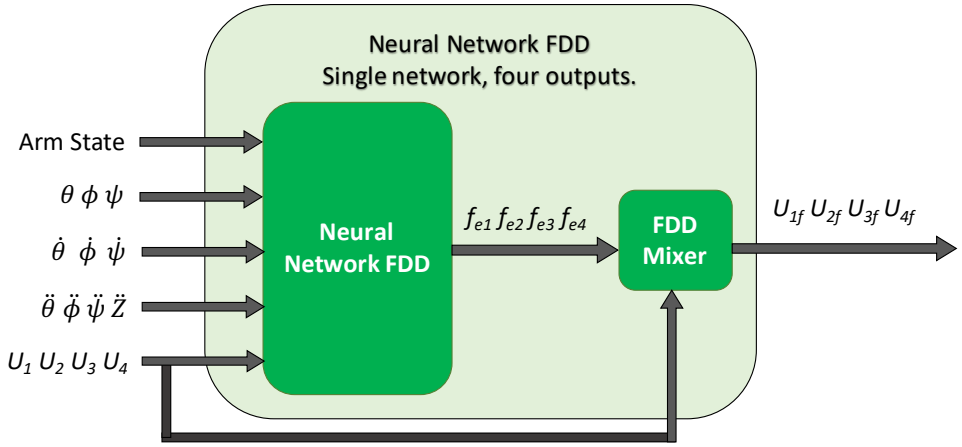


Figure 3.4: One neural network fault detection and diagnosis unit.

speeds used in equation (3.1) are computed using equation (2.47) and the control inputs obtained from the SMC. By multiplying the desired rotor speeds by $\frac{1}{f_{ei}}$, this increases the desired rotor speed and in turn the control inputs to compensate for the loss of effectiveness of the rotor.

$$\begin{bmatrix} U_{1f} \\ U_{2f} \\ U_{3f} \\ U_{4f} \end{bmatrix} = \begin{bmatrix} k & k & k & k \\ kl_f & -kl_f & -kl_f & kl_f \\ -kl_s & -kl_s & kl_s & kl_s \\ -c_Q & c_Q & -c_Q & c_Q \end{bmatrix} \begin{bmatrix} \left(\frac{\omega_{r1}}{f_{e1}}\right)^2 \\ \left(\frac{\omega_{r2}}{f_{e2}}\right)^2 \\ \left(\frac{\omega_{r3}}{f_{e3}}\right)^2 \\ \left(\frac{\omega_{r4}}{f_{e4}}\right)^2 \end{bmatrix} \tag{3.1}$$

where $0 < f_{ei} \leq 1$ is the estimated remaining effectiveness of rotor i.

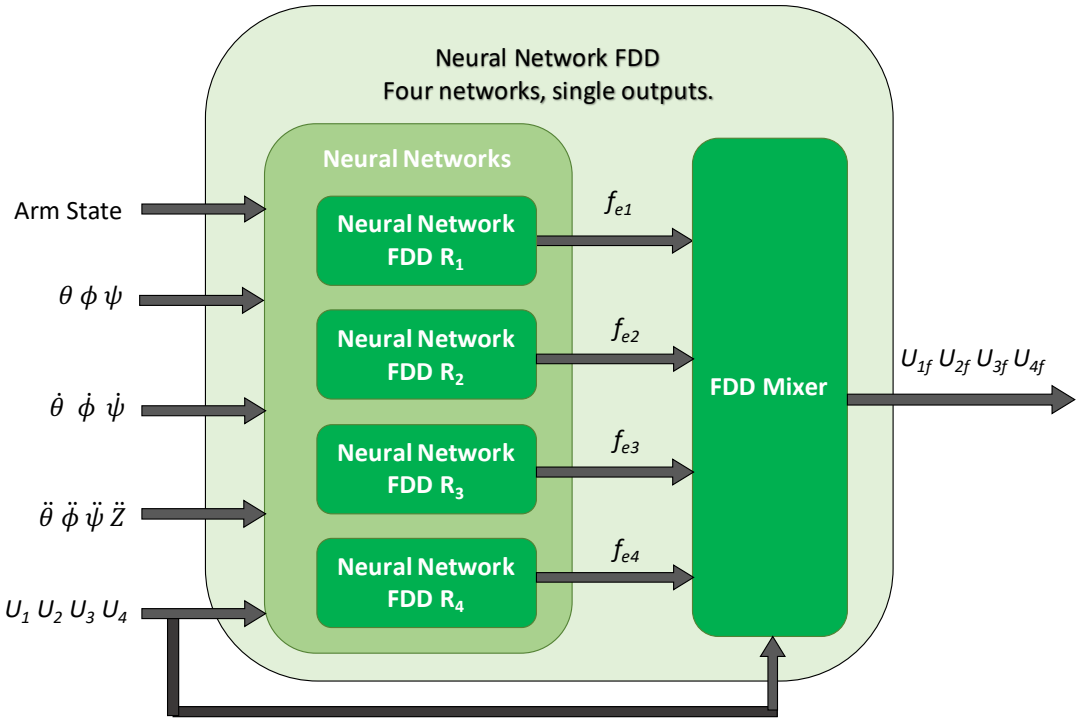


Figure 3.5: Four neural network fault detection and diagnosis unit.

Two types of neural networks are considered, the feed-forward, and cascade-forward neural networks. Examples of the neural networks, in abbreviated notation [139] are shown in Figure 3.6 and Figure 3.7. Each network includes the inputs, one or more hidden layers and the output layer. Each layer has a size representing the number of neurons in that layer. The neurons are connected to their inputs by the weighting matrix **W** and each neuron has a bias b, forming the bias vector **b** [139]. Each layer includes a summer for the weights and bias, and a transfer function for each neuron [139]. In the examples, a hyperbolic tangent sigmoid function is used for the hidden layers and a linear function for the output layer. For the feed-forward network, the input to the first hidden layer is the input vector to the neural network, thereafter, the output of a layer forms the input to the next layer. For the cascade-forward network, each layer has a connection to the neural network input vector as well as the outputs of all preceding layers [140].

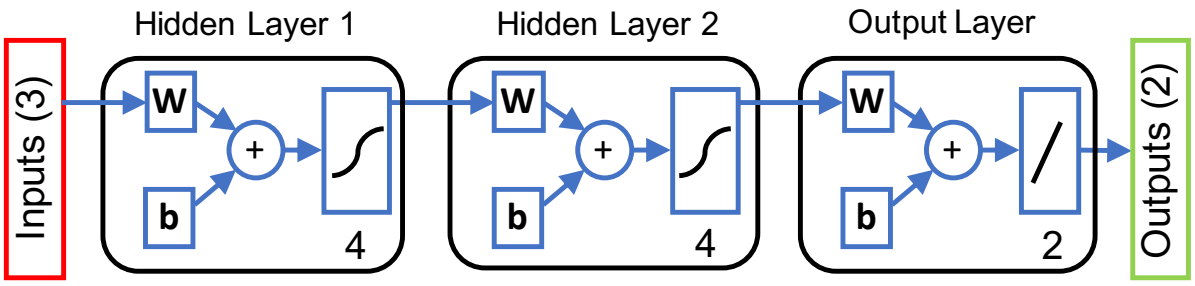


Figure 3.6: Feed-forward neural network with three inputs, two outputs, and two hidden layers of size four each.

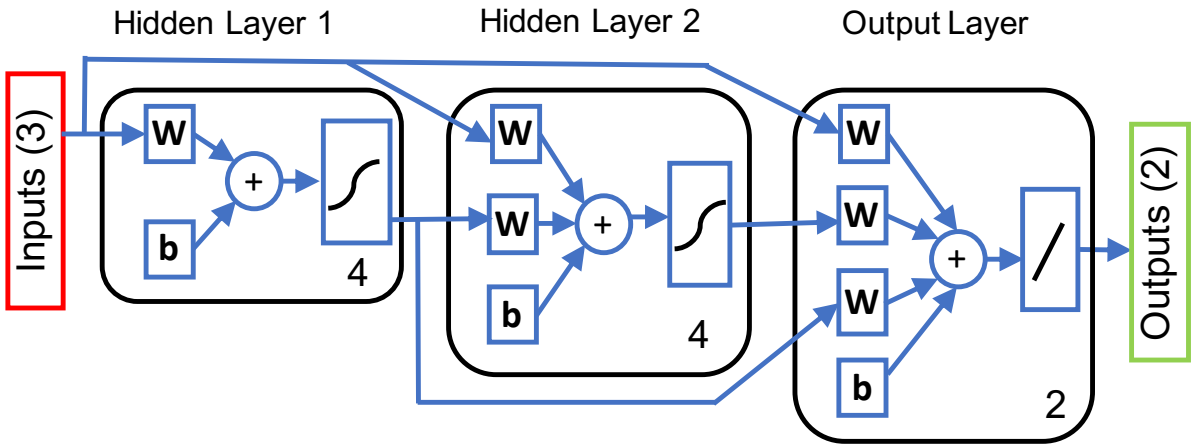


Figure 3.7: Cascade-forward neural network with three inputs, two outputs, and two hidden layers of size four each.

3.3 Quadrotor Trajectory Tracking Control

The SMCs presented in this chapter for the control of the quadrotor subsystem make use of the system model of the aerial manipulator presented in Chapter 2. To develop the SMCs for the quadrotor, the simplified system model given by equations (2.90) and (2.91) is used. For trajectory tracking, since the quadrotor is an underactuated system, the equations required to obtain the desired roll and pitch angles from the desired X and Y are presented. The trajectory generation method used is also presented.

3.3.1 Trajectory generation

A path gives a purely geometric description of the desired motion whereas a trajectory is a path with specified timing which could also be in terms of velocities and accelerations of each point in the path [141]. The trajectory can be generated by solving the following optimisation problem [141, 142]:

$$\mathbf{x}^{\star}(t) = \underset{\mathbf{x}(t)}{\operatorname{argmin}} \int_0^T (\mathbf{x}^n)^2 \, \mathrm{d}t \tag{3.2}$$

where n is the order of the state derivatives that are to be solved for. The higher the order of the state derivatives, the smoother the trajectory. In this case to solve for a minimum snap (the fourth derivative of position) trajectory, $n = 4$ is used.

The solution for the optimisation problem with $n = 4$ is a 7th-order polynomial [141, 142]:

$$\mathbf{x}(t) = \mathbf{c}_{p0} + \mathbf{c}_{p1}t + \mathbf{c}_{p2}t^2 + \mathbf{c}_{p3}t^3 + \mathbf{c}_{p4}t^4 + \mathbf{c}_{p5}t^5 + \mathbf{c}_{p6}t^6 + \mathbf{c}_{p7}t^7 \tag{3.3}$$

The boundary conditions for solving the polynomial are the initial and final times, positions, velocities, accelerations and jerk (the third derivative of position). For a simple point-to-point motion, the initial and final velocities, accelerations and jerk are all zero.

3.3.2 Trajectory tracking

A quadrotor is an under-actuated system, that is, it has only four control inputs for its six-DoFs. The X and Y positions are not controlled directly, but are controlled through the roll and pitch angles. For X and Y trajectory tracking, virtual control inputs U_x and U_y can be defined which are then used to determine the desired roll and pitch angles to track the specified trajectory.

$$U_x = (-\cos\psi \sin\theta \cos\phi + \sin\psi \sin\phi)\,\frac{U_1}{m} \tag{3.4}$$

$$U_y = (-\sin\psi \sin\theta \cos\phi - \cos\psi \sin\phi)\,\frac{U_1}{m} \tag{3.5}$$

For X and Y trajectory tracking, the desired roll ϕ_d and pitch θ_d angles are then obtained from the following equations:

$$\phi_d = \arcsin\left(\frac{m}{U_1}\left(\sin\psi_d U_x - \cos\psi_d U_y\right)\right) \tag{3.6}$$

$$\theta_d = \arcsin\left(\frac{-m}{U_1 \cos\phi_d}\left(\cos\psi_d U_x + \sin\psi_d U_y\right)\right) \tag{3.7}$$

The control inputs for the quadrotor can therefore be specified as:

$$\boldsymbol{U} = \begin{bmatrix} U_x & U_y & U_1 & U_2 & U_3 & U_4 \end{bmatrix}^T \tag{3.8}$$

Since the angular velocities of the quadrotor rotors are typically not measured directly, when designing the controller, the expressions containing the rotor angular velocities are treated as disturbances, along with the manipulator motion and any other disturbances experienced by the system. The aerial manipulator system can then be written as, (with the disturbances given by $\boldsymbol{\delta}$):

$$\ddot{\mathbf{x}} = f(\mathbf{x}) + b(\mathbf{x})\,\boldsymbol{U} + \boldsymbol{\delta} \tag{3.9}$$

where:

$$\mathbf{x} = \begin{bmatrix} X & Y & Z & \phi & \theta & \psi \end{bmatrix}^T \tag{3.10}$$

$$f(\mathbf{x}) = \begin{bmatrix} a_x & a_y & a_1 & a_2 & a_3 & a_4 \end{bmatrix}^T \tag{3.11}$$

$$b(\mathbf{x}) = \text{diag}\left(\begin{bmatrix} b_x & b_y & b_1 & b_2 & b_3 & b_4 \end{bmatrix}\right) \tag{3.12}$$

with:

$$\begin{aligned} a_x &= 0 & b_x &= 1 \\ a_y &= 0 & b_y &= 1 \\ a_1 &= -g & b_1 &= \frac{(\cos\theta\cos\phi)}{m} \\ a_2 &= \frac{I_{yy} - I_{zz}}{I_{xx}}\dot{\theta}\dot{\psi} & b_2 &= \frac{1}{I_{xx}} \\ a_3 &= \frac{I_{zz} - I_{xx}}{I_{yy}}\dot{\phi}\dot{\psi} & b_3 &= \frac{1}{I_{yy}} \\ a_4 &= \frac{I_{xx} - I_{yy}}{I_{zz}}\dot{\theta}\dot{\phi} & b_4 &= \frac{1}{I_{zz}} \end{aligned} \tag{3.13}$$

3.4 Sliding Mode Control Design

SMC is a type of robust nonlinear controller that uses a switching control law to force the controlled system's state trajectories to reach a specified sliding surface (or switching) surface s [143]. This is typically referred to as the reaching phase. Once the sliding surface is reached, the controller then maintains the system's state trajectories on this surface and the system states 'slide' along the line $s = 0$ [143]. If a disturbance forces the system's trajectory away from the sliding surface, the controller forces the system states back onto the sliding surface [144].

There are two important steps in the design of SMCs. The first step is to design the sliding surface so that the system has desired behaviour when it is constrained to the sliding surface [143]. The next step is constructing a control law necessary to drive the system's state trajectory to the sliding surface and to maintain it there [143]. The SMC control law U is:

$$U = U_{eq} + U_D \tag{3.14}$$

The SMC control law consists of two parts: a corrective control part (also referred to as the discontinuous control or the reaching control law) U_D and an equivalent control

part U_{eq} [117, 145]. The reaching control law allows the sliding surface to be reached by compensating for variations from the sliding surface of the state trajectories [144]. The reaching control part determines the robustness of the controller. The equivalent control ensures that the time derivative of the surface is maintained at zero, in order for the state trajectories to stay on the sliding surface [143, 144].

3.4.1 Classical first-order sliding surface

The classical sliding surface s and its derivative $\dot{s}$, are defined as [143]:

$$\begin{aligned} \boldsymbol{s} &= \boldsymbol{c}_1\boldsymbol{e} + \dot{\boldsymbol{e}} \\ \dot{\boldsymbol{s}} &= \boldsymbol{c}_1\dot{\boldsymbol{e}} + \ddot{\mathbf{x}} - \ddot{\mathbf{x}}_d \end{aligned} \tag{3.15}$$

where:

$$\boldsymbol{c}_1 = \text{diag}\left(\begin{bmatrix} c_{1X} & c_{1Y} & c_{1Z} & c_{1\phi} & c_{1\theta} & c_{1\psi} \end{bmatrix}\right) \tag{3.16}$$

and the error vector $\boldsymbol{e}$ is defined as:

$$\boldsymbol{e} = \mathbf{x} - \mathbf{x}_d \tag{3.17}$$

where $\mathbf{x}_d$ is the desired state value.

3.4.2 First-order corrective control

The corrective control law for the first-order CSMC is selected as [143]:

$$\boldsymbol{U}_D = b^{-1}\left(\mathbf{x}\right)\left(-\boldsymbol{k}_1\boldsymbol{\sigma} - \boldsymbol{k}_2\boldsymbol{s}\right) \tag{3.18}$$

where the discontinuous signum function is used:

$$\boldsymbol{\sigma} = \boldsymbol{\sigma}_D = \text{sgn}\left(\boldsymbol{s}\right) \tag{3.19}$$

where sgn is the signum function:

$$\text{sgn}\left(s\right) = \begin{cases} -1 & \texttt{if} \quad s < 0; \\ 0 & \texttt{if} \quad s = 0; \\ 1 & \texttt{if} \quad s > 0. \end{cases} \tag{3.20}$$

and:

$$\boldsymbol{k}_i = \text{diag}\left(\begin{bmatrix} k_{iX} & k_{iY} & k_{iZ} & k_{i\phi} & k_{i\theta} & k_{i\psi} \end{bmatrix}\right) ; i = 1, 2 \tag{3.21}$$

The proportional rate term $\boldsymbol{k}_2\boldsymbol{s}$, forces the system states to approach the sliding surface faster when $\boldsymbol{s}$ is large [143]. A higher value of $\boldsymbol{k}_2$ leads to faster reaching speed. When $\boldsymbol{s}$ is close to zero, the term $\boldsymbol{k}_1\boldsymbol{\sigma}_D$ then allows for fast convergence speed [143]. As the term contains a discontinuous component $\boldsymbol{\sigma}_D$, a low value of $\boldsymbol{k}_1$ should be used to reduce chattering.

3.4.3 Pseudo/quasi sliding mode control

Due to the discontinuous signum function used in the reaching law $\boldsymbol{\sigma}_D$, first-order SMCs typically experience the chattering phenomena. Replacing the signum function with a continuous approximation, such as the hyperbolic tangent $\boldsymbol{\sigma}_H$ or the saturation function $\boldsymbol{\sigma}_B$, can aid in reducing chattering. The use of a continuous function in the corrective control law instead of the discontinuous signum function, results in a controller that approaches the sliding surface and remains in the vicinity of the sliding surface. Such a controller is sometimes termed a pseudo or quasi SMC. Since the discontinuous term is responsible for the robustness of the SMC, the decreased chattering of the pseudo SMCs is obtained at the expense of robustness of the controller.

Hyperbolic Tangent

Using the hyperbolic tangent function in the reaching control law, then [143]:

$$\boldsymbol{\sigma} = \boldsymbol{\sigma}_H = \tanh\left(\frac{\boldsymbol{s}}{\lambda}\right) \tag{3.22}$$

where λ is a user-defined number that determines the steepness of the hyperbolic tangent function.

Saturation Function

Alternatively, the saturation function can be used in the reaching control law [143]:

$$\boldsymbol{\sigma} = \boldsymbol{\sigma}_B = \text{sat}\left(\boldsymbol{s}\right) \tag{3.23}$$

where

$$\text{sat}\left(s\right) = \begin{cases} \dfrac{s}{\lambda} & \text{if} \quad |s| < \lambda \\ \text{sgn}\left(s\right) & \text{if} \quad |s| \geq \lambda \end{cases} \tag{3.24}$$

where λ is a user-defined number that determines the boundary layer. Figure 3.8 shows the difference between the continuous functions and the discontinuous function in the reaching law.

3.4.4 Classical first-order sliding mode control law

To obtain the equivalent control $\boldsymbol{U}_{eq}$, the derivative of the sliding surface is driven to zero, and the expression for $\ddot{\mathbf{x}}$ from equation (3.9), excluding disturbances, is substituted into equation (3.15). Thus the equivalent control becomes:

$$\boldsymbol{U}_{eq} = b^{-1}\left(\mathbf{x}\right)\left(-\boldsymbol{c}_1\dot{\boldsymbol{e}} + \ddot{\mathbf{x}}_d - f\left(\mathbf{x}\right)\right) \tag{3.25}$$

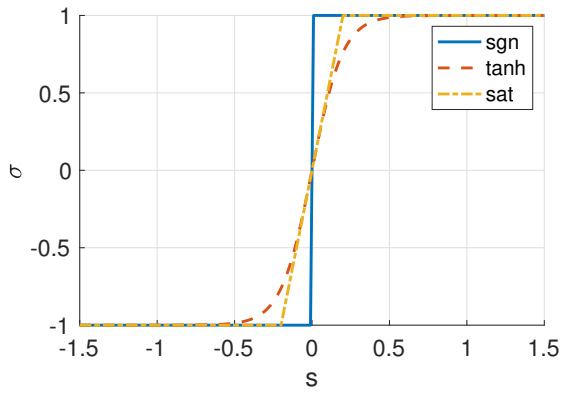


Figure 3.8: Discontinuous versus continuous functions.

The control inputs for the first-order CSMC are therefore calculated to be:

$$\boldsymbol{U} = b^{-1}(\mathbf{x})\left(-\boldsymbol{k}_1\boldsymbol{\sigma} - \boldsymbol{k}_2\boldsymbol{s} - \boldsymbol{c}_1\dot{\boldsymbol{e}} + \ddot{\mathbf{x}}_d - f(\mathbf{x})\right) \tag{3.26}$$

Controller Stability

Stability analysis of the CSMC can be found at [146, 147]. Selecting the positive-definite Lyapunov function as:

$$\begin{aligned} \mathbf{V} &= \frac{1}{2}\boldsymbol{s}^T\boldsymbol{s} \\ \dot{\mathbf{V}} &= \boldsymbol{s}^T\dot{\boldsymbol{s}} \end{aligned} \tag{3.27}$$

From equation (3.15) and equation (3.9), neglecting the rotor angular velocity expressions, the derivative becomes [146, 147]:

$$\begin{aligned} \dot{\mathbf{V}} &= \boldsymbol{s}^T\left(\boldsymbol{c}_1\dot{\boldsymbol{e}} + \ddot{\mathbf{x}} - \ddot{\mathbf{x}}_d\right) \\ \dot{\mathbf{V}} &= \boldsymbol{s}^T\left(\boldsymbol{c}_1\dot{\boldsymbol{e}} + f(\mathbf{x}) + b(\mathbf{x})\boldsymbol{U} + \boldsymbol{\delta} - \ddot{\mathbf{x}}_d\right) \end{aligned} \tag{3.28}$$

Substituting the control law (3.26) then [146, 147]:

$$\begin{aligned} \dot{\mathbf{V}} &= \boldsymbol{s}^T\left(-\boldsymbol{k}_1\mathrm{sgn}(\boldsymbol{s}) - \boldsymbol{k}_2\boldsymbol{s} + \boldsymbol{\delta}\right) \\ \dot{\mathbf{V}} &= -\boldsymbol{k}_1|\boldsymbol{s}| - \boldsymbol{k}_2\boldsymbol{s}^T\boldsymbol{s} + \boldsymbol{s}^T\boldsymbol{\delta} \end{aligned} \tag{3.29}$$

Since $\boldsymbol{k}_1$ and $\boldsymbol{k}_2$ are positive, and an upper bound is selected as $|\boldsymbol{\delta}| < \boldsymbol{k}_1$, $\dot{\mathbf{V}} \leq 0$ is satisfied.

3.4.5 Chattering simulations and experiments

To illustrate the chattering phenomena of CSMCs, a step input to the pitch DoF using discontinuous CSMCs is simulated in Matlab for the Crazyflie quadrotor using the system

dynamic equations presented in Chapter 2. The height, roll and yaw DoFs are controlled to remain at a constant height and at zero angles respectively. The gains used for the controllers are manually chosen and are shown in Tables 3.1 and 3.2. The k_1 gains for the height, roll and yaw are set to zero to prevent chattering in those DoFs which would influence the chattering experienced by the pitch DoF. CSMC_0 has a value of zero for the pitch gain k_1 and CSMC_1 has a k_1 value of 6 to determine the impact of the gain on the error, chattering and control input of the CSMC. A small constant-disturbance torque is added to the pitch DoF to compare the controllers' ability to reject disturbances. Sensor noise can also have an impact on the performance of CSMCs. $\text{CSMC}_{1\text{ (no noise)}}$ is simulated without a noise signal to determine the impact of noise on the controller. CSMC_1 and CSMC_0 have uniformly distributed, random sensor noise signals, bounded within specific intervals for the position, velocity and acceleration of each DoF. The Matlab '*rand* function is used to create the noise signal arrays.

Table 3.1: Quadrotor pitch control simulation gains.

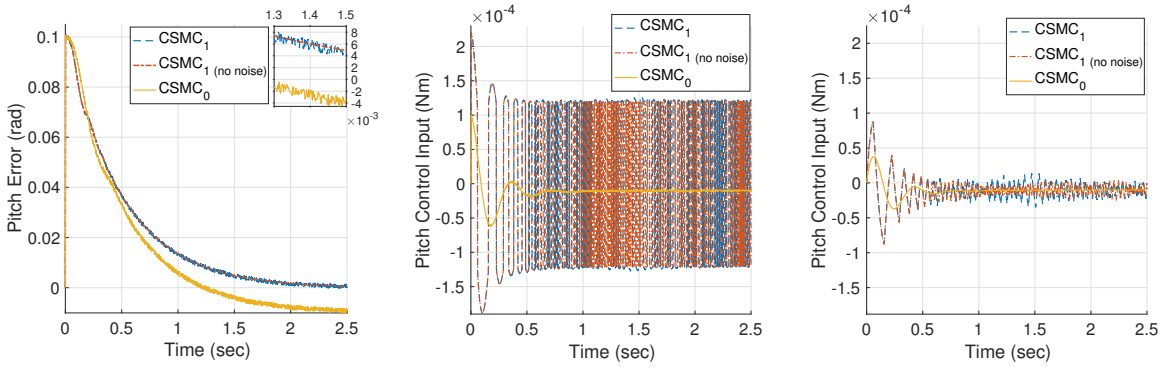
Pitch Gain	CSMC_0	CSMC_1
k_1	0	6
k_2	25	25
c_1	2	2

Table 3.2: Quadrotor altitude and attitude control gains.

Gain	Z	Roll	Yaw
k_1	0	0	0
k_2	8	30	15
c_1	2	2	2

Figure 3.9a shows the pitch angle errors, Figure 3.9b shows the desired control inputs and Figure 3.9c shows the actual control inputs, taking into account the rotor dynamics. Figure 3.10 shows the phase diagrams and the values of the sliding surface, error and change in error. As can be seen from the figures, for $\text{CSMC}_{1\text{ (no noise)}}$ and CSMC_1, high-frequency oscillations are observed in the control input when the sliding mode is reached, that is when the value of s reaches zero. As can be seen in Figure 3.9c, CSMC_1 has instances of higher control input than $\text{CSMC}_{1\text{ (no noise)}}$, due to the impact of sensor noise. CSMC_0 has a high pitch angle error and also fails to reach the sliding surface, showing the importance of the k_1 in reducing the pitch angle error, in particular when disturbances are experienced or due to system model inaccuracies. Since CSMC_0 has a k_1 value of zero, no chattering is observed in its control input.

The simulations performed successfully show the chattering phenomena of SMCs. However, unmodelled system dynamics of some physical systems can sometimes mean that chattering is actually not experienced by the physical system. A set of experiments is therefore performed to verify that chattering is experienced by the physical Crazyflie



(a) Error. (b) Desired control input. (c) Actual control input.

Figure 3.9: Pitch-angle step change response.

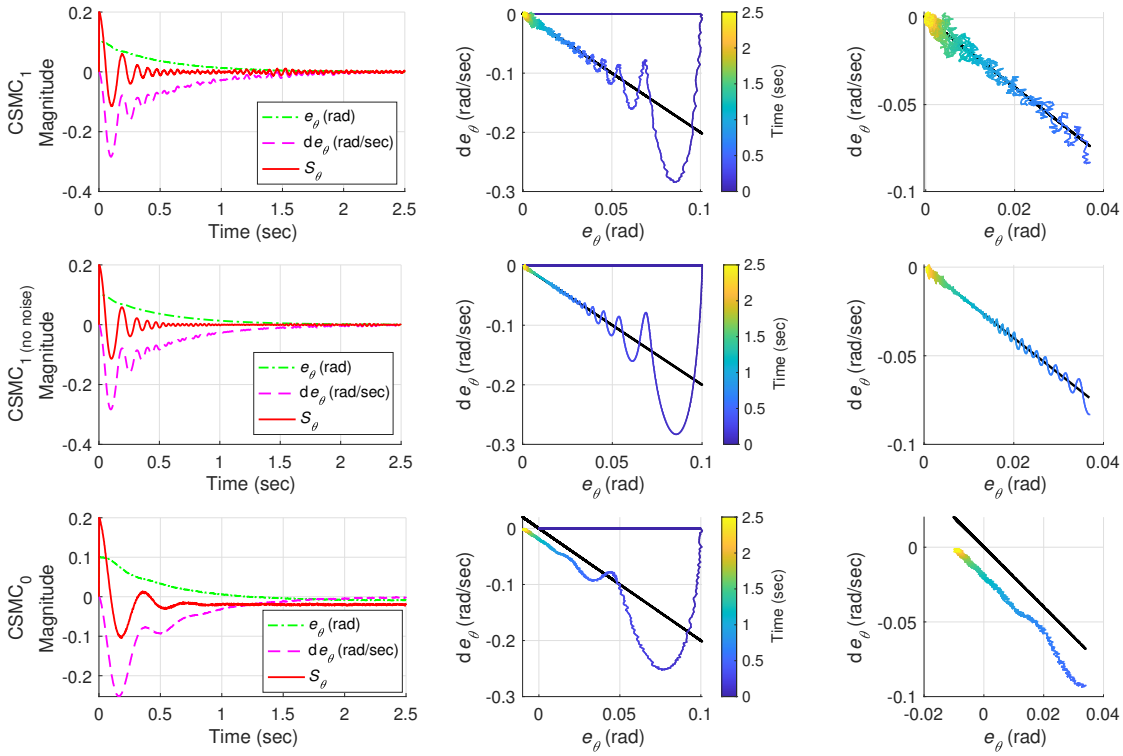


Figure 3.10: Phase diagram for a pitch-angle step change.

quadrotor when the discontinuous CSMC is used. The experiments are also used to verify that the nonlinear component of the CSMCs, given by the k_1 gain, aids in reducing the trajectory tracking error. The pitch DoF control for the Crazyflie quadrotor is tested under altitude and attitude control. The Crazyflie firmware is modified to allow for altitude and attitude control of the quadrotor and to allow for output of the actual control input given the estimation of the rotor dynamics.

First, the quadrotor has to rise to a specified height while maintaining roll, pitch, and yaw angles of zero radians. The pitch DoF motion is then performed after a few seconds. At the end of the pitch motion the quadrotor must maintain the desired height, and roll, pitch, and yaw angles of zero radians for a few seconds. The quadrotor then lands. The

pitch controller gains for this set of experiments are manually chosen and are shown in Table 3.3. The Z, roll and yaw controller gains used are shown in Table 3.2. For this set of experiments, manually chosen gains are acceptable as the aim is to verify that chattering is experienced for a single DoF of the quadrotor. In Chapter 5, methods of obtaining optimally tuned gains that minimise chattering, control inputs, and trajectory tracking errors for all DoFs of the quadrotor subsystem of the aerial manipulator are presented.

Table 3.3: Quadrotor controller gains - chattering comparison.

Pitch Gain	CSMC_{2c}	CSMC_{3c}
k_1	8	0
k_2	35	35
c_1	2	2

The pitch angle and pitch angle error results for the chattering experiments are shown in Figure 3.11 and the desired control input, actual control input, s, and σ results are shown in Figure 3.12. The portions of the trajectory when the quadrotor is taking off and landing are omitted in order to focus solely on the pitch DoF results. CSMC_{3c}, which has the gain k_1 equal to zero, has the highest pitch angle errors, showing the importance of the k_1 gain in reducing the trajectory tracking errors. The chattering of CSMC_{2c} is evident from both the desired control input results and the actual control input results shown in Figure 3.12. It is important to note that although chattering is evident for CSMC_{2c} in Figure 3.12, due to the the data logging frequency used for the quadrotor, very high-frequency chattering which may be present is not necessarily captured in the figure. No chattering is observed for CSMC_{3c}. The high-switching action of the σ for CSMC_{2c} is seen in Figure 3.12. Although CSMC_{3c} also has high-switching action for σ, since the gain k_1 is equal to zero for this controller, this high-switching action is not transferred to the control input. For this set of experiments, chattering is analysed by visual comparison of the error and control input results. In Chapter 4, chattering performance indices for measuring and comparing chattering experienced by different SMCs are developed. Comparison of chattering for the different SMCs, given optimally tuned controller gains is presented in Chapter 6.

3.4.6 Integral first-order sliding mode control

ISMC makes use of the integral of the error in the design of the sliding surface with the aim of improving robustness and accuracy. The sliding surface is designed, with the added integral term, as [148]:

$$\begin{aligned} \boldsymbol{s} &= \boldsymbol{c}_1\boldsymbol{e} + \dot{\boldsymbol{e}} + \boldsymbol{c}_2 \int \boldsymbol{e}\ \mathrm{d}\tau \\ \dot{\boldsymbol{s}} &= \boldsymbol{c}_1\dot{\boldsymbol{e}} + \ddot{\mathbf{x}} - \ddot{\mathbf{x}}_d + \boldsymbol{c}_2\boldsymbol{e} \end{aligned} \tag{3.30}$$

Using the same method as described in Chapter 3 for the CSMC, the control inputs for

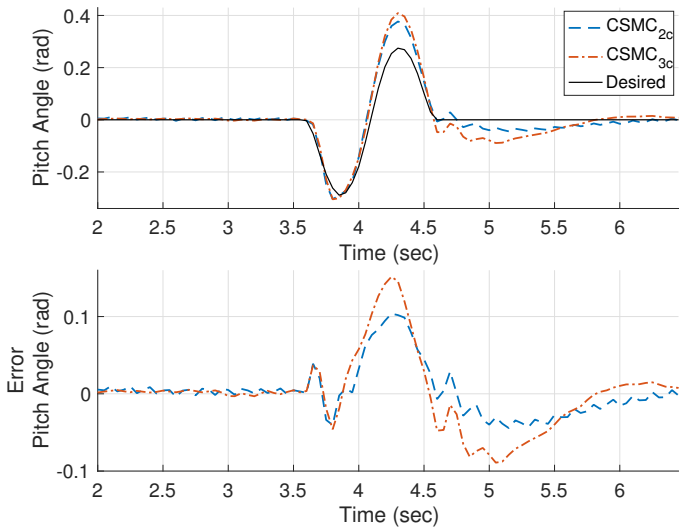


Figure 3.11: Pitch-angle chattering experiments - error.

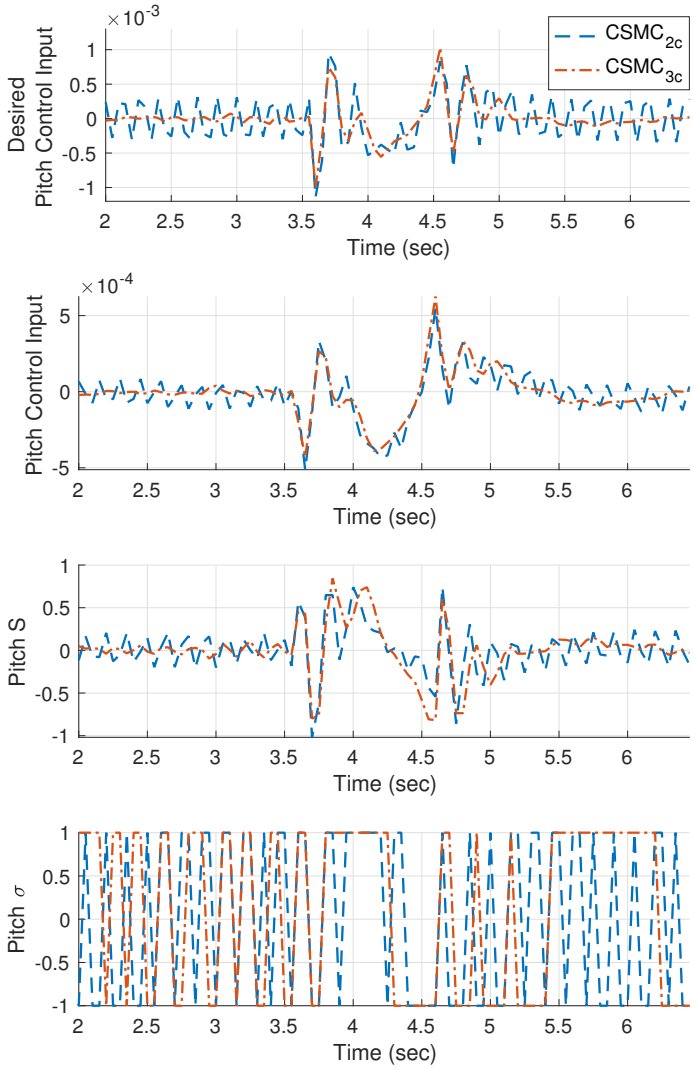


Figure 3.12: Pitch-angle chattering experiments - control input.

the ISMC are calculated to be:

$$\boldsymbol{U} = b^{-1}(\mathbf{x})\left(-\boldsymbol{k}_1\boldsymbol{\sigma} - \boldsymbol{k}_2\boldsymbol{s} - \boldsymbol{c}_1\dot{\boldsymbol{e}} - \boldsymbol{c}_2\boldsymbol{e} + \ddot{\mathbf{x}}_d - f(\mathbf{x})\right) \tag{3.31}$$

Controller Stability

Selecting the positive-definite Lyapunov function as:

$$\begin{aligned} V &= \frac{1}{2}s^2 \\ \dot{V} &= s\dot{s} \end{aligned} \tag{3.32}$$

From equation (3.30) and equation (3.9), the derivative becomes:

$$\begin{aligned} \dot{V} &= s\left(c_1\dot{e} + c_2 e + \ddot{\mathrm{x}} - \ddot{\mathrm{x}}_d\right) \\ \dot{V} &= s\left(c_1\dot{e} + c_2 e + f(\mathrm{x}) + b(\mathrm{x})\,U + \delta - \ddot{\mathrm{x}}_d\right) \end{aligned} \tag{3.33}$$

Substituting the control law (3.31) then:

$$\begin{aligned} \dot{V} &= s\left(-k_1 \mathrm{sgn}(s) - k_2 s + \delta\right) \\ \dot{V} &= -k_1|s| - k_2 s^2 + s\delta \end{aligned} \tag{3.34}$$

Since k_1 and k_2 are positive constants, and and an upper bound is selected as $|\delta| < k_1$, the condition $\dot{V} \leq 0$ is satisfied.

3.5 Adaptive Sliding Mode Control

A controller can be adapted either by adjusting the controller gains or by adding an additional adaptive term to the controller. The added adaptive term is typically computed based on estimation of external disturbances, while the adaptive controller gains for SMCs is computed based on the error and sliding surface values. This section deals with adaptive controller gains with the view of reducing chattering. Since the discontinuous term $\boldsymbol{\sigma}_D$ in equation (3.18) results in chattering, adapting the associated gain $\boldsymbol{k}_1$ appropriately, would lead to decreased chattering. Thus, instead of having fixed gain $\boldsymbol{k}_1$, the ASMC has adaptive gain $\boldsymbol{\alpha}$. For the general first-order ASMC, the reaching law becomes:

$$\boldsymbol{U}_D = b^{-1}(\mathbf{x})\left(-\boldsymbol{\alpha}\boldsymbol{\sigma} - \boldsymbol{k}_2\boldsymbol{s}\right) \tag{3.35}$$

Gain adaptation can be applied to both CSMC and ISMC. Without loss of generality, the ISMC is used to illustrate gain adaptation in this section. The adaptive integral sliding mode control (AISMC) becomes:

$$\boldsymbol{U} = b^{-1}(\mathbf{x})\left(-\boldsymbol{\alpha}\boldsymbol{\sigma} - \boldsymbol{k}_2\boldsymbol{s} - \boldsymbol{c}_1\dot{\boldsymbol{e}} - \boldsymbol{c}_2\boldsymbol{e} + \ddot{\mathbf{x}}_d - f(\mathbf{x})\right) \tag{3.36}$$

Different adaptation laws can be used to adapt the $\boldsymbol{\alpha}$ gains. The gain adaptation laws used in this chapter are based on those by Plestan et al., [149] (referred to as the Plestan method) and by Roy et al., [123] (referred to as the Roy method).

3.5.1 Adaptive control gains - Plestan method

The gain adaptation law given by Plestan et al., [149] is:

$$\dot{\alpha} = \begin{cases} \eta_\alpha \ |s| \ \text{sgn}\left(|s| - \mu\right) & \texttt{if} \quad \alpha > \alpha_m \\ \eta_D & \texttt{if} \quad \alpha \leq \alpha_m \end{cases} \tag{3.37}$$

where μ, α_m, η_α, and η_D are positive constants. This adaptation method works by increasing the gains until the sliding mode is reached. The gains are then decreased until the sliding surface starts to move away from equilibrium, in which case the gains start to increase again [149]. A 'detector', μ, is used to detect when the sliding mode is moving away from equilibrium [149]. It should, however be noted that the reaching law selected by Plestan et al., ($\boldsymbol{U}_D = b^{-1}\left(\mathbf{x}\right)\left(-\boldsymbol{k}_1\boldsymbol{\sigma}\right)$) differs from that used in this thesis.

Controller Stability

The controller stability analysis is similar to that presented in [149]. Selecting the following Lyapunov function:

$$V = \frac{1}{2}s^2 + \frac{1}{2\eta_\alpha}\left(\alpha - \alpha^*\right)^2 \tag{3.38}$$

where α^* is the maximum possible value of α.

The time derivative is:

$$\dot{V} = s\dot{s} + \frac{1}{\eta_\alpha}\left(\alpha - \alpha^*\right)\dot{\alpha} \tag{3.39}$$

Using equations (3.30), and (3.9) and substituting control law (3.36) :

$$\dot{V} = s\left(-\alpha\text{sgn}\left(s\right) - k_2 s + \delta\right) + \left(\alpha - \alpha^*\right)\dot{\alpha} \tag{3.40}$$

With the adaptive gain (3.76):

$$\dot{V} = s\left(-\alpha\text{sgn}\left(s\right) - k_2 s + \delta\right) + \left(\alpha - \alpha^*\right)|s|\,\text{sgn}\left(|s| - \mu\right) \tag{3.41}$$

For the case $|s| > \mu$:

$$\dot{V} = -k_2 s^2 + s\delta - \alpha^*|s| \tag{3.42}$$

Since $|\delta| \leq \alpha^*$

$$\dot{V} \leq -k_2 s^2 \tag{3.43}$$

Since k_2 is a positive constant, the condition $\dot{V} \leq 0$ is satisfied.

3.5.2 Adaptive control gains - Roy method

The Plestan gain adaptation method makes the assumption of a priori boundedness of the disturbance term [123]. As shown by Roy et al., [123], if the a priori boundedness assumption is violated and the initial gain α_i is not high enough, the Plestan gain adaptation method may lead to instability of the control system [123]. Roy et al., [123] therefore propose an adaptive method that, unlike the Plestan method, does not require a priori bounded uncertainty. The gain adaptation law given by Roy et al., [123] is:

$$\alpha = r_0 + r_1 |e| \tag{3.44}$$

which is adapted by:

$$\dot{r}_0 = |s| - \rho_0 r_0 \tag{3.45}$$

$$\dot{r}_1 = |s| |e| - \rho_1 r_1 \tag{3.46}$$

where ρ_0, and ρ_1 are positive constants.

Controller Stability

The controller stability analysis is as presented by Roy et al., [123] and is repeated here for completeness. Selecting the following Lyapunov function [123]:

$$V = \frac{1}{2}s^2 + \frac{1}{2}\left(r_0 - r_0^*\right)^2 + \frac{1}{2}\left(r_1 - r_1^*\right)^2 \tag{3.47}$$

where r_0^* and r_1^* are the maximum possible values of r_0 and r_1

The time derivative is:

$$\dot{V} = s\dot{s} + \left(r_0 - r_0^*\right)\dot{r}_0 + \left(r_1 - r_1^*\right)\dot{r}_1 \tag{3.48}$$

Using equations (3.30), and (3.9) and substituting control law (3.36) :

$$\dot{V} = s\left(-\alpha \mathrm{sgn}\left(s\right) - k_2 s + \delta\right) + \left(r_0 - r_0^*\right)\dot{r}_0 + \left(r_1 - r_1^*\right)\dot{r}_1 \tag{3.49}$$

considering the state-dependent upper bound:

$$|\delta| \leq r_0^* + r_1^* |e| \tag{3.50}$$

and the gain (3.44), then:

$$\dot{V} \leq -k_2 s^2 - \left(r_0 - r_0^*\right)\left(|s| - \dot{r}_0\right) - \left(r_1 - r_1^*\right)\left(|s||e| - \dot{r}_1\right) \tag{3.51}$$

Substituting equations (3.45) and (3.46):

$$\dot{V} \leq -k_2 s^2 + r_0 r_0^* \rho_0 - r_0^2 \rho_0 + r_1 r_1^* \rho_1 - r_1^2 \rho_1 \tag{3.52}$$

Using the fact that:

$$\begin{aligned} r_i r_i^* - r_i^2 &= -\left(\frac{r_i}{\sqrt{2}} - \frac{r_i^*}{\sqrt{2}}\right)^2 - \frac{r_i^2}{\sqrt{2}} + \frac{r_i^{*2}}{\sqrt{2}} \\ &\leq -\left(\frac{r_i}{\sqrt{2}} - \frac{r_i^*}{\sqrt{2}}\right)^2 + \frac{r_i^{*2}}{\sqrt{2}} \end{aligned} \tag{3.53}$$

and the definition of the Lyapunov function, then:

$$\dot{V} \leq -\overline{\varrho} V + \frac{1}{2}\rho_0 r_0^{*2} + \frac{1}{2}\rho_1 r_1^{*2} \tag{3.54}$$

where, by design:

$$\overline{\varrho} \triangleq 2\min_i\{k_2,\ \frac{\rho_i}{2}\} > 0 \tag{3.55}$$

Defining a scalar $0 < \overline{\kappa} < \overline{\varrho}$, then:

$$\dot{V} \leq -\overline{\kappa} V - (\overline{\varrho} - \overline{\kappa})\, V + \overline{\delta} \tag{3.56}$$

where:

$$\overline{\delta} \triangleq \frac{1}{2}\left(\rho_0 r_0^{*2} + \rho_1 r_1^{*2}\right) \tag{3.57}$$

Defining a scalar:

$$\overline{B} \triangleq \frac{\overline{\delta}}{\overline{\varrho} - \overline{\kappa}} \tag{3.58}$$

then $\dot{V} \leq -\overline{\kappa} V(t)$ when $V(t) \geq \overline{B}$, so that:

$$V \leq \max\{V(0),\ \overline{B}\} \quad \forall t \geq 0 \tag{3.59}$$

and the Lyapunov function enters the ball defined by $\overline{B}$, in finite time.

3.6 Super-Twisting Sliding Mode Control

STSMC is a type of second-order SMC, which is designed for systems with a relative degree of one with respect to the sliding surface. It is designed to try and decrease chattering. The standard reaching law $\boldsymbol{U}_D$ for the STSMC is [116–118]:

$$\begin{aligned} \boldsymbol{U}_D &= b^{-1}(\mathbf{x})\left(-\boldsymbol{k}_1\sqrt{|\boldsymbol{s}|}\ \boldsymbol{\sigma} + \boldsymbol{U}_B\right) \\ \dot{\boldsymbol{U}}_B &= -\boldsymbol{k}_2\ \boldsymbol{\sigma} \end{aligned} \tag{3.60}$$

3.6.1 Classical super-twisting sliding mode control

Using the classical sliding surface from equation (3.15), the control law for the classical super-twisting sliding mode control (CSTSMC) is:

$$\boldsymbol{U} = b^{-1}\left(\mathbf{x}\right)\left(-\boldsymbol{k}_1\sqrt{|\boldsymbol{s}|}\boldsymbol{\sigma} - \boldsymbol{k}_2\int_0^t \boldsymbol{\sigma}\ \mathrm{d}\tau - \boldsymbol{c}_1\dot{\boldsymbol{e}} + \ddot{\mathbf{x}}_d - f\left(\mathbf{x}\right)\right) \tag{3.61}$$

Controller Stability

The stability of the controller is as presented by Moreno et al., [150] and González-Hernández et al., [151] using the following Lyapunov function [150] [151]:

$$\mathrm{V} = 2k_2|\mu_1| + \frac{1}{2}\mu_2^2 + \frac{1}{2}\left(k_1|\mu_1|^{\frac{1}{2}}\mathrm{sgn}\left(\mu_1\right) - \mu_2\right)^2 \tag{3.62}$$

where:

$$\begin{aligned}\mu_1 &= s\\ \mu_2 &= -k_2\textstyle\int_0^t \mathrm{sgn}\left(s\right)\ \mathrm{d}\tau\end{aligned} \tag{3.63}$$

The Lyapunov function written in quadratic form [150] [151] is:

$$\mathrm{V} = \zeta^T P \zeta \tag{3.64}$$

where:

$$\zeta^T = \begin{bmatrix}\ |\mu_1|^{\frac{1}{2}}\,\mathrm{sgn}\left(\mu_1\right) & \mu_2\ \end{bmatrix} \tag{3.65}$$

and:

$$P = \frac{1}{2}\begin{bmatrix} 4k_2 + k_1^2 & -k_1 \\ -k_1 & 2 \end{bmatrix} \tag{3.66}$$

V is continuous, however it is not differentiable at $\mu_1 = 0$. V is positive definite and radially bounded if $k_2 > 0$, thus:

$$\lambda_{min}(P)\|\zeta\|_2^2 \leq \mathrm{V} \leq \lambda_{max}(P)\|\zeta\|_2^2 \tag{3.67}$$

where $\lambda_{min}(P)$ and $\lambda_{max}(P)$ represent the minimum and maximum eigenvalue of matrix P respectively. $\|\zeta\|_2^2 = |\mu_1| + \mu_2^2$ represents the Euclidean norm of ζ.

The time derivative of the Lyapunov function is [152]:

$$\dot{\mathrm{V}} = -\frac{1}{|\mu_1|^{\frac{1}{2}}}\zeta^T Q \zeta + \delta_2 q_1^T \zeta \tag{3.68}$$

where

$$q_1^T = \begin{bmatrix}\ k_1 & 2\ \end{bmatrix} \tag{3.69}$$

Assuming the perturbation term of the system is bounded by:

$$\delta_2 \leq l_1 \tag{3.70}$$

where the constant $l_1 > 0$, then:

$$\dot{\mathrm{V}} = -\frac{1}{|\mu_1|^{\frac{1}{2}}}\zeta^T \tilde{Q}\zeta \tag{3.71}$$

where

$$\tilde{Q} = \frac{k_1}{2}\begin{bmatrix} 2k_2 + k_1^2 - 2l_1 & k_1 \\ -\left(k_1 + \dfrac{2l_1}{k_1}\right) & 1 \end{bmatrix} \tag{3.72}$$

$\dot{\mathrm{V}}$ is negative definite if $\tilde{Q} > 0$, which is true if:

$$\begin{aligned} &k_1 > 0 \\ &k_2 > \frac{l_1\|q_1\|}{\dfrac{1}{2}k_1^2} \end{aligned} \tag{3.73}$$

3.7 Adaptive Super-Twisting Sliding Mode Control

Adapting the STSMC, the reaching law for the ASTSMC becomes [122]:

$$\begin{aligned} \boldsymbol{U}_D &= b^{-1}(\mathbf{x})\left(-\boldsymbol{\alpha}\sqrt{|\boldsymbol{s}|}\ \boldsymbol{\sigma} + \boldsymbol{U}_B\right) \\ \dot{\boldsymbol{U}}_B &= -\boldsymbol{\beta}\ \boldsymbol{\sigma} \end{aligned} \tag{3.74}$$

The adaptive classical super-twisting sliding mode control (ACSTSMC) law is then:

$$\boldsymbol{U} = b^{-1}(\mathbf{x})\left(-\boldsymbol{\alpha}\sqrt{|\boldsymbol{s}|}\boldsymbol{\sigma} - \boldsymbol{\beta}\int_0^t \boldsymbol{\sigma}\ \mathrm{d}\tau - \boldsymbol{c}_1\dot{\boldsymbol{e}} + \ddot{\mathbf{x}}_d - f(\mathbf{x})\right) \tag{3.75}$$

3.7.1 Adaptive control gains - Shtessel method

The gain adaptation law given by Shtessel et al., [122] is:

$$\dot{\alpha} = \begin{cases} \eta_\alpha\ \mathrm{sgn}\left(|s| - \mu\right) & \texttt{if} \quad \alpha > \alpha_m \\ \eta_D & \texttt{if} \quad \alpha \leq \alpha_m \end{cases} \tag{3.76}$$

$$\beta = \eta_\beta\ \alpha \tag{3.77}$$

where μ, α_m, η_α, η_β, and η_D are positive constants. This adaptation method works by increasing the gains until the sliding mode is reached. The gains are then decreased until the sliding surface starts to move away from equilibrium, in which case the gains start to increase again [122]. A 'detector', μ, is used to detect when the sliding mode is moving away from equilibrium [122].

Controller Stability

The controller stability analysis is similar to that presented in [122] and [153]. Selecting the following Lyapunov function:

$$V = V_0 + \frac{1}{2\gamma_1}(\alpha - \alpha^*)^2 + \frac{1}{2\gamma_2}(\beta - \beta^*)^2 \tag{3.78}$$

where α^* and β^* are the maximum possible values of α and β, and [153]:

$$\mathrm{V}_0 = \zeta^T P \zeta \tag{3.79}$$

where:

$$\zeta^T = \begin{bmatrix} |\mu_1|^{\frac{1}{2}} \operatorname{sgn}(\mu_1) & \mu_2 \end{bmatrix} \tag{3.80}$$

where:

$$\begin{aligned} \mu_1 &= s \\ \mu_2 &= -k_2 \textstyle\int_0^t \operatorname{sgn}(s)\ \mathrm{d}\tau \end{aligned} \tag{3.81}$$

and:

$$P = \frac{1}{2}\begin{bmatrix} \lambda + 4\varepsilon^2 & -2\varepsilon \\ -2\varepsilon & 1 \end{bmatrix} \tag{3.82}$$

where $\lambda > 0$ and $\varepsilon > 0$ are any real numbers, in which case P is positive definite.

The time derivative of the Lyapunov function V_0 is [153]:

$$\dot{\mathrm{V}}_0 \leq -\frac{1}{|\mu_1|^{\frac{1}{2}}}\zeta^T Q \zeta \tag{3.83}$$

where

$$\tilde{Q} = \begin{bmatrix} 2\lambda\alpha - 2\rho(\lambda + 4\varepsilon^2) & \lambda + 4\varepsilon^2 - 2\varepsilon\rho \\ \lambda + 4\varepsilon^2 - 2\varepsilon\rho & 4\varepsilon \end{bmatrix} \tag{3.84}$$

where ρ is a bounded function such that $|\delta| \leq \rho|s|^{\frac{1}{2}}$

$\dot{\mathrm{V}}_0$ is negative definite if $\tilde{Q} > 0$, which is true if [122]:

$$\alpha > \frac{\rho(\lambda + 4\varepsilon^2)}{\lambda} + \frac{(\lambda + 4\varepsilon^2 - 2\varepsilon\rho)^2}{8\lambda\varepsilon} \tag{3.85}$$

Given the adaptive gains, then [122]:

$$\dot{\mathrm{V}}_0 \leq -r\mathrm{V}_0^{\frac{1}{2}} \tag{3.86}$$

where:

$$r = \frac{\varepsilon\lambda_{\min}^{\frac{1}{2}}(P)}{\lambda_{\max}(P)} \tag{3.87}$$

The derivative of the Lyapunov function V is then [122] :

$$\dot{\mathrm{V}} \leq -r\mathrm{V}_0^{\frac{1}{2}} + \frac{1}{\gamma_1}\varepsilon_\alpha\dot{\alpha} + \frac{1}{\gamma_2}\varepsilon_\beta\dot{\beta} \tag{3.88}$$

with $\varepsilon_\alpha = \alpha - \alpha^*$ and $\varepsilon_\beta = \beta - \beta^*$

$$\dot{\mathrm{V}} \leq -r\mathrm{V}_0^{\frac{1}{2}} + \frac{1}{\gamma_1}\varepsilon_\alpha\dot{\alpha} + \frac{1}{\gamma_2}\varepsilon_\beta\dot{\beta} - \frac{\tilde{\omega}_1}{\sqrt{2\gamma_1}}|\varepsilon_\alpha| + \frac{\tilde{\omega}_1}{\sqrt{2\gamma_1}}|\varepsilon_\alpha| - \frac{\tilde{\omega}_2}{\sqrt{2\gamma_2}}|\varepsilon_\beta| + \frac{\tilde{\omega}_2}{\sqrt{2\gamma_2}}|\varepsilon_\beta| \tag{3.89}$$

where $\tilde{\omega}_1$ and $\tilde{\omega}_2$ are arbitrary positive constants. Using the inequality:

$$\left(x^2 + y^2 + z^2\right)^{\frac{1}{2}} \leq |x| + |y| + |z| \tag{3.90}$$

and from the Lyapunov function, then:

$$-r\mathrm{V}_0^{\frac{1}{2}} - \frac{\tilde{\omega}_1}{\sqrt{2\gamma_1}}|\varepsilon_\alpha| - \frac{\tilde{\omega}_2}{\sqrt{2\gamma_2}}|\varepsilon_\beta| \leq -\eta_0\sqrt{\mathrm{V}} \tag{3.91}$$

with $\eta_0 = \min\left(r, \tilde{\omega}_1, \tilde{\omega}_2\right)$. Then:

$$\dot{\mathrm{V}} \leq -\eta_0\sqrt{\mathrm{V}} - |\varepsilon_\alpha|\left(\frac{1}{\gamma_1}\dot{\alpha} - \frac{\tilde{\omega}_1}{\sqrt{2\gamma_1}}\right) - |\varepsilon_\beta|\left(\frac{1}{\gamma_2}\dot{\beta} - \frac{\tilde{\omega}_2}{\sqrt{2\gamma_2}}\right) \tag{3.92}$$

Given the adaptive gains (3.76), then convergence of the system is guaranteed.

3.8 Manipulator Closed-Loop Inverse Kinematics Control

When designing the aerial manipulator controller in this thesis, the quadrotor and manipulator subsystem controllers are decoupled. The manipulator subsystem controller is therefore designed independently of the quadrotor subsystem controller. Closed-loop inverse kinematics is used for the manipulator subsystem control, given a desired manipulator end-effector trajectory. To obtain the manipulator's joint angles given a desired end-effector position and orientation, inverse kinematics modelling is used. Differential kinematics inversion, which presents a linear mapping between the manipulator joint variables and the manipulator link variables is used to develop the closed-loop inverse kinematics controller. From equation (2.54) [110, 133]:

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{-1}\boldsymbol{v} \tag{3.93}$$

Given the joint angles for the initial or starting posture of the manipulator, the manipulator joint angles at each time instant for a desired manipulator trajectory can then be calculated numerically using [110, 133]:

$$\boldsymbol{q}\left(t_{k+1}\right) = \boldsymbol{q}\left(t_k\right) + \dot{\boldsymbol{q}}\left(t_k\right)\Delta t \tag{3.94}$$

where Δt is the time interval.

3.8.1 Weighted damped least-squares pseudo-inverse Jacobian

For redundant manipulators, there are many solutions to the inverse kinematics. The right pseudo-inverse of $\boldsymbol{J}$ can be used to select a solution to $\dot{\boldsymbol{q}}$ that minimises the cost function of the joint velocities. For the inversion of $\boldsymbol{J}$ to be computed, the Jacobian relating the manipulator link linear velocities to the manipulator joint angular velocities has to be of full rank, that is, the manipulator is not near a singular configuration. To overcome the problem of inverting the differential kinematics when the manipulator is near a singularity, the damped least-squares (DLS) pseudo-inverse can be used [110,133]. Weighing the DLS pseudo-inverse Jacobian by the positive definite matrix $\boldsymbol{W}$ allows constraints such as joint limit constraints and self collision avoidance constraints to be added to the solution. The weighted DLS pseudo-inverse Jacobian is [135–137]:

$$\boldsymbol{J}^{\star} = \boldsymbol{W}^{-1}\boldsymbol{J}^{T}(\boldsymbol{J}\boldsymbol{W}^{-1}\boldsymbol{J}^{T} + \lambda^{2}\mathbf{1})^{-1} \tag{3.95}$$

where:
λ is a damping constant,
$\mathbf{1}$ is an identity matrix, and

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{\star}\boldsymbol{v} \tag{3.96}$$

3.8.2 Joint limit weighing matrix

The positive definite matrix $\boldsymbol{W}$ can be made up of a combination of a number of different weighing matrices. In this thesis, only the joint limit weighing matrix $\boldsymbol{W}_{JL}$, which keeps the manipulator from exceeding its joint motion limits, is used, thus:

$$\boldsymbol{W} = \boldsymbol{W}_{JL} \tag{3.97}$$

The joint limit avoidance is based on a weighted least-norm solution, with $\boldsymbol{H}(\boldsymbol{q})$ chosen as the candidate joint limit function [136]. $\boldsymbol{H}(\boldsymbol{q})$ produces higher values when the joints are near to their limit and it tends to infinity at the joint limits [136]. The joint limit function $\boldsymbol{H}(\boldsymbol{q})$ is used to compute the joint limit weighing matrix. The gradient of $\boldsymbol{H}$, $\nabla\boldsymbol{H}$, is a vector representing the joint limit gradient function and it points in the direction of the fastest rate of increase of $\boldsymbol{H}$ [135–137]:

$$\nabla\boldsymbol{H} = \frac{\partial\boldsymbol{H}(\boldsymbol{q})}{\partial\boldsymbol{q}} = \left[\frac{\partial H}{\partial q_1}, \cdots, \frac{\partial H}{\partial q_n}\right] \tag{3.98}$$

where

$$\boldsymbol{H}(\boldsymbol{q}) = \frac{1}{4}\sum_{i=1}^{n}\frac{(q_{iM} - q_{im})^2}{(q_{iM} - q_i)(q_i - q_{im})} \tag{3.99}$$

and

$$\frac{\partial H}{\partial q_i} = \frac{1}{4}\frac{(q_{iM} - q_{im})^2(2q_i - q_{iM} - q_{im})}{(q_{iM} - q_i)^2(q_i - q_{im})^2} \tag{3.100}$$

q_{iM} is the maximum joint limit, q_{im} is the minimum joint limit.

The joint is at the middle of its motion range when the gradient is equal to zero, and as the gradient goes to infinity, the joint is at either limit of its motion [135–137]. The joint limit weighing matrix $\boldsymbol{W}_{JL}$ is then an $n \times n$ diagonal matrix with diagonal elements W_{JLi} [135–137]:

$$\boldsymbol{W}_{JL} = \begin{bmatrix} W_{JL1} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & W_{JLn} \end{bmatrix} \tag{3.101}$$

where

$$W_{JLi} = \begin{cases} 1 + \left|\dfrac{\partial H}{\partial q_i}\right| & \text{if} \quad \nabla\left|\dfrac{\partial H}{\partial q_i}\right| \geq 0 \\ 1 & \text{if} \quad \nabla\left|\dfrac{\partial H}{\partial q_i}\right| < 0 \end{cases} \tag{3.102}$$

$\nabla\left|\frac{\partial H}{\partial q_i}\right|$ is the gradient function's change in magnitude. A positive value means that the joint is moving towards its joint limit. When this occurs, the weighing factor goes towards infinity causing the joint motion to slow down. A negative value means the joint is moving away from its motion limit [135–137].

3.9 Summary

In this chapter, quadrotor-based aerial manipulator passive and active FTC structures were presented. Three different FDD structures, making use of neural networks to detect and estimate the magnitude of quadrotor rotor faults were proposed for the active FTCs. Minimum snap trajectory generation for the aerial manipulator motion was presented. Decoupled control for the quadrotor and manipulator subsystems was presented. For position trajectory tracking control of the quadrotor subsystem, virtual control inputs were introduced which are used to determine the desired roll and pitch angles, given desired X and Y trajectories. The basics of SMC design were then presented. Important aspects of SMC formulation involve the design of the sliding surface and the corrective control law. The discontinuity in the CSMC corrective control law results in the undesirable chattering phenomena that SMCs are prone to. Simulations and experiments were conducted to verify that the quadrotor does indeed suffer from chattering under CSMC control. The

simulations and experiments show the importance of the k_1 gain in reducing the trajectory tracking errors. The simulations and experiments also show that a high k_1 gain results in chattering of the controlled system. To reduce chattering, different continuous approximations can be used in the control law at the expense of decreased robustness of the controller. Adaptive SMC and second-order SMC were also presented with the aim of reducing chattering. Finally, in this chapter, given that decoupled control was used for the quadrotor and manipulator subsystems, closed-loop inverse kinematics control was presented for the control of the manipulator subsystem.

In Chapter 4, different SMC chattering performance indices will be developed to measure and compare the levels of chattering that SMCs experience. In Chapter 5, controller gains optimisation methods will be presented and compared with the aim of obtaining optimal controller gains that minimise chattering, control inputs, and trajectory tracking errors of the SMCs. Comparison of the different fault-tolerant SMCs, with optimally tuned controller gains, for farm coverage and plant sample collection will be presented in Chapter 6.

4 Sliding Mode Control Chattering Performance Indices

SMCs are prone to chattering, which is a high-frequency switching action that results in high-frequency oscillations of the controlled system. Chattering can typically be observed as high-frequency oscillations of the control input. Chattering occurs due to various factors such as unmodelled dynamics and noise. When designing and tuning SMCs, chattering reduction is therefore of importance. In addition to this, the objectives of minimising error and minimising control input need to be met when designing SMCs. Performance indices exist that are useful for analysing and comparing the errors and control inputs of controllers. However, similar performance indices that measure chattering are still required. This chapter first details several performance indices that are typically used to evaluate the error and control input of controllers. Chattering performance indices that can be used during the controller gains-tuning process are then developed and validated.

4.1 Related Work

Some work has been done on SMC chattering analysis, however with some significant limitations. Levant [154] proposed a formal definition of chattering that classified chattering into three levels: infinitesimal, bounded and unbounded. The method, however, as stated by Levant, might have debatable practical interpretation [154]. The describing function method is a popular approach that has been used by Martínez-Fuentes et al., [155], Castillo and Freidovich [100], Pérez-Ventura and Fridman [101], Asad et al., [156], Boiko et al., [102–104], Rosales et al., [105], and Lee and Utkin [106] to estimate SMC chattering. The describing function method, however, is an approximate method which is accurate only for certain classes of systems that can either be represented as an interconnection of linear systems or can be linearised about a certain point [100]. In addition, the linear part of the system must have low-pass filter characteristics [100] in order for the describing function method to be applied. In this chapter, alternative methods for chattering analysis, making use of the discrete Fourier transform for frequency analysis of the control input, are proposed that are not restricted by these considerations.

4.2 Error and Control Input Performance Indices

Before describing the chattering performance criteria formulation, first an overview of some of the existing error and control input performance criteria is presented. There are

several different error and control input performance indices that can be used to tune controller gains and to compare the performance of the controllers. The choice of error or control input performance index to be used typically depends on the desired behaviour of the system.

4.2.1 Error indices

Some of the common performance indices that are used for the trajectory tracking errors are the [1]:

- integral time-weighted absolute error (ITAE),
- integral absolute error (IAE),
- integral squared error (ISE),
- integral time-weighted squared error (ITSE).

For the error $e(t)$ the indices are defined as, [1]:

$$\begin{aligned} \text{ITAE} &= \int t\,|e(t)|\;\mathrm{d}t \\ \text{IAE} &= \int |e(t)|\;\mathrm{d}t \\ \text{ISE} &= \int e^2(t)\;\mathrm{d}t \\ \text{ITSE} &= \int te^2(t)\;\mathrm{d}t \end{aligned} \tag{4.1}$$

For the IAE performance index, the absolute error is integrated over time. This gives the same weight to all errors over the entire time span, thus, given a step input, large errors are not minimised quickly, however, small errors tend not to persist over time. With the ITAE index, the absolute error is multiplied by time and integrated over time. The ITAE thus gives a greater weight to errors occurring at a later time. Errors therefore tend to settle faster than for the IAE. With the ISE index, the square of the error is integrated over time. This therefore gives a much greater weight to very large errors. The ISE would thus minimise very large errors quickly. However, given a step input, the ISE would allow smaller errors to persist over time. With the ITSE index, the squared error is multiplied by time and integrated over time. Large errors that persist over time are therefore penalised.

4.2.2 Control input indices

Similar to the error indices, some commonly used performance indices for the control input are the [1]:

- integral time-weighted absolute control input (ITACI),
- integral absolute control input (IACI),
- integral squared control input (ISCI),
- integral time-weighted squared control input (ITSCI).

The control input indices aim to minimise the control input and in some cases can also assist in lowering oscillations in the control input. As with the error indices, given a step input, the ITACI index gives greater weight to control inputs that persist over time. As chattering can be observed as a high-frequency control input that persists after steady state has been attained, the ITACI could possibly give an indication of chattering. However, the ITACI would not necessarily be able to distinguish between a constant control input that persists over time versus an oscillating control input that persists over time. The ITSCI index gives higher weight to high control inputs that persist over time. The IACI index gives the same weight to all control inputs irrespective of time and can thus allow for large control inputs to exist. The ISCI on the other hand penalises large control inputs.

4.3 Chattering Performance Indices

Although the control indices that are typically used for controller gains tuning are useful in reducing the control input, they do not necessarily provide sufficient information on the chattering that SMCs are prone to. To minimise chattering, which can be observed as high-frequency oscillation of the control input, frequency information of the control input can be used to develop a performance index. Instead of using indices obtained directly from the control inputs, it is therefore proposed that chattering indices based on the discrete Fourier transform frequency analysis of the control input be used.

4.3.1 Indices based on the fast Fourier transform

The fast Fourier transform (FFT) [157, 158] is a well-known method for computing the discrete Fourier transform for transforming signals from the time domain to the frequency domain. Using the FFT, the frequency components and amplitudes of the control input signal can be obtained. The chattering performance indices can then be obtained from

the amplitudes A of the frequencies f. The discrete-time Fourier transform is defined as [157]:

$$y_{fft}[k] = \sum_{j=0}^{n-1} \omega_n^{jk} x[j] \tag{4.2}$$

where x is the input vector; n is the number of uniformly spaced samples; $0 \leq k < n$, $0 \leq j < n$ and ω_n is one of n complex roots of unity [157]:

$$\omega_n = e^{-2\pi i/n} \tag{4.3}$$

where i is the imaginary unit.

For control input frequency f (where $0 \leq f \leq \frac{F_s}{2}$ and F_s is the sampling frequency), the amplitude A is then:

$$A[k] = \begin{cases} |y_{fft}[k]| & \texttt{if} \quad k = 1 \\ 2 \times |y_{fft}[k]| & \texttt{if} \quad 2 \leq k \leq \left(\frac{n}{2} - 1\right) \end{cases} \tag{4.4}$$

In a similar manner to the error and control input indices, below are the possible chattering indices:

- integral frequency-weighted absolute control input frequency amplitude (IFAFA),
- integral absolute control input frequency amplitude (IAFA),
- integral squared control input frequency amplitude (ISFA),
- integral frequency-weighted squared control input frequency amplitude (IFSFA).

To exclude information related to lower frequencies, the indices are only computed over a specified percentage (a% to b%) of the frequency span. The IAFA index gives the same weighting to all amplitude values, whereas the ISFA gives greater weighting to higher amplitude values. The IFAFA index gives higher weighting to higher frequencies, and the IFSFA gives higher weighting to both higher amplitudes and higher frequencies. For control input frequencies f, with amplitudes A, the chattering indices are:

$$\begin{aligned} \text{IFAFA} &= \sum_{k=a}^{k=b} f A[k] \\ \text{IAFA} &= \sum_{k=a}^{k=b} A[k] \\ \text{ISFA} &= \sum_{k=a}^{k=b} (A[k])^2 \\ \text{IFSFA} &= \sum_{k=a}^{k=b} f (A[k])^2 \end{aligned} \tag{4.5}$$

4.3.2 Indices based on short-time Fourier transform

An alternative method to using the FFT is to use the short-time Fourier transform (STFT) [158,159]. The STFT analyses how the frequency content of a signal changes over time by sliding window over the signal and computing the discrete-time Fourier transform of the windowed section of the signal. In contrast, the FFT returns the highest amplitude per frequency for the entire signal range. The window size of the STFT affects the time and frequency resolution for the STFT. A wide window length produces higher resolution in the frequency domain while a narrow window length produces a higher resolution in the time domain [158]. The STFT matrix with m rows and k columns is defined as [158,159]:

$$y_{stft}[m,k] = \sum_{n_t=0}^{M-1} x[n_t]\, h[n_t - mR]\, e^{-j2\pi k n_t/M} \tag{4.6}$$

where $x[n_t]$ is the input signal at time n_t; $h[n_t]$ is the window function of length M; $y_{stft}[m,k]$ is the discrete-time Fourier transform of the windowed data, centred about time mR and R is the hop size (the difference between window length M and overlap length L), between successive discrete-time Fourier transforms.

Examples of commonly used windows include the: triangular, Hann, and Hamming windows [160]. These windows all taper off at the edges and the window's overlap compensates for the effect of this tapering off. The Hann window is [160]:

$$h_{hn}[n_t] = 0.5\left(1 - \cos\left(2\pi\frac{n_t}{N}\right)\right), \quad 0 \leq n_t \leq N \tag{4.7}$$

where the window length $M = N + 1$. The Hamming window is [160]:

$$h_{hm}[n_t] = 0.54 - 0.46\cos\left(2\pi\frac{n_t}{N}\right), \quad 0 \leq n_t \leq N \tag{4.8}$$

The Bartlett window, which is a type of triangular window and is [160]:

$$h_{br}[n_t] = \begin{cases} \dfrac{2n_t}{N}, & 0 \leq n_t \leq \dfrac{N}{2} \\ 2 - \dfrac{2n_t}{N}, & \dfrac{N}{2} \leq n_t \leq N \end{cases} \tag{4.9}$$

Similar to the indices obtained from the FFT, the following indices can thus be obtained from the STFT.

- integral frequency-weighted absolute chattering frequency power spectrum (IFAP),
- integral absolute chattering frequency power spectrum (IAP),

- integral squared chattering frequency power spectrum (ISP),
- integral frequency-weighted squared chattering frequency power spectrum (IFSP).

However, unlike the FFT indices, for the STFT indices, since time is taken into account, the average of the STFT output values at times when the sliding surface is reached is used to compute the chattering indices. The STFT indices are:

$$\begin{aligned} \text{IFAP} &= \sum_{k=a}^{k=b} f\overline{P}\,[k] \\ \text{IAP} &= \sum_{k=a}^{k=b} \overline{P}\,[k] \\ \text{ISP} &= \sum_{k=a}^{k=b} \left(\overline{P}\,[k]\right)^2 \\ \text{IFSP} &= \sum_{k=a}^{k=b} f\left(\overline{P}\,[k]\right)^2 \end{aligned} \tag{4.10}$$

where:

$$\overline{P}\,[k] = \frac{1}{r}\sum_{m=m_1}^{m=m_r} |y_{stft}\,[m,k]| \tag{4.11}$$

where m_1 to m_r represent the times when the sliding surface is reached, and r is the number of instances when the sliding surface is reached. In practice, when the sliding surface is reached, the value of s is not exactly zero, instead the value oscillates about zero as shown in Figure 3.10. To determine the relevant time instances when the sliding surface is reached, the derivative of s is therefore computed and the times corresponding to a change in sign of s, including the time instant prior to the sign change are used as the desired time instants.

4.4 Control Input Oscillations Performance Indices

When considering the control input, chattering is not the only cause of undesirable control input oscillations. For instance, for controllers such as PID controllers, large values of the proportional gain can result in control input oscillations. For SMCs, control input oscillations that are not due to chattering can also occur. The control input indices presented in Section 4.2, aim at identifying high control inputs, not necessarily highly-oscillatory control inputs. Similar to the proposed chattering indices, control input oscillation indices based on the FFT and STFT are also proposed. For the FFT formulation, the control input oscillation indices are essentially the same as those presented in Section 4.3. The main difference being that in this case, the full frequency range is considered, rather than only the high frequencies. For the STFT formulation the indices are:

- integral frequency-weighted absolute control input oscillation (IFACO),
- integral absolute control input oscillation (IACO),
- integral squared control input oscillation (ISCO),
- integral frequency-weighted squared control input oscillation (IFSCO).

For the STFT formulation, again the full frequency range is considered, however the indices are computed as follows:

$$\begin{aligned} \text{IFACO} &= \sum fO\left[k\right] \\ \text{IACO} &= \sum O\left[k\right] \\ \text{ISCO} &= \sum \left(O\left[k\right]\right)^2 \\ \text{IFSCO} &= \sum f\left(O\left[k\right]\right)^2 \end{aligned} \tag{4.12}$$

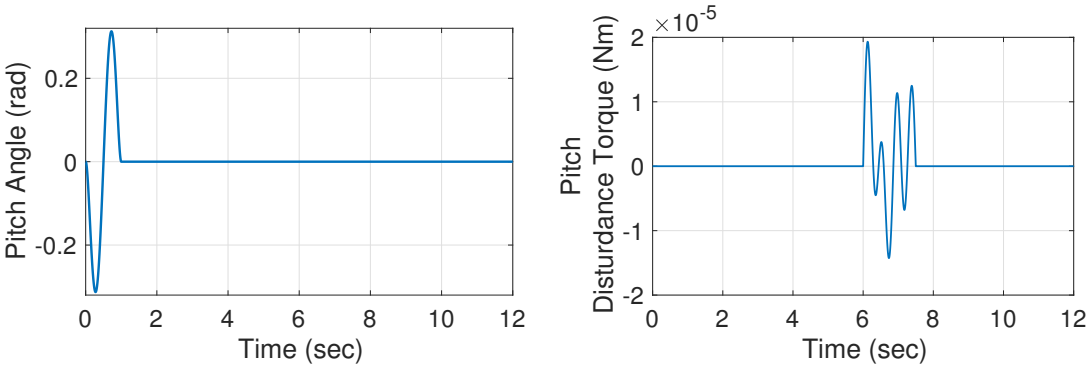
where:

$$O\left[k\right] = \left|y_{stft}\left[m,k\right]\right| \tag{4.13}$$

4.5 Quadrotor Control Simulations

To compare the two chattering indices' formulations, a variety of manually-selected gains are applied to the discontinuous CSMC, with the quadrotor simulated to track a simple pitch trajectory as shown in Figure 4.1a. Wind disturbances are added to the simulation as shown in Figure 4.1b. Sensor noise signals are also added to the simulations. The controller gains used in the simulations are shown in Tables 4.1 and 3.2. For this simulation, the controller gains are manually chosen to obtain different chattering levels, control input signals and trajectory tracking errors. The results of the simulations are used to validate the proposed chattering indices and to compare the chattering, control input and trajectory tracking error indices in order to select the most appropriate indices for the MOO gain-tuning process.

The quadrotor is simulated under altitude and attitude control. The roll and yaw DoF controllers are set to maintain angles of zero radians for this simulation. The Z DoF controller is set to maintain a constant height value. For simplicity, only the results of the pitch controller are presented and analysed in this section. However, similar results were obtained when also considering trajectory tracking of the other DoFs. The values used to simulate the quadrotor are given in Chapter 2. The control loop is run at a frequency



(a) Desired pitch trajectory.

(b) Disturbance torque - pitch.

Figure 4.1: Pitch angle simulation.

of 500 Hz, which is the frequency of the Crazyflie attitude control loop. The FFT [157] is applied to the quadrotor control inputs using the Matlab '*fft*' function and the STFT is applied using the Matlab '*stft*' function. The STFT windows make use of the Matlab '*hann*', '*hamming*' and '*bartlett*' functions. The chattering indices are computed starting at a frequency of 10% of the highest frequency. This effect of different starting frequencies can, however, be further investigated to determine their impact on the resultant chattering indices.

Table 4.1: Quadrotor control simulation gains.

Pitch Gain	$CSMC_0$	$CSMC_1$	$CSMC_2$	$CSMC_3$	$CSMC_4$
k_1	0	6	3	6	3
k_2	25	25	25	2	2
c_1	2	2	2	25	25

4.5.1 Trajectory tracking error results

The trajectory tracking error results are shown in Figure 4.2. From Figure 4.2, it can be seen that $CSMC_3$ and $CSMC_4$ have highly-oscillatory trajectory tracking errors compared to $CSMC_0$, $CSMC_1$ and $CSMC_2$. $CSMC_1$ has lower trajectory tracking errors than $CSMC_2$ and $CSMC_0$. $CSMC_0$ has high errors from around 0.55 seconds and 0.9 seconds. When the wind disturbances are experienced from a time of 6 seconds, $CSMC_4$ has the highest errors, followed by $CSMC_3$ and $CSMC_0$. As can be seen from a time of 6 seconds to 8.5 seconds, $CSMC_1$ and $CSMC_2$ are better able to reject disturbances than the other three controllers. The error results from $CSMC_0$, which has a k_1 gain of zero, show the importance of the k_1 gain in reducing trajectory tracking errors and rejecting disturbances as it had worse performance in both cases than $CSMC_1$ and $CSMC_2$ which both have non-zero k_1 gains. The selection of the k_2 and c_1 gains is also important as shown by the poor performance of $CSMC_4$ and $CSMC_3$.

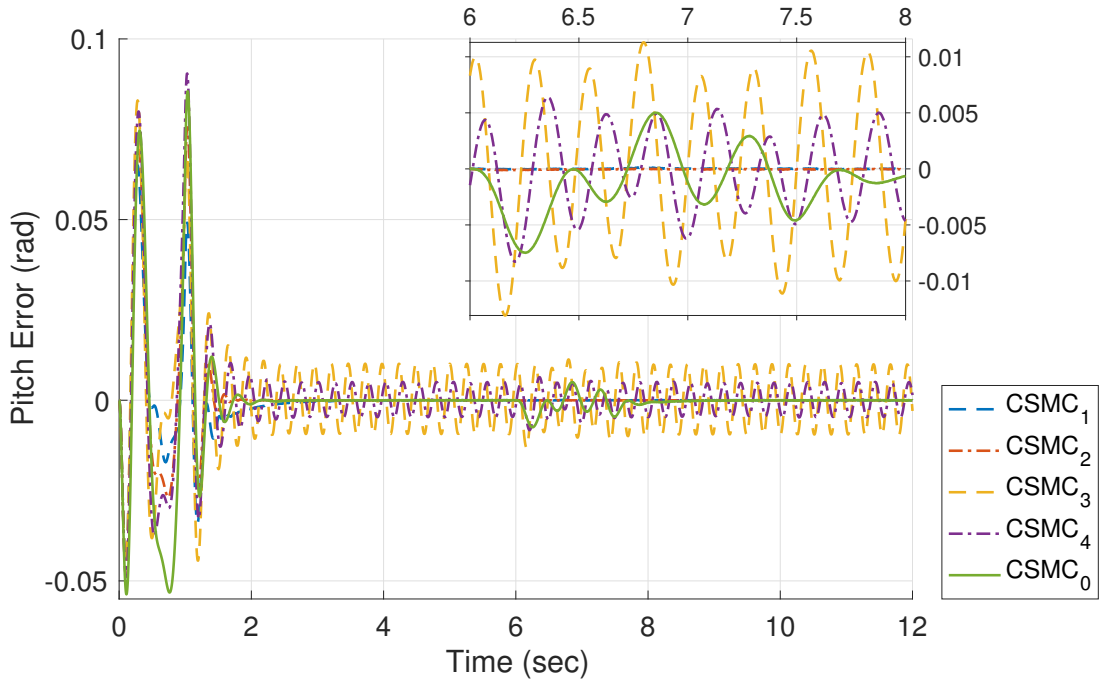


Figure 4.2: Trajectory tracking error results.

4.5.2 Control input results

Figure 4.3 shows the desired control inputs and Figure 4.4 shows the actual control inputs, taking into account the rotor dynamics. Overall, the actual control inputs are lower that the desired control inputs. As can be seen from Figure 4.3, $CSMC_1$ experiences significant, high-amplitude chattering and $CSMC_2$ has lower-amplitude chattering. From Figure 4.4, the actual chattering experienced by $CSMC_1$ and $CSMC_2$ is present, however it is much lower than the chattering observed from the desired control inputs. The lower chattering observed from the actual control inputs is due to the rotor dynamics.

Highly-oscillating desired control inputs as well as high-amplitude chattering can be observed from Figure 4.3 for $CSMC_3$. $CSMC_4$ also exhibits highly-oscillatory desired control inputs, albeit with lower amplitude than $CSMC_3$. In this case, highly-oscillatory control inputs refer to control input oscillations that are not due to chattering. For example, $CSMC_0$ does not exhibit any chattering throughout the entire time period, but from a time of about 6 seconds to 8 seconds, highly-oscillatory desired control is observed for $CSMC_0$. For the same time period of 6 seconds to 8 seconds, $CSMC_3$ experiences a combination of high-amplitude desired control input, and high-amplitude chattering. The chattering in the desired control input for $CSMC_3$ can be observed, for example, at a time of 11.7 seconds. Likewise the chattering in the desired control input for $CSMC_4$ can be observed, at a time of about 11.6 seconds. From Figure 4.4, highly-oscillatory actual control inputs are still observed for $CSMC_3$ and $CSMC_4$. The chattering in the actual control input is not as prominent as in the desired control input, but can still be observed, for example by the sharp change in the actual control input experienced by $CSMC_3$ at a time of 11.7 seconds.

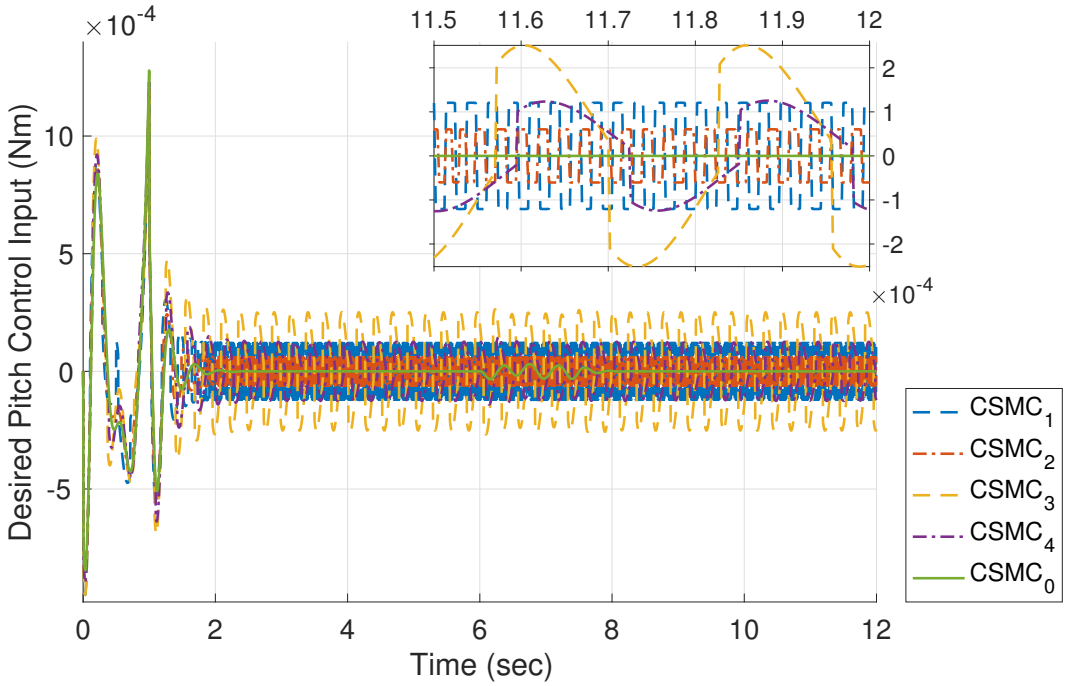


Figure 4.3: Desired control inputs.

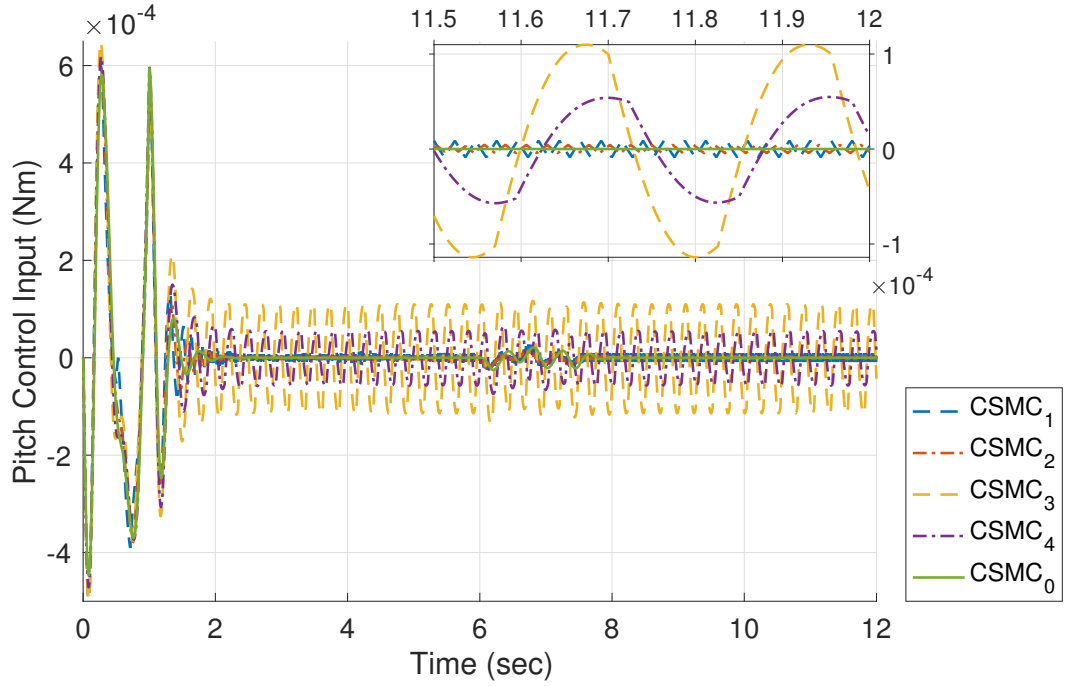


Figure 4.4: Actual control inputs.

From Figure 4.5 it can been seen that both $CSMC_4$ and $CSMC_3$ move far away from the sliding surface during the trajectory tracking manoeuvre as can be seen from a time of 0 second to 2 seconds. $CSMC_4$ and $CSMC_3$ both oscillate significantly about the sliding surface from a time of about 1 second, that is after the trajectory tracking manoeuvre is performed. Since these two controllers oscillate significantly about the sliding surface, the chattering is only experienced at the time instances when a sliding surface of zero is

reached. For example $CSMC_3$ reaches a sliding surface value of zero at 11.7 seconds, and chattering was observed at that time period for the control inputs in Figures 4.3 and 4.4. $CSMC_0$ has slight oscillations about the sliding surface from 1 second to 2 seconds and then remains in the vicinity of the sliding surface until the wind disturbance is experienced and oscillations from a time of 6 seconds to 8 seconds are experienced. Both $CSMC_1$ and $CSMC_2$ remain within the vicinity of the sliding surface from a time of about 2 seconds and also during the time period when the wind disturbance is experienced.

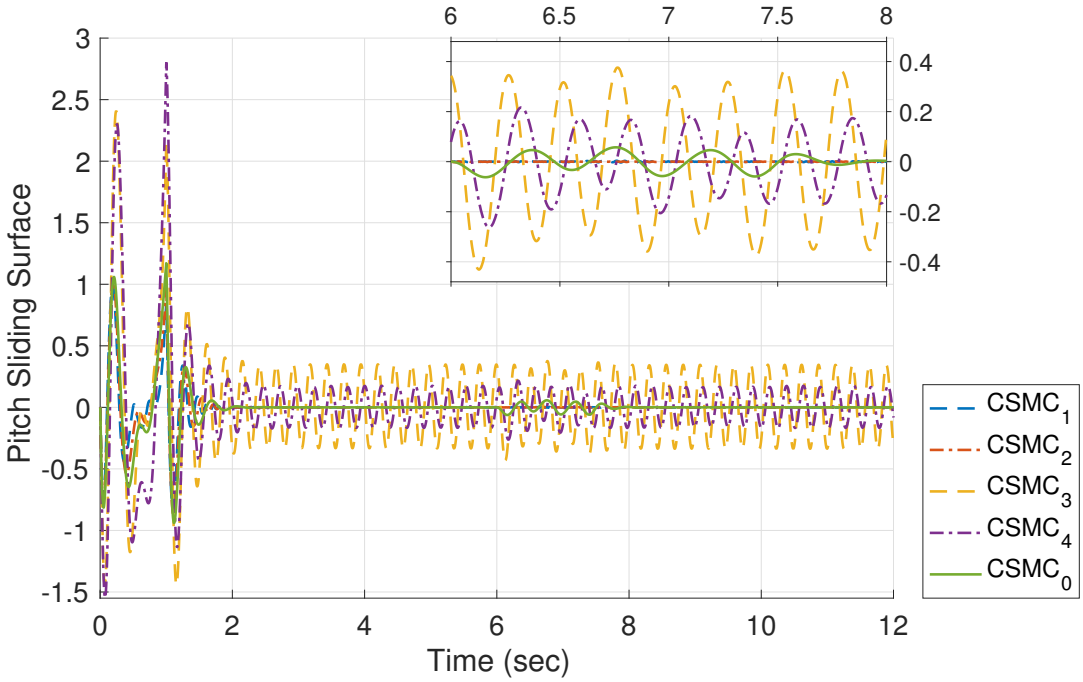


Figure 4.5: Sliding surface results.

4.5.3 Fast Fourier transform chattering indices results

The chattering indices are computed using the actual control inputs shown in Figure 4.4. From Figure 4.4, it can be seen that $CSMC_1$ and $CSMC_2$ have similar chattering frequencies and that $CSMC_1$ has a higher-chattering amplitude than $CSMC_2$. It is therefore expected that $CSMC_1$ should have higher-chattering index results than $CSMC_2$. The difference in chattering between $CSMC_3$ and $CSMC_4$ is difficult to discern from Figure 4.4, however, the desired control inputs in Figure 4.3, show $CSMC_3$ as having higher-chattering amplitude than $CSMC_4$. $CSMC_0$ does not experience chattering, however all of the simulations include sensor noise, which means that very low chattering values may be computed as a result of the sensor noise for $CSMC_0$. $CSMC_1$ and $CSMC_2$ have much higher-chattering frequencies than $CSMC_3$ and $CSMC_4$. It is therefore expected that $CSMC_1$ should have higher-chattering indices than $CSMC_3$. The same applies for $CSMC_2$ which should have higher-chattering indices than $CSMC_4$.

Figure 4.6 shows the control input frequency results obtained from the FFT. The FFT results show $CSMC_1$ as having the highest-amplitude oscillations, followed by $CSMC_2$,

$CSMC_3$, $CSMC_4$ and $CSMC_0$ respectively. All of the controllers, except for $CSMC_0$, have several instances of amplitude peaks at different frequencies.

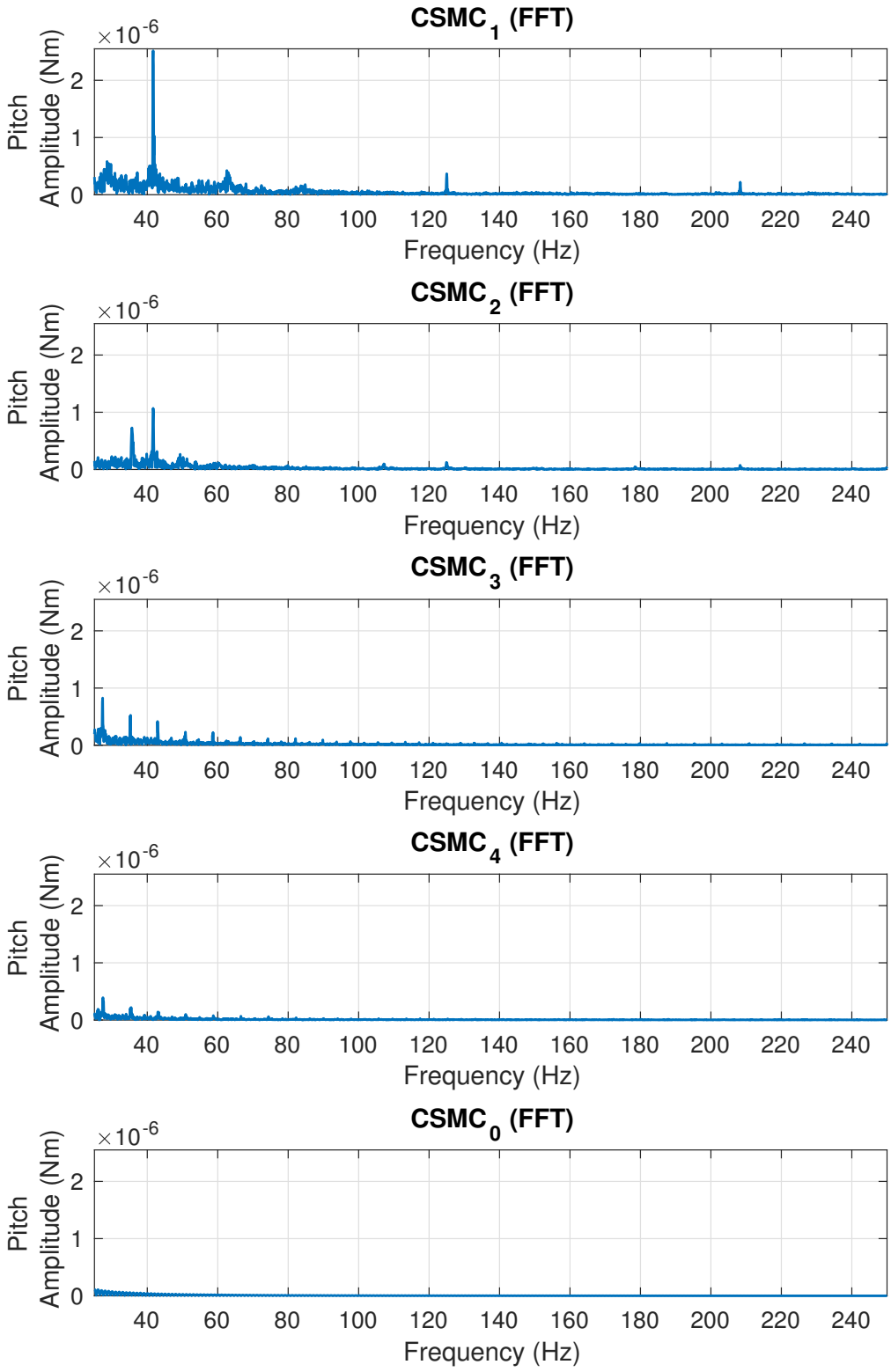


Figure 4.6: FFT results.

The chattering indices are computed for frequencies starting at 10% of the maximum frequency, since chattering is a high-frequency phenomena. The resultant FFT chattering

indices are shown in Figure 4.7. For the FFT formulation, $CSMC_1$ has the highest-chattering indices and $CSMC_0$ has the lowest-chattering indices. The ISFA and IFSFA show $CSMC_1$ as having significantly higher chattering than the other controllers. For the ISFA and IFSFA indices, $CSMC_4$ has slightly higher-chattering results than $CSMC_0$, whereas for the IAFA and IFAFA there is a bigger difference between the two. $CSMC_2$ has the second highest-chattering indices. For the IFAFA index, $CSMC_3$ is only slightly lower than $CSMC_2$.

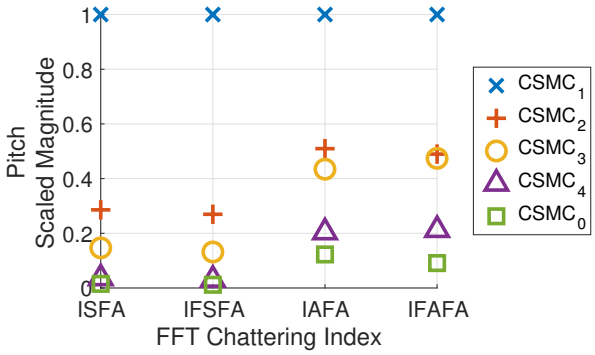


Figure 4.7: FFT chattering indices results.

The ISFA and IFSFA give greater weight to high amplitudes, thus $CSMC_1$ has significantly higher-chattering results than $CSMC_2$ for these two indices. The ISFA and IFSFA indices of the FFT formulation are in this case considered as not producing good results, as they show $CSMC_0$ and $CSMC_4$ as having similar chattering indices. The IAFA and IFAFA can be used to minimise the amplitude of the chattering oscillations, however the results are not considered as good as it is expected that $CSMC_3$ should have a slightly higher-chattering index than $CSMC_2$. In this case IAFA is the closest to the expected result.

4.5.4 Short-time Fourier transform chattering indices results

For the STFT, a small window size results in better time resolution, whereas a larger window size results in better frequency resolution. Good time resolution and frequency resolution are both important as the STFT-based chattering indices require an estimate of the chattering frequency and magnitude at specific time instances. The window overlap length also affects the results obtained since the overlap compensates for the effect of the tapered windows. To select an appropriate window size and overlap length, different window sizes and overlap lengths are simulated for the Hann, Hamming and Bartlett windows and the STFT results are compared.

The results of comparing the different window types, window sizes and overlap lengths using the actual control inputs from Figure 4.4 are shown in Figures 4.8 to 4.10. The three different window types produce a variety of different results. From the control input-results shown in Figure 4.4, $CSMC_1$ has the highest-chattering levels as it has both

high-amplitude and high-frequency chattering, and so it should result in the highest-chattering indices. $CSMC_3$ is expected to have the second highest chattering, followed by $CSMC_2$, $CSMC_1$, and $CSMC_0$.

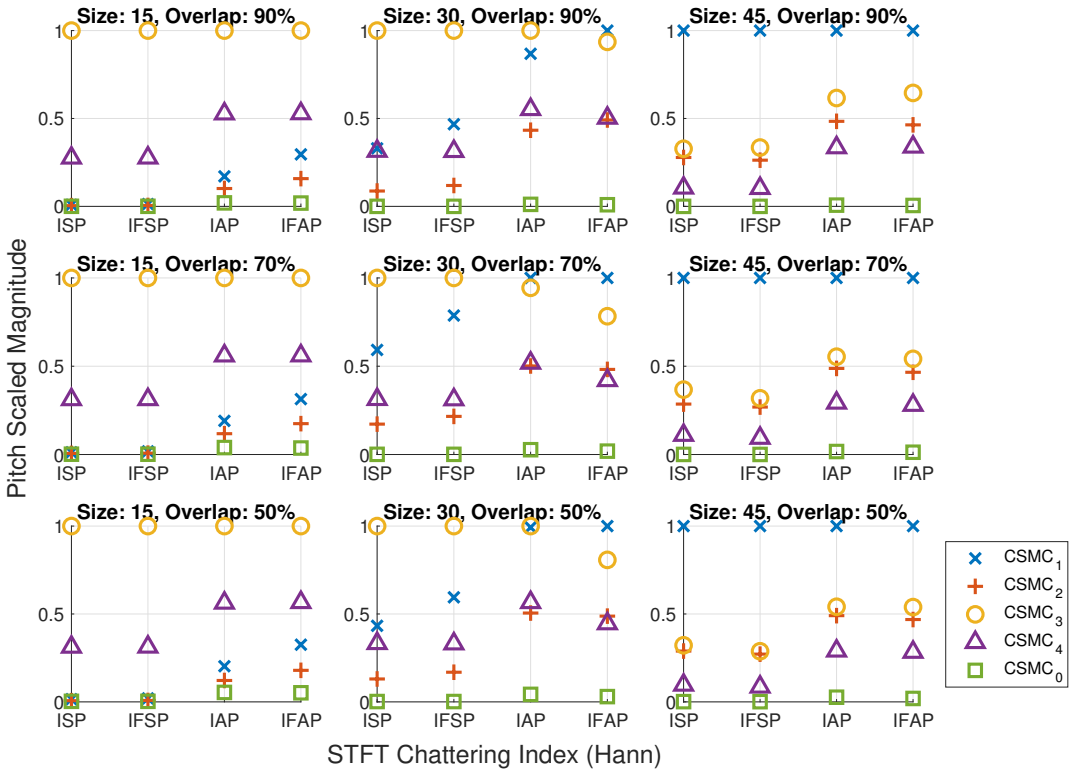


Figure 4.8: Hann window size and overlap results.

For the Hann window of size 15, $CSMC_3$ has the largest chattering, followed by $CSMC_4$, for all three overlap lengths. A window size of 30, in most cases shows $CSMC_3$ as having the highest chattering, except for the IFAP indices which have $CSMC_1$ with largest chattering. The IAP for the 50% and 75% overlaps show $CSMC_3$ and $CSMC_1$ as having similar chattering. A Hann window of size 45, for all of the overlap lengths, shows $CSMC_1$ as having the highest chattering, followed by $CSMC_3$, $CSMC_2$, $CSMC_4$, and $CSMC_0$. For the smaller overlap lengths, $CSMC_3$ and $CSMC_2$ have similar results. For an overlap length of 90%, there is a slightly larger difference between $CSMC_3$ and $CSMC_2$. The IAP and the IFAP for a window size of 45 and overlap length of 90% both have good results, however the IFAP is slightly better than the IAP. The IFAP index for the Hann window with window size 45 and overlap length of 90% shows that $CSMC_2$ experiences approximately half the chattering levels of $CSMC_1$. Likewise, $CSMC_4$ has approximately half the chattering compared to $CSMC_3$.

For the Hamming window of size 15, for all the overlap lengths, $CSMC_3$ has the highest chattering, followed by $CSMC_4$. For a window size of 30, $CSMC_3$ has the highest chattering, followed by $CSMC_1$. A window size of 45 produces a variety of results for the different overlap lengths and indices. The ISP indices for a window size of 45, show the expected order of chattering for the controllers, however the differences in chattering magnitude are not within the expected ranges.

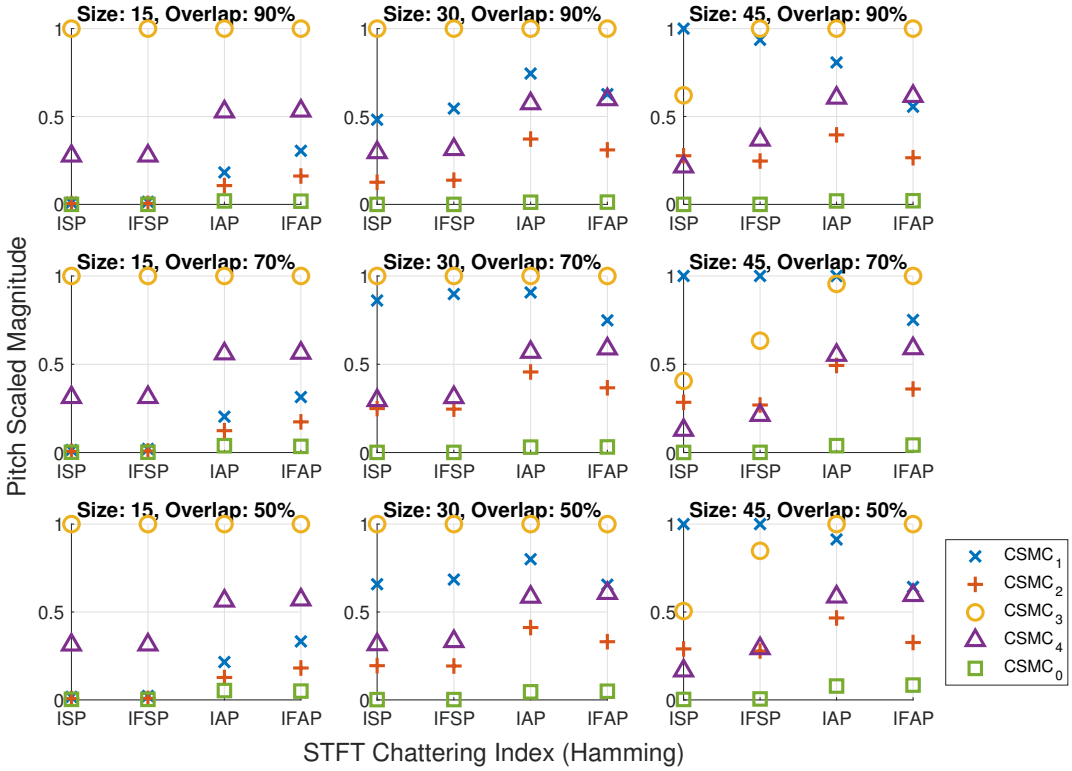


Figure 4.9: Hamming window size and overlap results.

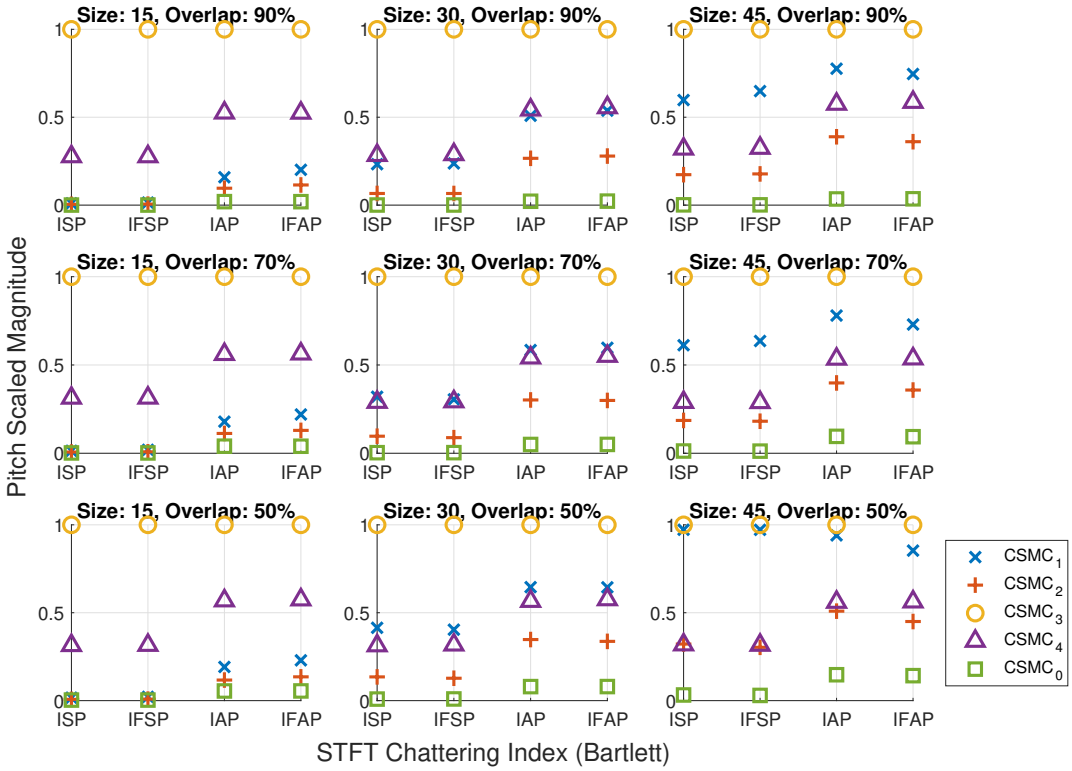


Figure 4.10: Bartlett window size and overlap results.

For the Bartlett window, all of the window sizes and overlap lengths show $CSMC_3$ as having the highest chattering. For a window size of 15, $CSMC_4$ has the second highest

chattering. For a window size of 30, $CSMC_1$ and $CSMC_4$ have similar chattering, with very small differences. For a window size of 45, $CSMC_1$ has the second highest chattering.

Although further investigation and analysis on the effect of window type, size and overlap in different scenarios can be conducted, the results obtained are sufficient to allow for the window parameters to be selected for this thesis. For the remaining STFT simulations, a Hann window size of 45 samples and an overlap length of 90% (41 samples) is used. The STFT results for the Hann window size of 45 samples and an overlap length of 90% are shown in Figure 4.11. Figure 4.12 shows the results of the obtained mean values $\overline{P}$.

Overall, the STFT with a Hann window of size 40 and overlap length 90% performs better than the FFT at determining the chattering levels of the controllers that were investigated. Both formulations have $CSMC_1$ as having the highest chattering and $CSMC_0$ as having the lowest chattering. The STFT shows $CSMC_0$ as having much lower chattering than the other controllers. This is expected as the frequency information for the $CSMC_0$ is due to the effects of sensor noise since it has a k_1 gain of zero. The STFT has $CSMC_3$ with the second highest chattering, whereas the FFT has $CSMC_2$ with the second highest chattering. The IFAP of the STFT formulation with a Hann window of size 40 and overlap length of 90% is therefore selected as the most appropriate chattering index.

4.5.5 Trajectory tracking error indices results

Two sets of error indices are computed for the simulation. One set of error indices is computed for the first half of the time span (trajectory portion), from a time of 0 second to 6 seconds, to determine the trajectory tracking errors in the regular trajectory tracking portion of the simulation. The second set of error indices is computed for the second half of the time span (disturbance portion), from a time of 6 seconds to 12 seconds, to determine the errors when wind disturbances are experienced.

The trajectory tracking error-index results for the simulations are shown in Figure 4.14a and the wind disturbance error-index results are shown in Figure 4.14b. For ease of comparison, the indices have been scaled by dividing each index by the maximum value obtained for that index for all of the controllers and simulations. For the trajectory tracking portion, all of the error indices, except for the ISE, show $CSMC_3$ as having the highest trajectory tracking errors. $CSMC_3$ has the third highest ISE index. $CSMC_0$ has the highest ISE index. $CSMC_4$ has the second highest ISE, ITSE, IAE, and ITAE error indices. $CSMC_2$ has higher indices than $CSMC_1$, although the difference is very small for the ITAE index. $CSMC_1$ has the lowest errors for all of the indices. For the portion of the trajectory that experiences the disturbance, $CSMC_1$ has the lowest error indices, followed very closely by $CSMC_2$. $CSMC_3$ has the highest overall error indices, followed by $CSMC_4$ and $CSMC_0$.

The choice of the error index to be used for controller gains tuning depends on the desired trajectory tracking performance for this particular case being studied. For instance, to

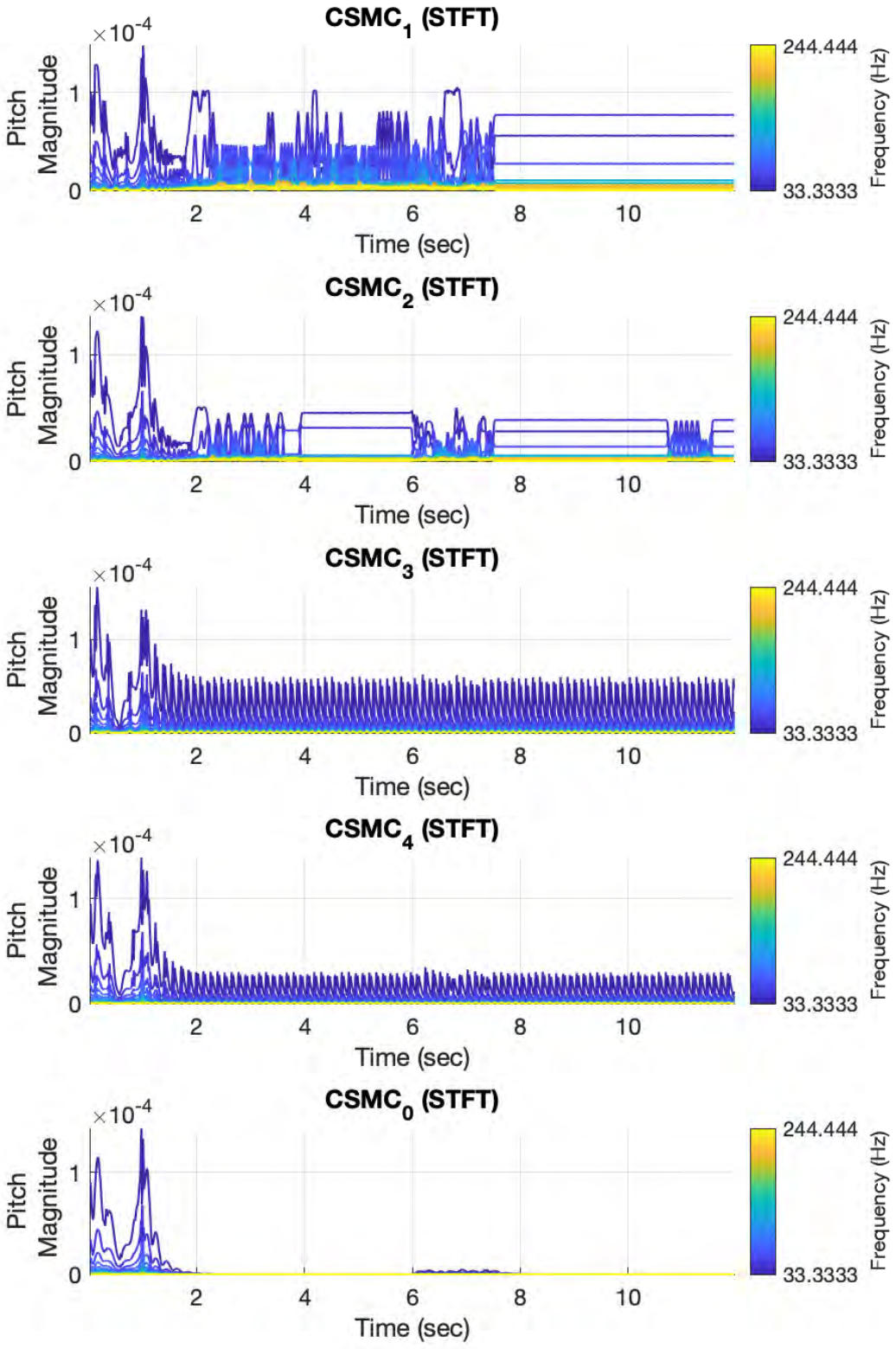


Figure 4.11: STFT results (Hann window size: 45, overlap: 90%).

try to minimise the trajectory tracking error oscillations, the ITAE can be used. This is because during the regular trajectory tracking portion, the ITAE results in relatively high indices for $CSMC_3$ and $CSMC_4$, which both have highly-oscillatory errors as shown in Figure 4.2. If the goal is to try to minimise large errors, regardless of oscillations, then

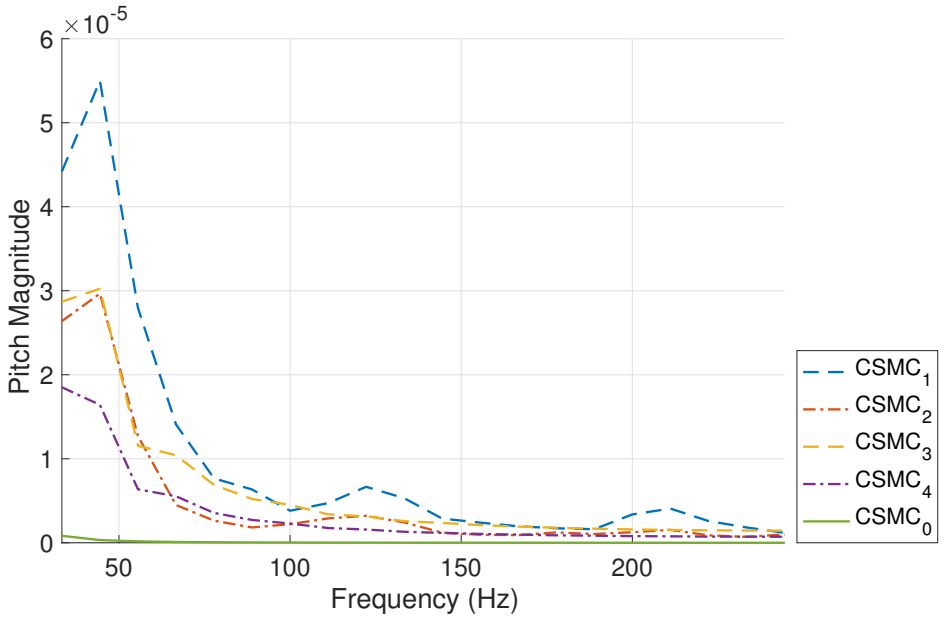


Figure 4.12: STFT mean results (Hann window size: 45, overlap: 90%).

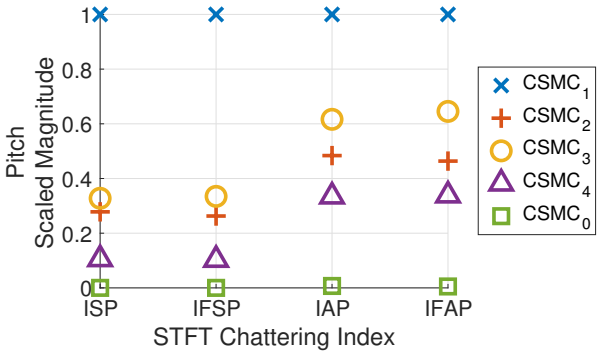
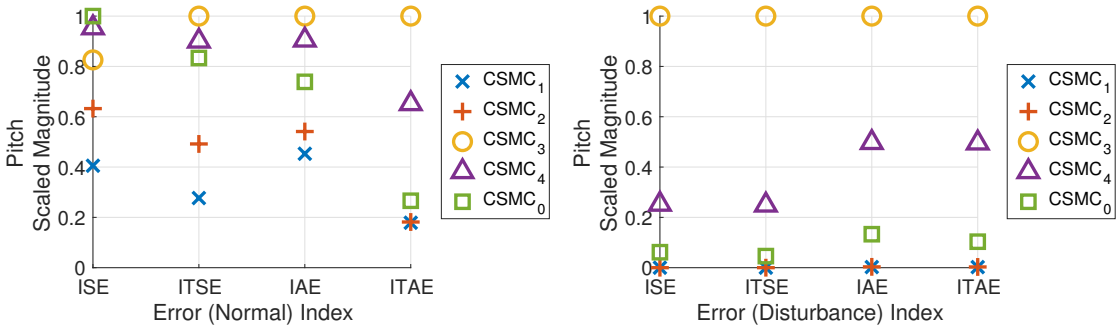


Figure 4.13: STFT chattering indices results (Hann window size: 45, overlap: 90%).



(a) Without disturbances.

(b) With disturbances.

Figure 4.14: Trajectory tracking error indices.

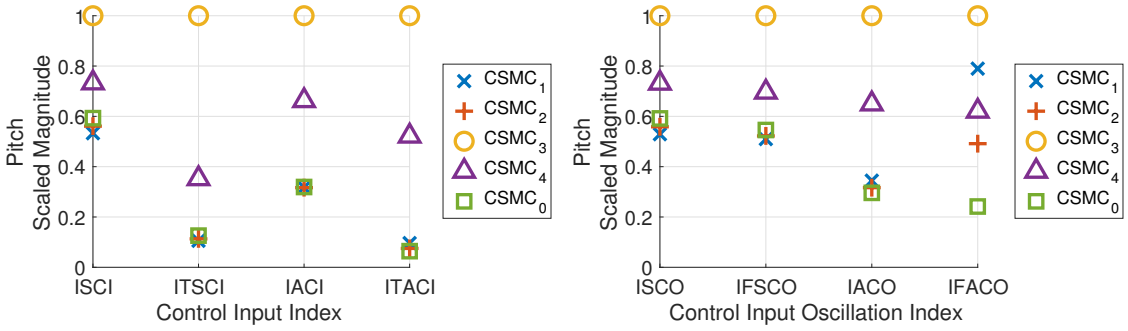
the ISE can be used for this trajectory tracking simulation. The ISE results in the higher indices for $CSMC_0$ which experiences very high errors at certain time instances during the trajectory tracking portion, followed by $CSMC_4$ which also has instances of high error and then by $CSMC_3$. For the disturbance portion, $CSMC_3$ has oscillating errors with

amplitude approximately half that of $CSMC_3$. The IAE and ITAE would be appropriate indices to use in this case.

4.5.6 Control input indices results

The control input-index results for the simulations, using the actual control inputs shown in Figure 4.4, are shown in Figure 4.15a. The control input indices show $CSMC_3$ as having the highest control effort, and $CSMC_4$ as having the second highest control effort. The IACI indices for $CSMC_2$, $CSMC_1$, and $CSMC_0$ are very similar. The ITACI shows $CSMC_0$ as having the lowest control input, followed by $CSMC_2$ and $CSMC_1$. For both the ISCI and ITSCI indices, $CSMC_1$ has the lowest control input, followed by $CSMC_2$, and $CSMC_0$.

The control input oscillation-index results for the simulations, using the actual control inputs are shown in Figure 4.15b. Based on the results of the chattering indices, the control input oscillation indices also make use of the STFT formulation with a Hann window of size 40 and overlap length 90%. The control input indices show $CSMC_3$ as having the highest control input oscillations, and $CSMC_4$ as having the second highest control input oscillations, except for the IFACO index. For the IFACO index, $CSMC_1$ has the second highest control input oscillations, followed by $CSMC_2$, and $CSMC_0$. For the ISCO and IFSCO indices, $CSMC_0$ has the third highest control input oscillations, followed by $CSMC_2$ and $CSMC_1$. For the IACO index, $CSMC_1$ has the third highest control input oscillations, followed by $CSMC_2$ and $CSMC_0$.



(a) Control input indices.

(b) Control input oscillation indices.

Figure 4.15: Control input indices results.

As with the trajectory tracking error indices, the choice of control input or control input oscillation index to be used for controller gains tuning depends on the desired control input performance. In this case, to try to minimise high amplitude, oscillatory control inputs, the ISCI, ITSCI, ISCO, and IFSCO indices can be used. However, they, to different extents allow for lower-amplitude oscillations to persist over time as is evidenced by the $CSMC_0$ having higher indices than $CSMC_2$ and $CSMC_1$. From Figure 4.4, $CSMC_4$ has control input oscillations of approximately half the amplitude of $CSMC_3$, thus the ITACI would be appropriate to use in this case. In addition, the ITACI also shows the

$CSMC_0$ as having the lowest control input. The IACO index is also appropriate as it shows that $CSMC_4$ has lower control input oscillations than $CSMC_3$, and that $CSMC_0$ has the lowest control input oscillations. The IFACO index gives high weighting to high amplitude, oscillatory control inputs as it shows $CSMC_3$ as having the highest control input oscillations. The IFACO index also gives high weight to high-frequency oscillations as it shows $CSMC_1$ as having the second highest control input oscillations. Compared to all the indices, the IFACO is the only one that shows $CSMC_0$ as having much lower control input oscillations than the rest of the controllers.

4.6 Summary

This chapter detailed the development and testing of several chattering performance indices that can be used to analyse the level of chattering of SMCs. Two formulations were developed, one based on the FFT and the other on the STFT. Control input oscillation indices based on the FFT and the STFT were also developed to measure control input oscillations that can be experienced due to factors other than chattering. Existing trajectory tracking error and control input indices were also presented. The indices were compared for the simulated pitch DoF trajectory tracking of a quadrotor under altitude and attitude control, with added wind disturbances for five different CSMCs. The choice of the trajectory tracking error and control input indices is dependant on the desired performance of the system. For chattering, the indices based on the STFT method produced better results than those based on the FFT. Different window types, window sizes and overlap lengths were used for the window function of the STFT. The Hann window, with a window size of 45 and and overlap length of 90% was shown to have the best results for the simulation. For the STFT formulation with the specified Hann window, the IAP index had good results, and IFAP index was shown to have the most accurate results for measuring control input chattering levels. This chattering index will be used in the remaining chapters of the thesis as the chattering performance criteria to be minimised in MOO controller gains tuning of SMCs, and for comparison of chattering of different SMCs.

5 Multi-Objective Optimisation for Controller Gains Tuning

The choice of controller gains significantly affects the performance of a controller. When tuning SMC gains, there is a trade-off between obtaining good tracking performance, minimising control input and minimising chattering. Manual tuning of controller gains with multiple conflicting objectives is time consuming and does not guarantee that optimal gains are found. Tuning of a quadrotor-based aerial manipulator is further complicated by the coupled nature of the system dynamics. Thus, the controller gains of one DoF can have some effect on the performance of a different DoF. MOO methods can be employed to obtain optimal controller gains in such cases. In addition, a better comparison of different controllers can then be made by comparing the performance obtained from optimised controller gains. This chapter details and compares two multi-objective metaheuristic computational intelligence methods for controller gains tuning.

5.1 Related Work

There are two approaches that can be used for optimising multiple objectives when tuning controller gains. The first approach combines the multiple objectives into a single composite objective thus reducing the problem to a single objective optimisation problem for which a single solution is obtained. Costa et al., [161] applied ant colony optimization and differential evolution to optimise the proportional-integral (PI) controller gains of a three-phase induction motor. They combined four objectives, minimising the error for speed, torque, flux, and linkage stator flux for the optimisation problem. Blondin et al., [162] used a combination of a simplified ant colony optimization algorithm and the Nelder-Mead method for the PID controller gains tuning of an automatic voltage regulator of a synchronous generator. The maximum peak value in the system response and the steady-state error were combined to form a single optimisation objective. Hassani and Lee, [163] used a quantum-behaved particle swarm optimisation to tune an LQR controller, considering three objectives, an LQR quadratic performance index, the transient response of the system and the steady-state error. They aggregated the objectives using dynamic weighted aggregation. They tested their method on an inverted pendulum system and an aircraft landing flare system. Mac et al., [164] used particle swarm optimisation to tune the PID controller gains of a quadrotor. They weighted and combined several objectives, namely the settling time and the rise time, the overshoot and the steady-state error into one objective to be optimised. Dangor et al., [165] used a single equation to combine several performance criteria into one performance criterion to be optimised for the PID control of a quarter-car suspension system. They compared manual, genetic algorithm,

particle swarm and differential evolution methods to optimise the combined performance objective. Pedro et al., [166] then applied particle swarm optimisation to the dynamic neural network-based feedback linearisation control of a full-car suspension system. The particle swarm optimisation was used in the training of the dynamic neural network and the tuning of the control parameters. The criterion to be optimised combined ride comfort and vehicle handling with the maximum permitted accelerations, tyre dynamic load, suspension rattle space, control input voltage and actuation force. The method of combining multiple objectives into a single composite objective, however, does not guarantee that the multiple objectives are combined in an optimal manner.

The second approach for optimising multiple objectives is to use multi-objective versions of single-objective optimisation algorithms. Evolutionary algorithms such as the genetic algorithm and swarm algorithms such as the particle swarm algorithm which optimise a single objective have been adapted to optimise multiple objectives [49, 50]. With this approach there is no unique solution to the optimisation problem, rather it results in a range of optimal solutions located on the Pareto optimal front. Multi-objective evolutionary algorithms are well suited for non-convex, nonlinear and constrained optimisation problems such as the controller gains tuning for the nonlinear controller of a nonlinear, coupled system [167]. Coello et al., [50] performed a comparison of the non-dominated sorting genetic algorithm (NSGA-II), Pareto archived evolution strategy (PAES) and MOPSO algorithms for a variety of test functions. They found that the MOPSO had good performance and had lower computational time compared to the PAES and NSGA-II methods. Deb et al., [49] found that the NSGA-II had better performance than the PAES.

Reynoso-Meza et al., [167] compared various MOO methods for controller gains tuning. Of the studies considered in the survey, only two dealt with more than nine objectives (i.e., 10 and 15 objectives) to be optimised. None of the studies in the survey considered the tuning of SMCs and only two studies made use of MOPSO for the controller gains-tuning process. Rodríguez-Molina et al., [168] surveyed various multi-objective metaheuristic optimisation methods used for controller gains tuning. They found that the most popular methods used were variations of the MOGA, the MOPSO and the multi-objective differential evolution (MODE) algorithms. The most popular controller used, making up 62% of the surveyed papers, were variants of the PID controller. In their survey, Rodríguez-Molina et al., [168] found that only 20% of the papers surveyed dealt with more than three objectives to be optimised. They termed these 'many-objective' problems. Of these, all except for one study, dealt with nine or less objectives. A single study had 18 objectives to be optimised with six gains to be tuned.

Gutiérrez-Urquídez et al., [169] used a type of MOGA, namely the NSGA-II for the tuning of model predictive controllers. The three criteria to be optimised were to maximise the feasibility region, minimise the performance loss, and minimise the online computational cost of the controller. They tested the method on two systems, a discrete-time state-space model with one input and one output and two parameters to be tuned, and a discrete-time state-space model with two inputs and two outputs and two parameters to be tuned. Zhou and Zhang [51] developed a multi-objective optimisation tuning method for the PID

control of an aerial manipulator. They used a NSGA-II with two conflicting objectives, that is good tracking performance (obtained by the summation of the ITSE of the system outputs) and smooth, low-oscillation control signals (obtained by the summation of a smooth control index for the system's control inputs). Qin et al., [52] used a parallel simple cell mapping multi-objective optimisation method to tune the proportional and derivative gains of the PID controller of a twin-rotor model helicopter. They tuned eight gains in total. The six objectives were to minimise the peak time, overshoot/maximum amplitude, and ITAE for both DoFs of the system. Mahmoodabadi et al., [170] proposed a variant of the MOPSO that they termed the ingenious-MOPSO. They used this for the SMC gains tuning of a three-link-planar biped robot walking in the lateral plane on slope. Two objectives were optimised, the normalised summation of angle errors and the normalised summation of angle control effort. They did not consider chattering minimisation as one of the objectives.

Having both a large number of gains/parameters to be tuned and a large number of objectives to be optimised increases the complexity of the MOO controller gains-tuning process. No studies have as yet, investigated the effect of having both a large number of gains to tune and a large number of objectives to optimise on the MOO gaining tuning outputs. Few studies have made use of MOO for controller gains tuning of SMCs, and those that have, did not consider chattering as one of the objectives to be minimised. Two multi-objective metaheuristic computational intelligence methods, the MOPSO and MOGA methods, are therefore simulated and compared in this chapter for MOO controller gains tuning of a quadrotor-based aerial manipulator. The MOPSO and MOGA methods are selected as they are the most commonly used methods found in the literature.

5.2 Multi-Objective Particle Swarm Optimisation

To tune the gains for the controllers with multiple objectives, the MOPSO algorithm as presented by Coello and Lecuga, [50] and implemented by Martínez-Cagigal [171] is modified and used. The MOPSO formulation allows for conflicting objectives to be specified and the algorithm then generates Pareto fronts based on the selected objectives. The MOPSO algorithm is as described by Coello and Lecuga, [50] and is repeated here and summarised in Algorithm (1) for completeness. A swarm contains a certain number of particles in its population. First, a user-specified population size of random particles is initialised for the swarm. The values assigned to each particle (in this case the values of the controller gains) is termed the position of the particle. The performance of each particle is evaluated by computing the objective functions for that particle, in this case the error, chattering and control input performance indices. The positions and performance indices of the non-dominated particles are then stored in the repository, and the position of each particle is stored in the particle's memory. Next, the following sequence is repeated until the specified maximum number of cycles has been reached. The particle speed, which determines the new position of the particle is computed as shown in equation (5.1). The new position for each particle is calculated using the computed particle speed

and by applying mutation. The objective functions of the the new particle positions are evaluated. The repository is then updated. If the current position of the particle is better than that stored in memory, the particle's position in its memory is updated.

The particle speed equation is [50]:

$$VEL[i] = W_I \times VEL[i] + R_1 \times (PBESTS[i] - POP[i]) + R_2 \times (REP[h] - POP[i]) \quad (5.1)$$

where:
W_I is the inertia weight;
R_1 and R_2 are random numbers in the range [0..1];
$PBESTS[i]$ is the best overall position obtained by particle i;
$REP[h]$ is a value that is taken from the repository based on the spread of the objective function values obtained thus far. The spread of values is computed by splitting the search area into hypercubes.

Algorithm 1 MOPSO algorithm (adapted from [50])

```
1:  procedure MOPSO
2:      Initialise random swarm population POP with particle positions POS
3:      for <each particle in POP> do
4:          Initialise particle speed VEL
5:          Evaluate performance criteria IND
6:      Store positions POS and performance IND of non-dominated particles in repository
    REP
7:      g ← 0
8:      while g < g_max do
9:          for <each particle in POP> do
10:             Compute particle speed VEL
11:             Compute new POS = POS + VEL
12:             Mutation
13:             Evaluate performance criteria IND
14:             if current IND better than memory IND then
15:                 Update best IND and POS of particle
16:         Update REP
17:         g = g + 1
```

5.3 Multi-Objective Genetic Algorithm

A MOGA, namely the controlled elitist NSGA-II [172] is also explored for MOO controller gains tuning. The NSGA-II works as described by Deb et al., [49, 172–174] and is summarised here. First, the initial population of particles is randomly generated. The objective functions are evaluated and the particles assigned a rank. Parents for the next

generation of particles are selected based on the particle rank and a measure of the crowding distance of the particles. The crowding distance is the distance from a particle to its closest neighbours. The crowding distance function helps to ensure the diversity of the particles. With the controlled elitist NSGA-II, diversity is further maintained by selecting particles that help to increase the diversity of the population, even if those particles have high rank. A child generation of the population is then computed using crossover and mutation. Crossover means the combination of parts of two parent particles and mutation is a random modification of the particle. The new population is evaluated and the next generation is created until the stopping condition is met.

Algorithm 2 NSGA-II algorithm (adapted from [175])

```
1: procedure MOGA
2:     Initialise random population POP
3:     for <each particle in POP> do
4:         Evaluate performance
5:         Assign Rank
6:     Generate child population using crossover and mutation
7:     g ← 0
8:     while g < g_max do
9:         for <each parent and child in POP> do
10:            Assign Rank
11:            Generate sets of nondominated solutions
12:            Determine crowding distance
13:            Add solutions to next generation
14:        Select points on the lower front with high crowding distance
15:        Create next generation using crossover and mutation
16:        g = g + 1
```

The overall structure of the SMC controller, including the MOO controller gains tuning structure for a quadrotor is shown in Figure 5.1 and that for a quadrotor-based aerial manipulator is shown in Figure 5.2.

5.4 Quadrotor Controller Gains Tuning Simulations

For comparison of the two MOO methods, the ISMC_b, making use of the saturation function is used. The Crazyflie quadrotor, simulated in Matlab, is used for the MOO controller gains tuning validation and comparison for the sake of simplicity. The simulations are run on a computer with a 2.6 GHz Dual-Core Intel Core i5 processor and 8 GB 1600 MHz Double Data Rate version 3 (DDR3) memory. For the simulation, the quadrotor first has to track a trajectory from an initial to a final height, followed by simultaneous motions for the X and Y positions and then a yaw angle motion while the quadrotor maintains the final height position. The desired path and desired trajectories are shown in Figures

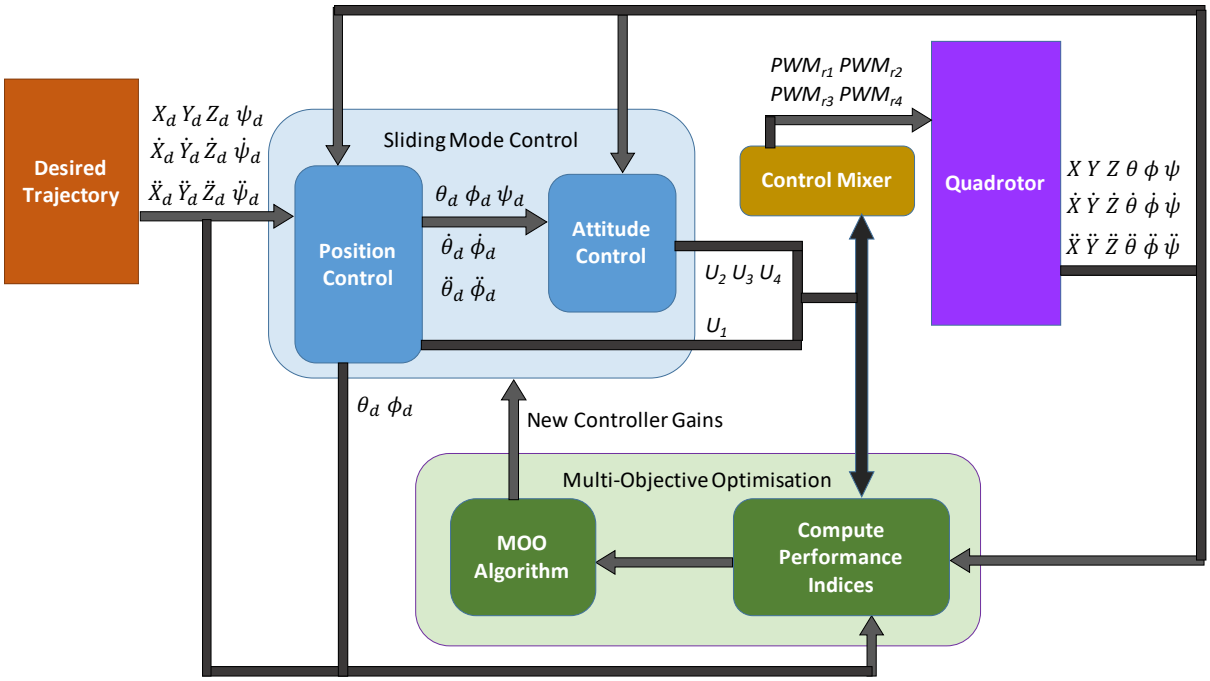


Figure 5.1: SMC and MOO controller gains tuning structure for a quadrotor.

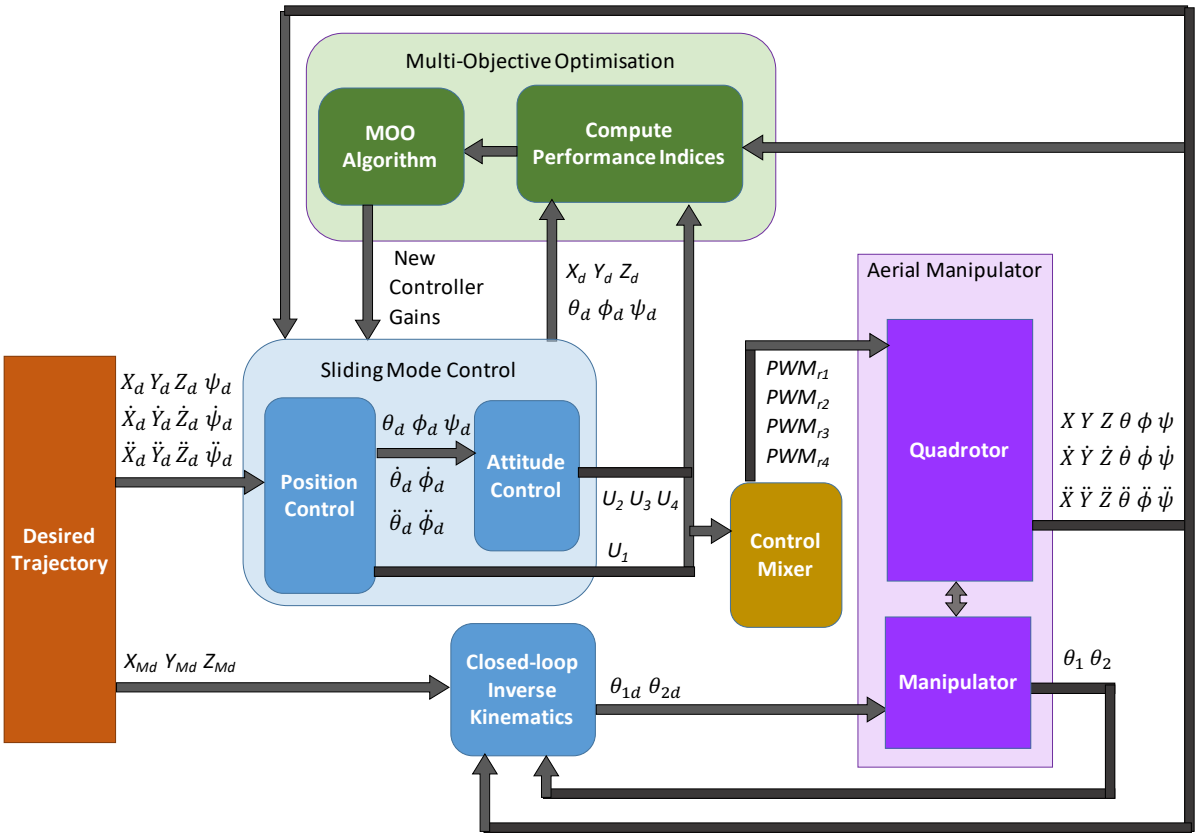


Figure 5.2: SMC and MOO controller gains tuning structure for an aerial manipulator.

5.3 and 5.4 respectively. Small, constant-disturbance forces and moments are added to the simulations to approximate uncertainties in the quadrotor modelling.

From the test cases in Chapter 4, the IFAP index of the STFT formulation gives consis-

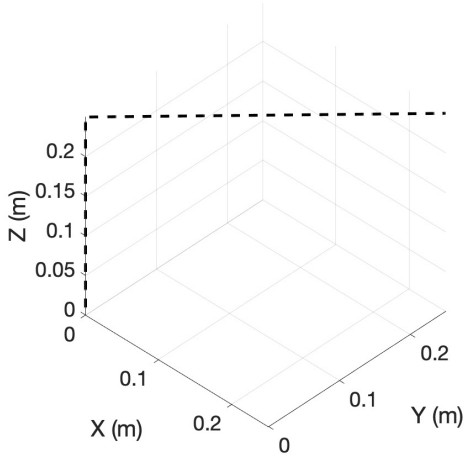


Figure 5.3: Desired path.

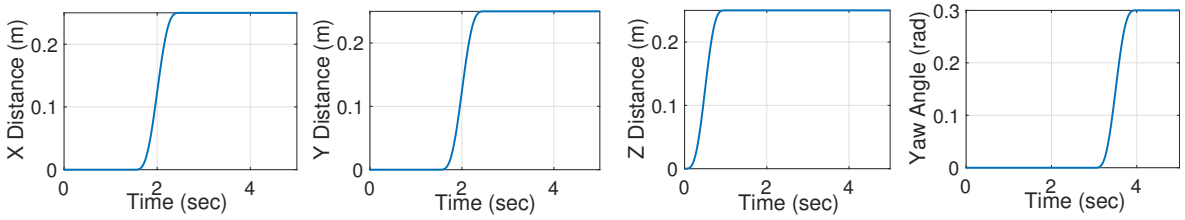


Figure 5.4: Desired trajectories.

tently accurate results for chattering analysis. The IFAP index is therefore selected as the chattering performance criteria for use in MOO controller gains tuning of the ISMC_b controller. There are three objectives to be met for each of the six-DoFs of the quadrotor when tuning the ISMC_b, that is low trajectory tracking error, low control input and low chattering. The ITSE and IACI indices are therefore also used for the controller gains tuning. The ITSE index will minimise high errors that persist over time and the IACI will give the same weighting to the control inputs regardless of time. In general, the objectives are conflicting in that for a controller with low trajectory tracking errors, high chattering can be experienced as can be observed in Figures 4.2 and 4.4 for CSMC_1. For a controller with low control input, high errors can be experienced as can be observed in Figures 4.2 and 4.4 for CSMC_0.

The parameters shown in Table 5.1 were used for the MOPSO controller gains-tuning process. The parameter values are based on the ranges suggested by Coello et al., [50]. The recommended population size is between 20 to 80 particles and the recommended number of generations is between 80 to 120 generations. The maximum repository size determines the quality of the Pareto front produced [50].

For the NSGA-II, a low population size could lead to the algorithm becoming stuck at a local minima while a larger population size leads to an increase in computational time. To ensure that the same number of function evaluations is performed for each algorithm, the maximum number of generation for the NSGA-II differs from that of the MOPSO. A tournament selection function is used for the NSGA-II to select parents of crossover

and mutation children. The parameters shown in Table 5.2 were used for the NSGA-II controller gains-tuning process.

Table 5.1: MOPSO parameters.

Description	**Value**
Inertia weight	0.4
Individual confidence factor	2
Swarm confidence factor	2
Number of grids in each dimension	40
Maximum velocity in percentage	5
Uniform mutation percentage	0.5

Table 5.2: NSGA-II parameters.

Description	**Value**
Pareto fraction	0.5
Cross over fraction	0.8

For the ISMC_b controller there are five controller gains to be tuned for each of the six-DoFs of the quadrotor, resulting in a total of 30 controller gains to be tuned. Table 5.3 shows the maximum and minimum controller gains that are used for the MOO controller gains-tuning process. As the controller gains should be positive non-zero numbers, the minimum values for the gains are chosen to be a small value above zero. The maximum gains are chosen through trial and error. Initially very high maximum gain values are chosen and simulations performed with a random sample of gains. The results of this gave an indication of which gain values led to unstable outputs and thus suitable maximum gains can be selected.

Table 5.3: Controller gains limits for MOO gains tuning.

	Minimum	**Maximum**	
Gain	All	X, Y	Z, ϕ, θ, ψ
k_1	0.1	5	10
k_2	0.1	10	125
c_1	0.1	5	15
c_2	0.1	5	40
λ	0.1	0.5	0.5

5.5 Multi-Objective Optimisation Controller Gains Tuning Results

The number of particles obtained for the Pareto front from the controller gains tuning, as well as the time taken for each simulation are presented in Table 5.4 and shown in Figure 5.5. Overall, the MOPSO algorithm results in a higher number of particles on the Pareto fronts than those obtained from the MOGA controller gains-tuning method. The MOPSO algorithm also has much lower completion time for the same number of function evaluations than the MOGA. For the MOPSO algorithm, a larger initial population size leads to both a higher final number of particles on the Pareto front and a higher completion time. Likewise, a larger number of generations leads to a higher number of particles and higher completion times for the MOPSO algorithm. For the MOGA algorithm, a larger initial population size leads to very high completion times. A significant amount of time is spent generating the initial population, since the MOGA requires a large population to avoid getting stuck at a local minimum, and only gains that result in feasible outputs are selected for the initial population. A larger initial population leads to a higher final number of particles on the Pareto front for the MOGA method. For this set of simulations, for the MOGA algorithm, a higher number of generations leads to a slightly lower number of final particles on the Pareto front. The MOPSO controller gains-tuning method, overall, has better results than the MOGA method in terms of time taken and the size of the Pareto front produced.

Table 5.4: MOO controller gains tuning results.

MOO	**Population**	**Generations**	**Evaluations**	**Time (sec)**	**Final Size**
$MOPSO_1$	50	120	6000	5655.6	2128
$MOPSO_2$	50	80	4000	4495.3	1381
$MOPSO_3$	25	120	3000	2746.8	1126
$MOPSO_4$	25	80	2000	2097.3	910
$MOGA_1$	400	15	6000	17908.6	131
$MOGA_2$	400	10	4000	16050.9	154
$MOGA_3$	200	15	3000	8090.9	86
$MOGA_4$	200	10	2000	8043	100

Figures 5.6 to 5.11 show the Pareto fronts for the controller gains tuning using the MOPSO and NSGA-II methods. Only the simulation that provided the best result in terms of the size of Pareto front produced is shown for each controller gains-tuning method. As a total of 18 objectives (three for each of the quadrotor's six-DoFs) were optimised, each Pareto front is a hyper-surface. For ease of visualisation, each of the two Pareto fronts is split into six figures representing the quadrotor DoFs. For visual clarity, only a portion of each Pareto front is shown in the figures. As can be seen from the figures, tuning the controller gains using the MOPSO method results in a wider spread of particles on the

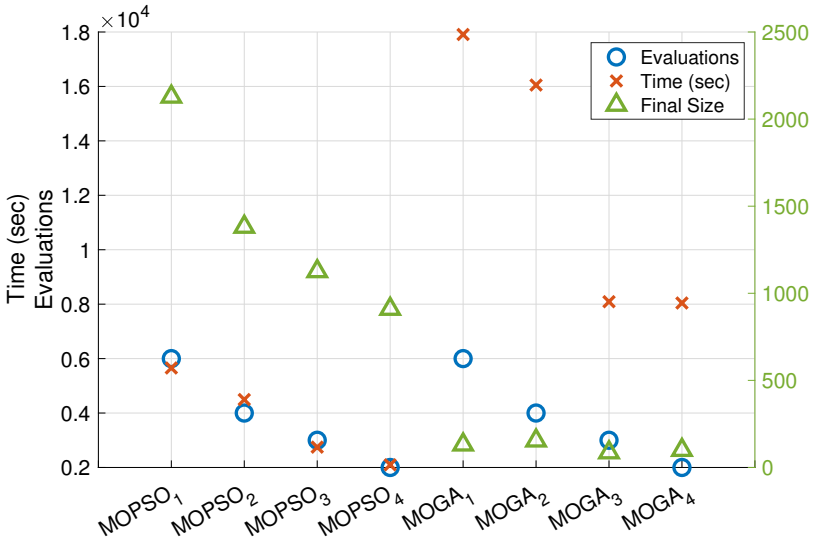


Figure 5.5: MOPSO and NSGA-II results.

Pareto front. The MOPSO formulation results in lower values for the chattering indices than the MOGA formulation as can be seen for the X, Y, and Z results.

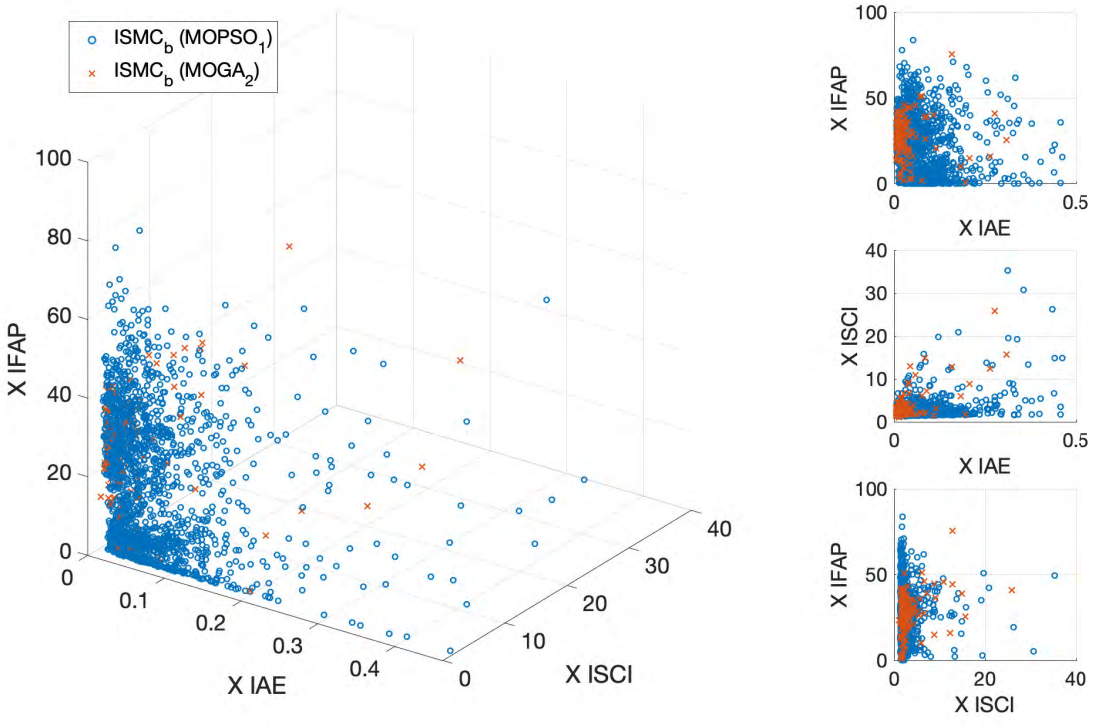


Figure 5.6: Portion of MOPSO and NSGA-II Pareto fronts - X.

The controller gains of all six-DoFs of the quadrotor were tuned simultaneously and the desired performance for all six-DoFs must be taken into consideration when selecting a solution from the Pareto front. Although similar figures were generated for all six-DoFs of the quadrotor, as an example, only the tuned height DoF controller gains obtained using

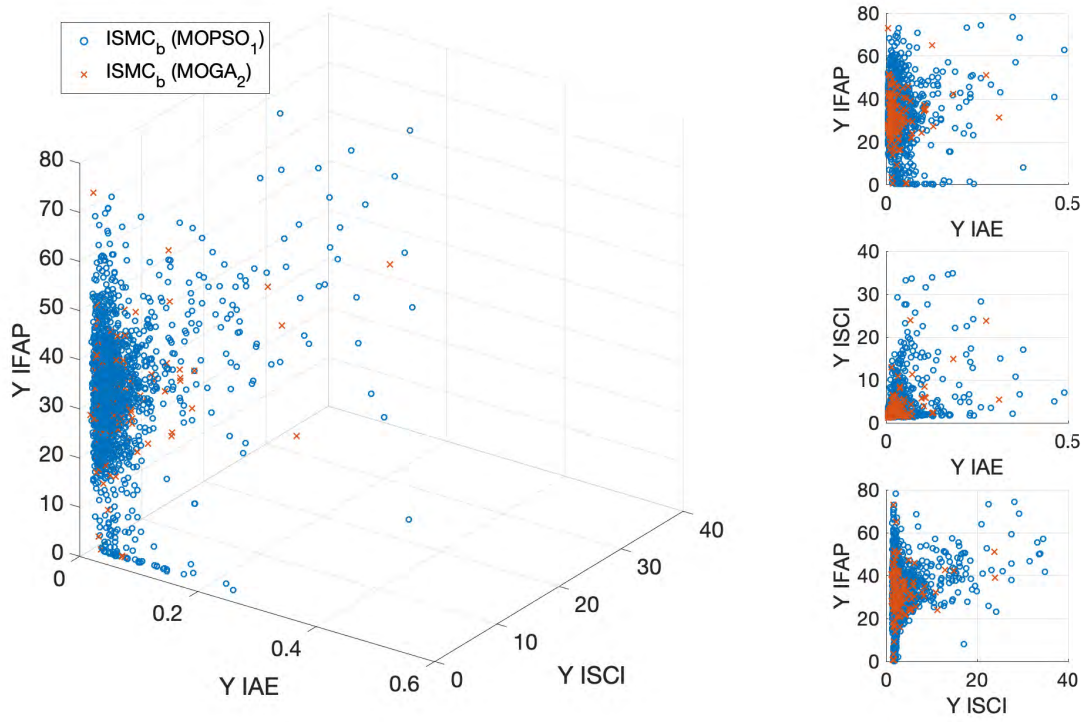


Figure 5.7: Portion of MOPSO and NSGA-II Pareto fronts - Y.

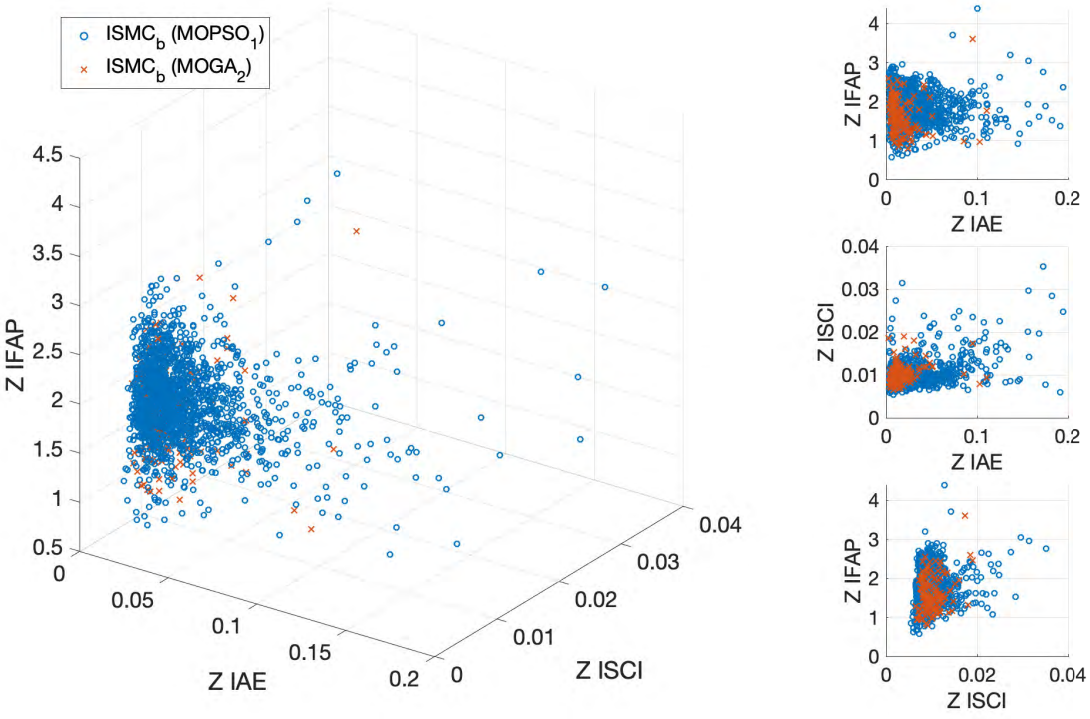


Figure 5.8: Portion of MOPSO and NSGA-II Pareto fronts - Z.

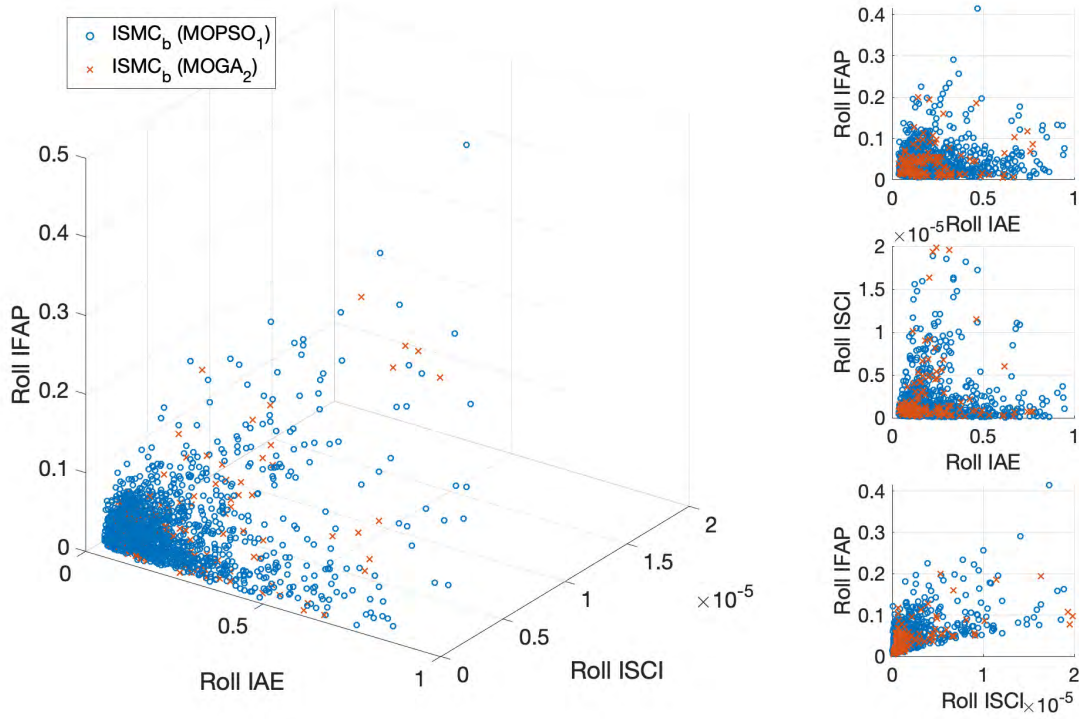


Figure 5.9: Portion of MOPSO and NSGA-II Pareto Fronts - Roll.

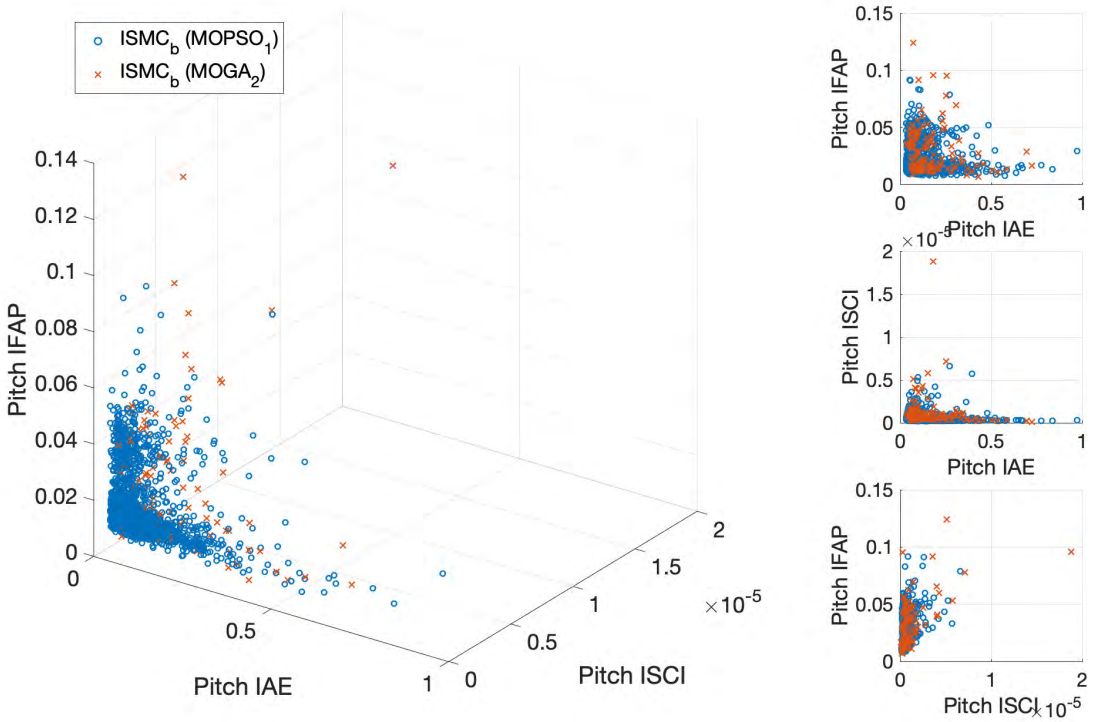


Figure 5.10: Portion of MOPSO and NSGA-II Pareto fronts - Pitch.

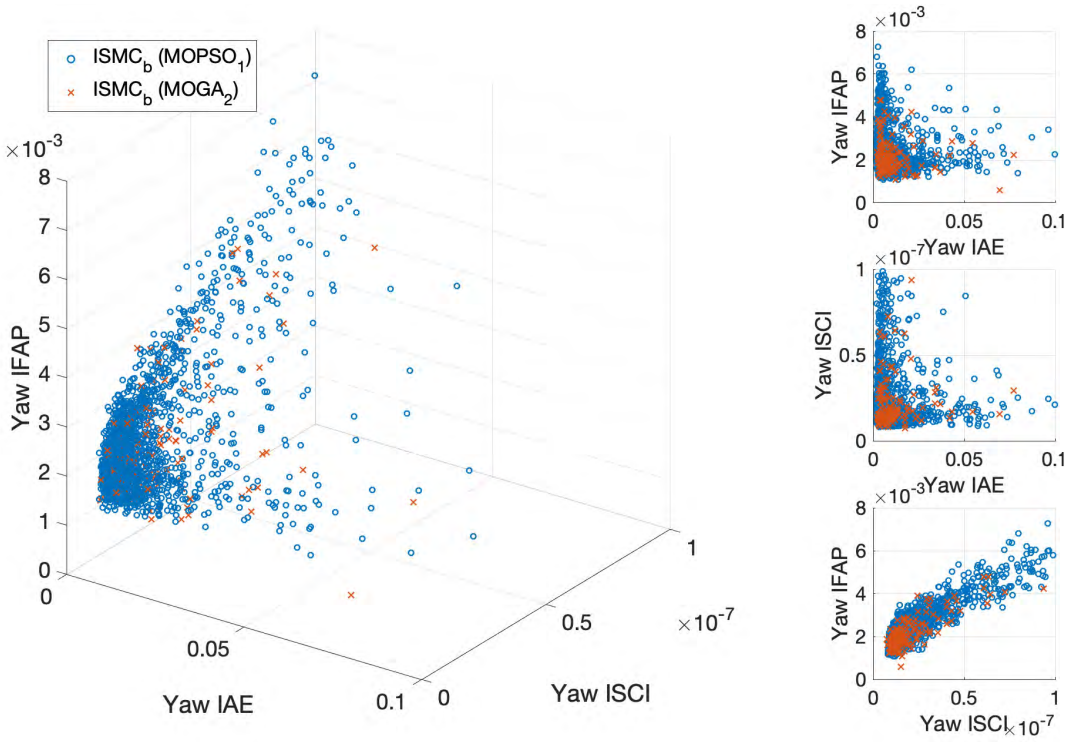


Figure 5.11: Portion of MOPSO and NSGA-II Pareto fronts - Yaw.

the MOPSO and NSGA-II algorithms are shown in Figure 5.12. The desired performance of the system can be selected by selecting a particle from the Pareto front with a particular set of values for the IFAP, IAE and ISCI indices for each DoF of the quadrotor, that results in the desired performance specified by the user. For example, a desired error range for each DoF could be selected using the IAE indices. A desired range of the control input indices for each DoF could then be selected using the ISCI indices. From this range, the particle resulting in the low chattering could then be chosen by selecting the particle with the low IFAP indices for each DoF. The controller gains for all six-DoFs corresponding to the selected particle would then be used for the quadrotor control.

5.6 Summary

In this section, two multi-objective metaheuristic computational intelligence methods, the MOPSO and NSGA-II methods were presented for the MOO controller gains tuning of SMCs, taking into account multiple conflicting objectives. The objectives to be minimised were chattering using the IFAP index, control input using the ISCI index, and trajectory tracking error using the IAE index. Quadrotor trajectory tracking simulations showed the effectiveness of the methods for obtaining optimised controller gains. Overall, the MOPSO method was shown to produce better Pareto fronts than the NSGA-II method when tuning a large number of controller gains and performance objectives. The NSGA-II method had significantly higher completion times compared to the MOPSO method for

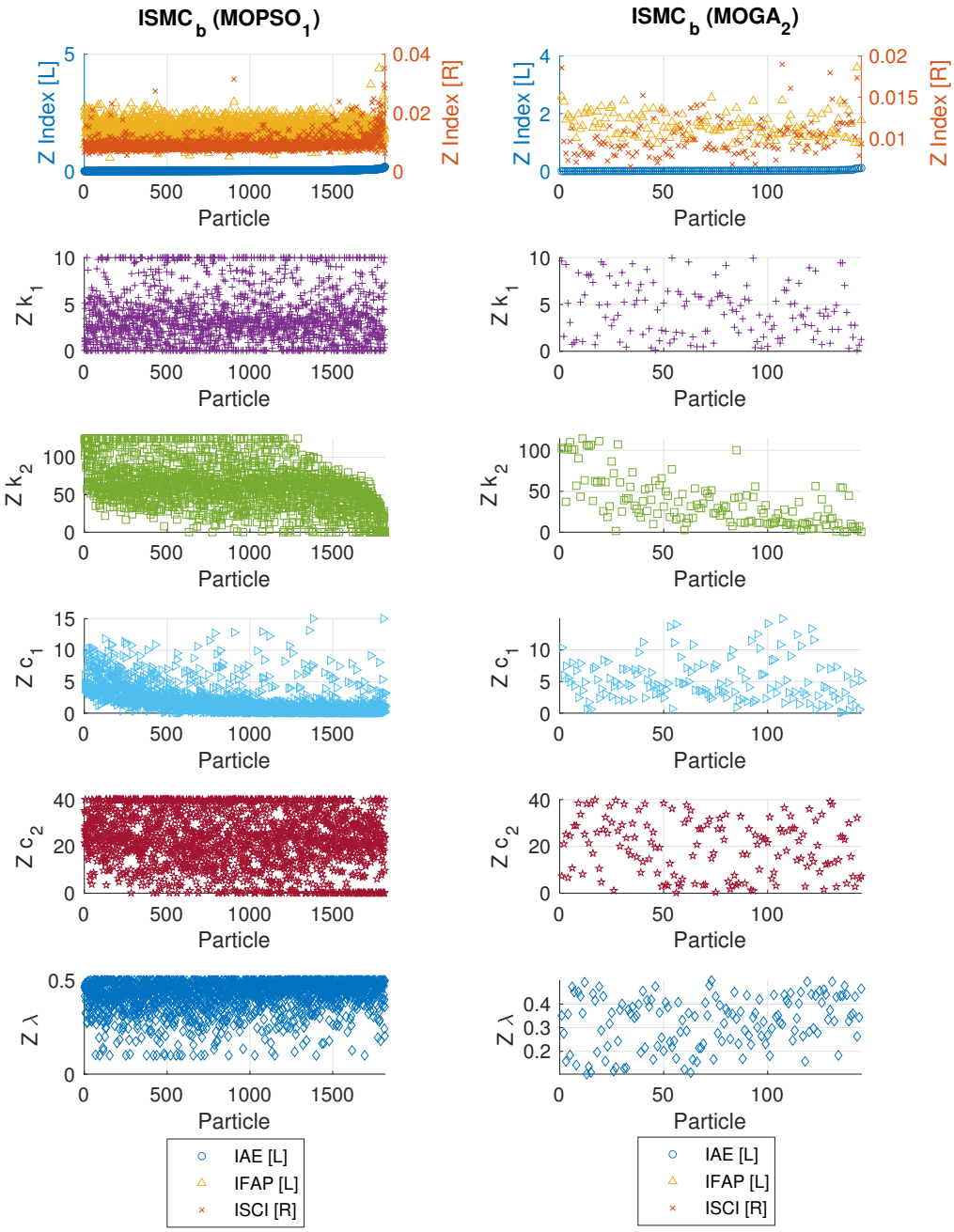


Figure 5.12: MOPSO and NSGA-II tuned gains - Z.

the same number of function evaluations. The final number of particles on the Pareto front for the NSGA-II method was, overall, much lower than that for the MOPSO method. The MOPSO method is therefore selected for controller gains tuning and comparison in Chapter 6 and Chapter 7.

6 Aerial Manipulator Farm Coverage and Plant Sample Collection FTC Simulations

Variations of SMC techniques have successfully been used for the FTC of quadrotors and can likewise be investigated for the FTC of quadrotor-based aerial manipulators. For precision agriculture remote sensing and plant sample collection, the aerial manipulator is required to cover and sense the entire area of the field. Commonly used field coverage path planning algorithms are therefore presented. In this chapter, several different SMC formulations are simulated for the passive and active FTC of quadrotor-based aerial manipulators. Variations of the controllers include the use of classical versus integral sliding surfaces, discontinuous versus continuous SMCs, gain adaptation using different adaptation laws, and first-order versus second-order SMC as presented in Chapter 3. The controller gains are tuned using the MOPSO controller gains-tuning method presented in Chapter 5 and the performance indices developed in Chapter 4. The results are used to compare the trajectory tracking, control input and chattering performance of the different passive and active FTCs. Finally, two controllers and their gains are selected for farm coverage and plant sample collection simulations.

6.1 Farm Coverage Path Planning

Coverage path planning involves planning the path to be followed by a ground or aerial robot in order to fully cover a specified area such as a farm [176]. Cabreira et al., [176] performed a survey on coverage path planning for aerial robots. They found that two commonly used coverage path planning patterns are the spiral and the back-and-forth coverage patterns. Cabreira et al., [177] developed an energy-aware spiral coverage planner that sets different optimal speeds for each line segment to improve energy usage. Figure 6.1 shows an example of a spiral coverage pattern for a hexagonal field. The quadotor takes off from a starting point and rises to a desired height. It then follows the first line in the spiral pattern and yaws at a vertex to follow the next line. Once the final vertex in the middle of the field is reached, the quadrotor returns to the starting point and lands. The return path passes some regions that have already been covered.

Di Franco and Buttazzo [178] developed a back-and forth coverage planner that aims at reducing energy consumption while achieving the desired image resolution for the remote sensing. They also improved the coverage planner by including a return path that allows the quadrotor to return to its starting position without passing regions of the field that have already been covered. Figure 6.2 shows an example of field coverage for a hexagonal shaped field using the back-and-forth coverage planner, including a return path. The path

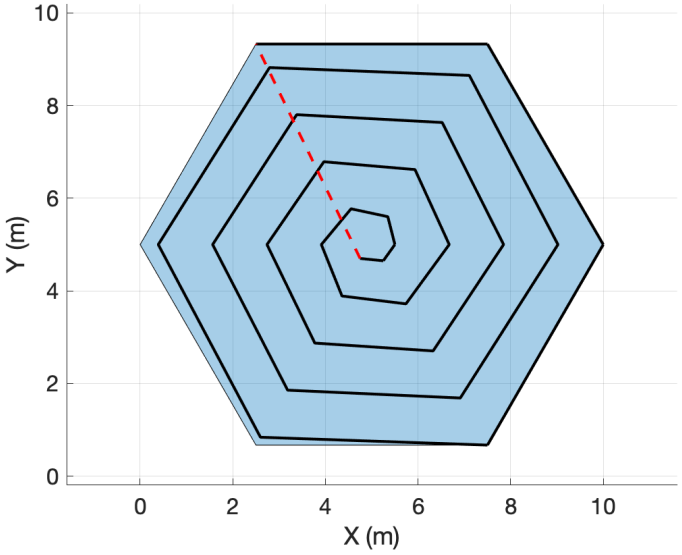


Figure 6.1: Field coverage path - spiral coverage.

consists of a series of long parallel lines connected by shorter lines. The distance between the long parallel lines is determined by the desired sensor resolution as well as ensuring that a return path can be generated. The quadrotor would start at its initial position and ascend to a desired height. The quadrotor would then move forward along the first long line in the pattern until it reaches a vertex. A yawing motion is then performed such that the quadrotor can go forward along the next short line in the pattern. The quadrotor then yaws and follows the next long line, parallel to the initial long line. This is repeated forming a back-and forth motion until the quadrotor follows the return path and reaches the starting point and lands.

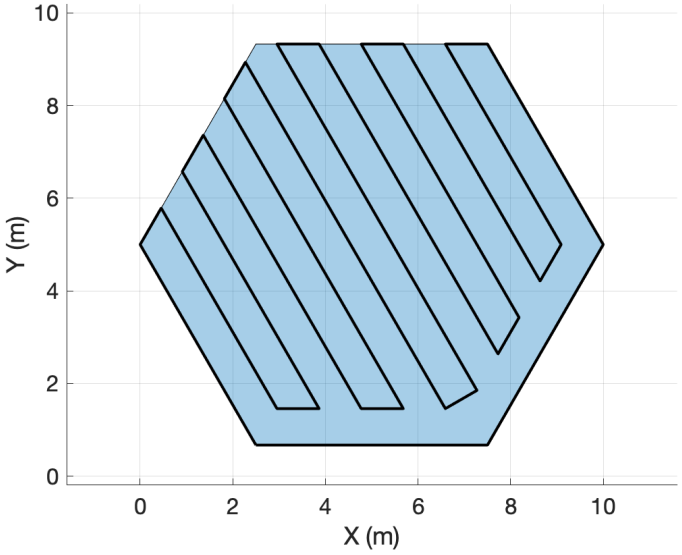


Figure 6.2: Field coverage path - back-and-forth coverage with return path.

6.2 Controller Gains Tuning Simulations

For the controller gains-tuning, the larger quadrotor-based aerial manipulator with a two-DoF manipulator is simulated in Matlab. The aim of the simulations is to obtain optimum gains for each controller and to use the results to compare the performance of the controllers. The desired path used to compare the controllers is shown in Figure 6.3 and the generated trajectories are shown in Figure 6.4. For both the spiral and back-and-forth area coverage paths, the four main motions are: taking off to a desired height, moving forwards along a straight line, yawing, and landing. For the aerial manipulator, hovering with the manipulator in motion is also required. The aerial manipulator position manoeuvres are chosen in order to have position tracking under normal operation as well as position tracking in the presence of an actuator fault. There are therefore several sections to the path chosen for the controller gains-tuning process. The aerial manipulator system first rises to a desired height and then tracks a desired trajectory from an initial position to a specified intermediate position along the X-axis. For the sample collection phase it is assumed that the location of the sample to be collected is known and is at the specified intermediate position. The aerial manipulator will therefore hover directly above the sample location for a brief period of time of 5.6 seconds and the manipulator subsystem will extend down to reach the desired sample location and then retract to store the plant sample. The mass of the plant sample (for example a leaf) collected is considered negligible when compared to the aerial manipulator mass. The aerial manipulator then moves to a final X position and performs a yaw motion. An actuator fault is applied after a short time while the aerial manipulator is hovering. The system then tracks a desired trajectory from an initial position to a specified intermediate position along the Y-axis. The aerial manipulator hovers at the intermediate position for a brief period of time while the manipulator subsystem performs an extension and retraction motion to collect and store another plant sample. The system then moves to a final Y position and performs a yaw motion, after which the system lands. This trajectory is therefore a shorter version of a farm coverage path, but contains all elements required for a coverage path and for plant sample collection.

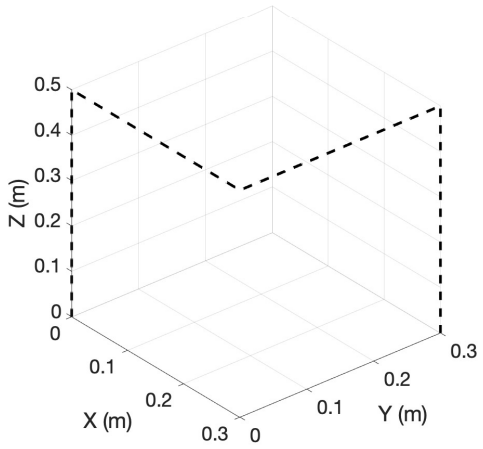


Figure 6.3: Desired path.

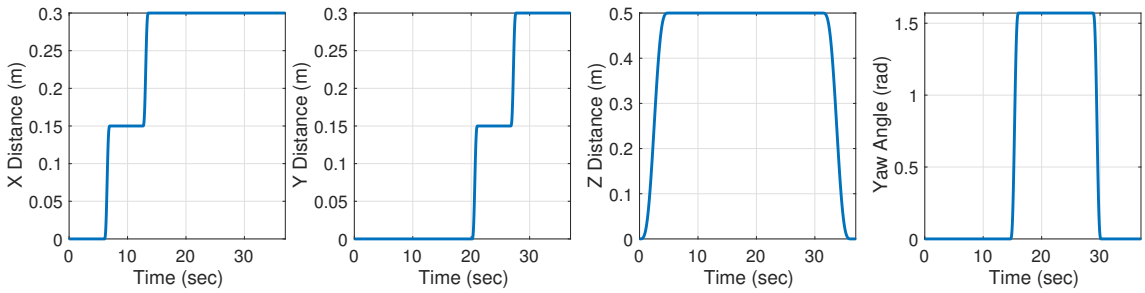


Figure 6.4: Desired trajectories.

For the extension and retraction of the manipulator, the manipulator end-effector has to move from its rest position and reach a desired final position with the end-effector following a specified end-effector trajectory from the initial to the final position. The manipulator joint angles obtained from the closed-loop inverse kinematics presented in Chapter 3 are shown in Figure 6.5. The initial and final manipulator configurations are shown in Figure 6.6.

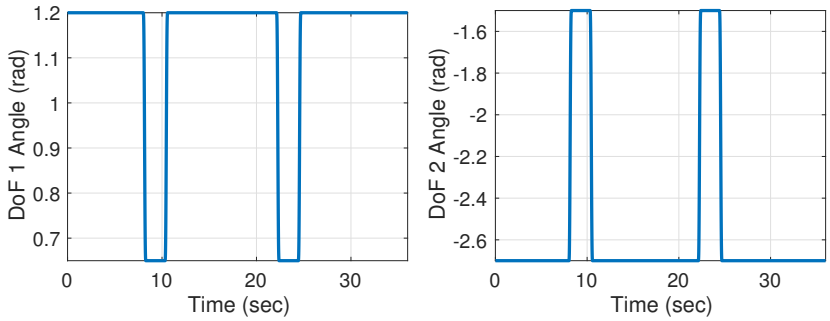
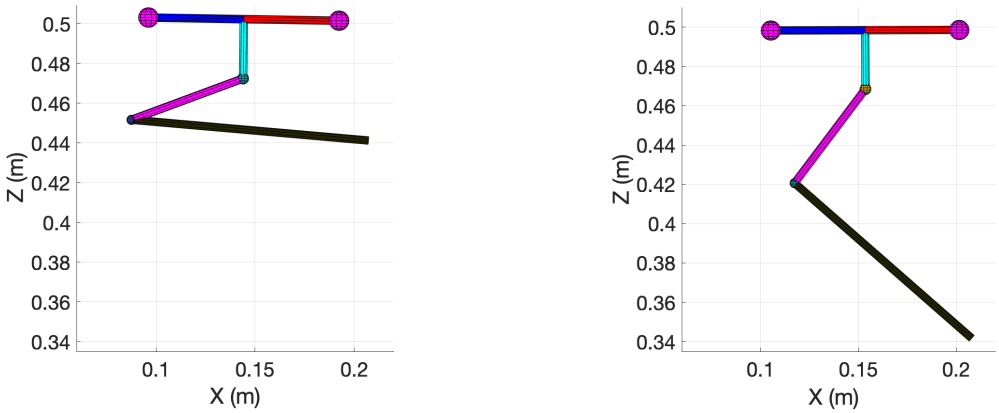


Figure 6.5: Manipulator joint angles.



(a) Manipulator initial state.

(b) Manipulator final state.

Figure 6.6: Manipulator states for a quadrotor-based aerial manipulator with a two-DoF manipulator.

To determine the FTC capabilities of the passive and active fault-tolerant SMCs for the aerial manipulator, an abrupt fault is added to rotor 3. For the 'cross' configuration of the quadrotor subsystem, a fault in one of the rotors affects each DoF of the quadrotor

subsystem, thus the simulations are conducted with only a single rotor fault. The fault magnitude is a loss of 10% of the rotor's effectiveness as shown in Figure 6.7.

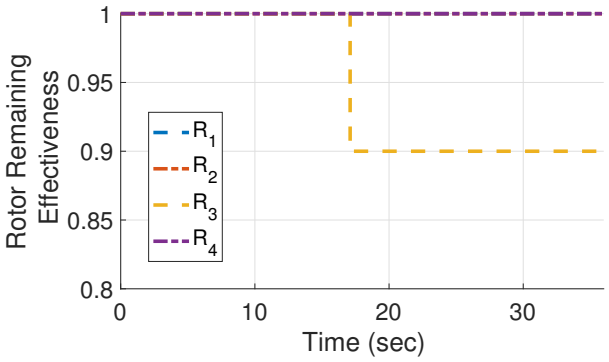


Figure 6.7: Rotor remaining effectiveness.

The MOPSO parameters used for the controller gains tuning are the same as those used in Chapter 5 and are shown in Table 5.1. The remaining parameters are shown in Table 6.1. Each controller is given the same population size, however the maximum number of generations is dependent on the number of controller gains being tuned per DoF of the controller. More generations are given to controllers with a larger number of tuned controller gains, to allow for a good Pareto front to be obtained.

Table 6.1: MOPSO parameters - aerial manipulator gains tuning.

Description	**Value**
Population size	20
Maximum number of generations - 3-DoF	80
Maximum number of generations - 4-DoF	90
Maximum number of generations - 5-DoF	100
Maximum number of generations - 6-DoF	110
Maximum number of generations - 7-DoF	120
Maximum number of generations - 8-DoF	130
Maximum repository size	3000

To tune the controller gains using the MOPSO method detailed in Chapter 5, four indices are used as the objectives to be optimised. Two separate IAE indices, one for the fault-free portion of the trajectory (labelled as IAE_n) and one for the faulty portion of the trajectory (labelled as IAE_f) are used. This allows for controllers to be tuned such that the trajectory tracking error performance in both the fault-free and faulty scenarios can be optimised and compared. The IAE is selected to give the same weight to errors regardless of time. The IACI index computed over the entire trajectory is used as the control input index. The IFAP is used for the chattering index and is computed for frequencies greater than or equal to 10% of the maximum frequency. Given the coupled nature of the

system dynamics, all DoFs of the quadrotor subsystem are tuned simultaneously using the controller gains limits given in Tables 6.2 and 6.3.

Table 6.2: Controller gains tuning maximum limits - first-order SMCs.

Controllers	Gain	X	Y	Z	ϕ	θ	ψ
CSMC, ISMC	k_1	1	1	4	4	4	4
ISMC_h, ISMC_b	k_1	5	5	20	20	20	20
All	k_2	10	10	20	70	70	70
	c_1	10	10	10	20	20	20
ISMC_h, ISMC_b, AISMC_ph, AISMC_rh, AISMC_p, AISMC_r	c_2	10	10	10	10	10	10
ISMC_h, ISMC_b, AISMC_ph, AISMC_rh	λ	0.5	0.5	0.5	0.5	0.5	0.5
AISMC_p	α_i	2.5	2.5	10	10	10	10
AISMC_ph	α_i	5	5	20	20	20	20
AISMC_p, AISMC_ph	η_α	0.1	0.1	0.1	0.1	0.1	0.1
	μ	1	1	1	1	1	1
AISMC_r	r_{0i}	2.5	2.5	10	10	10	10
	r_{1i}	2.5	2.5	10	10	10	10
AISMC_rh	r_{0i}	5	5	20	20	20	20
	r_{1i}	5	5	20	20	20	20
AISMC_r, AISMC_rh	ρ_0	40	40	40	40	40	40
	ρ_1	40	40	40	40	40	40

Table 6.3: Controller gains tuning maximum limits - STSMCs.

Controllers	Gain	X	Y	Z	ϕ	θ	ψ
CSTSMC	k_1	5	5	10	25	25	25
	k_2	15	15	30	60	60	60
CSTSMC, ACSTSMC_t	c_1	15	15	50	30	30	30
ACSTSMC_t	η_α	0.1	0.1	0.1	0.1	0.1	0.1
	η_β	5	5	5	5	5	5
	μ	1	1	1	1	1	1
	α_i	5	5	10	25	25	25

6.3 Passive Fault-Tolerant Control Results

Passive sliding mode FTC makes use of the robust properties of SMCs to mitigate the effects of faulty rotors of the quadrotor subsystem of the aerial manipulator. This section compares the MOPSO controller gain-tuning results for several passive SMCs for the short coverage and sample collection trajectory, with a rotor fault as described in Section 6.2.

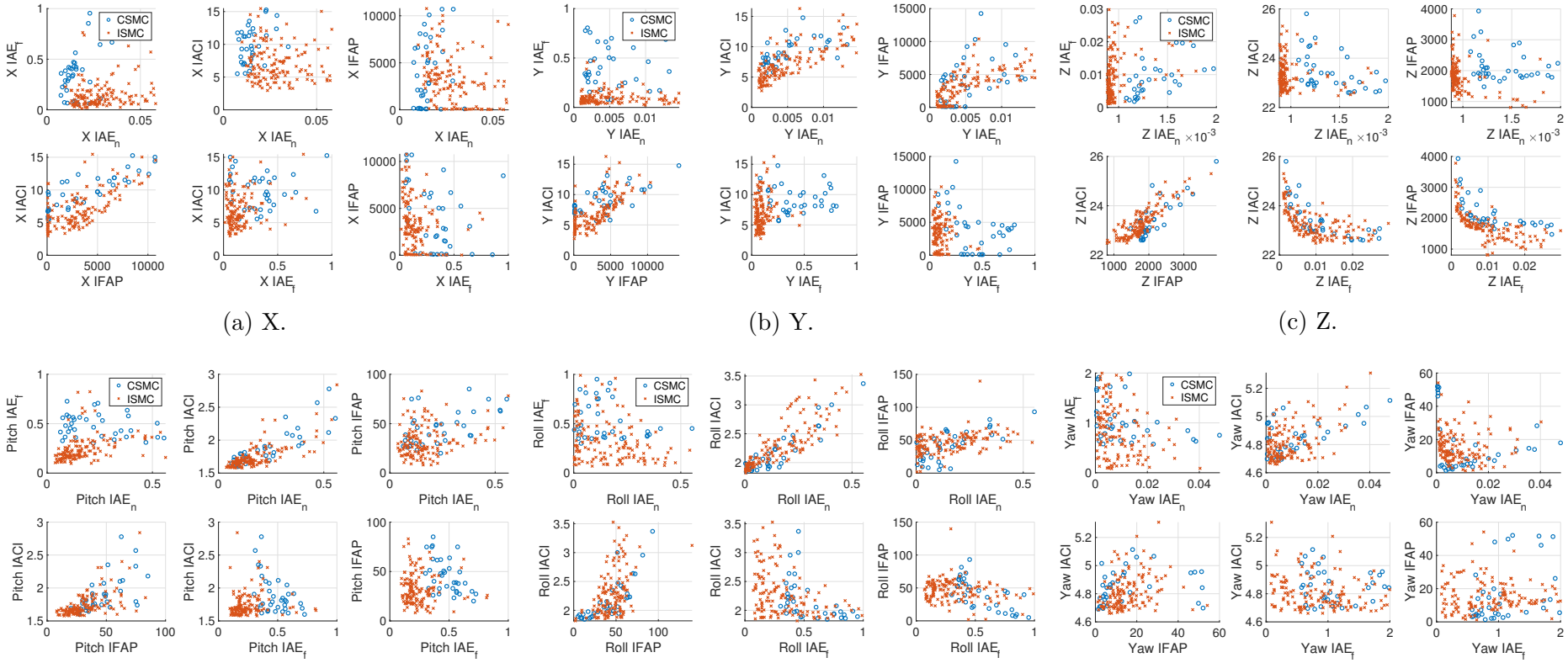
6.3.1 First-order sliding mode control results

First, the CSMC and the ISMC controller gains are tuned using the MOPSO method. The resulting Pareto fronts are shown in Figure 6.8. For the sake of better visual clarity, only a portion of each Pareto front is shown. This is done by limiting the maximum IAE_n and IAE_f indices for each DoF. As can be seen on the sub-figures, the ISMC results in lower IAE_f indices for all of the DoFs, showing that it is better than the CSMC at decreasing the trajectory tracking errors caused by rotor loss of effectiveness faults. The two controllers have similar chattering indices. The ISMC in some cases has lower IACI indices, thus it can have lower control inputs than the CSMC.

Since the ISMC produces better results than the CSMC, it is chosen for further investigation to determine the effects of the discontinuous versus the continuous reaching laws on the controller. Two controllers, one making use of the hyperbolic tangent function ISMC_h, and another making use of the saturation function ISMC_b, as the continuous approximation of the discontinuous signum function, are investigated. Controllers making use of a continuous approximation of the discontinuous signum function are termed 'continuous controllers' in this thesis. The resulting Pareto fronts are shown in Figure 6.9. Both the ISMC_h and ISMC_b are able to reduce the chattering experienced by the ISMC, as evidenced by the sub-figures showing the IFAP indices. There does not appear to be a significant difference between the results of the ISMC_h and ISMC_b.

6.3.2 Adaptive sliding mode control results

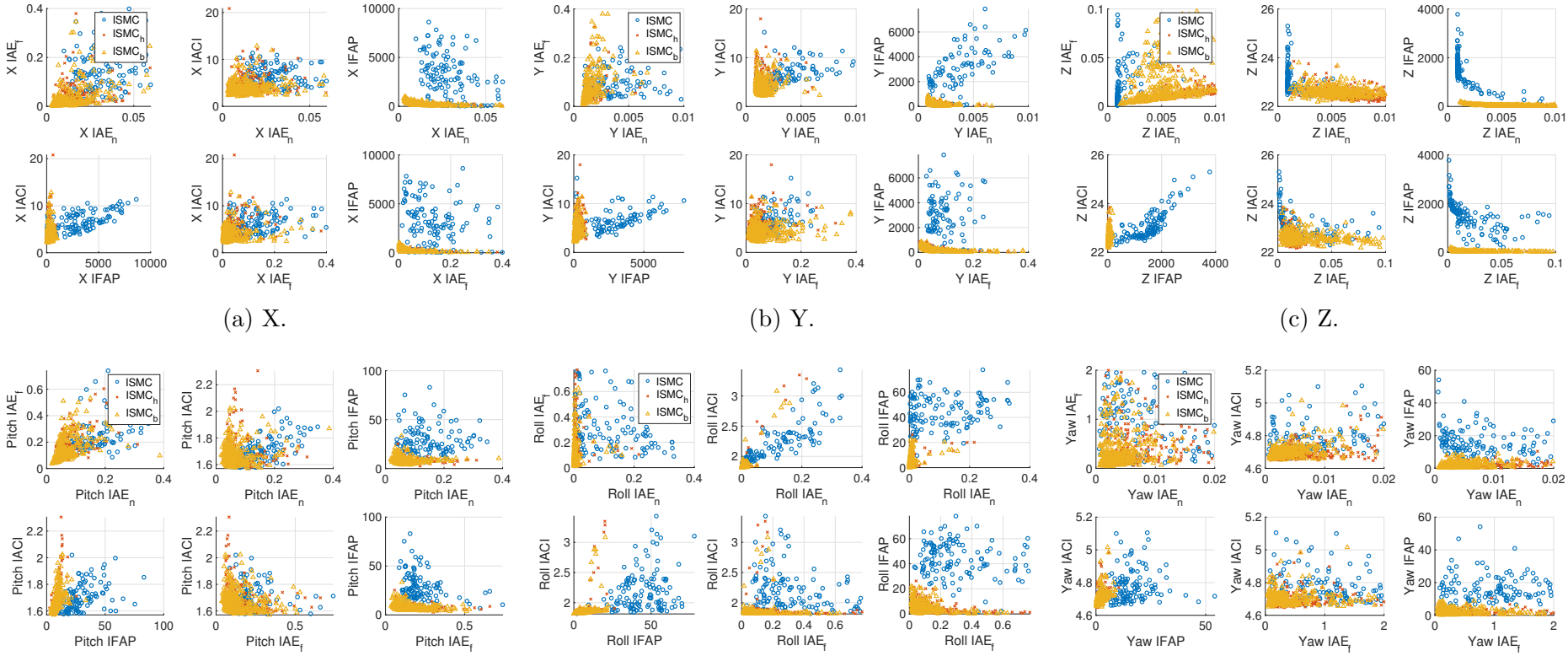
Figure 6.10 shows the Pareto fronts for the AISMCs where the α gains are adapted using the Plestan method (AISMC_p) and the Roy method (AISMC_r). For the Z, roll, pitch, and yaw DoFs, the adaptive controllers significantly reduce the control inputs experienced by the ISMC as shown by the sub-figures with the IACI indices. The AISMC_r is in some cases able to reduce chattering as shown by the IFAP indices for the roll and yaw DoFs. Although the AISMC_p is capable of producing low chattering, in some cases it can also produce higher chattering than the ISMC as shown by the roll, pitch and yaw DoFs which produce both lower and higher IACI indices than the ISMC.



(a) X. (b) Y. (c) Z.

(d) Pitch. (e) Roll. (f) Yaw.

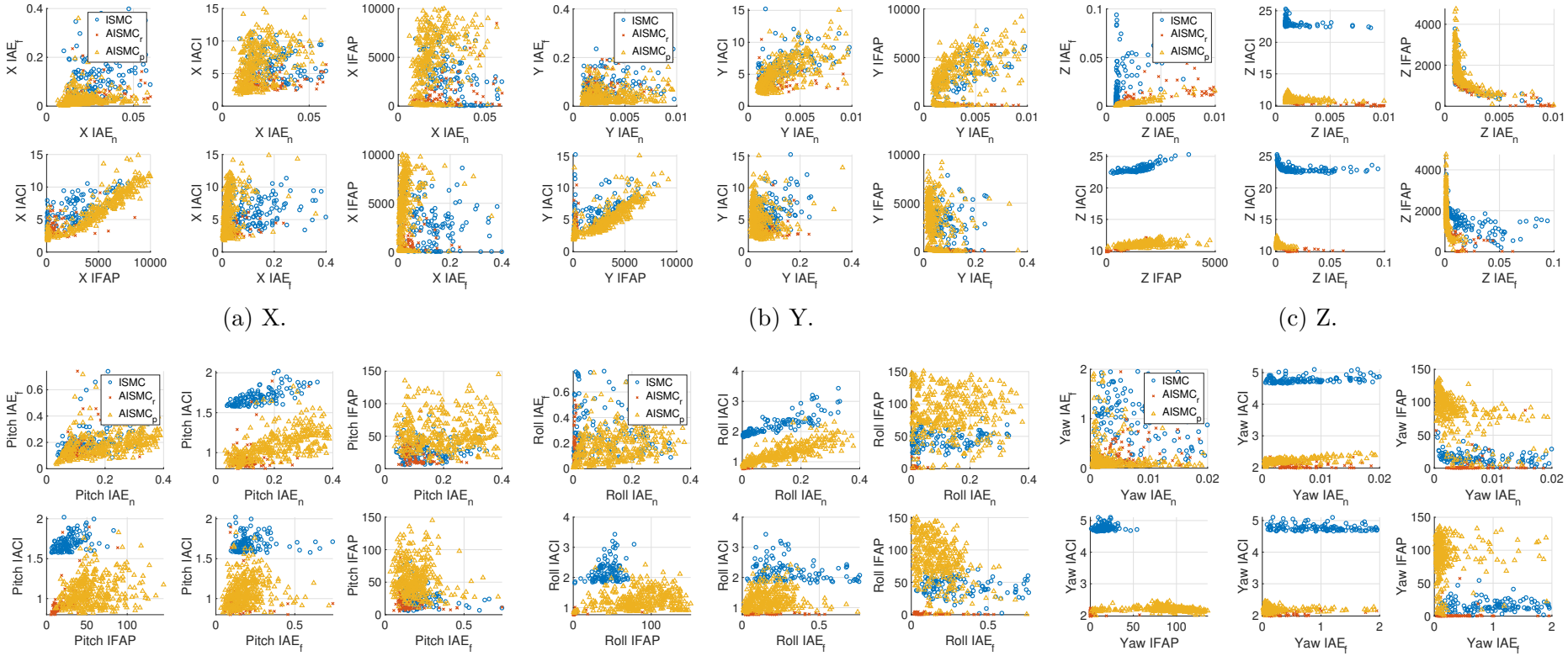
Figure 6.8: MOPSO Pareto fronts - CSMC and ISMC. Tuned gains are: CSMC - k_1, k_2, c_1 and ISMC - k_1, k_2, c_1, c_2.



(a) X. (b) Y. (c) Z.

(d) Pitch. (e) Roll. (f) Yaw.

Figure 6.9: MOPSO Pareto fronts - ISMC, ISMC$_h$ and ISMC$_b$. Tuned gains are: ISMC - k_1, k_2, c_1, ISMC$_h$ and ISMC$_b$ - k_1, k_2, c_1, c_2, λ.



(a) X. (b) Y. (c) Z.

(d) Pitch. (e) Roll. (f) Yaw.

Figure 6.10: MOPSO Pareto fronts - ISMC, AISMC$_\mathrm{r}$ and AISMC$_\mathrm{p}$. Tuned gains are: ISMC - k_1, k_2, c_1, AISMC$_\mathrm{r}$ - k_2, c_1, c_2, ρ_0, ρ_1, r_{0i}, r_{1i} , and AISMC$_\mathrm{p}$ - k_2, c_1, c_2, η_α, μ, α_i.

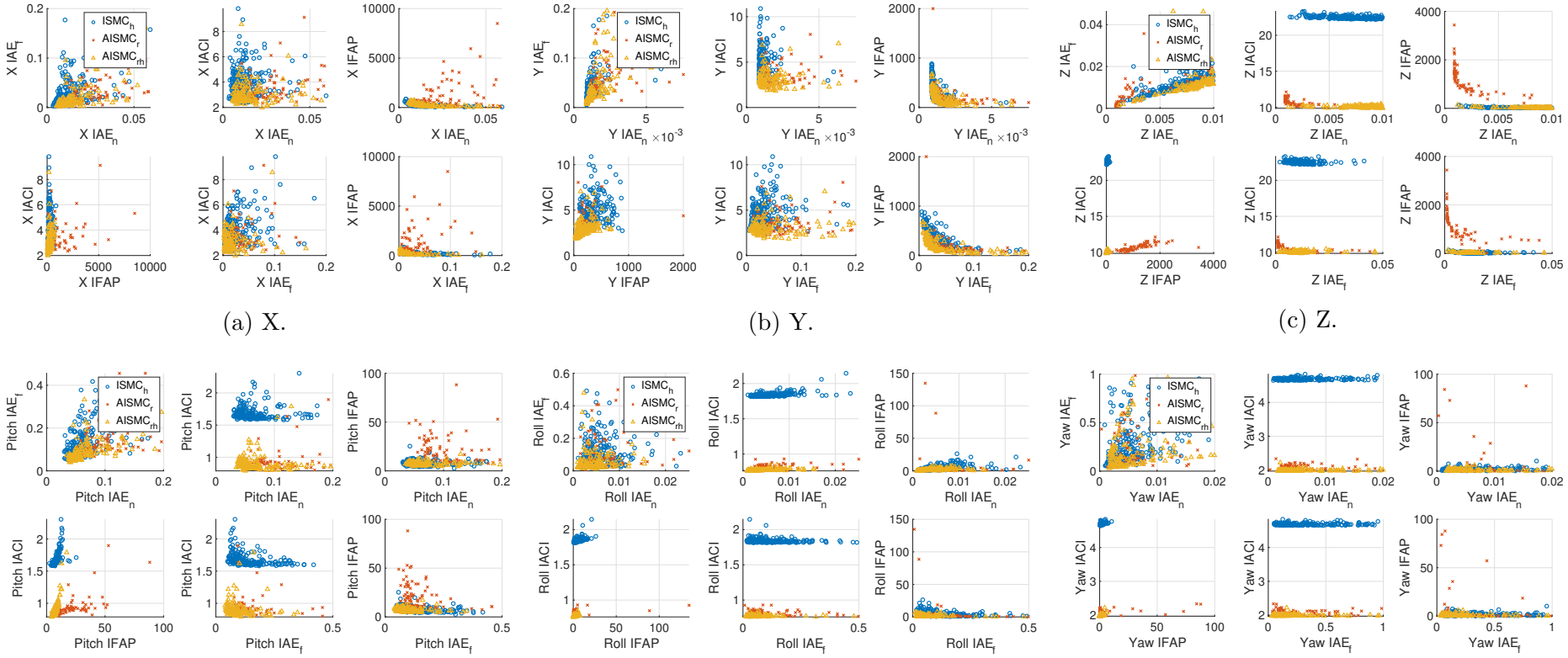
From the previous results, it can be seen that the continuous SMCs are able to significantly reduce chattering, whereas the adaptive SMCs are able to significantly reduce the control input. A combination of adaptive and continuous controllers is therefore investigated to determine if both of the benefits of the continuous and the adaptive SMCs can be achieved. There is no significant difference between the use of the hyperbolic tangent function and the saturation function for the continuous controllers. The hyperbolic tangent function is simply chosen for this set of experiments.

Figure 6.11 shows the Pareto fronts for the continuous, adaptive SMC, where the α gains are adapted using Roy adaptation method, AISMC_{rh}. This is compared against the continuous controller, ISMC_{h} and the discontinuous adaptive controller, AISMC_{r} to determine if the AISMC_{rh} combines the benefits of both those controllers. The AISMC_{rh} is able to reduce the control inputs experienced by the ISMC_{h} as can be seen most evidently by the sub-figures with IACI indices for the Z, roll, pitch and yaw DoFs. The AISMC_{rh} is able to reduce the chattering experienced by the AISMC_{r}. This is especially evident from sub-figures showing the IFAP indices for the X, Z, roll, pitch and yaw DoFs.

Figure 6.12 shows the Pareto fronts for the continuous controller, AISMC_{ph}, where the α gains are adapted using the Plestan adaptation method. This is compared against the continuous controller, ISMC_{h} and the discontinuous adaptive controller, AISMC_{p}. The AISMC_{ph} reduces the control inputs experienced by the ISMC_{h} as can be seen by the sub-figures showing the IACI indices for the Z, roll, pitch and yaw DoFs. The AISMC_{ph} reduces the chattering experienced by the AISMC_{p} as can be seen from sub-figures of the IFAP indices for the X, Z, roll, pitch and yaw DoFs, although it can still experience slightly higher chattering than the ISMC_{h} as can be seen from the IFAP indices for the roll, pitch and yaw DoFs.

6.3.3 Super-twisting sliding mode control results

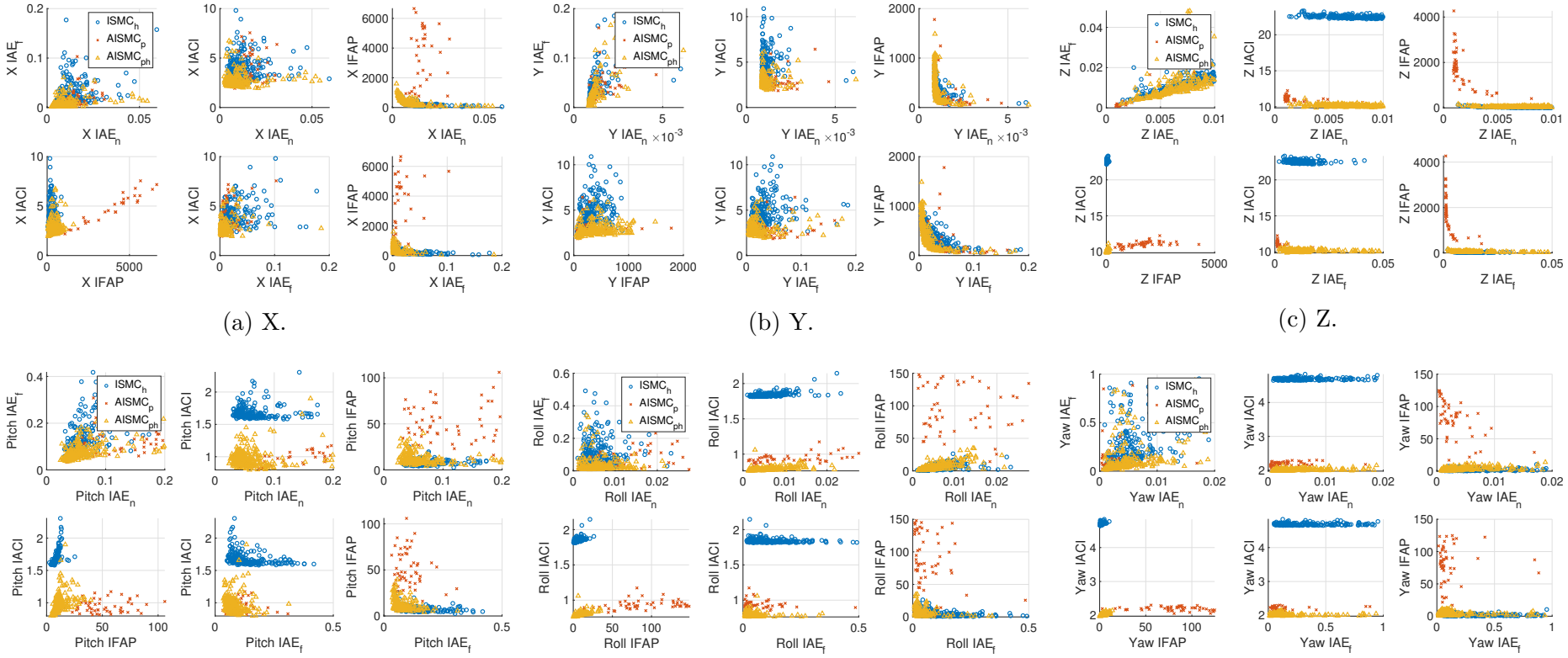
Two second-order SMCs, the CSTSMC and the adaptive second-order controller, $\text{ACSTSMC}_{\text{t}}$, making use of the Shtessel adaptation method, are simulated and compared against the ISMC to determine whether they can reduce the chattering experienced by the first-order ISMC. Figure 6.13 shows the Pareto fronts for the controllers. With the exception of the yaw DoF, the CSTSMC and $\text{ACSTSMC}_{\text{t}}$ are able to reduce chattering as shown by sub-figures for the IFAP indices. The CSTSMC and $\text{ACSTSMC}_{\text{t}}$, however, increase the control input as can be seen by the the sub-figures for the IACI indices for the Z, roll, pitch and yaw DoFs.



(a) X. (b) Y. (c) Z.

(d) Pitch. (e) Roll. (f) Yaw.

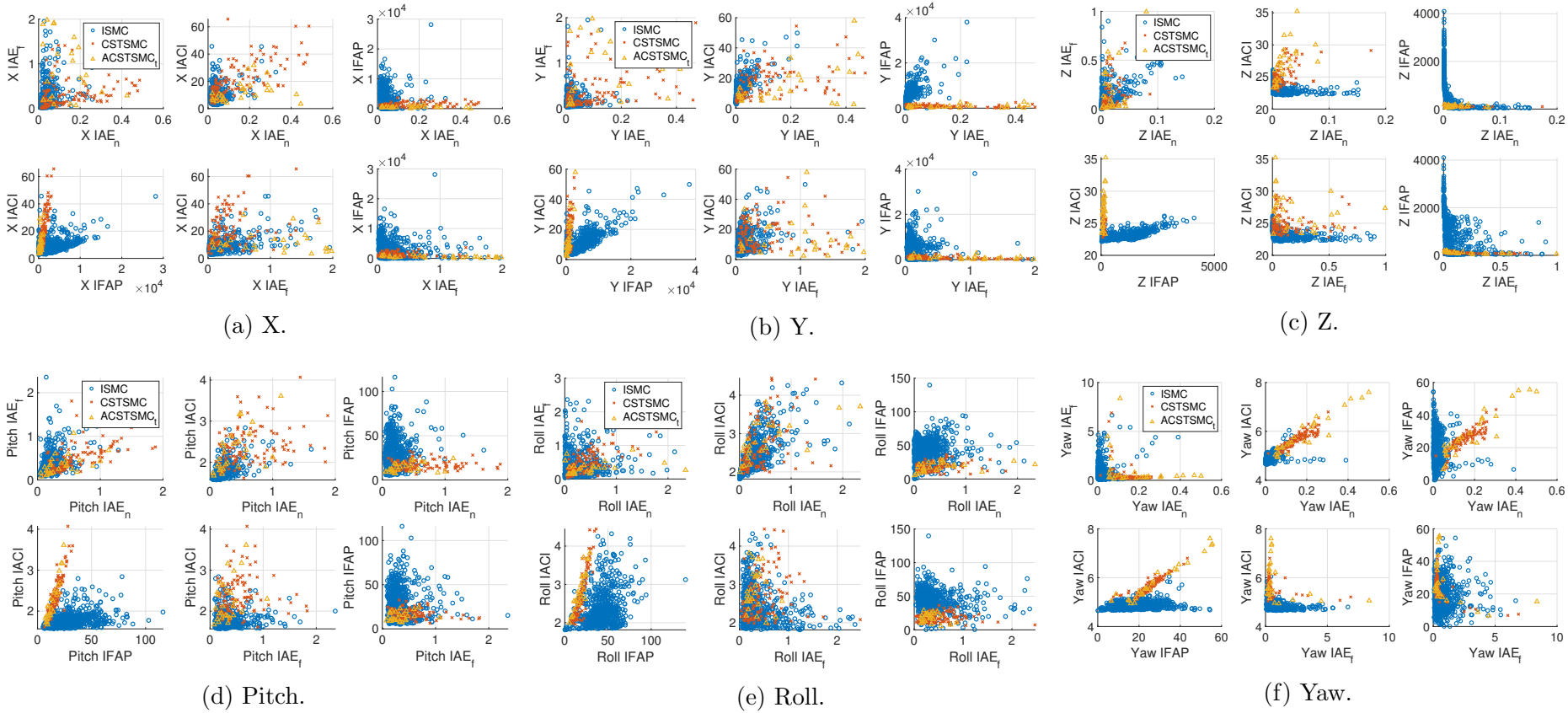
Figure 6.11: MOPSO Pareto fronts - ISMC_h , AISMC_r and AISMC_rh. Tuned gains are: ISMC_h - k_1, k_2, c_1, λ, AISMC_r - k_2, c_1, c_2, ρ_0, ρ_1, r_{0i}, r_{1i}, and AISMC_rh - k_2, c_1, c_2, ρ_0, ρ_1, r_{0i}, r_{1i}, λ.



(a) X. (b) Y. (c) Z.

(d) Pitch. (e) Roll. (f) Yaw.

Figure 6.12: MOPSO Pareto fronts - ISMC_h, AISMC_p and AISMC_ph. Tuned gains are: ISMC_h, AISMC_p - k_1, k_2, c_1, c_2, η_α, μ, α_i and AISMC_ph - k_2, c_1, c_2, η_α, μ, α_i, λ.



(a) X. (b) Y. (c) Z.

(d) Pitch. (e) Roll. (f) Yaw.

Figure 6.13: MOPSO Pareto fronts - CSTSMC and ACSTSMC$_t$. Tuned gains are: CSTSMC - k_1, k_2, c_1 and ACSTSMC$_t$ - c_1, η_α, μ, η_β, α_i.

6.4 Active Fault-Tolerant Control Gains Tuning Simulation Results

Active FTC uses a FDD unit to determine the occurrence of a fault and to estimate the magnitude of the fault. This information is used by the controller to mitigate the fault. For the controller gains-tuning simulations in this section, it is assumed that a perfectly accurate FDD unit is available.

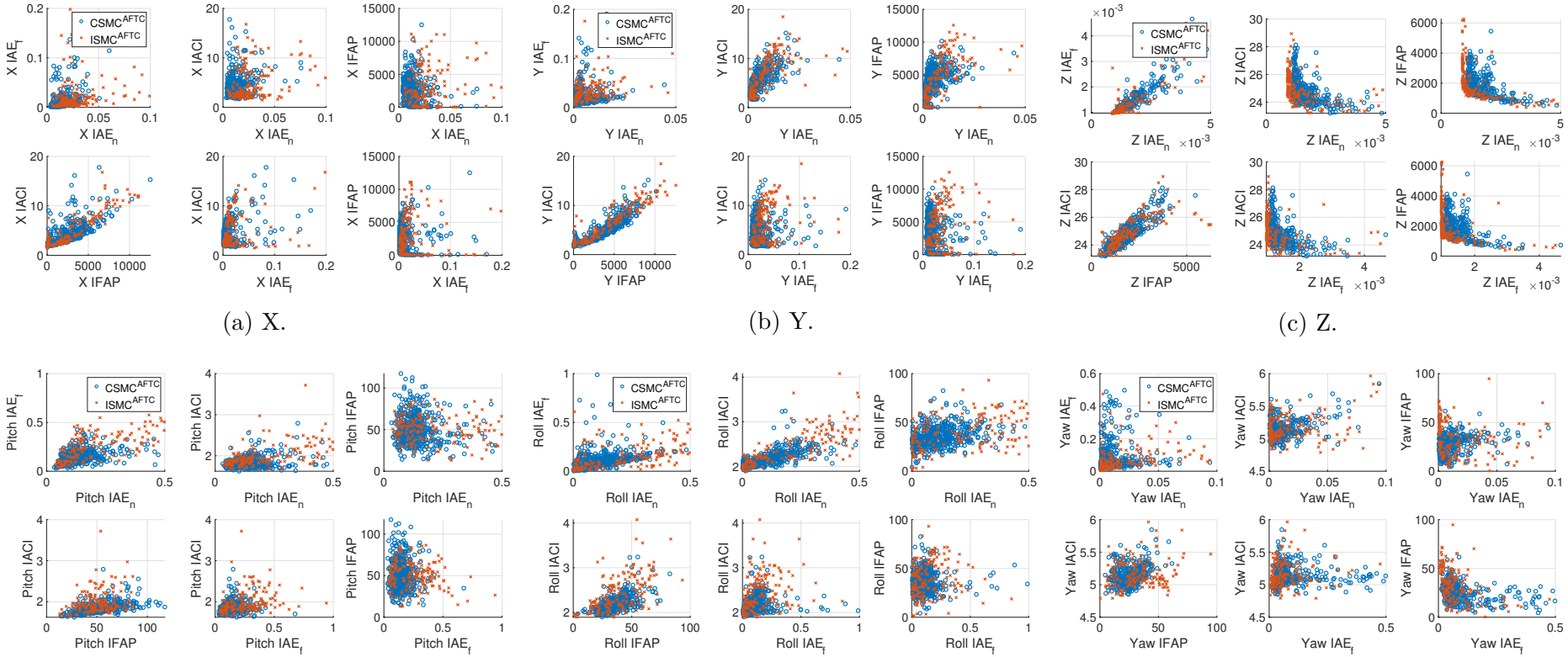
6.4.1 Active first-order sliding mode control results

The controller gains-tuning results for the active classical controller, $\text{CSMC}^{\text{AFTC}}$ and active integral controller, $\text{ISMC}^{\text{AFTC}}$, assuming the availability of an accurate FDD unit are shown in Figure 6.14. From the Z DoF results, it can be seen that the $\text{ISMC}^{\text{AFTC}}$ is capable of producing lower errors than the $\text{CSMC}^{\text{AFTC}}$, as is shown by the sub-figures for the IAE_{n} and IAE_{f} indices. For the X DoF, the IAE_{n} index is able to achieve slightly lower results for the $\text{CSMC}^{\text{AFTC}}$. There are no other significant differences between the two controllers.

As the $\text{ISMC}^{\text{AFTC}}$ produces better results than the $\text{CSMC}^{\text{AFTC}}$ for the IAE_{n} and IAE_{f} indices of the Z DoF, with only minor differences for the other DoFs, the $\text{ISMC}^{\text{AFTC}}$ is chosen for further investigation. The discontinuous $\text{ISMC}^{\text{AFTC}}$ is compared against the continuous $\text{ISMC}_{\text{h}}^{\text{AFTC}}$ which makes use of the hyperbolic tangent, and the $\text{ISMC}_{\text{b}}^{\text{AFTC}}$ which makes use of the saturation function as the continuous approximation of the discontinuous signum function. The results of the Pareto fronts are shown in Figure 6.15. Both the $\text{ISMC}_{\text{h}}^{\text{AFTC}}$ and $\text{ISMC}_{\text{b}}^{\text{AFTC}}$ are able to reduce the chattering experienced by the discontinuous $\text{ISMC}^{\text{AFTC}}$ as can be seen from the sub-figures for the IFAP indices. The Z, roll, and yaw DoFs also show a reduction in the control input for both the $\text{ISMC}_{\text{h}}^{\text{AFTC}}$ and $\text{ISMC}_{\text{b}}^{\text{AFTC}}$, as can be seen from the sub-figures for the IACI indices.

6.4.2 Active adaptive first-order sliding mode control results

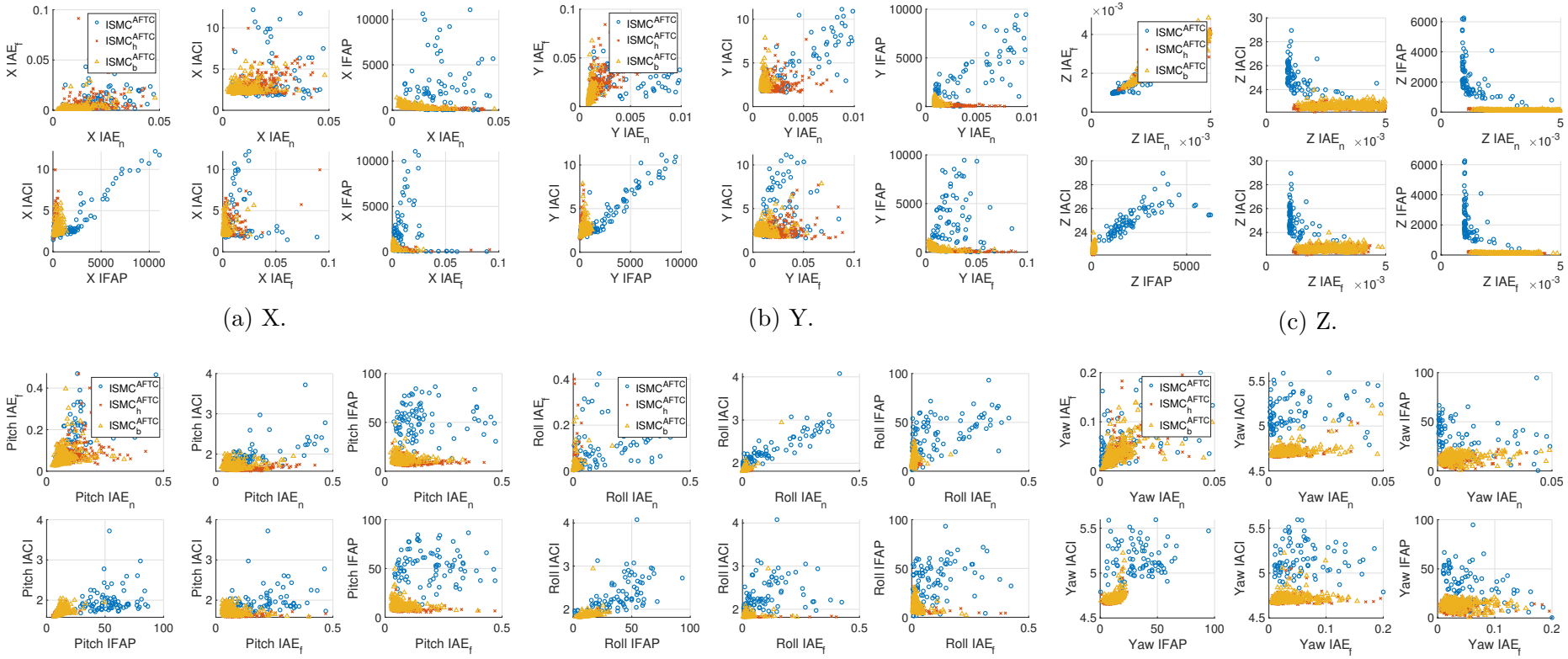
Figure 6.16 shows the Pareto fronts for the $\text{AISMC}_{\text{p}}^{\text{AFTC}}$ where the α gain is adapted using the Plestan method, and the $\text{AISMC}_{\text{r}}^{\text{AFTC}}$ that uses the Roy adaptation method. These controllers are compared against the discontinuous $\text{ISMC}^{\text{AFTC}}$. Both the $\text{AISMC}_{\text{r}}^{\text{AFTC}}$ and $\text{AISMC}_{\text{p}}^{\text{AFTC}}$ produce lower control inputs than the $\text{ISMC}^{\text{AFTC}}$ as is shown by the sub-figures for the IACI indices for the Z, roll, pitch and yaw DoFs. The $\text{AISMC}_{\text{r}}^{\text{AFTC}}$ reduces the chattering experienced by the $\text{ISMC}^{\text{AFTC}}$, as can be seen by the IFAP indices of the roll, pitch and yaw DoFs. Although the $\text{AISMC}_{\text{p}}^{\text{AFTC}}$ can produce low IFAP indices, it is also capable of producing higher IFAP indices than the $\text{ISMC}^{\text{AFTC}}$ as is evidenced by the roll, pitch and yaw DoFs.



(a) X. (b) Y. (c) Z.

(d) Pitch. (e) Roll. (f) Yaw.

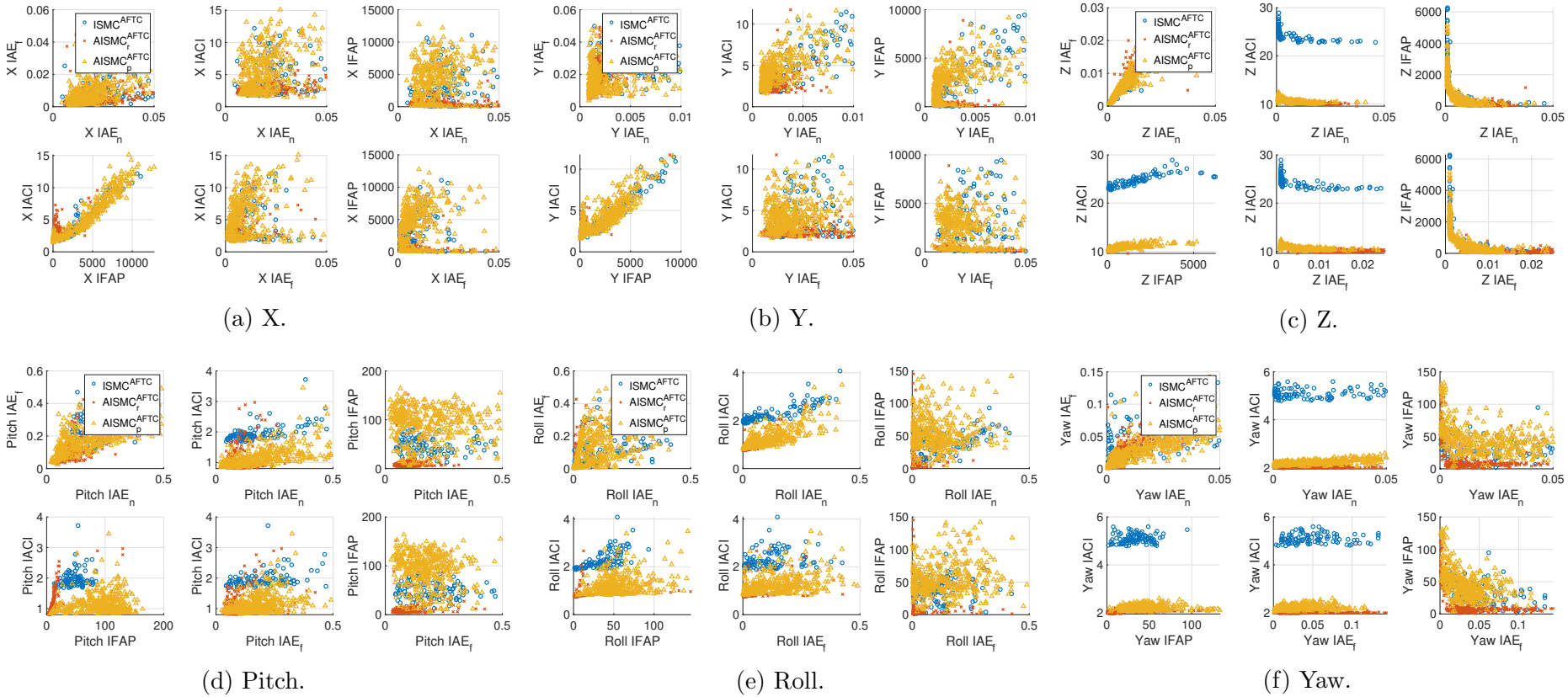
Figure 6.14: MOPSO Pareto fronts - CSMC$^{\text{AFTC}}$ and ISMC$^{\text{AFTC}}$. Tuned gains are: CSMC$^{\text{AFTC}}$ - k_1, k_2, c_1 and ISMC$^{\text{AFTC}}$ - k_1, k_2, c_1, c_2.



(a) X. (b) Y. (c) Z.

(d) Pitch. (e) Roll. (f) Yaw.

Figure 6.15: MOPSO Pareto fronts - $\text{ISMC}^{\text{AFTC}}$, $\text{ISMC}_{\text{h}}^{\text{AFTC}}$ and $\text{ISMC}_{\text{b}}^{\text{AFTC}}$. Tuned gains are:$\text{ISMC}^{\text{AFTC}}$ - k_1, k_2, c_1, $\text{ISMC}_{\text{h}}^{\text{AFTC}}$ and $\text{ISMC}_{\text{b}}^{\text{AFTC}}$ - k_1, k_2, c_1, c_2, λ.



(a) X. (b) Y. (c) Z.

(d) Pitch. (e) Roll. (f) Yaw.

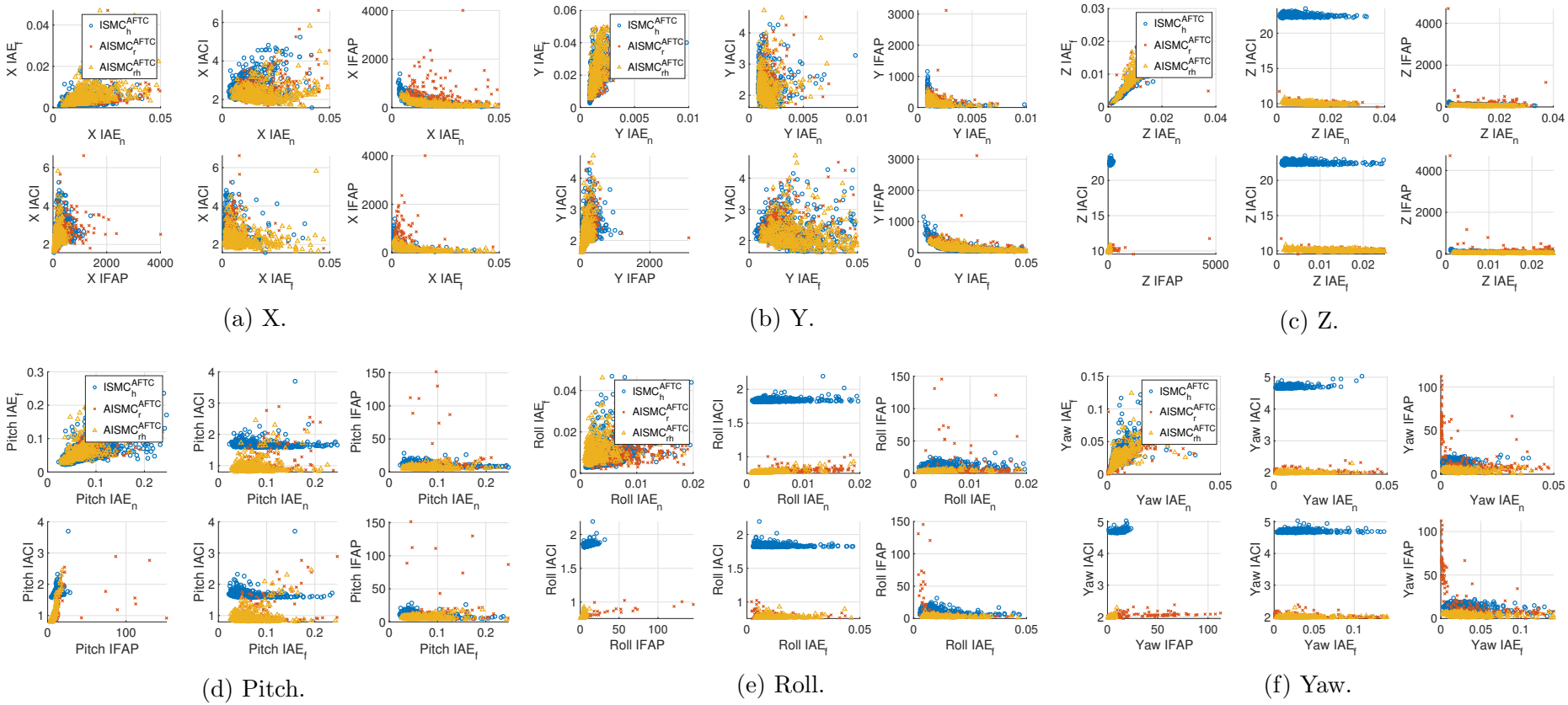
Figure 6.16: MOPSO Pareto fronts - ISMC$^{\text{AFTC}}$, AISMC$_{\text{r}}^{\text{AFTC}}$ and AISMC$_{\text{p}}^{\text{AFTC}}$. Tuned gains are: ISMC$^{\text{AFTC}}$ - k_1, k_2, c_1, AISMC$_{\text{r}}^{\text{AFTC}}$ - k_2, c_1, c_2, ρ_0, ρ_1, r_{0i}, r_{1i} , and AISMC$_{\text{p}}^{\text{AFTC}}$ - k_2, c_1, c_2, η_α, μ, α_i.

As was done with the passive FTCs, the continuous and adaptive active FTCs are combined and compared to determine whether the benefits of both types of controller can be attained. The hyperbolic tangent function is once again selected for the continuous controllers. Both the $\text{AISMC}_{\text{rh}}^{\text{AFTC}}$ and $\text{AISMC}_{\text{ph}}^{\text{AFTC}}$ are compared against their discontinuous versions, and against the continuous $\text{ISMC}_{\text{h}}^{\text{AFTC}}$. The results of the Pareto fronts for the $\text{AISMC}_{\text{rh}}^{\text{AFTC}}$, $\text{AISMC}_{\text{r}}^{\text{AFTC}}$, and $\text{ISMC}_{\text{h}}^{\text{AFTC}}$ are shown in Figure 6.17. The $\text{AISMC}_{\text{rh}}^{\text{AFTC}}$ significantly reduces the control input compared to the $\text{ISMC}_{\text{h}}^{\text{AFTC}}$ as is shown by the sub-figures of the IACI indices for the Z, roll, pitch and yaw DoFs. The $\text{AISMC}_{\text{rh}}^{\text{AFTC}}$ is also able to reduce the high chattering that can be experienced by the $\text{AISMC}_{\text{r}}^{\text{AFTC}}$ as can be seen by the sub-figures for the IFAP indices.

The results of the Pareto fronts for the $\text{AISMC}_{\text{ph}}^{\text{AFTC}}$, $\text{AISMC}_{\text{p}}^{\text{AFTC}}$, and $\text{ISMC}_{\text{h}}^{\text{AFTC}}$ are shown in Figure 6.18. The $\text{AISMC}_{\text{ph}}^{\text{AFTC}}$ is able to significantly reduce the control input compared to the $\text{ISMC}_{\text{h}}^{\text{AFTC}}$ as is shown by the sub-figures of the IACI indices for the Z, roll, pitch and yaw DoFs. The $\text{AISMC}_{\text{ph}}^{\text{AFTC}}$ also significantly reduces the high chattering that can be experienced by the $\text{AISMC}_{\text{p}}^{\text{AFTC}}$ as can be seen from the sub-figures for the IFAP indices.

6.4.3 Active super-twisting sliding mode control results

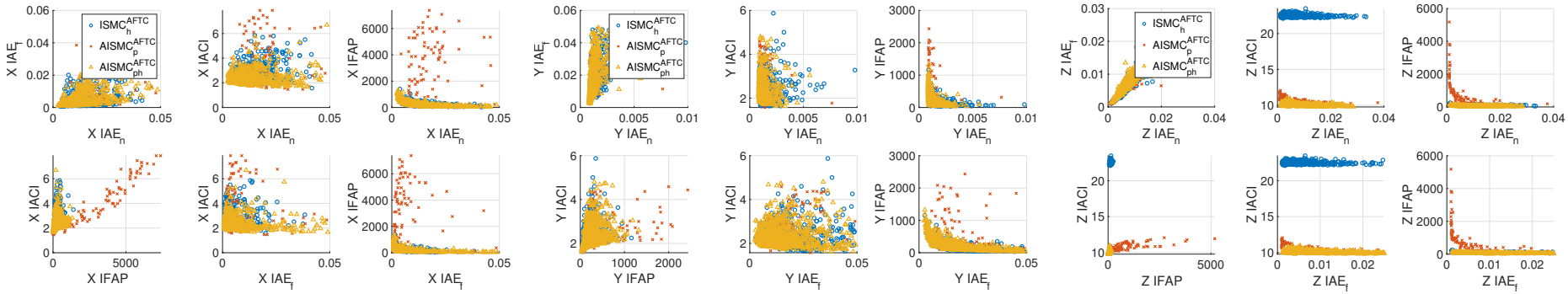
The active STSMCs, that is the $\text{CSTSMC}^{\text{AFTC}}$, and the $\text{ACSTSMC}_{\text{t}}^{\text{AFTC}}$ which is adapted using the Shtessel adaptation method, are compared against the discontinuous, first-order $\text{ISMC}^{\text{AFTC}}$ to determine their ability to reduce chattering experienced by the discontinuous, first-order controller. Figure 6.19 shows the Pareto fronts for the $\text{CSTSMC}^{\text{AFTC}}$, $\text{ACSTSMC}_{\text{t}}^{\text{AFTC}}$ and $\text{ISMC}^{\text{AFTC}}$. With the exception of the yaw DoF, the $\text{CSTSMC}^{\text{AFTC}}$ and $\text{ACSTSMC}_{\text{t}}^{\text{AFTC}}$ are able to reduce the chattering experienced by the $\text{ISMC}^{\text{AFTC}}$ as evidenced by the sub-figures showing the IFAP indices. In some cases, for example for the Z DoF, the $\text{CSTSMC}^{\text{AFTC}}$ and $\text{ACSTSMC}_{\text{t}}^{\text{AFTC}}$ can produce high control inputs as shown by the sub-figures for the IACI indices.



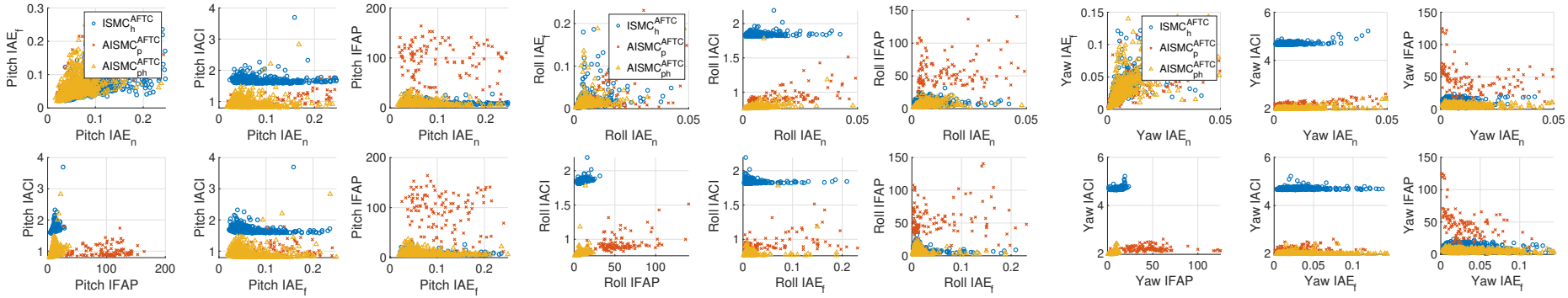
(a) X. (b) Y. (c) Z.

(d) Pitch. (e) Roll. (f) Yaw.

Figure 6.17: MOPSO Pareto fronts - $\text{ISMC}_{\text{h}}^{\text{AFTC}}$, $\text{AISMC}_{\text{r}}^{\text{AFTC}}$ and $\text{AISMC}_{\text{rh}}^{\text{AFTC}}$. Tuned gains are: $\text{ISMC}_{\text{h}}^{\text{AFTC}}$ - k_1, k_2, c_1, λ, $\text{AISMC}_{\text{r}}^{\text{AFTC}}$ - k_2, c_1, c_2, ρ_0, ρ_1, r_{0i}, r_{1i}, and $\text{AISMC}_{\text{rh}}^{\text{AFTC}}$ - k_2, c_1, c_2, ρ_0, ρ_1, r_{0i}, r_{1i}, λ.

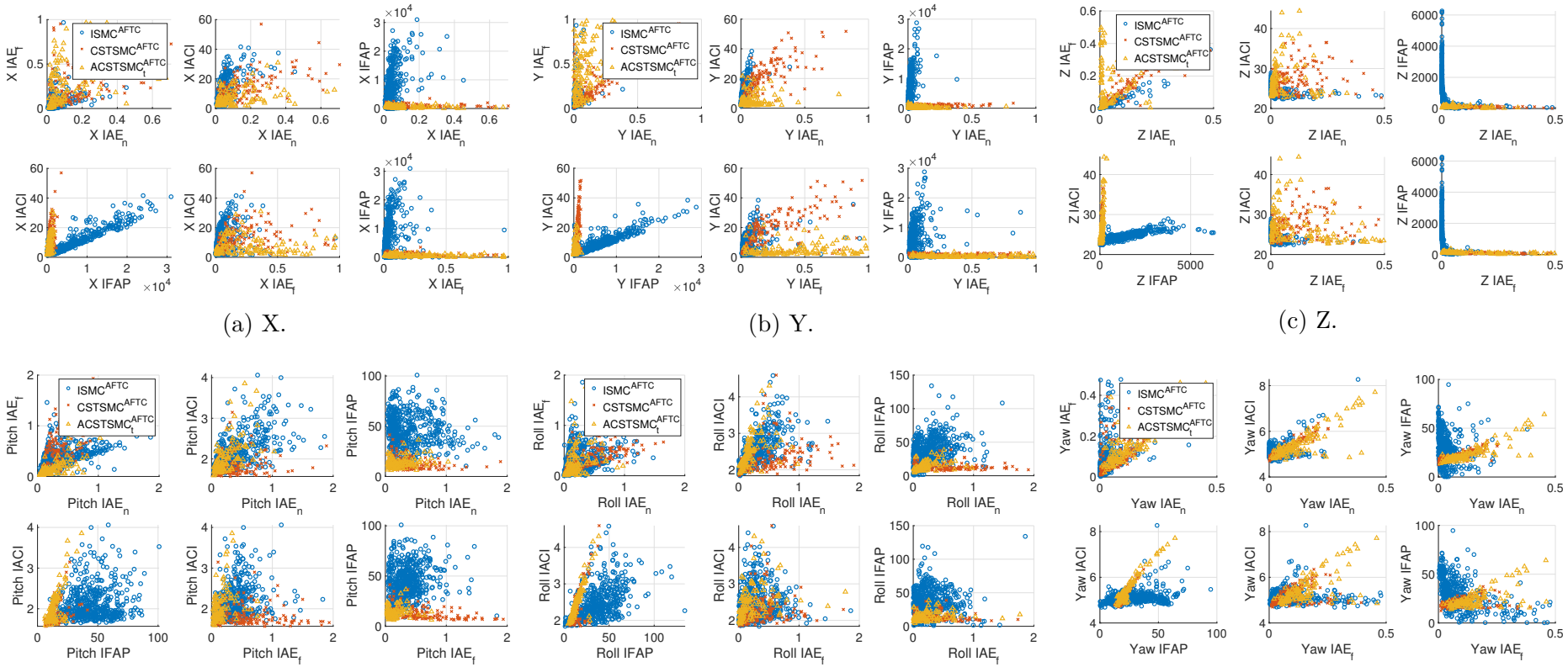


(a) X. (b) Y. (c) Z.



(d) Pitch. (e) Roll. (f) Yaw.

Figure 6.18: MOPSO Pareto fronts - ISMC$_{\text{h}}^{\text{AFTC}}$, AISMC$_{\text{p}}^{\text{AFTC}}$ and AISMC$_{\text{ph}}^{\text{AFTC}}$. Tuned gains are: ISMC$_{\text{h}}^{\text{AFTC}}$, AISMC$_{\text{p}}^{\text{AFTC}}$ - k_2, c_1, c_2, η_α, μ, α_i and AISMC$_{\text{ph}}^{\text{AFTC}}$ - k_2, c_1, c_2, η_α, μ, α_i, λ.



(a) X. (b) Y. (c) Z.

(d) Pitch. (e) Roll. (f) Yaw.

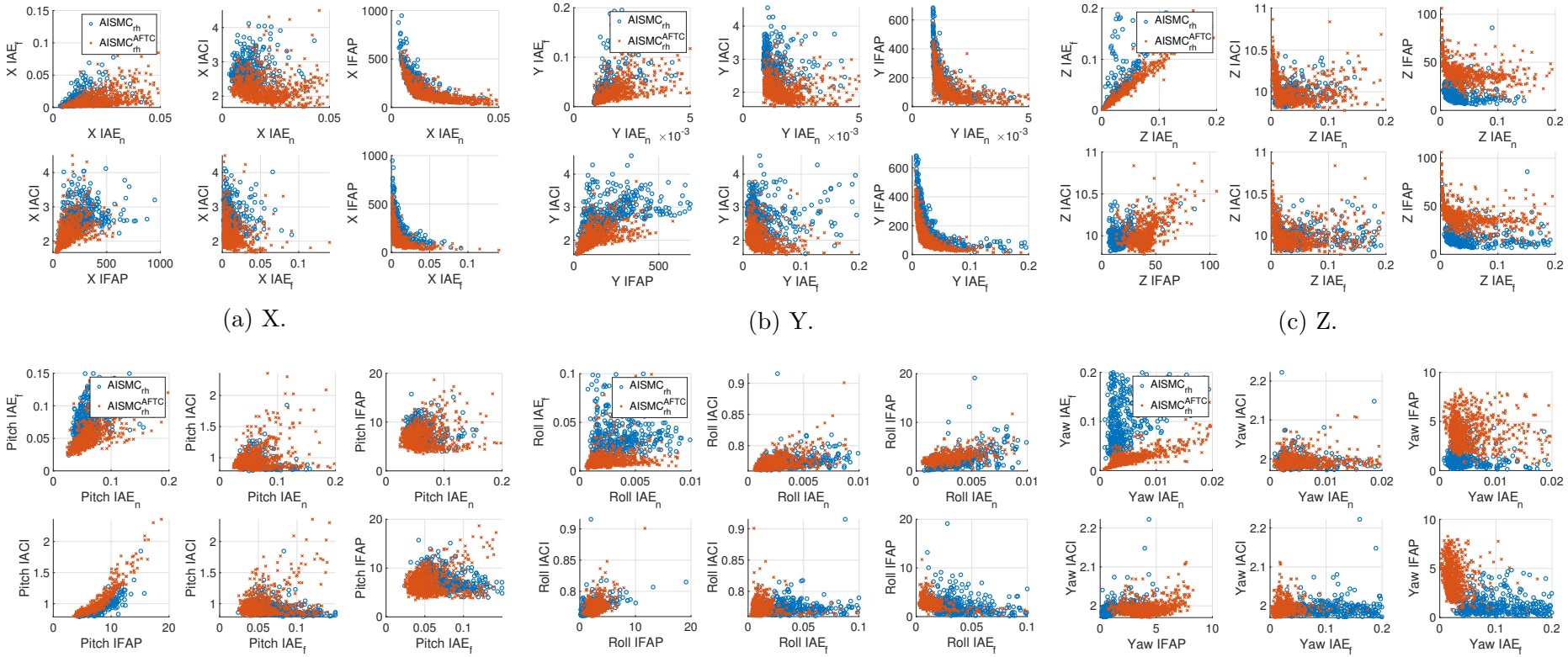
Figure 6.19: MOPSO Pareto fronts - $\text{ISMC}^{\text{AFTC}}$, $\text{CSTSMC}^{\text{AFTC}}$ and $\text{ACSTSMC}_{\text{t}}^{\text{AFTC}}$. Tuned gains are: $\text{ISMC}^{\text{AFTC}}$ - k_1, k_2, c_1, c_2, $\text{CSTSMC}^{\text{AFTC}}$ - k_1, k_2, c_1 and $\text{ACSTSMC}_{\text{t}}^{\text{AFTC}}$ - c_1, η_α, μ, η_β, α_i.

6.5 Active versus Passive Fault-Tolerant Control

For the passive FTCs, the continuous, adaptive integral controllers, AISMC_{rh} and AISMC_{ph} produce the best results in terms of lowering chattering and lowering control inputs. Likewise for the active controllers, $\text{AISMC}_{\text{rh}}^{\text{AFTC}}$ and $\text{AISMC}_{\text{ph}}^{\text{AFTC}}$ also produce the best chattering and control input results. The best performing passive and active sliding mode FTCs are now compared. The passive AISMC_{rh} is compared against the active $\text{AISMC}_{\text{rh}}^{\text{AFTC}}$, and the passive AISMC_{ph} is compared against the active $\text{AISMC}_{\text{ph}}^{\text{AFTC}}$.

The Pareto fronts for the AISMC_{rh} and $\text{AISMC}_{\text{rh}}^{\text{AFTC}}$ are shown in Figure 6.20. As can be seen from the sub-figures for the IAE_{f} indices, the $\text{AISMC}_{\text{rh}}^{\text{AFTC}}$ is better able to reduce the error during the faulty portion of the trajectory than the AISMC_{rh}. The X and Y DoFs have lower control inputs for the $\text{AISMC}_{\text{rh}}^{\text{AFTC}}$ as can be seen by the sub-figures for the IACI indices. The remaining DoFs have similar control inputs for the $\text{AISMC}_{\text{rh}}^{\text{AFTC}}$ and AISMC_{rh}. The X and Y DoFs have lower chattering for the $\text{AISMC}_{\text{rh}}^{\text{AFTC}}$ as can be seen by the sub-figures for the IFAP indices, however, the Z, roll and yaw DoFs for the $\text{AISMC}_{\text{rh}}^{\text{AFTC}}$ have higher chattering than the AISMC_{rh}.

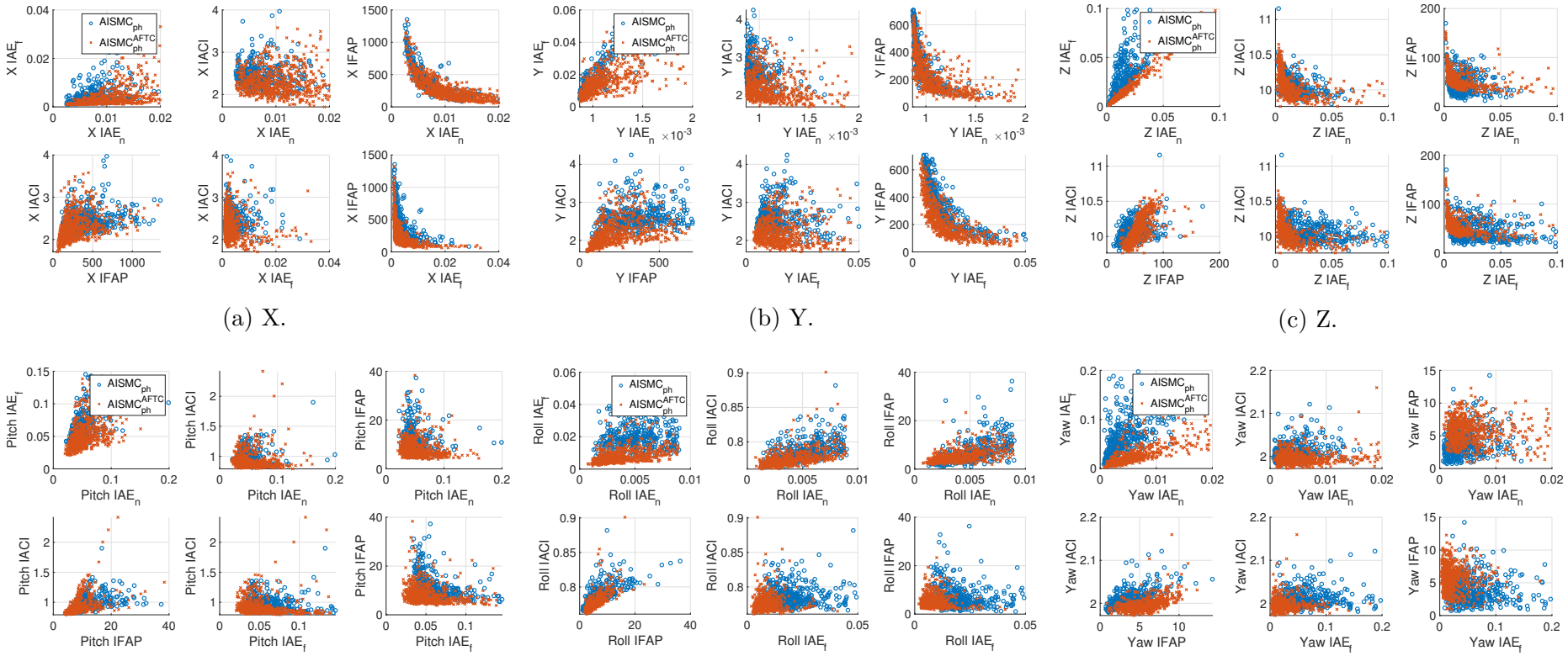
The Pareto fronts for the AISMC_{ph} and $\text{AISMC}_{\text{ph}}^{\text{AFTC}}$ are shown in Figure 6.21. As can be seen from the sub-figures for the IAE_{f} indices, the $\text{AISMC}_{\text{ph}}^{\text{AFTC}}$ is better able to reduce the error during the faulty portion of the trajectory than the AISMC_{ph}. The $\text{AISMC}_{\text{ph}}^{\text{AFTC}}$ can produce slightly lower control inputs than the AISMC_{ph} as can be seen by the sub-figures for the IACI indices. The X, Y and pitch DoFs have lower chattering for the $\text{AISMC}_{\text{ph}}^{\text{AFTC}}$ as can be seen by the sub-figures for IFAP indices, however, the Z and yaw DoFs for the $\text{AISMC}_{\text{ph}}^{\text{AFTC}}$ have higher chattering than the AISMC_{ph}.



(a) X. (b) Y. (c) Z.

(d) Pitch. (e) Roll. (f) Yaw.

Figure 6.20: MOPSO Pareto fronts - AISMC_{rh} and $\text{AISMC}_{\text{rh}}^{\text{AFTC}}$. Tuned gains are: AISMC_{rh} and $\text{AISMC}_{\text{rh}}^{\text{AFTC}}$ - k_2, c_1, c_2, ρ_0, ρ_1, r_{0i}, r_{1i}, λ.



(a) X. (b) Y. (c) Z.

(d) Pitch. (e) Roll. (f) Yaw.

Figure 6.21: MOPSO Pareto fronts - AISMC_{ph} and $\text{AISMC}_{\text{ph}}^{\text{AFTC}}$. Tuned gains are: AISMC_{ph} and $\text{AISMC}_{\text{ph}}^{\text{AFTC}}$ - k_2, c_1, c_2, η_α, μ, α_i, λ.

6.6 Farm Coverage and Plant Sample Collection Simulations

Based on the MOPSO controller gains-tuning results in the previous sections, the $\text{AISMC}_{\text{ph}}^{\text{AFTC}}$ and $\text{AISMC}_{\text{rh}}^{\text{AFTC}}$ are selected for full farm coverage and plant sample collection simulations of the aerial manipulator, with an added rotor fault. For this simulation it is assumed that a perfect FDD unit is available. The controller gains are selected from the MOPSO controller gains-tuning results by first selecting a small error range, based on the IAE_{n} and IAE_{n} indices for each DoF and then choosing controller gains that result in low chattering using the IFAP indices, and low control inputs using the IACI indices for all of the DoFs. The controller gains used for the simulation are shown in Table 6.4. Figure 6.22 shows the top view of the farm, with the back-and-forth coverage path, starting point for the aerial manipulator, and the sample collection locations shown. The back-and-forth coverage path, including a return path, is chosen due to its simplicity compared to the spiral coverage path. Figure 6.23 shows the rotor remaining effectiveness and Figure 6.24 shows the manipulator joint angles for the plant sample collection. It is assumed that the plant samples are all at the same height below the hovering aerial manipulator, thus the same manipulator motion is used for each plant sample collection.

Table 6.4: Controller gains for farm coverage and sample collection simulations.

Controllers	Gain	X	Y	Z	ϕ	θ	ψ
$\text{AISMC}_{\text{ph}}^{\text{AFTC}}$	k_2	4.9	6	4.8	42.9	57.7	53.3
	c_1	1.5	1.6	3.4	5.7	7.2	1
	c_2	1.1	4.3	9.1	7.4	2.5	0.001
	λ	0.49	0.5	0.37	0.31	0.39	0.1
	α_i	3.2	0.57	8.5	1×10^{-4}	18	20
	η_α	0.053	0.011	0.075	0.011	0.024	0.05
	η_β	1×10^{-5}	1×10^{-5}	1×10^{-5}	1×10^{-5}	1×10^{-5}	1×10^{-5}
	μ	0.74	0.22	0.01	0.49	0.76	0.93
$\text{AISMC}_{\text{rh}}^{\text{AFTC}}$	k_2	1.1	5.6	15.9	13.9	48.3	38
	c_1	10	2.8	8.6	15.3	8.1	14
	c_2	7.6	3.1	4.3	5.2	6.2	5.9
	λ	0.1	0.38	0.47	0.45	0.1	0.4
	r_{0i}	2.6	1.4	13	19.2	3.9	16.4
	r_{1i}	1×10^{-6}	4.6	18.3	16.9	20	1×10^{-6}
	ρ_0	24	24.2	9.1	6.5	5.7	1×10^{-4}
	ρ_1	34.3	38.2	29.6	38.4	9.6	40

The farm coverage results are shown in Figure 6.25. As can be seen from the figure,

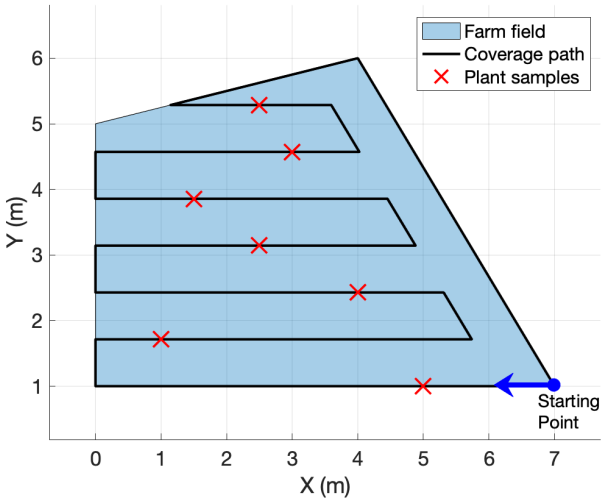


Figure 6.22: Aerial manipulator farm coverage and plant sample collection: back-and-forth coverage with return path, showing plant sample locations.

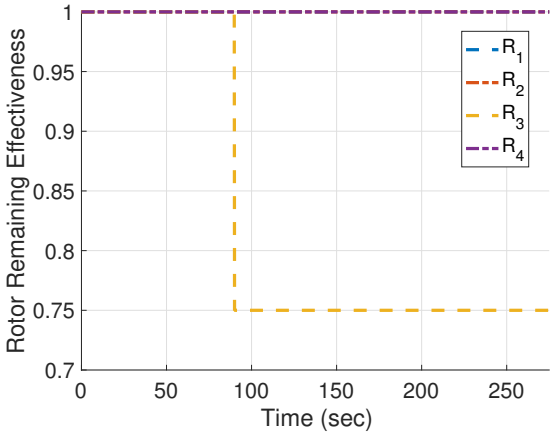


Figure 6.23: Aerial manipulator farm coverage and plant sample collection: rotor remaining effectiveness.

both the controllers are able to accurately track the desired coverage path, even when the rotor fault is experienced. The trajectory tracking errors for the quadrotor subsystem are shown in Figure 6.26. The trajectory tracking errors for all of the DoFs are small as can be seen from the figure. The $\text{AISMC}_{\text{rh}}^{\text{AFTC}}$ has slightly larger errors than the $\text{AISMC}_{\text{ph}}^{\text{AFTC}}$. The quadrotor subsystem control inputs are shown in Figure 6.27. The large changes in control input due to compensating for the rotor fault are evident for the Z, roll, pitch, and yaw DoFs. The control inputs for the quadrotor subsystem, required to compensate for the manipulator motion, are evident from the Z and pitch DoF control input results at the times when manipulator motion is experienced. Overall, the aerial manipulator successfully tracks the desired coverage path, recovers from the fault, and collects all the desired plant samples.

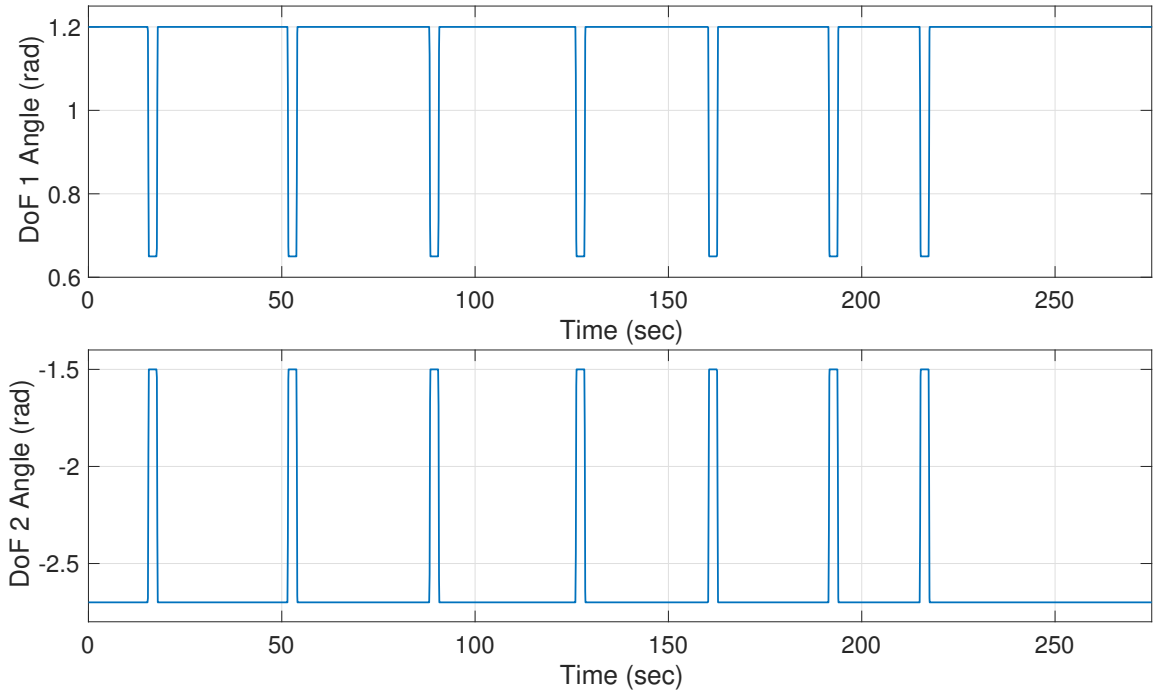


Figure 6.24: Aerial manipulator farm coverage and plant sample collection: manipulator joint angles.

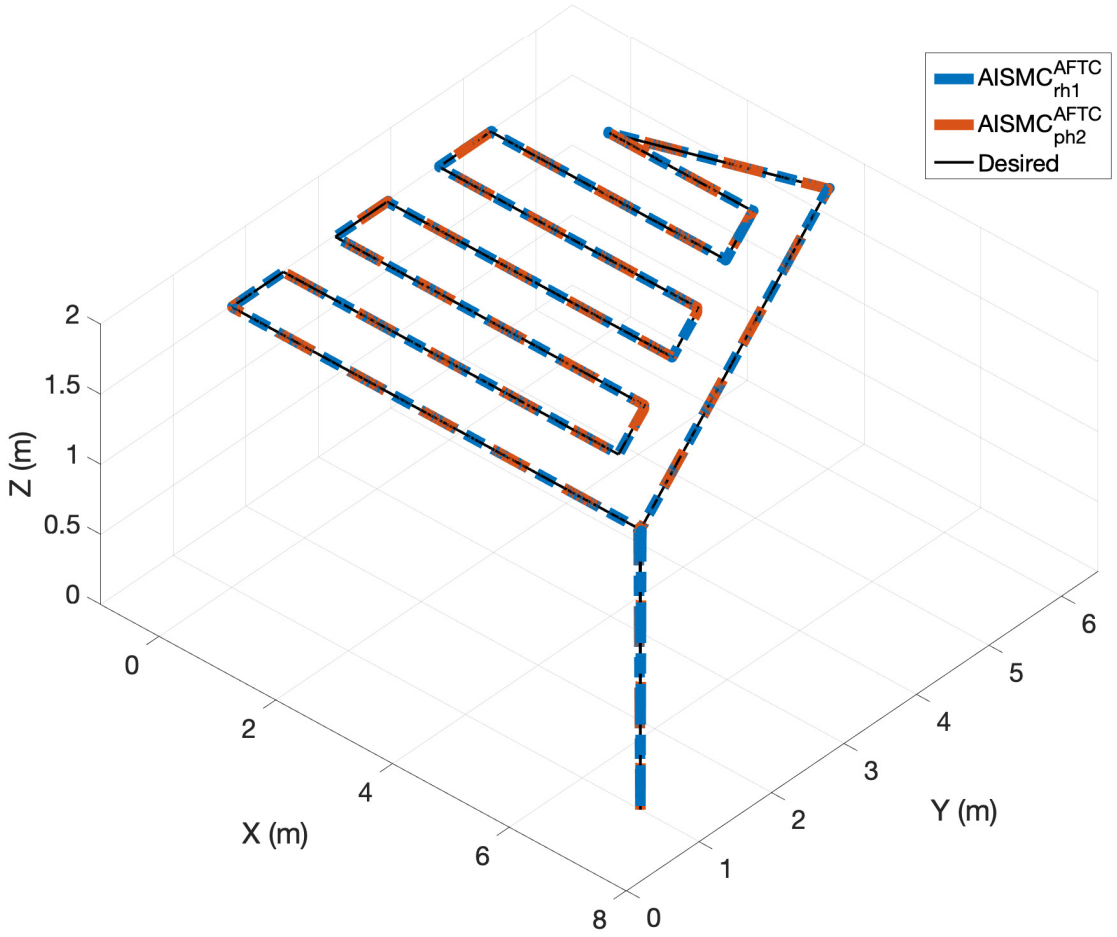


Figure 6.25: Aerial manipulator farm coverage path following results.

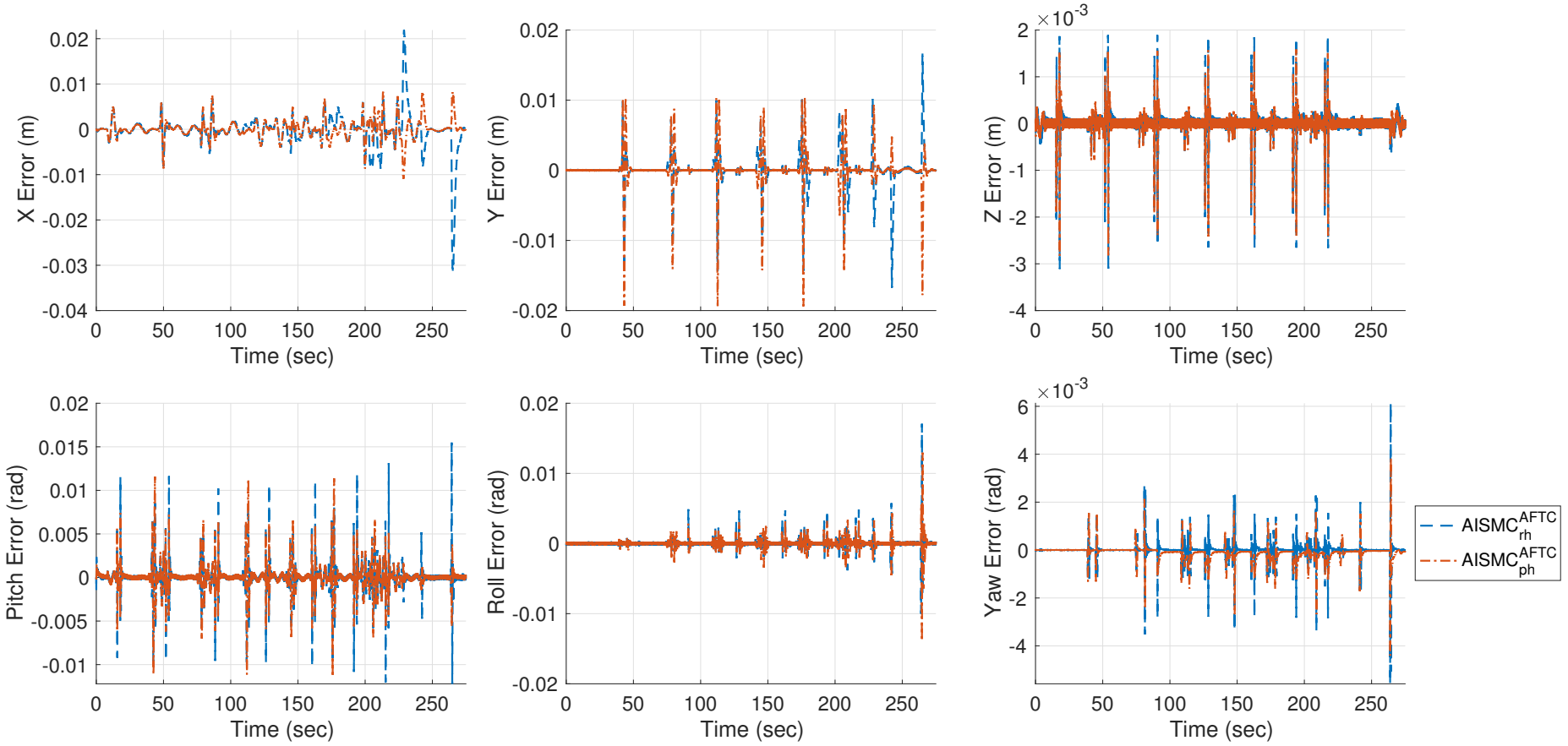


Figure 6.26: Aerial manipulator farm coverage and plant sample collection: quadrotor trajectory tracking errors.

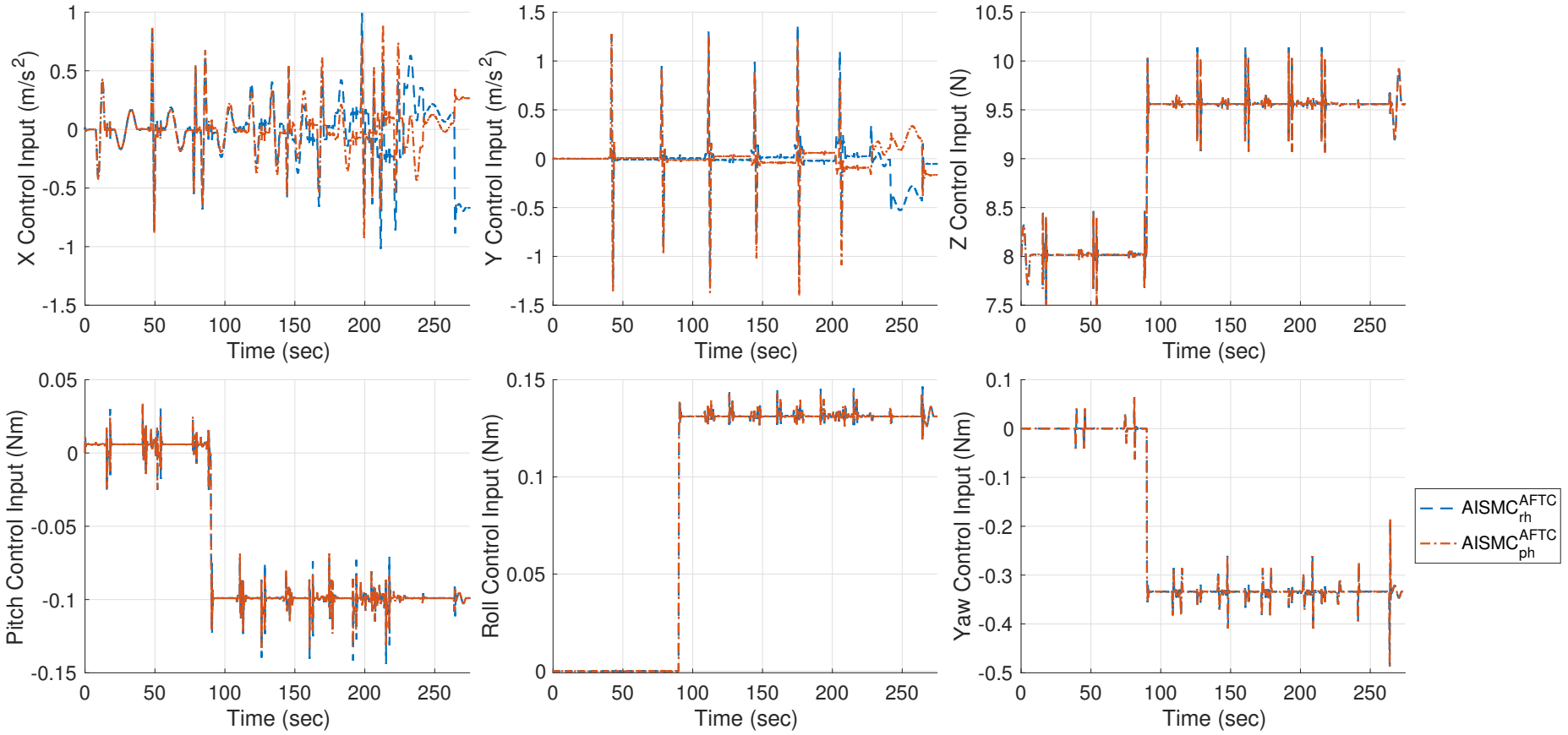


Figure 6.27: Aerial manipulator farm coverage and plant sample collection: quadrotor control inputs.

6.7 Summary

In this chapter, various SMC formulations were compared for passive and active FTC of an aerial manipulator, namely discontinuous and continuous SMCs, non-adaptive and adaptive SMCs, classical and integral SMCs, and first-order and second-order SMCs. The controller gains were optimised using the MOPSO formulation and the resultant Pareto fronts were used to compare the chattering, trajectory tracking, and control input performance of the controllers. The passive and active, adaptive, integral, continuous first-order SMCs,; AISMC_{rh}, AISMC_{ph}, $\text{AISMC}_{\text{rh}}^{\text{AFTC}}$ and $\text{AISMC}_{\text{ph}}^{\text{AFTC}}$, proved to have the best results in terms of trajectory tracking error, control input, and chattering. The active FTCs in most instances, had better results than the passive FTCs, although there were some instances where they had higher chattering indices than the passive FTCs. Farm coverage, using the back-and-forth coverage path, and plant sample collections were simulated for the aerial manipulator, including a rotor fault. The aerial manipulator successfully completed the farm coverage and plant sample collection task in the presence of the rotor fault.

7 Fault-Tolerant Control Experiments

This chapter details FTC experiments performed with a physical quadrotor and a physical quadrotor-based aerial manipulator. First, the passive and active FTC structures of the Crazyflie quadrotor-based aerial manipulator are presented. MOPSO controller gains tuning is performed to obtain a suitable set of controller gains for the quadrotor and for the aerial manipulator. Passive FTC experiments are performed to determine the effectiveness of the controllers at handling faults. The neural network-based FDD units are then trained using flight data with and without rotor faults. Experiments on the active FTC of the controllers using neural network-based FDD are then conducted. The results of the experiments for the quadrotor and for the quadrotor-based aerial manipulator are discussed.

7.1 Crazyflie Aerial Manipulator FTC Structures

The aerial manipulator used for the experiments consists of a Crazyflie quadrotor with a one-DoF manipulator as detailed in Chapter 2. The Crazyflie quadrotor is an open source system and has three on-board controllers that can be used for position and yaw trajectory tracking. The first controller is a cascaded PID controller that consists of a PID position controller, a PID velocity controller, an attitude PID controller, and an angle rate PID controller [179]. The second controller is the Mellinger controller [179, 180], which is a type of PID controller consisting of PID position control, PID yaw angle control and PD control with an additional angular velocity term for the roll and pitch angle control. The third controller uses PD position and velocity control with added incremental nonlinear dynamic inversion (INDI) acceleration control, and INDI control for the attitude rate control [179, 181]. The on-board Mellinger controller is used as a baseline to compare the performance of the SMCs as it has the best trajectory tracking results of the three on-board controllers as illustrated in Figure 7.1.

It should be noted that the Crazyflie quadrotor control system is set up such that the controllers for the four controllable DoFs must output dimensionless Z, roll, pitch, and yaw control commands (cmd_Z, cmd_ϕ, cmd_θ, cmd_ψ), with limits applied to the roll, pitch and yaw control commands. The control commands are then sent to the Crazyflie 'power distribution' where the control commands are translated into the desired rotor PWM duty cycle values. The relationship between the control commands and the PWM values found in the 'power distribution' of the Crazyflie control system (and corrected to give

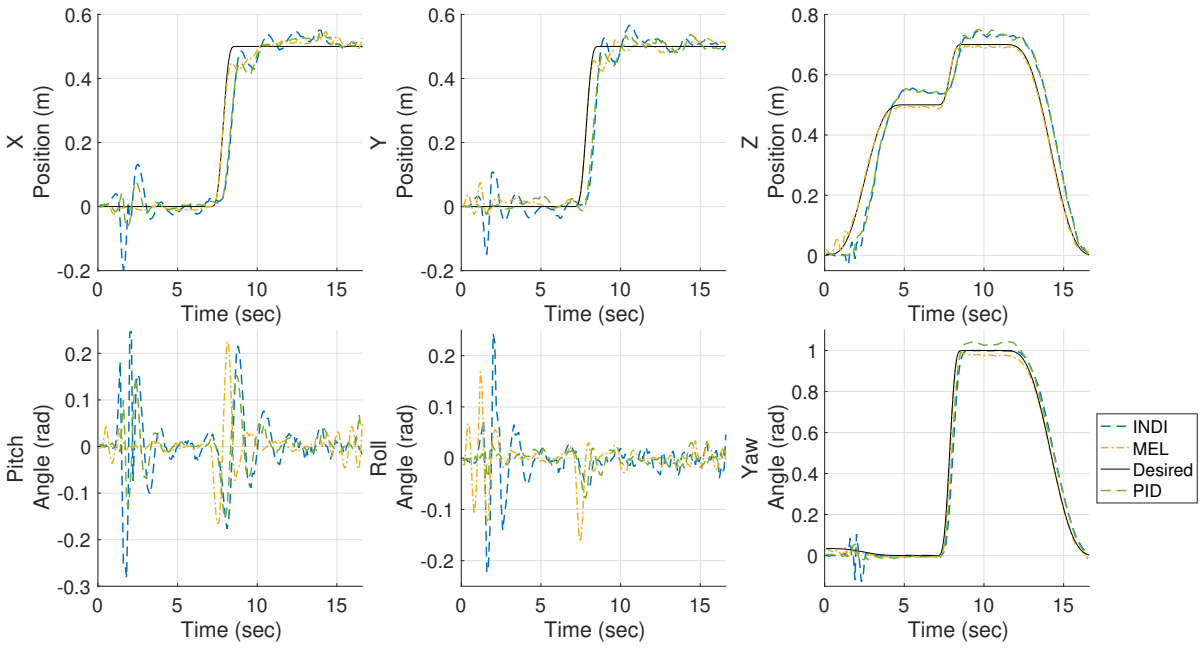


Figure 7.1: Crazyflie quadrotor on-board controllers: trajectory tracking.

the correct yaw orientation) is:

$$\begin{bmatrix} PWM_{r1} \\ PWM_{r2} \\ PWM_{r3} \\ PWM_{r4} \end{bmatrix} = \begin{bmatrix} 1 & -0.5 & +0.5 & -1 \\ 1 & -0.5 & -0.5 & +1 \\ 1 & +0.5 & -0.5 & -1 \\ 1 & +0.5 & +0.5 & +1 \end{bmatrix} \begin{bmatrix} cmd_Z \\ cmd_\phi \\ cmd_\theta \\ cmd_\psi \end{bmatrix} \tag{7.1}$$

The nature of the Crazyflie's control and 'power distribution' system mean that it is not possible to directly implement the fault-tolerant SMC structures presented in Chapter 3 since the SMCs compute control inputs in Newton for the Z DoF, and Newton metres for the roll, pitch and yaw DoFs. In order for the SMCs to be implemented for position and yaw trajectory tracking control on the Crazyflie quadrotor, the control inputs for the controllable Z, roll, pitch and yaw DoFs need to be converted into the dimensionless control commands. For the Z DoF, the on-board Mellinger controller uses a 'massthrust' factor to convert the control input to the control command cmd_Z. This same method is used for the Z DoF of SMCs.

For the Mellinger roll, and pitch DoFs PD controllers and the yaw DoFs PID controller, controller gains are used that result in the cmd_ϕ, cmd_θ, and cmd_ψ control commands being directly calculated for those DoFs. For SMC of the yaw DoF, a 'yawthrust' factor is calculated to convert the SMC control input into the control command cmd_ψ. The same method, however, does not produce good results for the SMC of the pitch and roll DoFs, for the position control of the system. This is possibly due to the fast changes in roll and pitch angles that are computed based on the X and Y virtual control inputs.

As was discussed in Chapter 3, a quadrotor is an underactuated system and for X and Y position control to be achieved, virtual control inputs U_X and U_X are computed for the X and Y DoFs, which are then used to compute the desired roll and pitch angles to achieve the X and Y position trajectory tracking. Using 'rollthrust' and 'pitchthrust' factors would likely amplify the fast changes in the control inputs in such a way that stable control would not be attained for the roll and pitch DoFs when the quadrotor is under position and yaw control.

The SMCs are therefore applied to four DoFs, the X, Y, Z, and yaw DoFs. The pitch and roll DoFs are controlled using the on-board Mellinger pitch and roll controllers. To obtain the pitch and roll angle errors, the Mellinger controller [180] makes use of the desired yaw angle, the SMC Z control input, (in this case it is denoted as U_{TZ}), and the U_{TX} and U_{TY} inputs obtained by multiplying the X and Y DoF virtual control inputs by the system mass. The resultant passive FTC structure used for the Crazflie quadrotor-based aerial manipulator is shown in Figure 7.2 and the active FTC structure is shown in Figure 7.3. For the active FTC structure, since the neural network-based FDD unit makes use of control inputs U_1, U_2, U_3, and U_4, the control commands cmd_Z, cmd_ϕ, cmd_θ, cmd_ψ are converted to control inputs by calculating the PWM duty cycle values using equation (7.1). The PWM duty cycle values are converted into control inputs using equations (2.86), (2.88), and (2.47). The FDD unit outputs are converted into control commands using equations (2.47), (2.86) and (7.1).

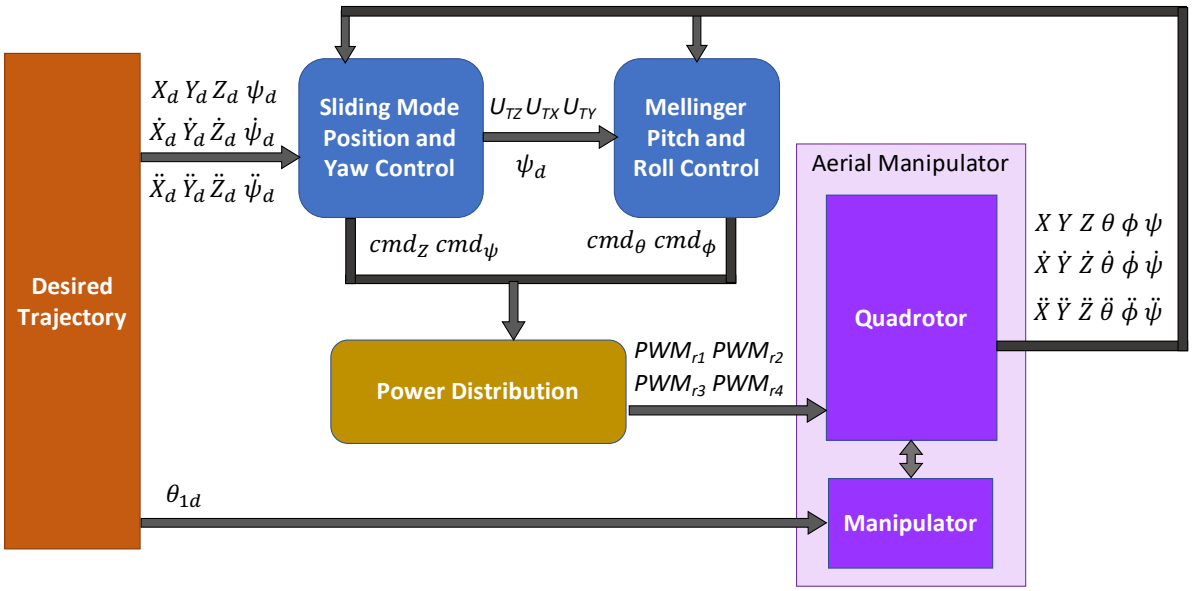


Figure 7.2: Crazyflie quadrotor-based aerial manipulator passive fault-tolerant control scheme

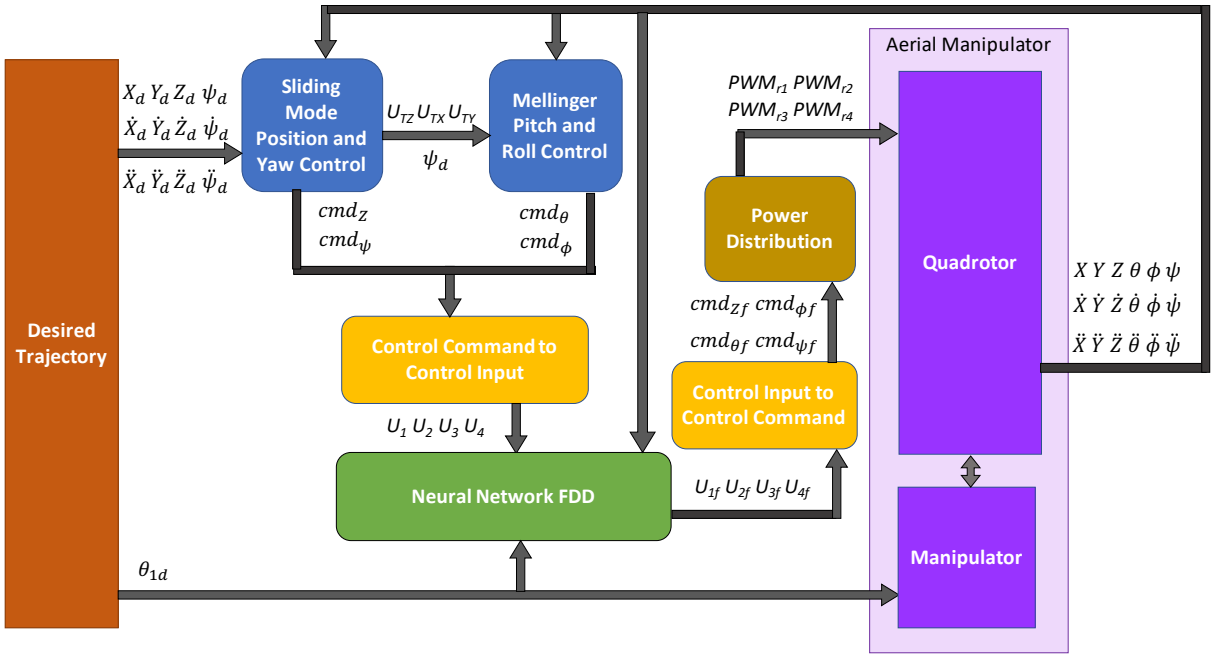


Figure 7.3: Crazyflie quadrotor-based aerial manipulator active fault-tolerant control scheme

7.2 Controller Gains Tuning

Experiments using both the quadrotor and the quadrotor-based aerial manipulator are conducted. The Crazyflie quadrotor-based aerial manipulator has a mass of 42 g, which is the maximum allowable take-off mass for the Crazyflie. In addition, the manipulator subsystem is powered by the Crazyflie quadrotor battery. This means that the aerial manipulator very quickly drains the battery and cannot sustain long flights. The quadrotor motors also heat up considerably as the rotors rotate close to their upper speed limits when the manipulator is attached. In order to preserve the battery and motors, initial tests are conducted using the quadrotor without the manipulator. Final tests are then conducted using the quadrotor-based aerial manipulator.

The system models and physical parameters from Chapter 2 are used to simulate the quadrotor and the aerial manipulator. The MOPSO control gains-tuning method detailed in Chapter 5 is used to obtain a set of possible controller gains for the quadrotor and for the aerial manipulator. Although tuning all of the controller gains for all the DoFs of the system simultaneously allows for convenient comparison of the performance of the controllers, it can complicate the selection of controller gains to be used on the physical system as there are a large number of gains and a large number of performance indices that need to be taken into account when selecting the controller gains. To simplify the process, first, the value of the λ gain is kept constant for all the continuous controllers and the λ value is therefore not tuned as part of the MOPSO controller gains-tuning process. This is because the effect of the λ gain is easy to understand, that is a smaller value of λ provides more robustness than a larger value of λ, whilst the smaller λ value increases the likelihood of chattering compared to a larger λ value. A value of $\lambda = 0.3$ is chosen for all

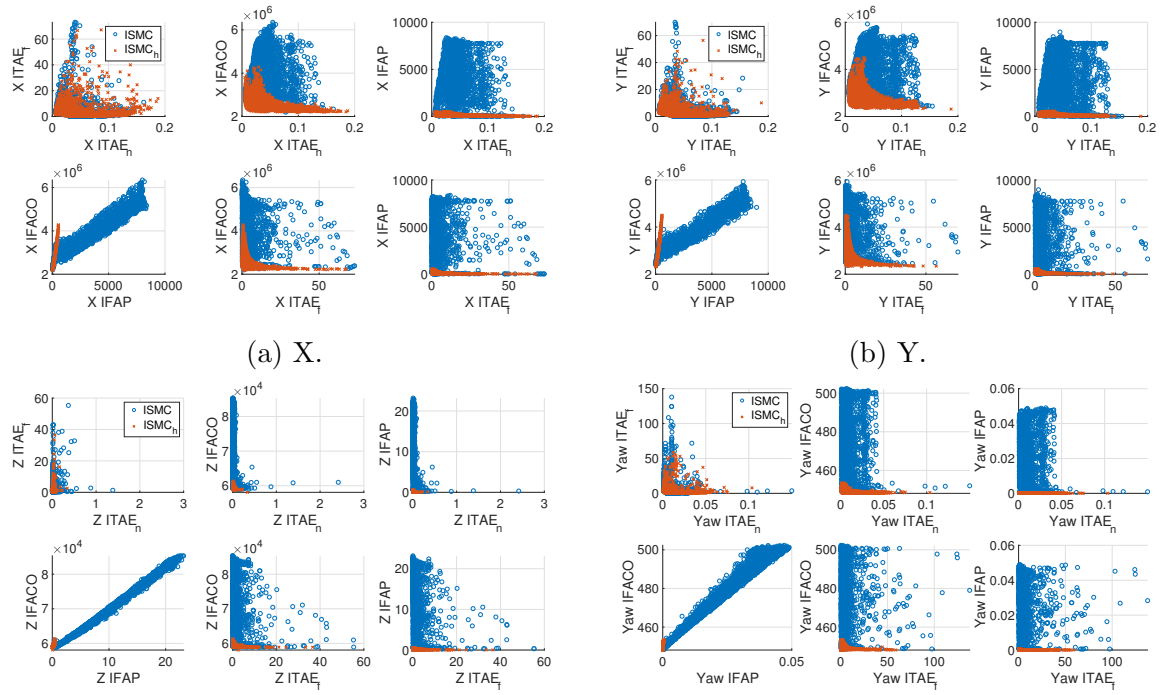
of the DoFs to provide sufficient robustness while lowering chattering. Other controller gains that are kept as constant to simplify the controller gain-tuning process are, for the Plestan adaptation method, the α_m and η_β gains, and for the Roy adaptation method, the r_{0m} and r_{1m} gains.

From the simulations presented in Chapter 6, it was shown that the continuous ISMC_h reduced the chattering experienced by the ISMC. The adaptive controllers, AISMC_r and AISMC_p were also able to reduce chattering, although high chattering could still be experienced with certain controller gain values. The adaptive, continuous controllers, AISMC_rh and AISMC_ph were shown to be able to further reduce chattering. Six controllers are therefore first tested on the quadrotor to verify their performance on the physical quadrotor. Thereafter, the most suitable controllers are tested on the aerial manipulator.

For the controller gains-tuning simulations, the quadrotor has to take off and reach a desired height. The quadrotor then hovers and a fault is introduced in one of the rotors. After several seconds, the quadrotor then lands. The initial position and orientation are offset as typically, with a physical quadrotor, the initial position and orientation are not exactly at zero values. As an example of the results obtained, Figure 7.4 shows the results for the quadrotor controller gains tuning for the ISMC and the ISMC_h. In this case, the control input oscillation index IFACO is used to reduce control input oscillations that are not related to chattering, the error indices ITAE_n and ITAE_f are used to minimise errors that persist over time, and the chattering index, IFAP, is used to reduce chattering. As with the controller gains-tuning results obtained in Chapter 6, the ISMC_h reduces chattering and control inputs compared to the ISMC.

7.2.1 Controller gains selection

To select the required controller gains from the MOPSO controller gains-tuning process, particles on the Pareto front that result in low errors during normal and the faulty trajectory portions are first selected using the ITAE_n and ITAE_f indices. From these particles, those that result in low chattering based on the IFAP index, and low control input oscillation based on the IFACO index, are then selected. All of the selected particles fall within a small range of the MOPSO performance indices. The controller gains for the selected particles are then used to simulate the desired position and yaw trajectory with an added rotor fault, and the results are compared. Varied error and control input results can be obtained, since particles from a small range of performance indices are selected, and also because the same performance index can be obtained for different error and control input plots. Comparing the resultant outputs gives an indication of which controller gains should be selected for the physical system. For example, Figure 7.5 shows the trajectory tracking error results from the simulation using three sets of gains obtained from the ISMC gains tuning and selection process, ($\text{ISMC}_{1\text{q}}$, $\text{ISMC}_{2\text{q}}$, and $\text{ISMC}_{3\text{q}}$), and Figure 7.6 shows the control input results. The $\text{ISMC}_{4\text{q}}$ gains set shows the effect of adjusting the gains obtained from the gains-tuning process. Table 7.1 shows the values of the controller gains.



(a) X. (b) Y.

(c) Z. (d) Yaw.

Figure 7.4: MOPSO Pareto fronts - ISMC and ISMC_h of the quadrotor. Tuned gains are: ISMC and ISMC_h - k_1, k_2, c_1, c_2.

Table 7.1: Quadrotor simulation MOPSO controller gains.

DoF	Gain	ISMC_{1q}	ISMC_{2q}	ISMC_{3q}	ISMC_{4q}
X	k_1	1×10^{-4}	4.84×10^{-2}	1.74×10^{-2}	1×10^{-4}
	k_2	2.94	5.23	2.24	5.23
	c_1	1.9	2.35	2.94	2.35
	c_2	0.754	0.296	0.353	0.296
Y	k_1	1×10^{-4}	1×10^{-4}	1×10^{-4}	1×10^{-4}
	k_2	6.43	6.98	6.54	6.98
	c_1	4.9	3.11	3.27	3.11
	c_2	0.467	0.497	0.378	0.497
Z	k_1	1.45×10^{-4}	1×10^{-2}	0.178	1×10^{-2}
	k_2	10	8.66	9.15	10
	c_1	3.63	4.81	4.04	3.63
	c_2	0.549	0.99	0.972	0.449
Yaw	k_1	0.184	0.333	0.3	9×10^{-2}
	k_2	11.42	9.45	11.42	11.42
	c_1	7.21	4.67	2.99	7.21
	c_2	0.983	0.595	1.05	0.983

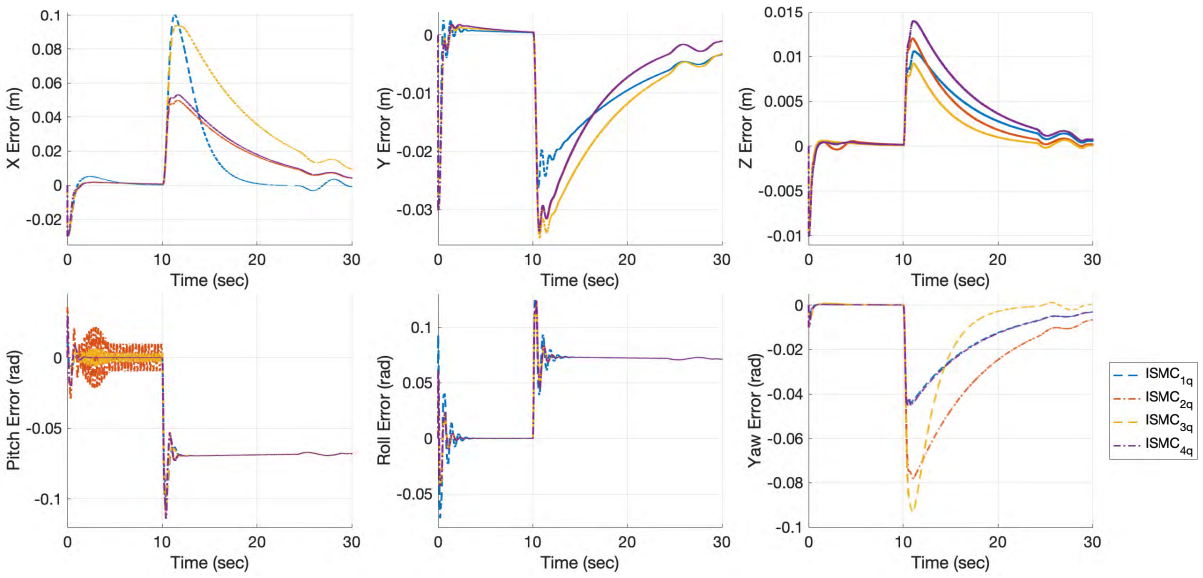


Figure 7.5: ISMC simulation: trajectory tracking errors for the quadrotor.

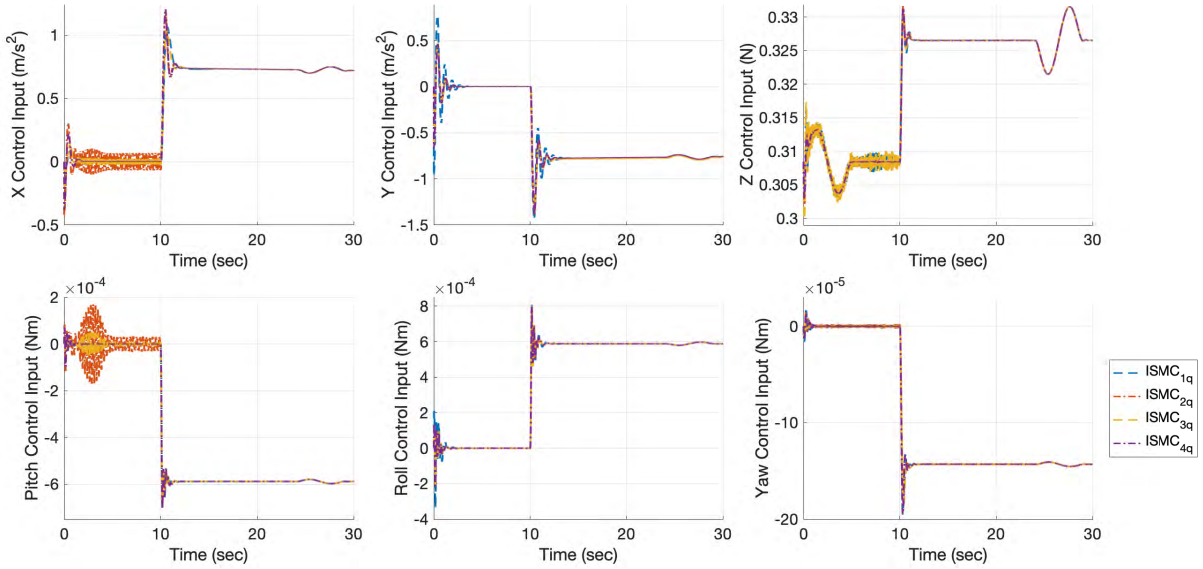


Figure 7.6: ISMC simulation: control inputs for the quadrotor.

As can be seen from Figure 7.5, for the X DoF, ISMC_{1q} has a very high trajectory tracking error when the fault is experienced from a time of 10 seconds. The error quickly reduces and remains close to zero after a time of about 19 seconds. For ISMC_{3q}, the X DoF has a very high error that reduces very slowly. The X DoF for ISMC_{2q} has a low error that reduces slowly after the fault. In this case, ISMC_{2q} has a desirable error response compared to the other two controllers for the X DoF. From Figure 7.6, it can be seen that the control inputs for the X DoF (and the related pitch DoF) for ISMC_{2q} and ISMC_{3q} experience high chattering. Even though the Mellinger controller is used for the roll and pitch DoFs and their controller gains are not tuned, due to the quadrotor being an underactuated system, with coupled dynamics, the pitch and roll results are still affected by the other DoFs. The X and Y DoFs, in particular, affect the roll and pitch

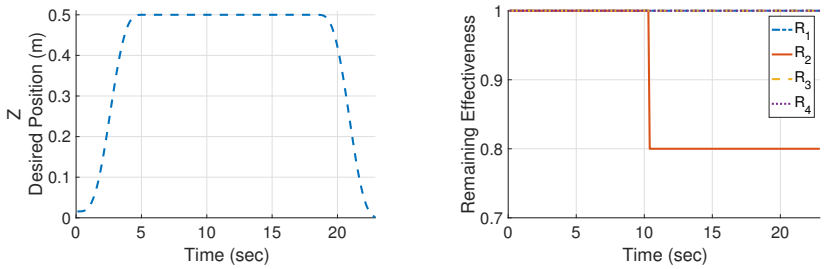
DoF results as their virtual control inputs are used to compute the desired roll and pitch angles. This is why high chattering is observed for the pitch DoF, along with the X DoF, for ISMC_{2q} and ISMC_{3q}. From Chapter 3, it was shown that high k_1 values result in high chattering. Lowering the k_1 value significantly reduces the chattering and slightly increases the error as shown by the X and pitch DoF results for ISMC_{4q}, which uses the same X DoF controller gains as ISMC_{2q}, with a lower value for the k_1 gain. Following a similar process, the controller gains for the remaining DoFs are adjusted as needed and different DoF controller gains sets can also be combined. Each set of controller gains is then tested on the physical quadrotor. The same process is conducted for each of the SMCs that are used for the physical experiments.

Care needs to be taken when testing the controller gains obtained from the MOPSO controller gains-tuning process as the simulated system is a simplification of the physical system. Thus, some adjustments to the tuned controller gains may need to be made and in some cases, controller gains obtained from the simulations may not be suitable at all for the physical system. Testing the gains on the physical system should be done carefully to avoid damage to the system. This can be done by building a test rig to isolate some of the system's DoFs, allowing for one or a small number of the the DoFs to be tested prior to actual flight tests. In this case, since the Crazyflie quadrotor is an open source system, advantage is taken of one of the on-board controllers, namely the Mellinger controller as it has a similar control structure to that used in this thesis. The DoFs of the SMCs are tested sequentially by replacing the relevant Mellinger DoF controller with the SMC controller for that DoF. Flight tests are conducted and the controller gains for the DoF are adjusted as necessary. For example, it was found that lower k_2 gains were needed for the ISMCs of the X, Y, and Z DoFs and that higher k_1 gains were required for the physical quadrotor to account for the actual disturbances experienced and for other unmodelled effects on the quadrotor. Although there is still some risk of the system crashing, this process makes it easier to find and correct the controller gain that may be causing unwanted behaviour. Tests are first conducted using the Crazyflie quadrotor before being conducted on the Crazyflie quadrotor-based aerial manipulator. This is because there is greater risk of damage occurring with the aerial manipulator system.

7.3 Quadrotor Fault-Tolerant Control Experiments

For the physical FTC experiments conducted, the quadrotor must hover with a fault in one rotor. Due to the limitations of the battery and the Crazyflie motors, full area coverage experiments are not conducted. The quadrotor must first move to a specified height and then maintain that height. A fault in one of the rotors is triggered by reducing the rotor effectiveness. This is accomplished through modifying the Crazyflie firmware such that the rotor's desired PWM value is set to a percentage of the actual desired PWM value. Figure 7.7a shows the desired height trajectory and Figure 7.7b shows the rotor remaining effectiveness were a value of 1 is full effectiveness and a value of 0 is complete

failure of the rotor. As chattering and rotor faults can lead to the rotors quickly wearing out, faults in different rotors are applied for different sets of experiments.



(a) Desired trajectory.

(b) Rotor remaining effectiveness, showing rotor R_2 loss of effectiveness.

Figure 7.7: Crazyflie quadrotor experiment: hovering with a rotor fault.

First, passive FTC experiments are conducted with the controllers. Active FTC experiments are then conducted assuming a perfect FDD unit. Finally, data is collected for training of neural networks for the FDD of the rotors, and active FTC experiments are conducted with the neural network-based FDD unit. Six controllers are investigated for the FTC of the Crazyflie quadrotor, and are compared to the on-board Mellinger controller. The six controllers are the ISMC, ISMC_h, AISMC_p, AISMC_ph, AISMC_r, and AISMC_rh. The discontinuous ISMC is selected in order to compare the chattering reduction achieved by the other five controllers. The continuous ISMC_h is used to determine the level of improvements obtained from using the adaptive controllers compared to the non-adaptive controller. Continuous and discontinuous AISMCs are selected based on the results obtained in Chapter 6 and are used to determine the benefits obtained from using the continuous versions of the AISMCs. From the controller gains tuning and selection process described in Section 7.2, controller gains that have good trajectory tracking, with low chattering, and low control inputs are selected. The gains presented in Table 7.2 are selected for the Crazyflie quadrotor.

7.3.1 Discontinuous versus continuous passive fault-tolerant control experiments

Passive FTC experiments using the Crazyflie quadrotor are first performed to verify the chattering reduction of the continuous controllers versus the discontinuous controllers, with a fault induced in rotor R_3. The quadrotor is an under-actuated system, with the X and Y DoFs providing virtual control inputs which are used to compute the desired roll and pitch angles to track a specified X and Y trajectory. This means that even though the same Mellinger roll and pitch controllers are used for each experiment, the results obtained for the roll and pitch DoFs will differ based on the X and Y controllers used. Figure 7.8 shows the trajectory tracking errors for the ISMC_h and the ISMC and Figure 7.9 shows the control inputs. Only control inputs for the controllable DoFs are shown,

Table 7.2: Crazyflie FTC controller gains.

Controller	Gain	X	Y	Z	ψ
All	k_2	5	5	7	16.8
	c_1	2	2	2	6.9
	c_2	0.3	0.3	0.8	1.5
ISMC	k_1	0.15	0.1	0.3	0.6
ISMC_h	k_1	0.6	0.4	0.9	1.2
Continuous	λ	0.3	0.3	0.3	0.3
AISMC_p	η_α	0.01	0.01	0.022	2×10^{-3}
	η_β	1×10^{-5}	1×10^{-5}	1×10^{-5}	1×10^{-5}
	μ	0.05	0.05	0.08	0.15
	α_m	5×10^{-4}	5×10^{-4}	1×10^{-3}	1×10^{-3}
	α_{max}	0.3	0.3	0.5	1
	α_i	0.05	0.05	0.3	0.3
AISMC_ph	η_α	0.015	0.015	0.022	3×10^{-3}
	η_β	1×10^{-5}	1×10^{-5}	1×10^{-5}	1×10^{-5}
	μ	0.05	0.05	0.08	0.15
	α_m	5×10^{-3}	5×10^{-3}	0.01	0.01
	α_{max}	2	2	4	4
	α_i	0.3	0.3	0.9	0.6
AISMC_r	ρ_0	0.5	0.5	0.5	0.5
	ρ_1	0.5	0.5	0.5	0.5
	r_{0m}	1×10^{-3}	1×10^{-3}	1×10^{-3}	1×10^{-3}
	r_{1m}	1×10^{-7}	1×10^{-7}	1×10^{-7}	1×10^{-7}
AISMC_rh	ρ_0	0.1	0.1	0.1	0.2
	ρ_1	0.1	0.1	0.1	0.2
	r_{0m}	1×10^{-3}	1×10^{-3}	1×10^{-3}	1×10^{-3}
	r_{1m}	1×10^{-7}	1×10^{-7}	1×10^{-7}	1×10^{-7}

that is the Z, roll, pitch and yaw DoFs, since the X and Y are virtual control inputs used to determine the desired roll and pitch angles. For the trajectory tracking of the X DoF, the ISMC_h shows a slight improvement in trajectory tracking error as can be seen from Figure 7.8. The most significant difference in trajectory tracking error for the two controllers is for the pitch DoF. The ISMC has highly-oscillatory pitch angle errors. For the Z DoF control inputs, both controllers experience high-frequency oscillations as can be seen from Figure 7.9 . The roll and pitch DoFs for the ISMC experience some chattering as evidenced by both the trajectory tracking errors and control inputs.

Figure 7.10 shows the trajectory tracking errors for the AISMC_p and AISMC_ph and Figure 7.11 shows the control inputs. For the X DoF, the AISMC_p experiences a large error at a time of about 17 seconds to 20 seconds as can be seen from Figure 7.10. For the Z DoF, the AISMC_ph experiences an overshoot after the rotor fault is induced. The AISMC_p

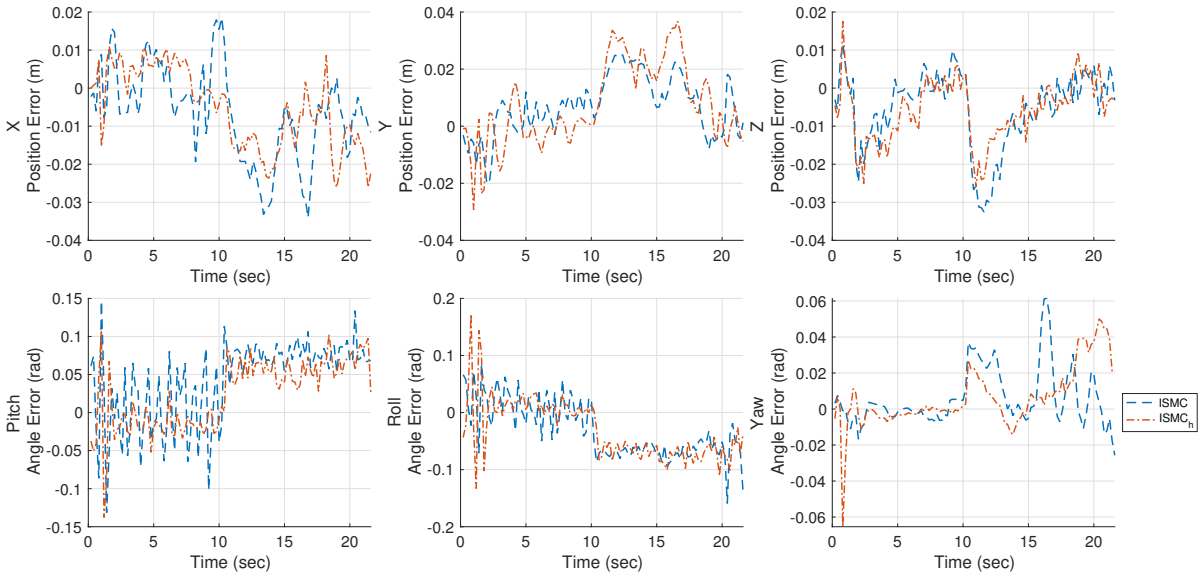


Figure 7.8: ISMC and ISMC$_h$ passive FTC trajectory tracking errors for the quadrotor.

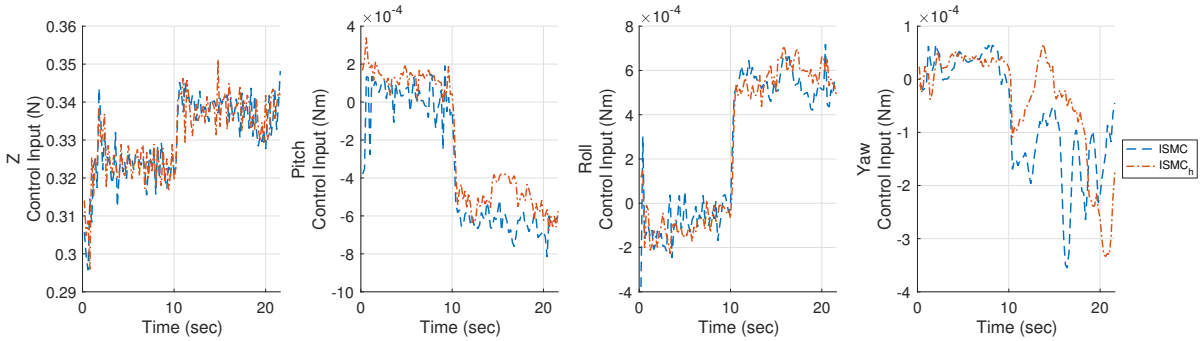


Figure 7.9: ISMC and ISMC$_h$ passive FTC control inputs for the quadrotor.

has lower errors for the yaw DoF. Both controllers experience some chattering for the Z DoF as can be seen from Figure 7.11. The roll DoF for the AISMC$_p$ experiences slight chattering from a time of about 0 second to 5 seconds as can be seen from Figure 7.11.

Figure 7.12 shows the trajectory tracking errors for the AISMC$_r$ and AISMC$_{rh}$ and Figure 7.13 shows the control inputs. As can be seen from Figure 7.12, for the Z DoF, the AISMC$_{rh}$ experiences an overshoot before the rotor fault is induced. For the yaw DoF the AISMC$_r$ has good performance before the fault is induced, thereafter it experiences some high error values. As can be seen from Figure 7.13, the AISMC$_{rh}$ has slightly lower chattering than the AISMC$_r$ for the Z DoF and pitch DoF.

The results of the physical experiments show that the ISMC$_h$ is able to reduce the chattering experienced by the ISMC. This is especially evident by the reduced pitch error oscillations experienced by the ISMC$_h$ compared to the ISMC. Although slightly better chattering reduction is experienced by the AISMC$_{rh}$ and AISMC$_{ph}$ compared to the AISMC$_r$ and AISMC$_p$ respectively, the difference in the chattering reduction is not very significant. The AISMC$_r$ and AISMC$_p$ have one less parameter to tune per DoF compared to the AISMC$_{rh}$ and AISMC$_{ph}$, which simplifies the gain-tuning process, and they

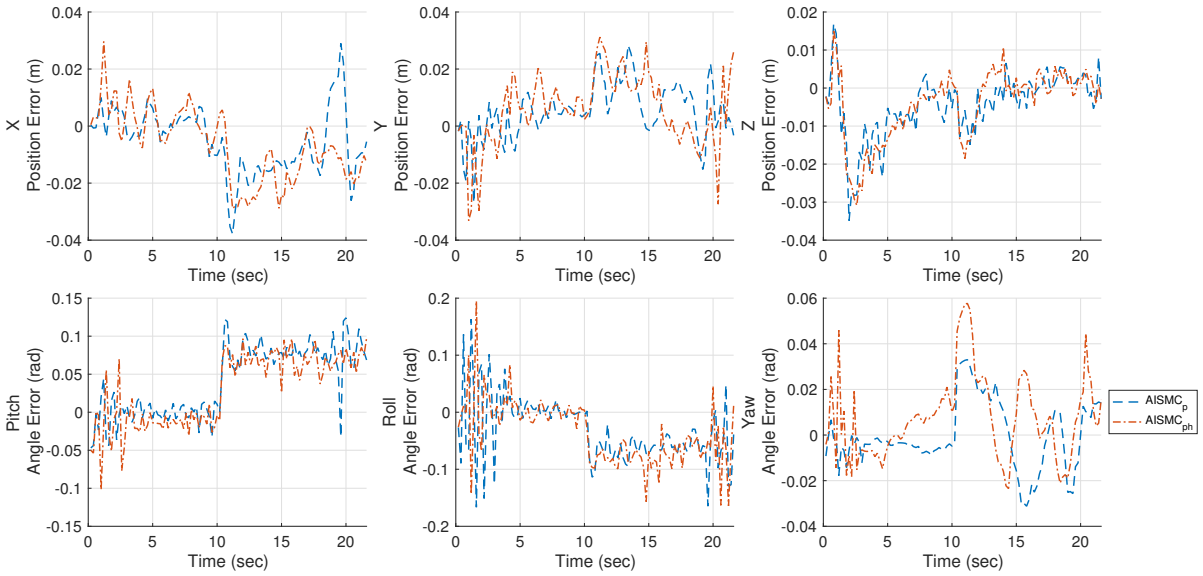


Figure 7.10: AISMC_p and AISMC_ph passive FTC trajectory tracking errors for the quadrotor.

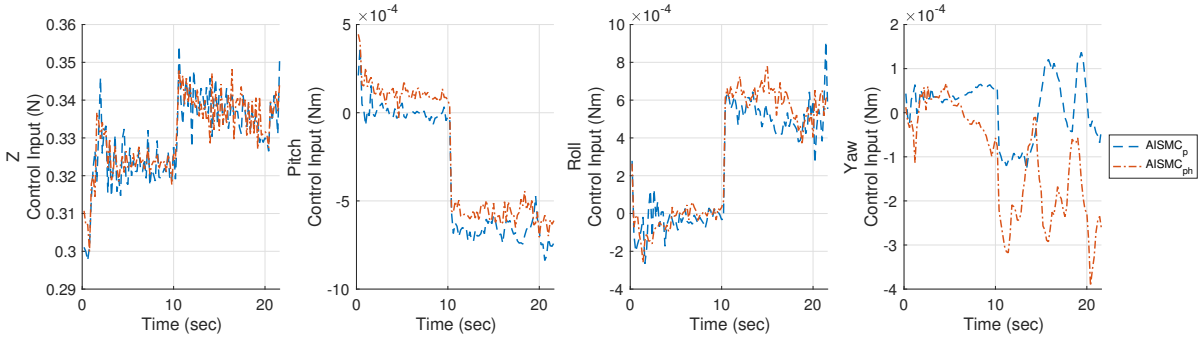


Figure 7.11: AISMC_p and AISMC_ph passive FTC control inputs for the quadrotor.

provide sufficient chattering reduction. Further experiments are therefore conducted with the AISMC_r and AISMC_p.

7.3.2 Quadrotor passive fault-tolerant control experiments

Passive FTC experiments are conducted on the quadrotor using the AISMC_r and AISMC_p. The ISMC is included as a benchmark for measuring the chattering reduction. The on-board Mellinger controller is also used as a benchmark to compare the trajectory tracking errors. The fault is induced in rotor R_2. Figure 7.14 shows the position and angle errors for the passive FTC experiments and Figure 7.15 shows the control inputs. As can been seen from Figure 7.14, the AISMCs are better able to recover from the fault than the on-board Mellinger controller. It should, however be noted that the Mellinger controller gains were not tuned from their original values. Tuning the Mellinger controller gains could possibly improve the controller's FTC performance. From Figures 7.14 and 7.15 it

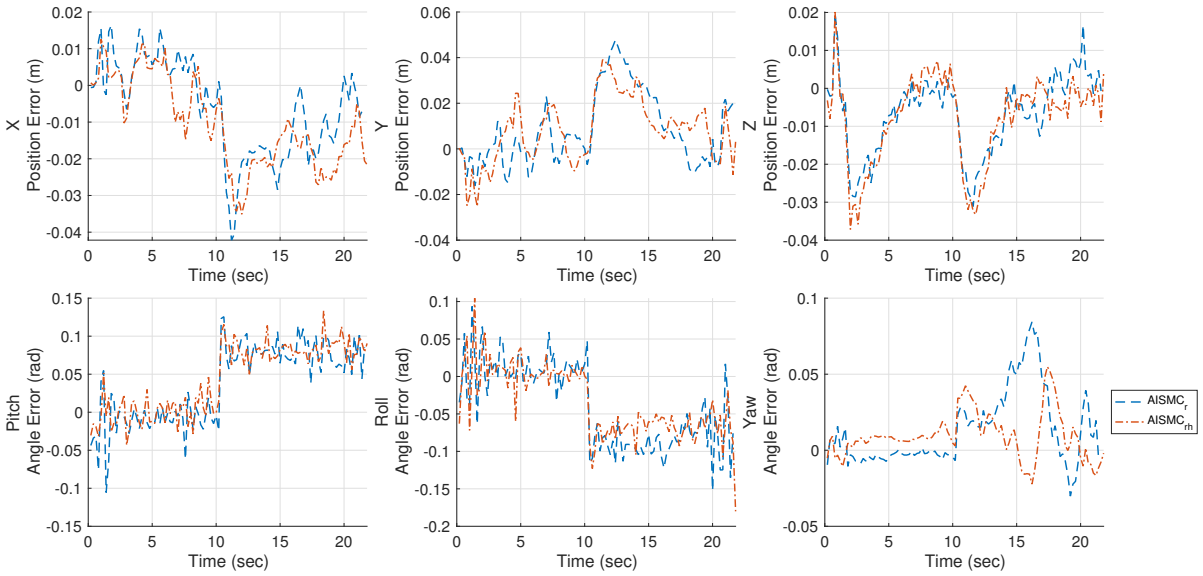


Figure 7.12: AISMC$_r$ and AISMC$_{rh}$ FTC trajectory tracking errors for the quadrotor.

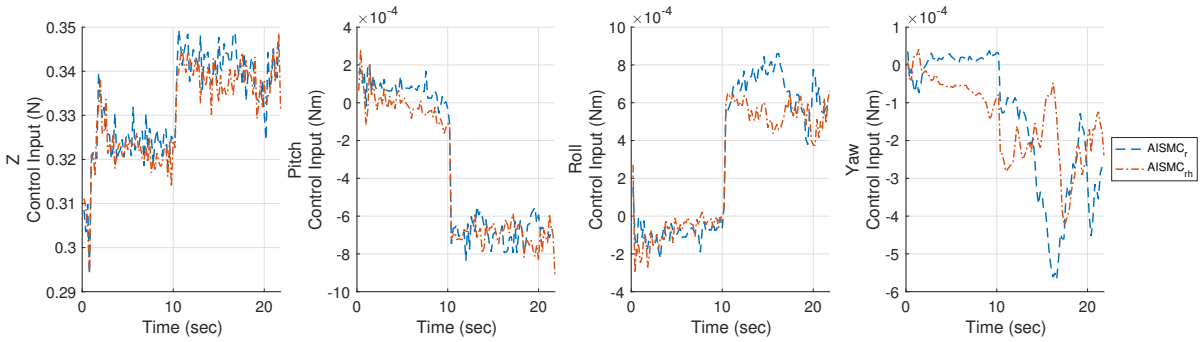


Figure 7.13: AISMC$_r$ and AISMC$_{rh}$ FTC control inputs for the quadrotor.

can be seen that the AISMCs manage to reduce chattering for this experiment compared to the ISMC. This is most evident for the roll and pitch DoFs.

7.3.3 Quadrotor active fault-tolerant control experiments

Figure 7.16 shows the position and angle errors for the active FTC experiments with a perfect FDD unit, and Figure 7.17 shows the control inputs. As can been seen from Figure 7.16, the AISMC$_r^{AFTC}$ and AISMC$_p^{AFTC}$ are better able to recover from the fault than the on-board Mellinger controller as they have lower trajectory tracking errors after the fault occurrence, particularly for the Z and yaw DoFs. From Figures 7.16 and 7.17 it can again be seen from the roll and pitch DoFs that the AISMC$_r^{AFTC}$ and AISMC$_p^{AFTC}$ manage to reduce chattering for this experiment compared to the ISMC.

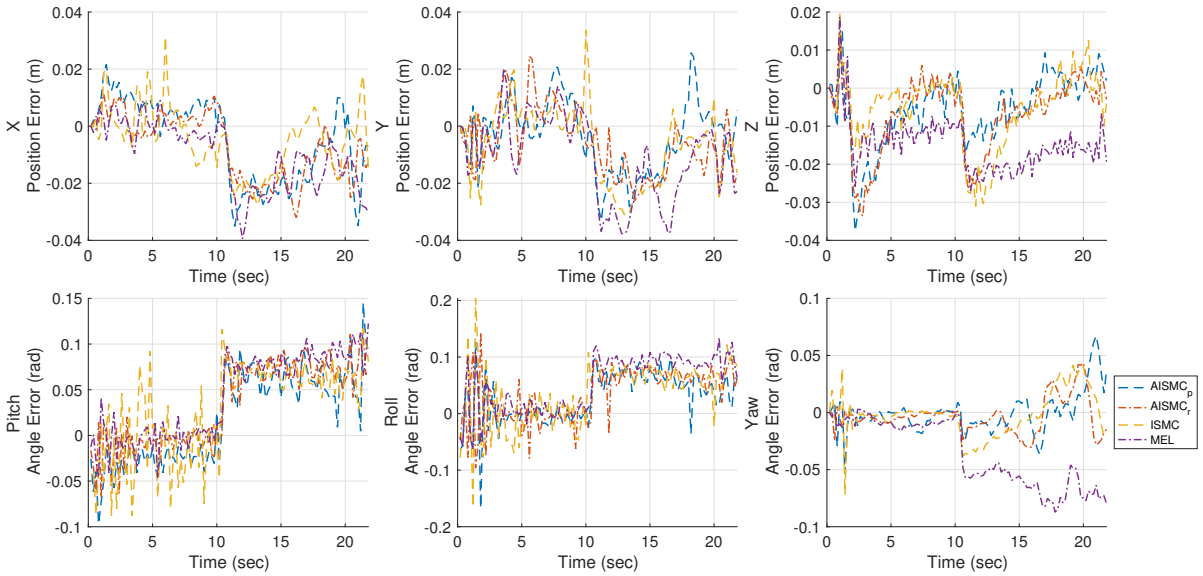


Figure 7.14: Passive FTC trajectory tracking errors for the quadrotor.

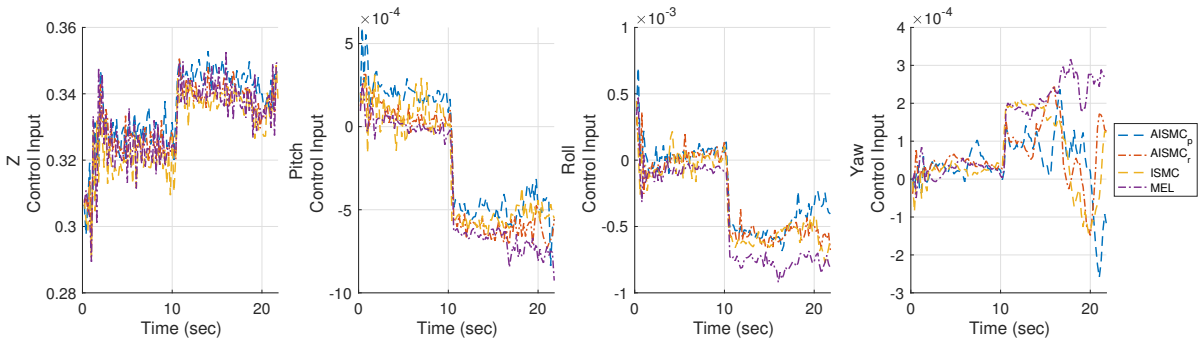


Figure 7.15: Passive FTC control inputs for the quadrotor.

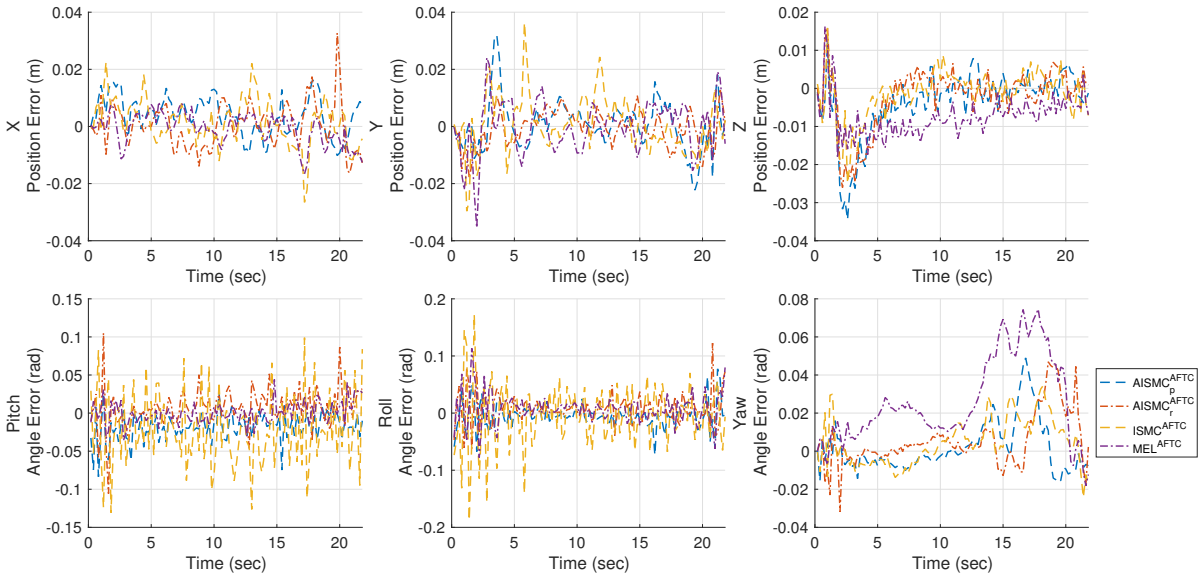


Figure 7.16: Active FTC trajectory tracking errors for the quadrotor.

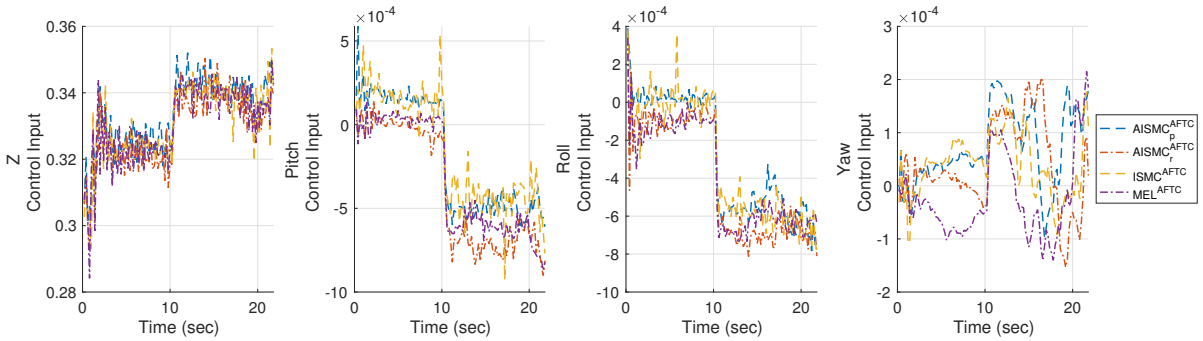


Figure 7.17: Active FTC control inputs for the quadrotor.

7.3.4 Quadrotor neural network-based FDD

Different types and configurations of neural networks are considered for the FDD unit as described in Chapter 3. This includes a two neural network structure, one neural network to estimate the system accelerations, and the second neural network which uses the actual versus the estimated accelerations to estimate the fault magnitude as shown in Figure 3.3. The other FDD structure is one that performs the entire FDD for all four rotors using a single neural network as shown in Figure 3.4. Another FDD structure has four neural networks, one neural network each for the FDD of a single rotor as shown in Figure 3.5. The inputs to and outputs from the neural network-based FDD units are shown in Figure 7.3. Two types of neural network are compared, the feed-forward and cascade-forward networks shown in Figure 3.6 and Figure 3.7 respectively. Two training methods are used for the neural networks, the Bayesian regularisation algorithm and the Levenberg-Marquardt algorithm. The data for training of the neural networks was obtained by logging the relevant data from the quadrotor while it performed several flights with different rotor fault magnitudes for each of the four rotors. The best results obtained from the neural network training for each type and structure of the neural network-based FDD units are shown in Figures 7.18 to 7.22.

The FDD unit comprised of two neural networks requires one neural network for the system identification of the quadrotor which outputs the estimated accelerations as described in Section 3.2.3 and shown in Figure 3.3. The second neural network then uses the difference between the actual and estimated accelerations to estimate the rotor faults. The system identification neural network requires a large amount of training data and a large network size. A variety of manoeuvres are performed to obtain the system identification data. Feed-forward and cascade-forward networks with three hidden layers of size 40 each are used for the system identification neural network. The Levenberg-Marquardt training algorithm produces better results than the Bayesian regularisation algorithm in this case. The feed-forward and cascade-forward FDD neural networks have two hidden layers of size 20 each. Figures 7.18 and 7.19 show the training results of the different system identification networks and Figure 7.20 shows the results of the different FDD networks. The final FDD results produced from the neural network training are poor. More training data could be used or larger neural network sizes used to improve the results.

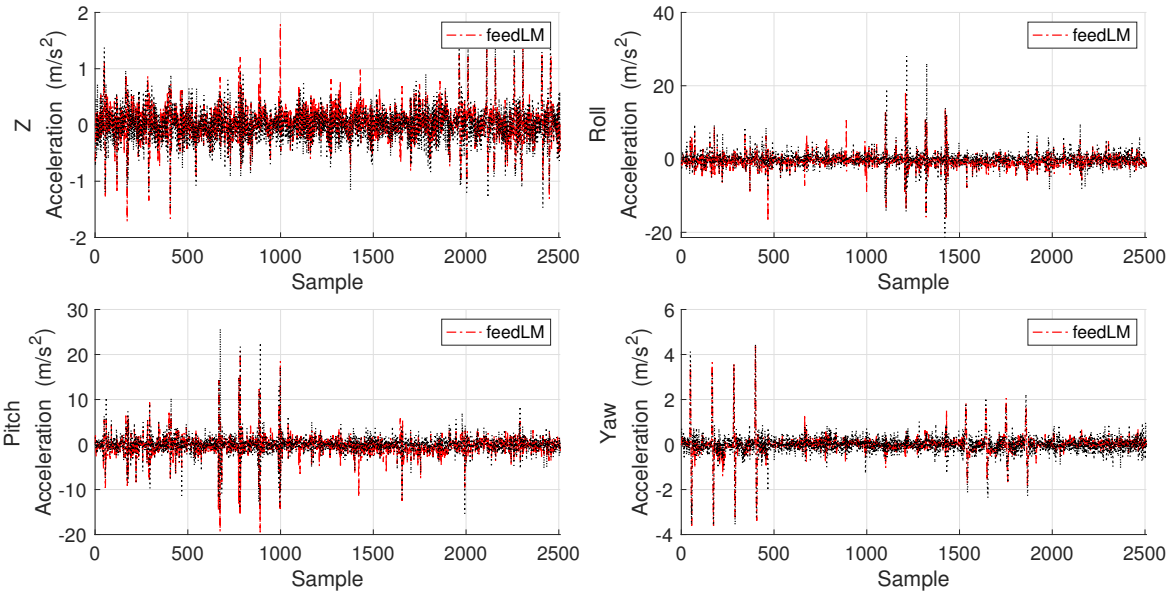


Figure 7.18: Training results for quadrotor system identification feed-forward neural networks. Key: feed - feed-forward neural network; LM - Levenberg-Marquardt training.

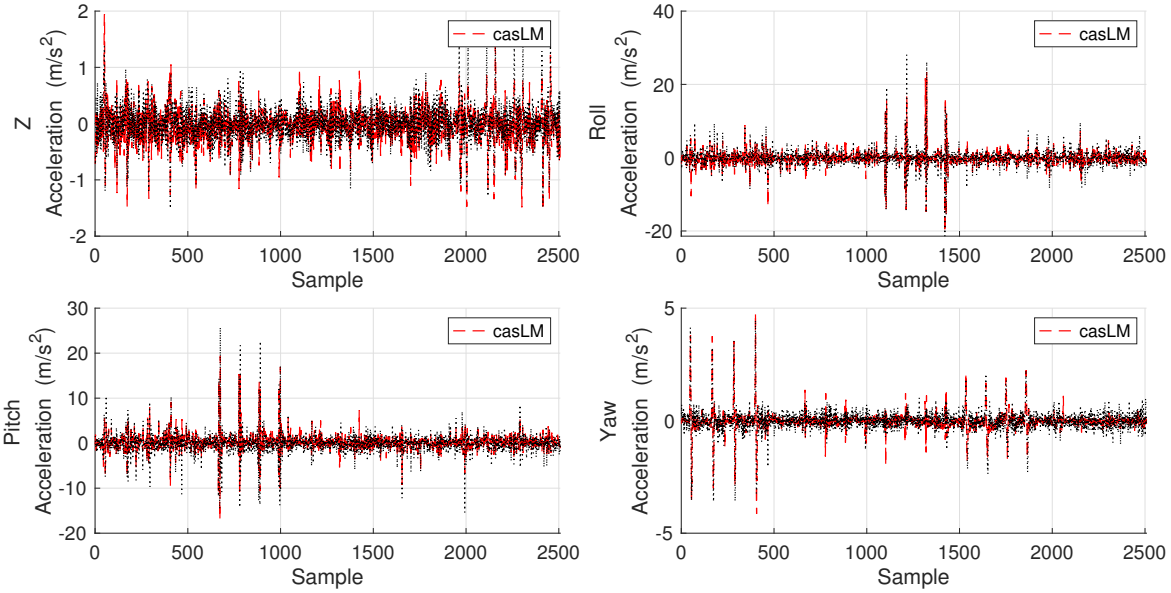


Figure 7.19: Training results for quadrotor system identification cascade-forward neural network. Key: cas - cascade-forward neural network; LM - Levenberg-Marquardt training.

For the FDD unit structured with one neural network producing four outputs as described in Section 3.2.3 and shown in Figure 3.4, the neural network training results are shown in Figure 7.21. The feed-forward and cascade-forward networks each consist of three hidden layers of size 30 each. The networks produce acceptable results, although for rotor R_3 there is a sample point that is badly estimated. For both network types, the Bayesian regularisation training algorithm produces the best results.

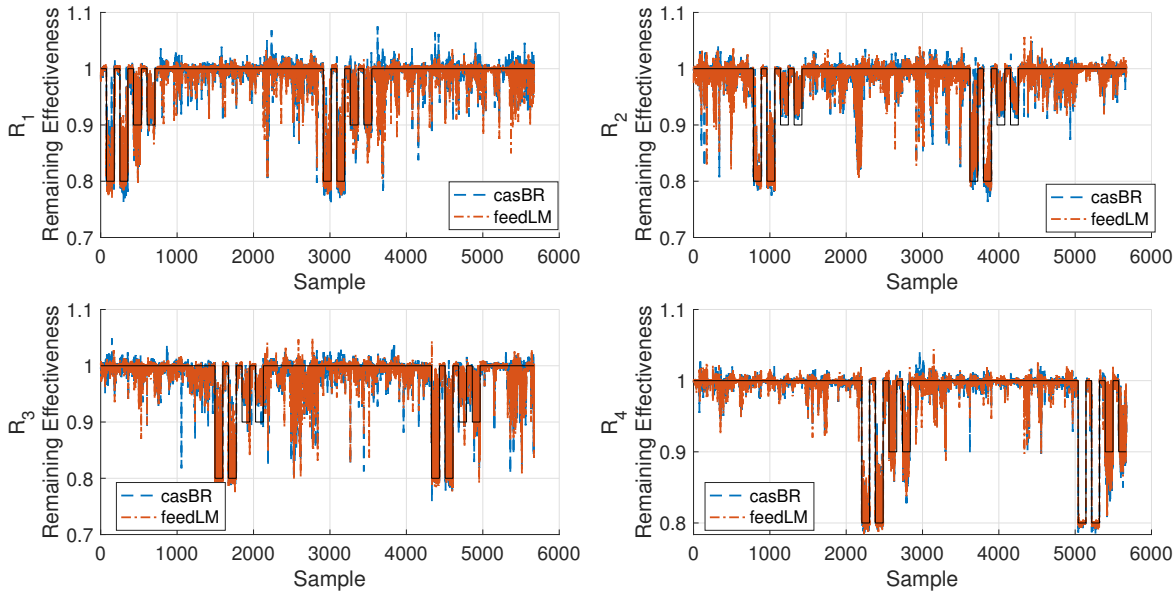


Figure 7.20: Training results for quadrotor FDD neural networks. Key: cas - cascade-forward neural network; feed - feed-forward neural network; LM - Levenberg-Marquardt training; BR - Bayesian regularisation training.

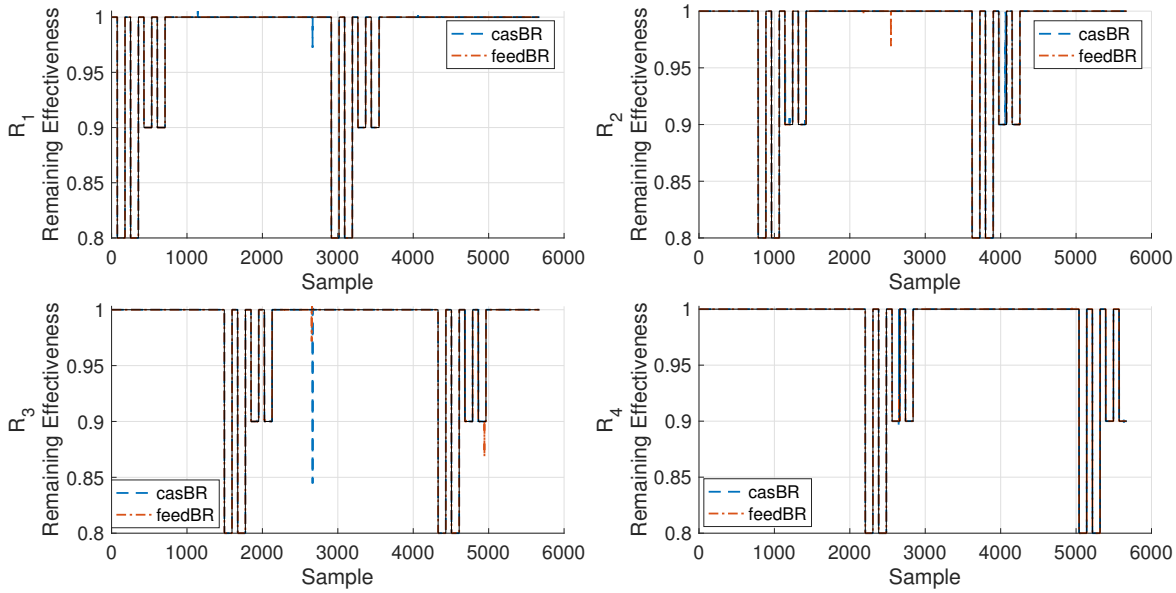


Figure 7.21: Training results for quadrotor FDD units comprised of a single neural network. Key: cas - cascade-forward neural network; feed - feed-forward neural network; BR - Bayesian regularisation training.

For the FDD unit structured with four neural networks with a single output each as described in Section 3.2.3 and shown in Figure 3.5, the neural network training results are shown in Figure 7.22. The feed-forward and cascade-forward networks each consist of two hidden layers of size 20 each. The networks produce good FDD results. For the cascade-forward neural network, the Bayesian regularisation training algorithm produces the best results. For the feed-forward network, both the Bayesian regularisation and

Levenberg-Marquardt training algorithms produce good results. However, on average for the number of training runs that were conducted, the Bayesian regularisation produces better results than the Levenberg-Marquardt training algorithm.

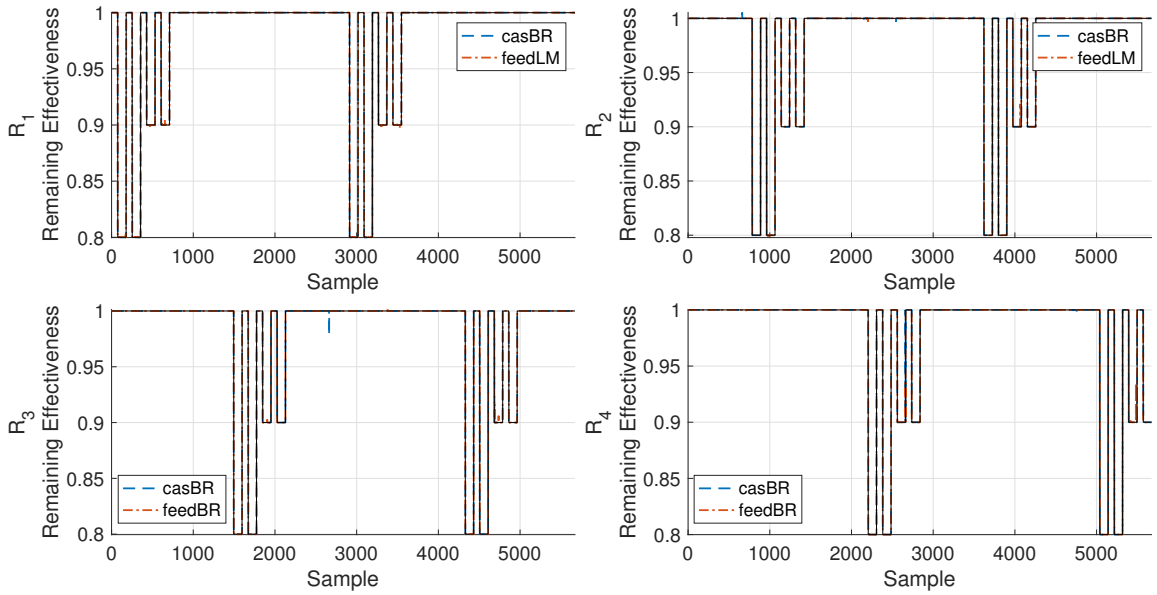


Figure 7.22: Training results for quadrotor FDD units comprising four neural networks. Key: cas - cascade-forward neural network; feed - feed-forward neural network; LM - Levenberg-Marquardt training; BR - Bayesian regularisation training.

First, the neural network-based FDD units are trained and tested for the Crazyflie quadrotor and the best network structure is then used for training and testing for the Crazyflie quadrotor-based aerial manipulator. When testing the neural networks on the physical system, the results show that larger networks and networks with more than one hidden layer result in poor performance of the overall system due to the computational limits of the Crazyflie quadrotor. Thus, a single hidden layer feed-forward neural network-based FDD unit, to estimate the remaining effectiveness of a single rotor, is implemented on the physical system. The feed-forward neural network is chosen as it has a simpler structure than the cascade-forward neural network. For training of the neural network for the FDD of a single rotor, the Levenberg-Marquardt training produced good results. Figure 7.23 shows the training results for FDD of rotor R_2, using a feed-forward neural network with a single hidden layer of size 15.

7.3.5 Quadrotor active fault-tolerant control experiments with neural network-based FDD

Figure 7.24 shows the quadrotor position and angle errors for the active FTC experiments with a neural network-based FDD unit, and Figure 7.25 shows the control inputs. As can been seen from Figure 7.38, the $\text{AISMC}_{\text{r}}^{\text{AFTCnn}}$ and $\text{AISMC}_{\text{p}}^{\text{AFTCnn}}$ are better able to recover from the fault than the on-board Mellinger controller. From Figure 7.25 it can

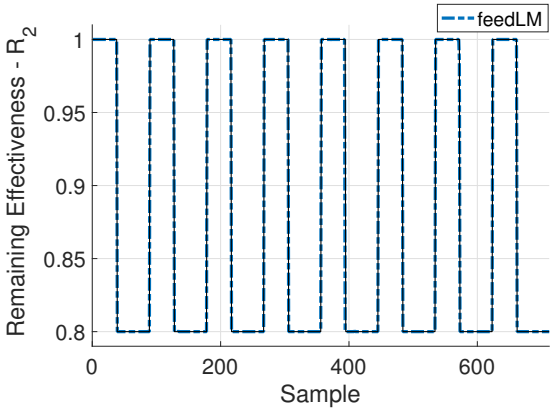


Figure 7.23: Training results for a quadrotor single rotor FDD unit comprised of one neural network. Key: feed - feed-forward neural network; LM - Levenberg-Marquardt training.

be seen that the AISMC$_{\mathrm{r}}^{\mathrm{AFTCnn}}$ and AISMC$_{\mathrm{p}}^{\mathrm{AFTCnn}}$ manage to reduce chattering for this experiment when compared to the ISMC$^{\mathrm{AFTCnn}}$.

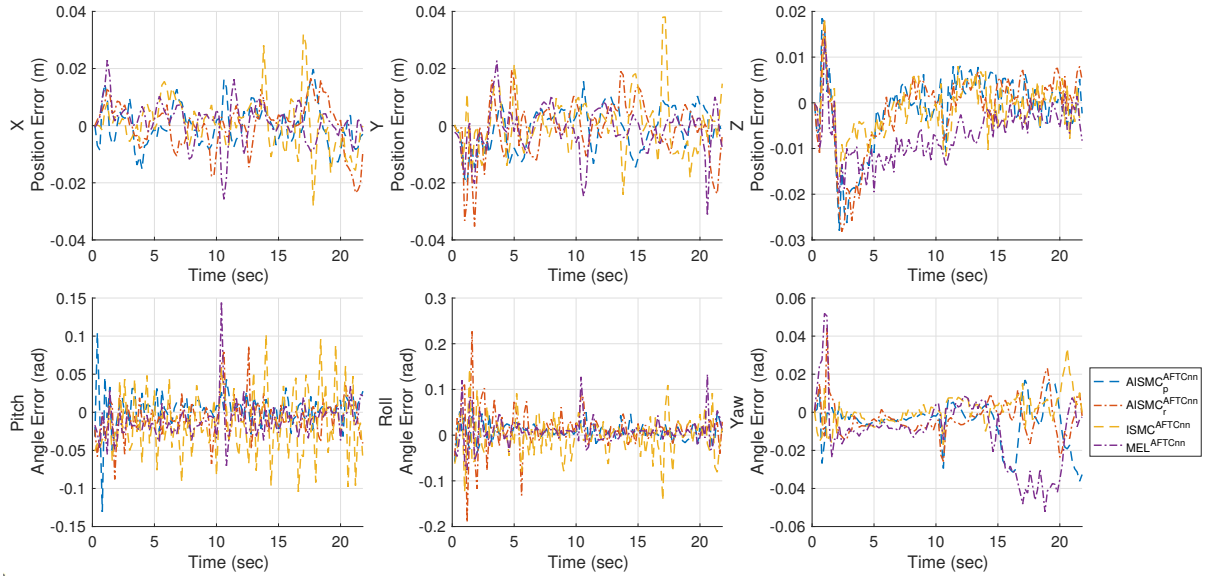


Figure 7.24: Active FTC with neural network-based FDD: trajectory tracking errors for the quadrotor.

Figure 7.26 shows the output of the neural network-based FDD unit. From the figure it can be seen that the FDD unit successfully detects the remaining effectiveness of rotor R_2. There is a slight delay before the fault is fully detected due to the filtering of the neural network outputs. The FDD unit does not perform as well for the ISMC$^{\mathrm{AFTCnn}}$ as can be seen by the spikes after the fault occurs where the rotor fault is underestimated.

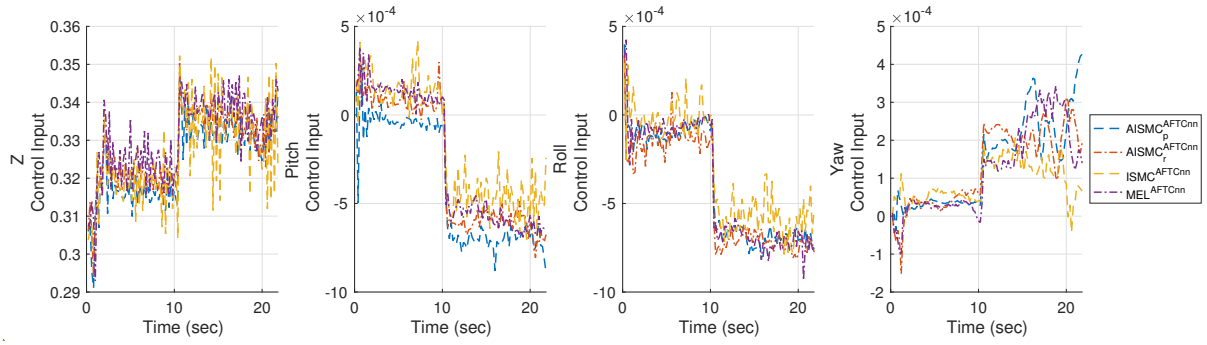


Figure 7.25: Active FTC with neural network-based FDD: control inputs for the quadrotor.

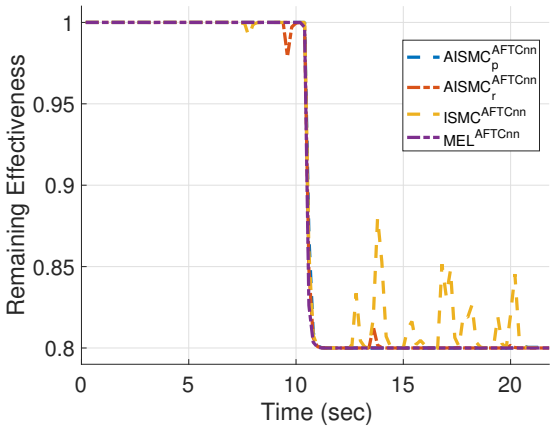


Figure 7.26: Neural network-based FDD outputs for the quadrotor.

7.3.6 Quadrotor passive and active fault-tolerant control comparison

The error results for passive FTC and the two active FTCs versions of each controller are now compared for the quadrotor. Figure 7.27 shows the passive and active FTC trajectory tracking error results for the Mellinger controller. Figure 7.28 shows the trajectory tracking error results for the ISMCs. Figure 7.29 shows the trajectory tracking error results for the AISMCs with Plestan adaptation, and Figure 7.30 shows the error results for the AISMCs with the Roy adaptation. From the figures for each of the controllers, it can be seen that the passive FTC versions of the controllers have the worst performance, taking a long time to recover from the rotor fault. The active FTC versions of the controllers with a perfect FDD have the best results for all of the controllers. The active FTC versions of the controllers that make use of the neural network-based FDD unit significantly improve on the results of the passive FTCs.

For the Mellinger controllers, as can be seen from Figure 7.27, for the passive FTC, MEL, the X and Y DoFs have large trajectory tracking errors compared to the active FTCs, MEL^{AFTC} and $\text{MEL}^{\text{AFTCnn}}$, when the fault is experienced. For the active controller making using of the neural network-based FDD unit, $\text{MEL}^{\text{AFTCnn}}$, there is an increase in

error for X and Y DoFs when the fault occurs, and they then quickly recover from the error. For the Z DoF, the passive controller, unlike the active controllers, has a distinct increase in error when the fault occurs and it recovers from the fault at a slow rate. For the roll and pitch DoFs, the passive controllers experience a large increase in error when the fault occurs and they do not recover from the error. Although the distinct increase in error for the yaw DoF is only experienced for the passive controller when the fault occurs, both active controllers also have poor performance after the fault occurs. The active controller with the neural network-based FDD unit gives the best yaw results for this experiment.

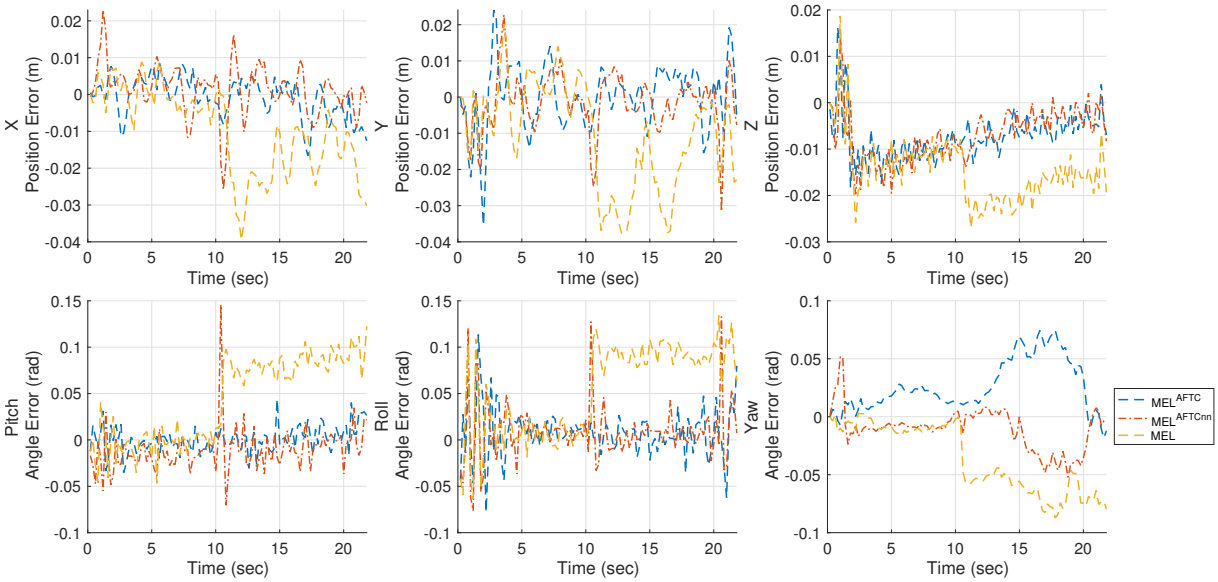


Figure 7.27: Mellinger passive and active FTC for the quadrotor.

For the ISMCs, as can be seen from Figure 7.28, the passive and active FTCs for the X and Y DoFs all experience trajectory tracking errors when the fault is experienced. The passive controller, ISMC, takes a longer time to recover from the X and Y DoF errors, whereas the active controllers, ISMC$^{\text{AFTC}}$ and ISMC$^{\text{AFTCnn}}$, experience some spikes in error after the fault occurs. For the Z DoF, the passive controller has a distinct large increase in error when the fault occurs and it recovers from the fault within 5 seconds. The Z DoF active controller with the neural network-based FDD unit, ISMC$^{\text{AFTCnn}}$, has a small increase in error when the fault occurs and it very quickly recovers from the error. For the roll and pitch DoFs, the passive controllers experience a large increase in error when the fault occurs and they do not recover from error. Chattering is experienced for roll and pitch DoFs for each of the ISMCs. For the passive controller, there is a decrease in chattering after the fault. This is due to the fact that the fault causes the X and Y DoFs to move away from their sliding surfaces, hence chattering is decreased for the X and Y DoFs, and in turn for the roll and pitch DoFs. A distinct large increase in error is experienced for the passive controller of the yaw DoF and a small increase in error is experienced for the active controller with the neural network-based FDD unit, ISMC$^{\text{AFTCnn}}$, when the fault occurs. The yaw DoF quickly rovers from the fault for the ISMC$^{\text{AFTCnn}}$.

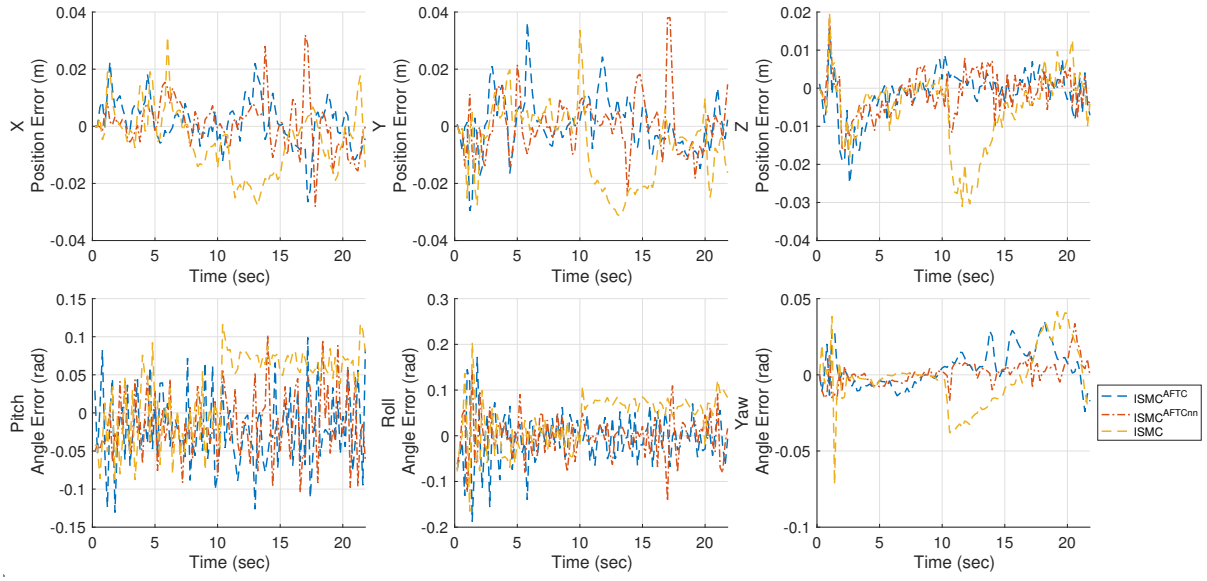


Figure 7.28: ISMC passive and active FTC trajectory tracking errors for the quadrotor.

For the AISMCs with the Plestan adaptation method, as can be seen from Figure 7.29, for the X and Y DoFs the passive FTC, AISMC_p, produces large trajectory tracking errors compared to the active FTCs, $\text{AISMC}_\text{p}^\text{AFTC}$ and $\text{AISMC}_\text{p}^\text{AFTCnn}$, when the fault is experienced. For the Z DoF, the passive controller has a distinct increase in error when the fault occurs and it recovers from the fault at a fast rate. The active controllers for the Z DoF both experience a slight overshoot after the fault occurs. For the roll and pitch DoFs, the passive controllers experience a large increase in error when the fault occurs and they do not recover from error. The yaw DoF has varied results with all of the controllers experiencing poor performance after the fault occurs.

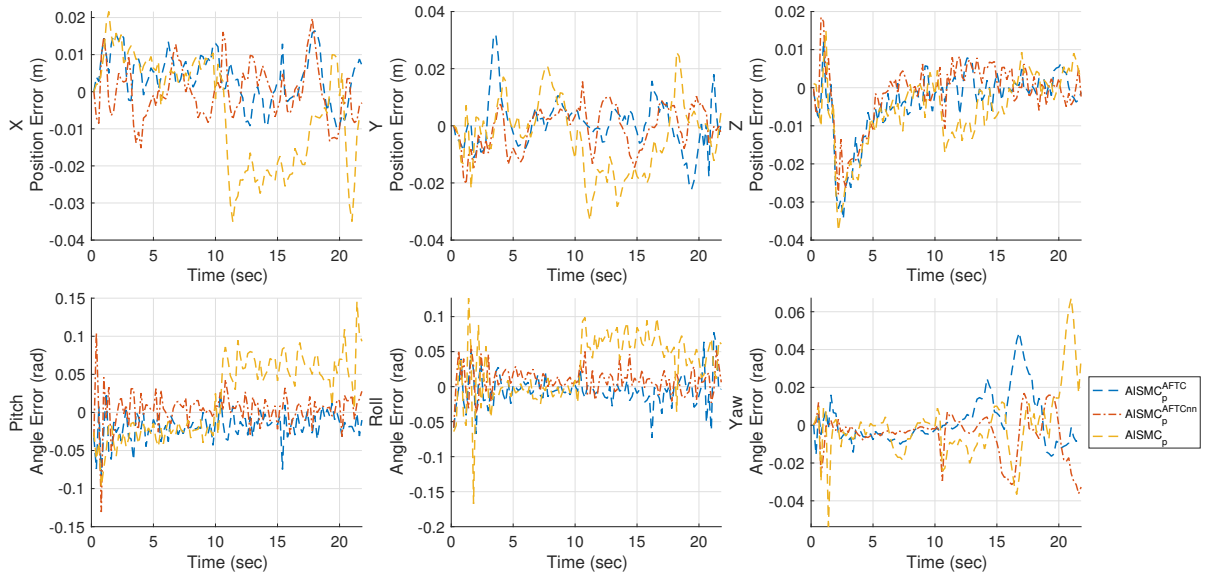


Figure 7.29: AISMC Plestan adaptation passive and active FTC trajectory tracking errors for the quadrotor.

For the AISMCs with the Roy adaptation method, as can be seen from Figure 7.30, for the passive FTC, AISMC_r, the X and Y DoFsexperience large trajectory tracking errors compared to the active FTCs, $\text{AISMC}_\text{r}^{\text{AFTC}}$ and $\text{AISMC}_\text{r}^{\text{AFTCnn}}$, when the fault is experienced. For the Z DoF, the passive controller, unlike the active controllers, has a distinct increase in error when the fault occurs and it recovers from the fault within a time of about 5 seconds. For the roll and pitch DoFs, the passive controllers experience a large increase in error when the fault occurs and they do not recover from the error. For the yaw DoF all of the controllers have poor performance after the fault occurs. The active controller with the neural network-based FDD unit, $\text{AISMC}_\text{r}^{\text{AFTCnn}}$, gives the best yaw results for this experiment.

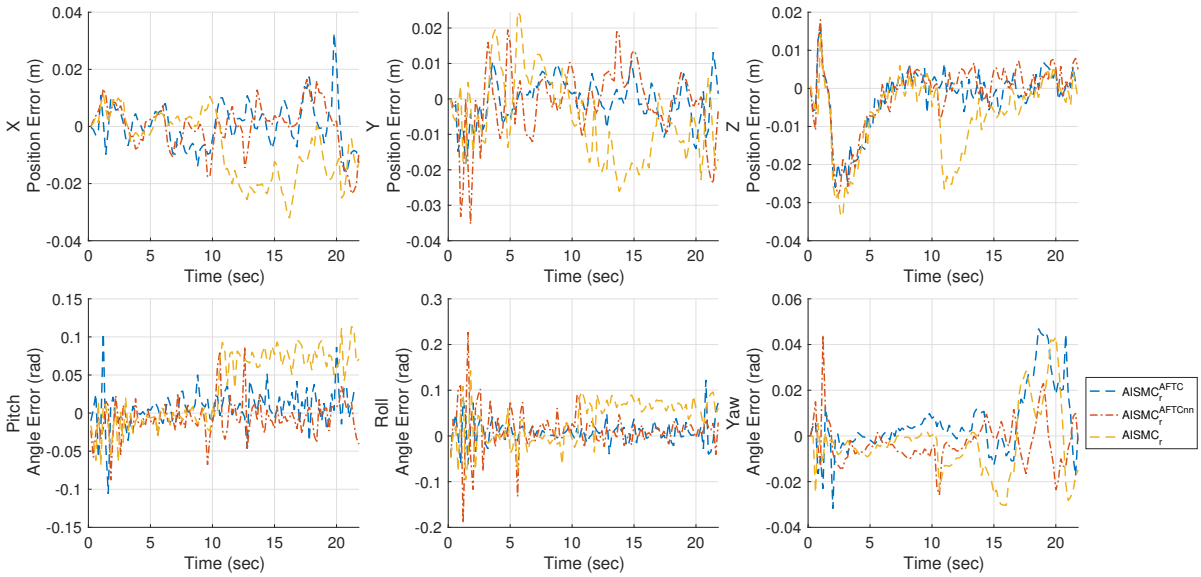


Figure 7.30: AISMC Roy adaptation passive and active FTC trajectory tracking errors for the quadrotor.

7.4 Aerial Manipulator Fault-Tolerant Control Experiments

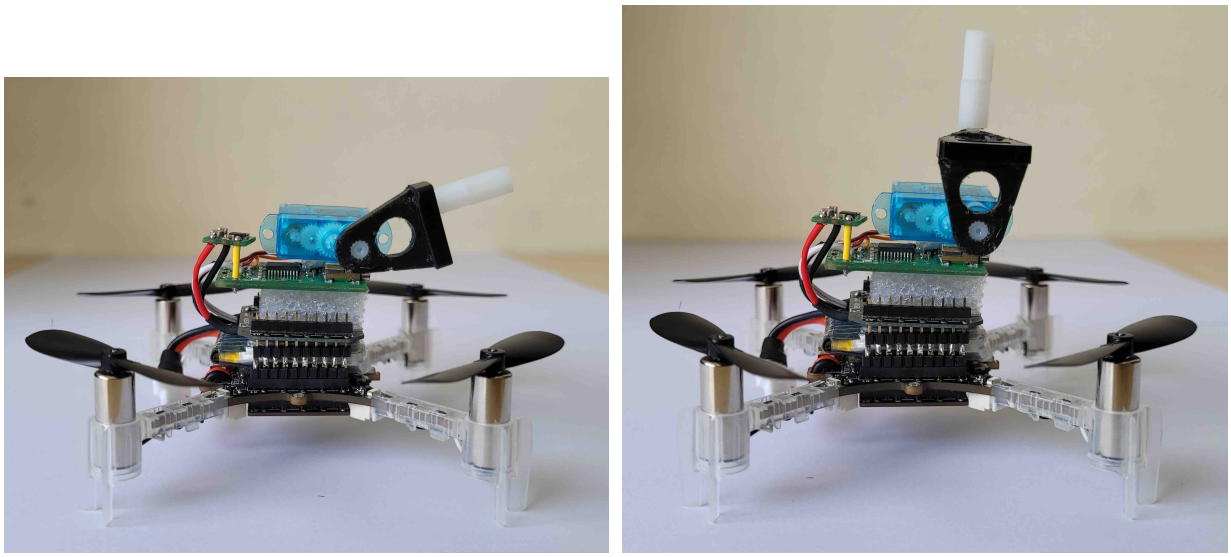
Three controllers are investigated for the FTC of the Crazyflie quadrotor-based aerial manipulator, the on-board Mellinger controller and two AISMCs, one with the Roy adaptation method, AISMC_r, and the other with the Plestan adaptation method, AISMC_p. The AISMCs are selected based on the results of the Crazyflie quadrotor FTC experiments and the Mellinger controller is used as a baseline to compare the performance of the controllers. For the aerial manipulator, the ISMC experiments are not conducted as it has already been established from the quadrotor experiments that the AISMCs successfully reduce chattering experienced by the ISMC. In addition, the quadrotor rotors operate close to their maximum limits for the quadrotor-based aerial manipulator. The ISMC experiments are therefore not conducted for the aerial manipulator in order to preserve the quadrotor rotors by avoiding chattering, which would lead to the rotors quickly

wearing out. From the controller gains tuning and selection process described in Section 7.2, controller gains that have good trajectory tracking, with low chattering and low control inputs are selected. The controller gains presented in Table 7.3 are selected for the Crazyflie quadrotor-based aerial manipulator.

Table 7.3: Crazyflie quadrotor-based aerial manipulator controller gains.

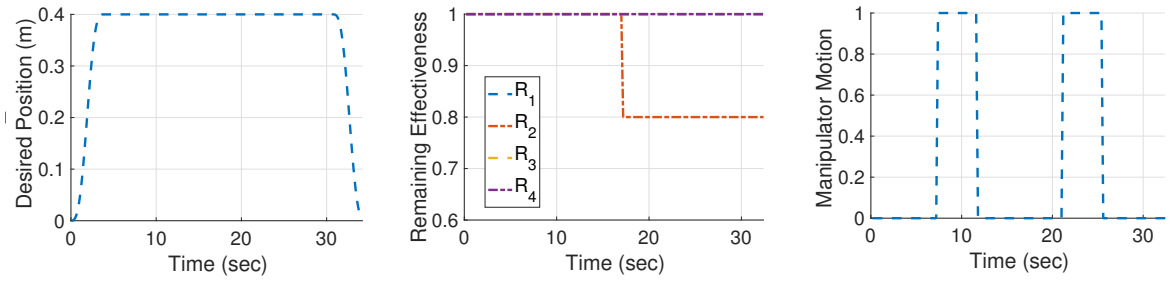
Controller	Gain	X	Y	Z	ψ
All	k_2	5	5	7	10
	c_1	2	2	2	5
	c_2	0.5	0.5	1	0.1
AISMC_p	η_α	1.5×10^{-3}	1.5×10^{-3}	0.012	1×10^{-3}
	η_β	1×10^{-5}	1×10^{-5}	1×10^{-5}	1×10^{-5}
	μ	0.05	0.05	0.05	0.15
	α_m	5×10^{-4}	5×10^{-4}	1×10^{-3}	1×10^{-3}
	α_{max}	0.3	0.3	0.45	1
	α_i	0.01	0.005	0.1	0.2
AISMC_r	ρ_0	0.5	0.5	0.5	0.5
	ρ_1	0.5	0.5	0.5	0.5
	r_{0m}	1×10^{-3}	1×10^{-3}	5×10^{-3}	1×10^{-3}
	r_{1m}	1×10^{-7}	1×10^{-7}	1×10^{-7}	1×10^{-7}

First, passive FTC experiments are conducted with the three controllers for the Crazyflie quadrotor-based aerial manipulator. Active FTC experiments are then conducted assuming a perfect FDD unit. Finally, data is collected for training of a neural network for FDD unit of one of the rotors and active FTC experiments are conducted with the neural network-based FDD unit. To preserve the quadrotor rotors, the experiments are conducted with faults applied to different rotors. For the experiments conducted, the aerial manipulator must hover with the manipulator in motion, including a fault in one rotor. Due to the limitations of the battery and the Crazyflie motors, full area coverage experiments are not conducted. The aerial manipulator must first move to a specified height and then maintain that height while the manipulator is extended and retracted. Figure 7.31 shows the manipulator in its retracted and extended positions. A fault in one of the rotors is then triggered by reducing the rotor effectiveness. The manipulator is then extended and retracted in the presence of the rotor fault. Figure 7.32a shows the desired height trajectory for the aerial manipulator and Figure 7.32b shows the rotor remaining effectiveness where a value of 1 is full effectiveness and a value of 0 is complete failure of the rotor. Figure 7.32c shows the manipulator motion where a value of 1 represents the manipulator in its extended position, and a value of 0 means the manipulator is retracted.



(a) Manipulator retracted (b) Manipulator extended

Figure 7.31: Manipulator positions for the Crazyflie quadrotor-based aerial manipulator.



(a) Desired trajectory. (b) Rotor 2 remaining effectiveness. (c) Manipulator motion.

Figure 7.32: Crazyflie quadrotor-based aerial manipulator experiment.

7.4.1 Passive fault-tolerant control experiments

Figure 7.33 shows the position and angle errors for the passive FTC experiments for the aerial manipulator and Figure 7.39 shows the control inputs. The rotor fault is applied to rotor R_4 for this experiment. As can been seen from Figure 7.33, the AISMCs are better able to recover from the rotor fault than the on-board Mellinger controller. From Figure 7.34 it can be seen that the AISMCs do not show signs of significant chattering for this experiment. Small levels of chattering are detected for the roll and pitch DoFs for the AISMC_r as can be seen from Figure 7.33 by the slight spikes in error for the roll and pitch DoFs.

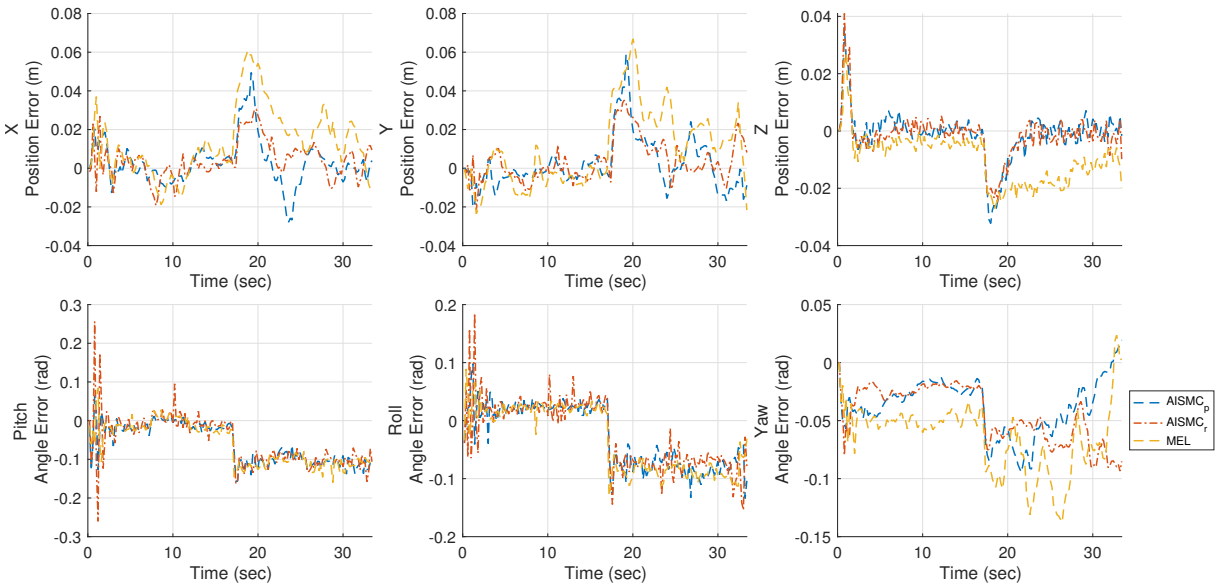


Figure 7.33: Passive FTC trajectory tracking errors for the aerial manipulator.

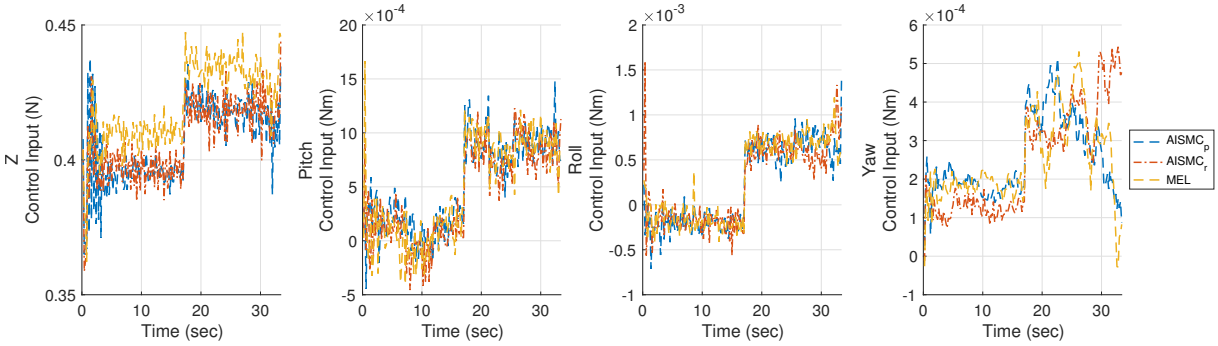


Figure 7.34: Passive FTC control inputs for the aerial manipulator.

7.4.2 Active fault-tolerant control experiments

Figure 7.35 shows the position and angle errors for the active FTC experiments with a perfect FDD unit, and Figure 7.36 shows the control inputs for the aerial manipulator. The rotor fault is applied to rotor R_1 for this experiment. As can been seen from Figure 7.38, for the Z DoF, the $\text{AISMC}_{\text{r}}^{\text{AFTC}}$ and $\text{AISMC}_{\text{p}}^{\text{AFTC}}$ are better able to recover from the fault than the on-board Mellinger controller, MEL. There is no significant difference in the fault recovery results for the controllers for the X and Y DoFs. For the yaw DoF, the AISMCs produce good error results up until a time of about 29 seconds. The Mellinger controller has the worst yaw DoF error results. From Figure 7.36 and Figure 7.35 it can be seen from the roll and pitch DoF results that the $\text{AISMC}_{\text{r}}^{\text{AFTC}}$ and $\text{AISMC}_{\text{p}}^{\text{AFTC}}$ experience slight chattering for this experiment as evidenced by the spikes in error and control input, with the $\text{AISMC}_{\text{r}}^{\text{AFTC}}$ having higher chattering.

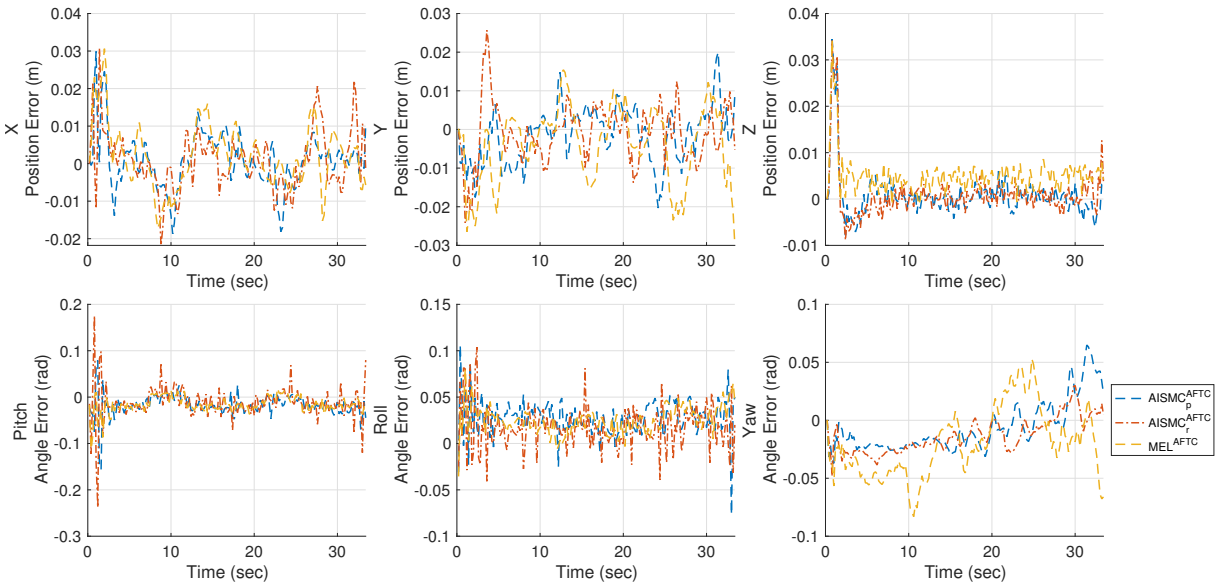


Figure 7.35: Active FTC trajectory tracking errors for the aerial manipulator.

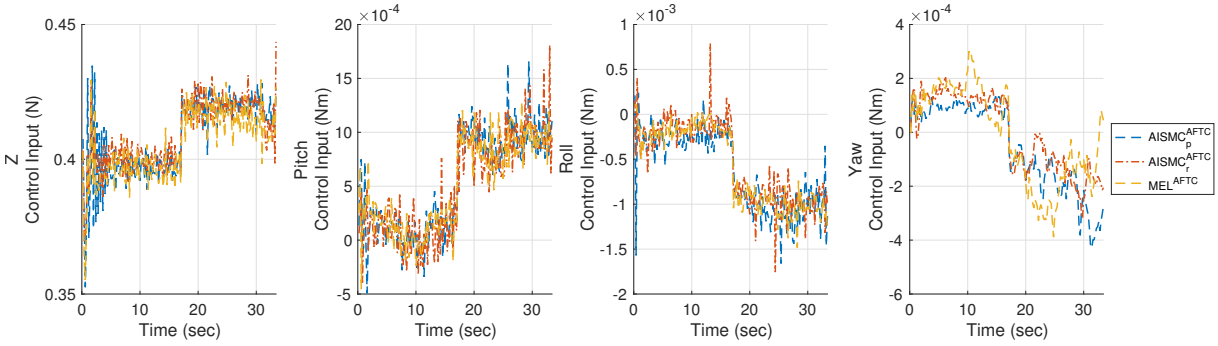


Figure 7.36: Active FTC control inputs for the aerial manipulator.

7.4.3 Neural network-based FDD

Based on the results obtained from the Crazyflie quadrotor FDD neural network training, a feed-forward neural network with a single hidden layer of size 16 is selected and trained for the FDD unit of a single rotor and is implemented on the Crazyflie quadrotor-based aerial manipulator. The neural network has 15 inputs - the Z, roll, pitch and yaw control inputs, the roll, pitch and yaw positions and velocities, the current Z, roll, pitch and yaw control accelerations and the manipulator position, that either is retracted or extended. The outputs from the neural network are then filtered using the exponential moving average filter to improve the FDD outputs. Figure 7.37 shows the training results for rotor R_2 using a feed-forward neural network with a single hidden layer of size 16. The Bayesian regularisation training produced good results for this particular case.

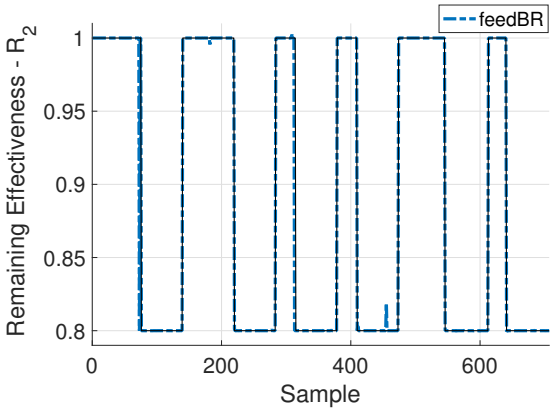


Figure 7.37: Training results for an aerial manipulator single rotor FDD unit comprised of one neural network. Key: feed - feed-forward neural network; BR - Bayesian regularisation training.

7.4.4 Active fault-tolerant control experiments with neural network-based FDD

Figure 7.38 shows the position and angle errors for the active FTC experiments with a neural network-based FDD unit, and Figure 7.39 shows the control inputs for the aerial manipulator. The rotor fault is applied to rotor R_2 for this experiment. As can been seen from Figure 7.38, for the Z DoF, the AISMC$_{\text{r}}^{\text{AFTCnn}}$ and AISMC$_{\text{p}}^{\text{AFTCnn}}$ produce better trajectory tracking error results than the on-board Mellinger controller. The X and Y DoFs have similar error results for the three controllers, with some small spikes in error experienced for the AISMCs. From Figure 7.39 and Figure 7.38 it can be seen from the roll and pitch DoF results that the AISMC$_{\text{r}}^{\text{AFTCnn}}$ and AISMC$_{\text{p}}^{\text{AFTCnn}}$ do not experience significant chattering for this experiment, although slight chattering is experienced for the pitch DoF of the AISMC$_{\text{r}}^{\text{AFTCnn}}$ and the roll DoF of the AISMC$_{\text{p}}^{\text{AFTCnn}}$.

Figure 7.40 shows the output of the neural network-based FDD unit for rotor R_2 of the aerial manipulator. From the figure it can be seen that the FDD unit successfully detects the remaining effectiveness of rotor R_2. There is a slight delay before the fault is fully detected due to the filtering of the neural network outputs. The FDD unit does not perform as well for the AISMC$_{\text{r}}^{\text{AFTCnn}}$ as can be seen by the spikes after the fault occurs where the rotor fault is underestimated.

7.4.5 Passive and active fault-tolerant control comparison

Figure 7.41 shows the passive and active FTC results for the Mellinger controller. Figure 7.42 shows the results for the passive and active AISMCs with Plestan adaptation, and Figure 7.43 shows the results for the passive and active AISMCs with the Roy adaptation. It should be noted that due to having faults induced in different rotors for the experiments, the positive and negative signs for the trajectory tracking errors of the DoFs, with the

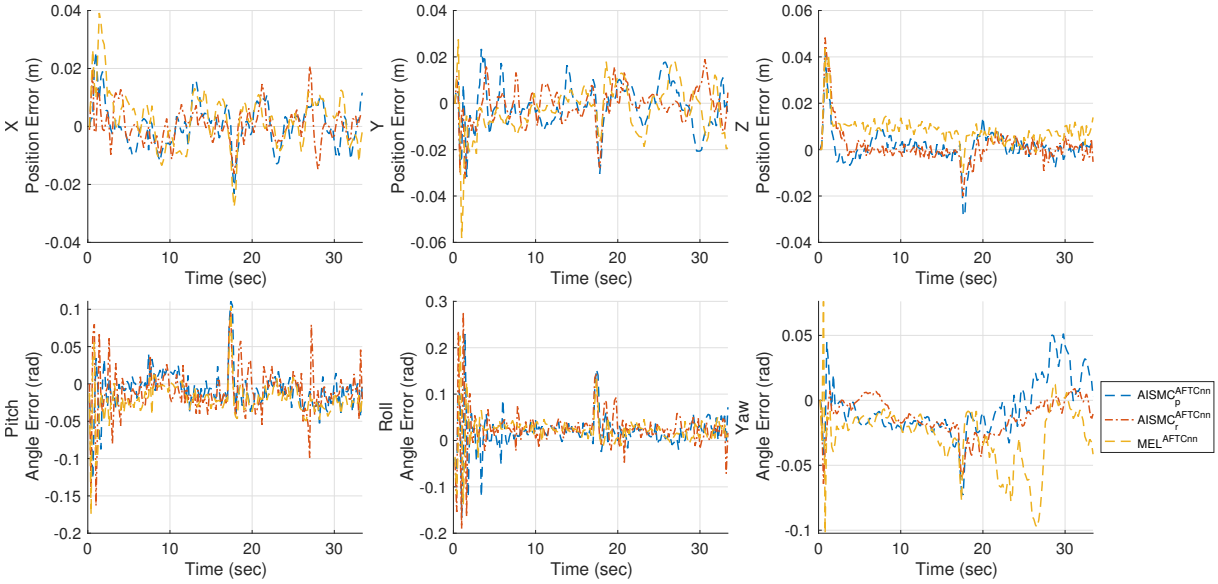


Figure 7.38: Active FTC with neural network-based FDD: trajectory tracking errors for the aerial manipulator.

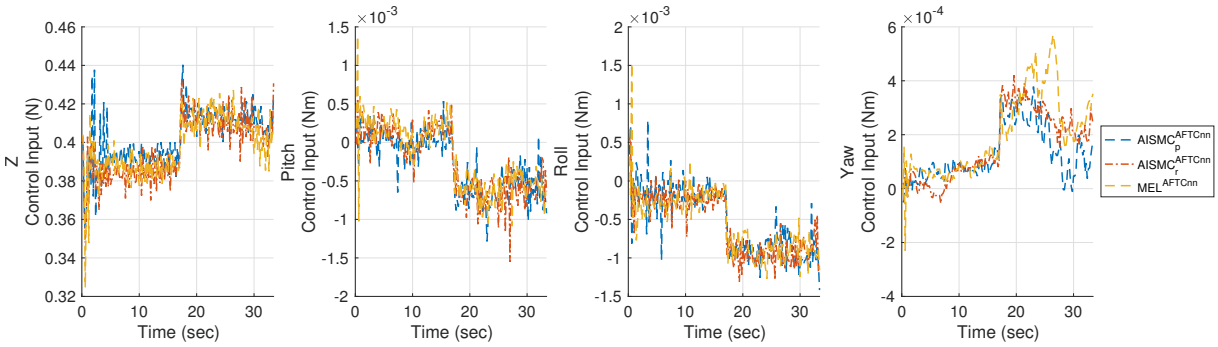


Figure 7.39: Active FTC with neural network-based FDD: control inputs for the aerial manipulator.

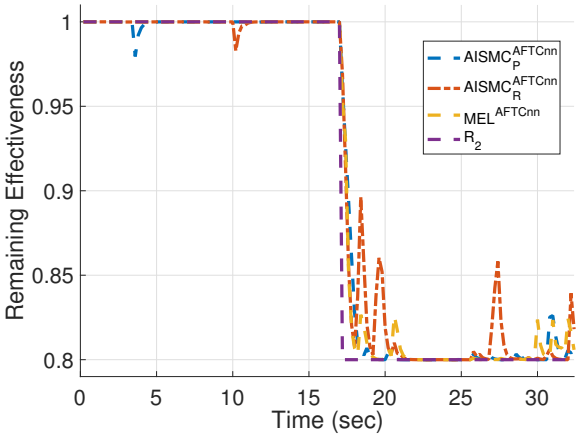


Figure 7.40: Neural network-based FDD results for the aerial manipulator.

exception of the Z DoF, will differ. This is because each faulty rotor has a different effect on the direction that the aerial manipulator travels before the system recovers form the

fault. For the Z DoF, the aerial manipulator will decrease in height before recovering from the rotor fault. The magnitude of the errors are therefore used for comparison of the passive and active controllers. From the figures, it can be seen that the passive controllers have the worst performance, taking a long time to recover from the rotor fault. The active controllers with a perfect FDD have the best results for all of the controllers. The active controllers that make use of the neural network-based FDD unit significantly improve on the results of the passive controllers.

For the aerial manipulator Mellinger controllers, as can be seen from Figure 7.41, for the passive FTC, MEL, the X and Y DoFs have large trajectory tracking errors compared to the active FTCs, MEL^{AFTC} and $\text{MEL}^{\text{AFTCnn}}$, when the fault is experienced. For the active controller making using of the neural network-based FDD unit, $\text{MEL}^{\text{AFTCnn}}$, there is an increase in error for X DoF when the fault occurs, and then it quickly recovers from the error. For the Z DoF, the passive controller has a distinct increase in error when the fault occurs and it recovers from the fault at a very slow rate. The Z DoF for the $\text{MEL}^{\text{AFTCnn}}$ has a slight increase in error that is very quickly recovered from. For the roll and pitch DoFs, the passive controllers experience a large increase in error when the fault occurs and they do not recover from the error. For the $\text{MEL}^{\text{AFTCnn}}$, the roll and pitch DoFs experience an increase in error when the fault occurs, however they very quickly recover from the error. The yaw DoF has poor performance for all of the controllers.

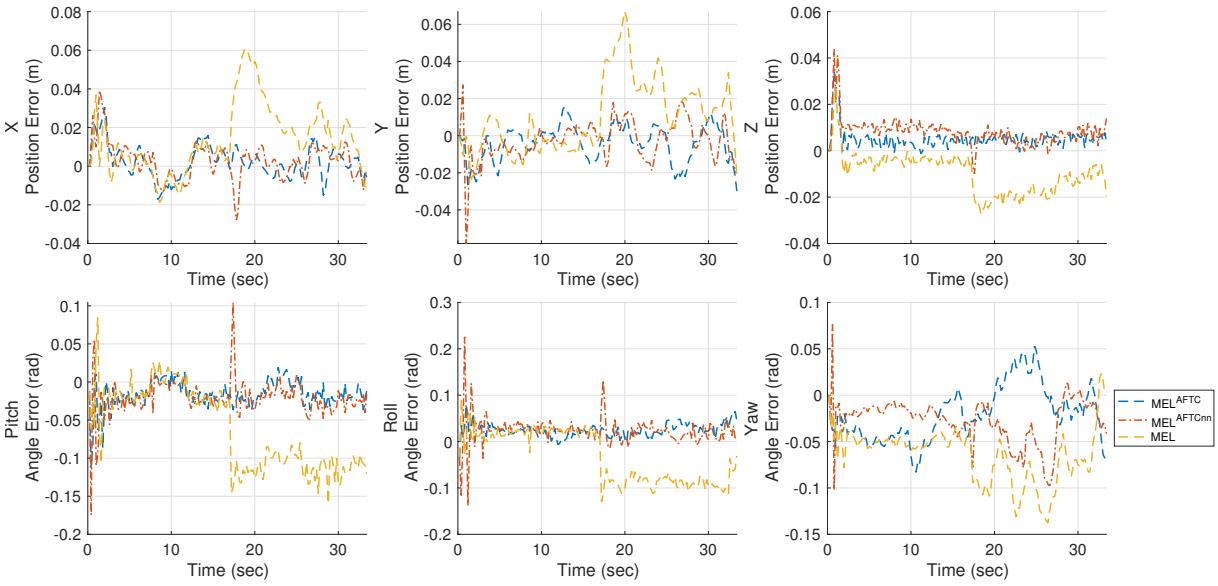


Figure 7.41: Mellinger passive and active FTC trajectory tracking errors for the aerial manipulator.

The passive and active FTC results for the AISMCs with the Plestan adaptation method are shown in Figure 7.42. As can be seen from the figure, the passive FTC, AISMC_{p}, for the X and Y DoFs experience large trajectory tracking errors compared to the active FTCs, $\text{AISMC}_{\text{p}}^{\text{AFTC}}$ and $\text{AISMC}_{\text{p}}^{\text{AFTCnn}}$, when the fault is experienced. For the $\text{AISMC}_{\text{p}}^{\text{AFTCnn}}$, the X and Y DoFs experience a small error when the fault occurs and they very quickly recover from the error. For the Z DoF, the AISMC_{p} and the $\text{AISMC}_{\text{p}}^{\text{AFTCnn}}$ have a distinct increase in error when the fault occurs. The Z DoF for the $\text{AISMC}_{\text{p}}^{\text{AFTCnn}}$

recovers from the fault at a faster rate than the for the AISMC_p. For the $\text{AISMC}_\text{p}^{\text{AFTCnn}}$, the Z DoF experiences a slight overshoot when recovering from the fault. For the roll and pitch DoFs, the passive controllers experience a large increase in error when the fault occurs and they do not recover from the error. For the $\text{AISMC}_\text{p}^{\text{AFTCnn}}$ roll and pitch DoFs, a large increase in error is experienced and is very quickly recovered from when the fault occurs. The yaw DoF results for the active controllers are better than the results for the passive controller.

The passive and active FTC results for the AISMCs with the Roy adaptation method are shown in Figure 7.43. As can be seen from the figure, for the passive FTC, AISMC_r, the X and Y DoFs experience large trajectory tracking errors compared to the active FTCs, $\text{AISMC}_\text{r}^{\text{AFTC}}$ and $\text{AISMC}_\text{r}^{\text{AFTCnn}}$, when the fault is experienced. For $\text{AISMC}_\text{r}^{\text{AFTCnn}}$, the X and Y DoFs experience an increase in error when the fault occurs and very quickly recover from the error. For the Z DoF, the AISMC_r and the $\text{AISMC}_\text{r}^{\text{AFTCnn}}$, have a distinct increase in error when the fault occurs. The $\text{AISMC}_\text{r}^{\text{AFTCnn}}$ recovers from the Z DoF error at a much faster rate than the AISMC_r. The $\text{AISMC}_\text{r}^{\text{AFTCnn}}$ experiences a slight overshoot for the Z DoF when recovering from the fault. For the passive controller, the roll and pitch DoFs, experience large increases in error when the fault occurs and they do not recover from the errors. For the $\text{AISMC}_\text{r}^{\text{AFTCnn}}$ roll and pitch DoFs, an increase in the errors is experienced when the fault occurs and the errors oscillates for a few seconds afterwards. For the yaw DoF the active controllers have better performance than the passive controller after the fault occurs.

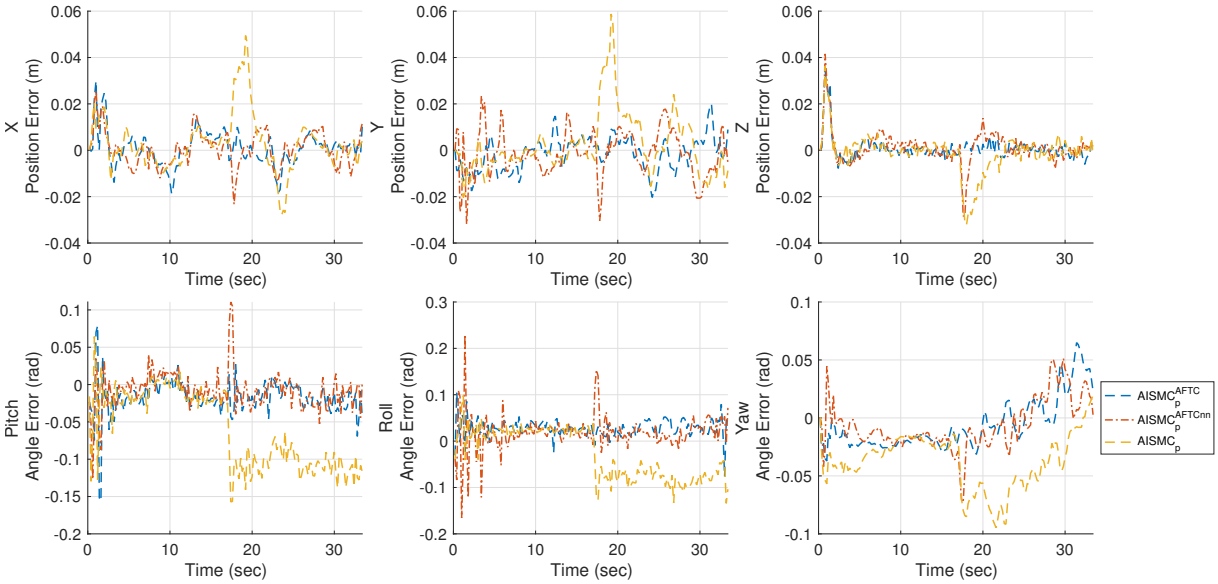


Figure 7.42: AISMC Plestan passive and active FTC trajectory tracking errors for the aerial manipulator.

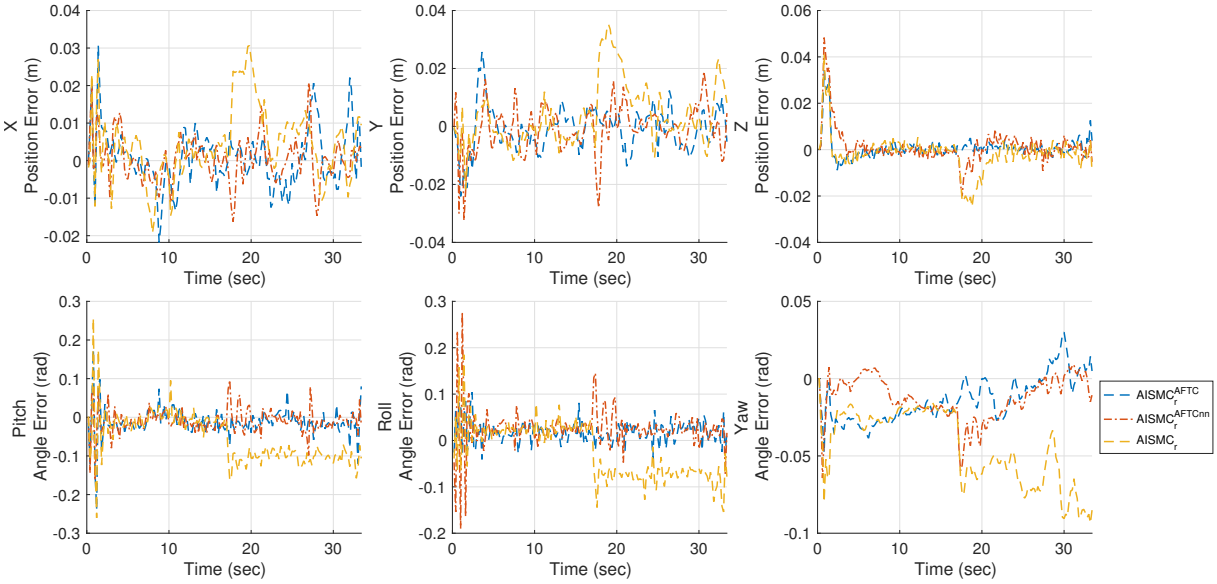


Figure 7.43: AISMC Roy passive and active FTC trajectory tracking errors for the aerial manipulator.

7.5 Summary

In this chapter, passive and active FTC experiments for a Crazyflie quadrotor and a Crazyflie quadrotor-based aerial manipulator were conducted. Due to the control system design of the Crazyflie quadrotor, it was only possible to apply SMCs to the X, Y, Z, and yaw DoFs. The roll and pitch DoFs were controlled using the on-board Mellinger controller, which uses a type of PD controller for those DoFs. It was found that some adjustments to the results obtained from the MOPSO controller gains-tuning process are necessary to make sure the controller gains were suitable for implementation on the physical systems. The FDD unit with four individual neural networks, one for each rotor, produced the best results for estimating the remaining effectiveness of each rotor, given different rotor faults. Due to the computational limitations of the Crazyflie quadrotor, the full FDD unit could not be implemented on the physical system. A simplified FDD unit that detected the fault for only a single rotor was used for the experiments and produced good results. The active FTCs produced better fault recovery results than the passive FTCs. The adaptive SMCs were successfully able to lower chattering.

8 Conclusions and Recommendations for Future Work

This chapter details the conclusions and recommendations for the development of intelligent nonlinear FTC of quadrotor-based aerial manipulators with application to precision agriculture.

8.1 Conclusions

This thesis sought to develop an intelligent nonlinear FTC for quadrotor-based aerial manipulators for the specific application of remote sensing of a farm and plant sample collection for precision agriculture. This involved the kinematic and dynamic modelling, and the system identification of the aerial manipulator system. Various nonlinear sliding mode FTCs for mitigating rotor faults were presented. Both passive FTC making use of the robust properties of SMCs, and active FTC making use of intelligent neural networks-based FDD units were investigated. Performance indices for chattering, an undesirable phenomena experienced by SMCs, were developed based on the frequency domain analysis of the control input signals using the FFT and the STFT. The STFT-based IFAP chattering index, together with control input and trajectory tracking indices were used as the objectives to be minimised for the MOO controller gains tuning, in order to obtain a set of optimal controller gains for the aerial manipulator system. Two multi-objective metaheuristic computational intelligence methods were compared for the MOO controller gains tuning, the MOPSO algorithm and the NSGA-II, which is a type of MOGA. It was found that the MOPSO had the best performance. The variations of sliding mode FTCs were tuned using the MOPSO method and the results were used to compare the chattering, trajectory tracking errors, and control inputs for the controllers. Two continuous (making use of the hyperbolic tangent function as a continuous approximation of the the discontinuous signum function), adaptive, integral SMCs were selected. These controllers, the AISMC using the Roy adaptation method and the AISMC using the Plestan adaptation method, were simulated for the active FTC of an aerial manipulator with a two-DoF manipulator for precision agriculture farm coverage and plant sample collection, with an added rotor fault. Finally, passive sliding mode FTC, and active sliding mode FTC with neural network-based FDD experiments were successfully conducted using a physical quadrotor and a physical quadrotor-based aerial manipulator system with a one-DoF manipulator. The overall aim of the thesis was therefore achieved.

The first objective of this thesis was to develop a model of the quadrotor-based aerial manipulator system. The general model of the quadrotor-based aerial manipulator was

developed using the Newton-Euler formulation. The quadrotor subsystem forces and torques were then modelled and the manipulator subsystem dynamics were modelled using the recursive Newton-Euler formulation. The two subsystem models were combined by treating the quadrotor as the floating base for the manipulator. The rotor loss of effectiveness faults were modelled as a percentage of the rotor effectiveness. Two different quadrotor-based aerial manipulator systems were modelled. One aerial manipulator had a larger quadrotor subsystem with a two-DoF manipulator subsystem mounted at the bottom of the quadrotor. This configuration allows for the collection of plant samples while the aerial manipulator hovers above the plant. This model was used for simulating the farm coverage and plant sample collection tasks. The second aerial manipulator model was based on a physical, miniature Crazyflie quadrotor with a one-DoF manipulator that was used for the physical FTC experiments. The manipulator was mounted on top of the quadrotor since an optical flow sensor, used for obtaining the position of the system, was mounted on the bottom of the quadrotor. System identification experiments were performed to obtain the relevant parameters such as the moments of inertia, the rotor thrust factor, the rotor dynamics time constants, and the relationship between the rotor speeds and PWM duty cycle values, that were used in the system models. The first objective of this thesis was therefore achieved.

The second objective of the thesis was to develop a sliding mode FTC to accurately control the motion of the quadrotor-based aerial manipulator in the presence of rotor faults. The third objective, which is related to the second objective, was to develop an intelligent FDD scheme for quadrotor-based aerial manipulator partial loss of rotor effectiveness. A decoupled control strategy was used for the aerial manipulator system. The quadrotor subsystem was controlled using SMCs and the manipulator subsystem was controlled using closed-loop inverse kinematics. Minimum snap trajectories were used to track the desired paths of the aerial manipulator system. Since the quadrotor is an underactuated system, X and Y virtual control inputs were used to compute the desired roll and pitch angles to track a desired X, Y, and Z position and yaw angle trajectory. Both passive and active sliding mode FTC structures were presented to mitigate rotor faults of the quadrotor subsystem. Several variation of SMCs, including first-order SMCs, continuous and discontinuous SMCs, integral SMCs, adaptive SMCs, and second-order STSMCs were investigated. The passive FTC systems use the robust properties of SMCs to recover from rotor faults. The active FTCs make use of a FDD unit to detect the fault occurrence and fault magnitude of each rotor. The passive and active FTC structures were tested through simulations and physical experiments which verified their effectiveness at providing accurate control of the aerial manipulator in the presence of rotor faults. Three different neural network-based FDD units were investigated and tested for the active FTCs. One FDD structure used a single neural to detect and diagnose faults for all three rotors. Another structure had four neural networks, one each for the FDD of a rotor. The other structure had a system identification neural network with the outputs being the estimated system accelerations. A second neural network used the difference between the actual and estimated acceleration to detect and diagnose the faults for all four rotors. Two types of neural network were investigated, a feed-forward neural network

and a cascade-forward neural network. The neural network-based FDD units were trained and tested using flight data. Two neural network training algorithms were investigated, the Levenberg-Marquardt training algorithm and the Bayesian regularisation training algorithm. The most effective structure was the FDD unit with four neural networks. Both the feed-forward neural network and the cascade-forward neural network produced good results. The training algorithms had varied results dependant on the FDD structure and the neural network used. The experimental results using a physical quadrotor-based aerial manipulator showed that the selected neural network-based FDD unit was able to accurately detect the occurrence and magnitude of a rotor fault. The second and third objectives of the thesis were therefore achieved. The development and application of intelligent FTC for aerial manipulators is one of the main contributions of this thesis.

The fourth objective of this thesis was to develop a sliding mode chattering performance criterion for MOO controller gains tuning. SMCs typically suffer from the undesirable chattering phenomena, which needs to be minimised when designing the controllers. Thus, it was important to develop a means of measuring the chattering levels of the SMCs. Chattering indices were developed based on the frequency domain analysis of the control inputs using the FFT and STFT formulations. The chattering indices were tested and compared for the pitch DoF under altitude and attitude control of a quadrotor, with and without the presence of sensor noise. It was shown that the developed STFT-based IFAP index using a Hann window of window size of 45 and an overlap of 90% provided a good measure of the sliding mode chattering for all of the test cases. When selecting controller gains, there are often conflicting performance objectives that need to be satisfied, such as minimising trajectory tracking errors, control input, and chattering. Manually tuning controller gains is time consuming and does not usually result in optimum gains. Part of the fourth objective of this thesis was therefore to apply MOO to the tuning of SMCs gains. The MOO process results in Pareto front of optimised performance indices. Based on the desired chattering, trajectory tracking error, and control input performance of the system, particles on the Pareto front were selected and their corresponding controller gains were used for the SMCs. The developed chattering index was used as the chattering performance objective to be minimised during the MOO controller gains-tuning process. For the trajectory tracking error and control input, different indices can be used as the objectives to be minimised, depending on the desired performance of the system. Two multi-objective metaheuristic computational intelligence methods, the MOPSO algorithm and the MOGA were successfully used for tuning of controller gains to optimise tracking performance and control effort, and to minimise chattering. The MOPSO method was found to produce better results than the MOGA method. The Pareto fronts produced were also successfully used as a means of comparing the performance of the different passive and active SMCs. The fourth objective of the thesis was therefore successfully met. The development of a novel SMC chattering performance index and the application of intelligent MOO to the controller gains tuning of quadrotor-based aerial manipulators form two main contributions of this thesis.

The final objective of the thesis was to evaluate the effectiveness of the FTC of the aerial manipulation system. Several different passive and active sliding mode FTCs were simu-

lated for the larger aerial manipulator with the two-DoF manipulator and were compared using the results of the MOPSO controller gains-tuning method. The developed chattering performance index, along with trajectory tracking error and control input indices were used as the objectives to be minimised. This meant that the effectiveness of controllers were compared based on a Pareto set of optimally tuned gains. The results showed that the passive fault-tolerant AISMCs had good fault rejection, low chattering and low control inputs, however they were outperformed by the active fault-tolerant AISMCs. Simulations were performed using the active fault-tolerant AISMCs, where the larger aerial manipulator was required to cover a farm for remote sensing and plant sample collection, in the presence of a rotor fault. A back-and-forth coverage algorithm with a return path was used for the system to cover the entire farm area. To collect plant samples, the aerial manipulator hovered over the plant, and then extended the two-DoF manipulator to reach and collect the plant sample, and retracted the manipulator to store the plant sample. The FTC of an aerial manipulator for precision agriculture was therefore successfully simulated. Passive and active FTC experiments were conducted using both a Crazyflie quadrotor and a Crazyflie quadrotor-based aerial manipulator with a one-DoF manipulator. The MOPSO controller gains-tuning method was used to obtain initial sets of gains to be tested on the physical systems. The controller gains were then tested and adjusted as necessary for use on the physical systems. For the experiments, the system had to take-off to a specified height and hover. A fault in a rotor was induced by adjusting the quadrotor firmware to output a percentage of the desired rotor PWM value, thus reducing the effectiveness of that rotor. For the aerial manipulator, manipulator extension and retraction motions were applied both before and after the fault was induced. The AISMCs were able to significantly reduce chattering when compared to the ISMC. The passive fault-tolerant AISMCs were able to mitigate the rotor fault but were outperformed by the active fault-tolerant AISMCs. The active fault-tolerant SMCs require a good FDD unit. Three different neural network-based FDD unit structures, that have been previously described, were trained and tested using flight data from the physical systems. The FDD unit with a bank of four neural networks, one network for each one of the rotors, was found to produce the best results and was able to accurately detect and diagnose rotor loss of effectiveness faults. Due to the computational limitations of the Crazyflie quadrotor, a simplified version of the neural network-based FDD unit was used for the physical FTC experiments for both the quadrotor and the quadrotor-based aerial manipulator. The FDD unit was able to successfully detect the occurrence and magnitude of the rotor fault. Intelligent FTC was therefore successfully applied and validated through simulations and physical experiments for the control of quadrotor-based aerial manipulators. The fifth objective of the thesis was therefore achieved. Application of aerial manipulators to precision agriculture farm coverage and plant sample collection forms one of the main contributions of this thesis. The experimental validation of the passive and active FTCs and of the intelligent MOO controller gains-tuning method, form another main contribution of this thesis.

8.2 Recommendations for Future Work

This thesis presented simulation and experimental studies of the FTC for quadrotor-based aerial manipulators. The FTC of a quadrotor-based aerial manipulator with a two-DoF manipulator was simulated for farm coverage and plant sample collection, in the presence of a rotor fault. A miniature quadrotor-based aerial manipulator with a one-DoF manipulator was used for physical FTC experiments. Future work will include physical experiments for the FTC of a quadrotor-based aerial manipulator with a multi-DoF manipulator for full farm coverage and plant sample collection. For the cross configuration of the quadrotor subsystem, a fault in a single actuator affects all DoFs of the quadrotor. Experiments conducted with a single rotor fault were therefore sufficient to validate the FTC performance of the system. However, future work will include FTC experiments where faults are experienced in more than one rotor. For the physical experiments, due the the nature of the quadrotor firmware, SMCs were applied to four DoFs of the quadrotor subsystem of the aerial manipulator. Future work includes conducting physical experiments with SMC applied to all six DoFs of the quadrotor subsystem. The effect of the starting frequency (given as a percentage of the maximum frequency) used, and the window type, window size and overlap length for the STFT-based chattering indices requires further investigation. The MOO controller gains tuning including sensor noise was simulated, however the sensor noise was kept at the same level for each iteration of the simulations. MOO controller gains tuning in the presence of sensor noise needs to be further investigated. The manipulator joint angle motion affects the centre of mass of the entire aerial manipulator system. This could be exploited to assist with fault recovery of the system. Future work therefore will include investigating the use of the manipulator motion to change the centre of mass position of the whole system to help to mitigate the effects of rotor faults.

References

[1] N. S. Özbek, M. Önkol, and M. Ö. Efe, "Feedback control strategies for quadrotor-type aerial robots: a survey," *Transactions of the Institute of Measurement and Control*, vol. 38, no. 5, pp. 529–554, 2016.

[2] A. Ollero, M. Tognon, A. Suarez, D. Lee, and A. Franchi, "Past, Present, and Future of Aerial Robotic Manipulators," *IEEE Transactions on Robotics*, vol. 38, no. 1, pp. 626–645, 2022.

[3] H. Bonyan Khamseh, F. Janabi-Sharifi, and A. Abdessameud, "Aerial manipulation. A literature survey," *Robotics and Autonomous Systems*, vol. 107, pp. 221–235, 2018.

[4] A. Suarez, V. M. Vega, M. Fernandez, G. Heredia, and A. Ollero, "Benchmarks for Aerial Manipulation," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2650–2657, 2020.

[5] A. Mohiuddin, T. Tarek, Y. Zweiri, and D. Gan, "A Survey of Single and Multi-UAV Aerial Manipulation," *Unmanned Systems*, vol. 8, no. 2, pp. 119–147, 2020.

[6] M. Orsag, C. Korpela, S. Bogdan, and P. Oh, "Dexterous aerial robots-mobile manipulation using unmanned aerial systems," *IEEE Transactions on Robotics*, vol. 33, no. 6, pp. 1453–1466, 2017.

[7] M. Orsag, C. Korpela, S. Bogdan, and P. Oh, "Valve Turning using a Dual-Arm Aerial Manipulator," in *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, (Orlando, FL), pp. 836–841, 27–30 May 2014.

[8] J. R. Kutia, K. A. Stol, and W. Xu, "Canopy Sampling Using an Aerial Manipulator : A Preliminary Study," in *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, (Denver, Colorado), pp. 1–8, 9–12 Jun 2015.

[9] S. Kim, H. Seo, and H. J. Kim, "Operating an Unknown Drawer Using an Aerial Manipulator," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, (Seattle, Washington), pp. 5503–5508, 26–30 May 2015.

[10] M. Fumagalli, R. Naldi, A. Macchelli, R. Carloni, S. Stramigioli, and L. Marconi, "Modeling and Control of a Flying Robot for Contact Inspection," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, (Vilamoura, Algarve), pp. 3532–3537, 7–12 Oct 2012.

[11] J. L. Scholten, M. Fumagalli, S. Stramigioli, and R. Carloni, "Interaction control of an UAV endowed with a manipulator," in *2013 IEEE International Conference on Robotics and Automation*, (Karlsruhe), pp. 4910–4915, IEEE, 6–10 May 2013.

[12] S. Hamaza, I. Georgilas, and T. Richardson, "Towards an Adaptive-Compliance Aerial Manipulator for Contact- Based Interaction," *IEEE International Conference on Intelligent Robots and Systems*, pp. 7448–7453, 2018.

[13] M. Tognon, H. A. Chavez, E. Gasparin, Q. Sable, D. Bicego, A. Mallet, M. Lany, G. Santi, B. Revaz, J. Cortes, and A. Franchi, "A Truly-Redundant Aerial Manipulator System with Application to Push-and-Slide Inspection in Industrial Plants," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1846–1851, 2019.

[14] X. Meng, Y. He, and J. Han, "Design and Implementation of a Contact Aerial Manipulator System for Glass-Wall Inspection Tasks," *IEEE International Conference on Intelligent Robots and Systems*, pp. 215–220, 2019.

[15] M. Á. Trujillo, J. R. Martínez-De Dios, C. Martín, A. Viguria, and A. Ollero, "Novel aerial manipulator for accurate and robust industrial NDT contact inspection: A new tool for the oil and gas inspection industry," *Sensors (Switzerland)*, vol. 19, no. 6, pp. 1–24, 2019.

[16] W. Zhang, Q. Liu, M. Wang, J. Jia, S. Lyu, K. Guo, X. Yu, and L. Guo, "Design of an Aerial Manipulator System Applied to Capture Missions," in *2021 International Conference on Unmanned Aircraft Systems, (ICUAS)*, (Athens), pp. 1063–1069, 2021.

[17] S. Kim, H. Seo, S. Choi, and H. J. Kim, "Vision-guided aerial manipulation using a multirotor with a robotic arm," *IEEE/ASME Transactions on Mechatronics*, vol. 21, no. 4, pp. 1912–1923, 2016.

[18] V. Ghadiok, J. Goldin, and W. Ren, "On the design and development of attitude stabilization, vision-based navigation, and aerial gripping for a low-cost quadrotor," *Autonomous Robots*, vol. 33, no. 1-2, pp. 41–68, 2012.

[19] G. Garimella and M. Kobilarov, "Towards Model-Predictive Control for Aerial Pick-and-Place," in *IEEE International Conference on Robotics and Automation*, (Seattle, Washington), pp. 4692–4697, 26–30 May 2015.

[20] D. Mellinger, Q. Lindsey, M. Shomin, and V. Kumar, "Design, modeling, estimation and control for aerial grasping and manipulation," in *IEEE International Conference on Intelligent Robots and Systems*, (San Francisco, CA), pp. 2668–2673, 25–30 Sep 2011.

[21] S. Hamaza, I. Georgilas, G. Heredia, A. Ollero, and T. Richardson, "Design, modeling, and control of an aerial manipulator for placement and retrieval of sensors in the environment," *Journal of Field Robotics*, no. December 2018, pp. 1–22, 2020.

[22] N. Delavarpour, C. Koparan, J. Nowatzki, S. Bajwa, and X. Sun, "A technical study on UAV characteristics for precision agriculture applications and associated practical challenges," *Remote Sensing*, vol. 13, no. 6, pp. 1–25, 2021.

[23] S. Candiago, F. Remondino, M. D. Giglio, M. Dubbini, and M. Gattelli, "Evaluating Multispectral Images and Vegetation Indices for Precision Farming Applications from UAV Images," *Remote Sensing*, vol. 7, no. 4, pp. 4026–4047, 2015.

[24] F. A. Vega, F. C. Ramirez, M. P. Saiz, and F. O. Rosua, "Multi-temporal imaging using an unmanned aerial vehicle for monitoring a sunflower crop," *Biosystems Engineering*, vol. 132, no. Apr 2015, pp. 19–27, 2015.

[25] R. Acevo-Herrera, A. Aguasca, X. Bosch-Lluis, A. Camps, J. Martinez-Fernandez, N. Sanchez-Martin, and C. Perez-Gutierrez, "Design and first results of an UAV-borne L-band radiometer for multiple monitoring purposes," *Remote Sensing*, vol. 2, no. 7, pp. 1662–1679, 2010.

[26] J. Bendig, A. Bolten, S. Bennertz, J. Broscheit, S. Eichfuss, and G. Bareth, "Estimating biomass of barley using crop surface models (CSMs) derived from UAV-based RGB imaging," *Remote Sensing*, vol. 6, no. 11, pp. 10395–10412, 2014.

[27] V. Gonzalez-Dugo, P. Zarco-Tejada, E. Nicolas, P. A. Nortes, J. J. Alarcon, D. S. Intrigliolo, and E. Fereres, "Using high resolution UAV thermal imagery to assess the variability in the water status of five fruit tree species within a commercial orchard," *Precision Agriculture*, vol. 14, no. 6, pp. 660–678, 2013.

[28] J. M. Peña, J. Torres-Sánchez, A. I. de Castro, M. Kelly, and F. López-Granados, "Weed Mapping in Early-Season Maize Fields Using Object-Based Analysis of Unmanned Aerial Vehicle (UAV) Images," *PLoS ONE*, vol. 8, no. 10, pp. 1–11, 2013.

[29] N. Maine, J. Lowenberg-DeBoer, and W. T. Nell, "Impact of variable-rate application of nitrogen on yield and profit : a case study from South Africa," *Precision Agriculture*, vol. 11, no. 5, pp. 448–463, 2010.

[30] B. Ncube, S. J. Twomlow, J. P. Dimes, M. T. Van Wijk, and K. E. Giller, "Resource flows, crops and soil fertility management in smallholder farming systems in semi-arid Zimbabwe," *Soil Use and Management*, vol. 25, no. 1, pp. 78–90, 2009.

[31] G. Debaene, J. Niedźwiecki, A. Pecio, and A. Zurek, "Effect of the number of calibration samples on the prediction of several soil properties at the farm-scale," *Geoderma*, vol. 214-215, no. Feb 2014, pp. 114–125, 2014.

[32] J. Huang, R. M. Lark, D. A. Robinson, I. Lebron, A. M. Keith, B. Rawlins, A. Tye, O. Kuras, M. Raines, and J. Triantafilis, "Scope to predict soil properties at within-field scale from small samples using proximally sensed γ-ray spectrometer and EM induction data," *Geoderma*, vol. 232-234, no. Nov 2014, pp. 69–80, 2014.

[33] R. A. Ortega, "Determination of management zones in corn (Zea mays L .) based on soil fertility," *Computers and Electronics in Agriculture*, vol. 58, no. 1, pp. 49–59, 2007.

[34] Y. M. Oliver, M. J. Robertson, and M. T. F. Wong, "Integrating farmer knowledge , precision agriculture tools , and crop simulation modelling to evaluate management options for poor-performing patches in cropping fields," *European Journal of Agronomy*, vol. 32, no. 1, pp. 40–50, 2010.

[35] M. A. Lee, Y. Huang, H. Yao, S. J. Thomson, and L. M. Bruce, "Determining the effects of storage on cotton and soybean leaf samples for hyperspectral analysis," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 7, no. 6, pp. 2562–2570, 2014.

[36] S. Kannan, M. Alma, M. A. Olivares-Mendez, and H. Voos, "Adaptive Control of Aerial Manipulation Vehicle," in *4th IEEE International Conference on Control System, Computing and Engineering, ICCSCE 2014*, (Penang), pp. 273–278, 28–30 Nov 2014.

[37] S. Di Lucia, G. D. Tipaldi, and W. Burgard, "Attitude Stabilization Control of an Aerial Manipulator using a Quaternion-Based Backstepping Approach," in *European Conference on Mobile Robots (ECMR) 2015*, (Lincoln), pp. 1–6, 2–4 Sep 2015.

[38] A. Khalifa, M. Fanni, A. Ramadan, and A. Abo-Ismail, "Modeling and control of a new quadrotor manipulation system," in *2012 International Conference on Innovative Engineering Systems, ICIES*, (Alexandria), pp. 109–114, 7–9 Dec 2012.

[39] C. Korpela, M. Orsag, M. Pekala, and P. Oh, "Dynamic stability of a mobile manipulating unmanned aerial vehicle," in *2013 IEEE International Conference on Robotics and Automation*, (Karlsruhe), pp. 4922–4927, 6–10 May 2013.

[40] M. Orsag, C. Michael, S. Bogdan, and P. Yu, "Hybrid Adaptive Control for Aerial Manipulation," *Journal of Intelligent and Robotic Systems*, vol. 73, no. 1, pp. 693–707, 2014.

[41] G. Heredia, A. Jimenez-Cano, I. Sanchez, D. Llorente, V. Vega, J. Braga, J. A. Acosta, and A. Ollero, "Control of a Multirotor Outdoor Aerial Manipulator," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2014)*, (Chicago, IL), pp. 3417–3422, 14–18 Sep 2014.

[42] A. Khalifa, M. Fanni, A. Ramadan, and A. Abo-Ismail, "Adaptive Intelligent Controller Design for a New Quadrotor Manipulation System," in *2013 IEEE International Conference on Systems, Man, and Cybernetics*, (Manchester), pp. 1666–1671, 13–16 Oct 2013.

[43] A. Khalifa, M. Fanni, A. Ramadan, and A. Abo-Ismail, "Controller Design of a New Quadrotor Manipulation System Based on Robust Internal-Loop Compensator," in

2015 IEEE International Conference on Autonomous Robot Systems and Competitions, (Vila Real), pp. 97–102, 8–10 Apr 2015.

[44] R. Mebarki and V. Lippiello, "Image-based control for dynamically cross-coupled aerial manipulation," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2014)*, vol. 16, (Chicago, IL), pp. 646–656, 14–18 Sep 2014.

[45] S. Kim, S. Choi, and H. J. Kim, "Aerial manipulation using a quadrotor with a two DOF robotic arm," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, (Tokyo), pp. 4990–4995, 3–7 Nov 2013.

[46] H. Yang and D. Lee, "Dynamics and Control of Quadrotor with Robotic Manipulator," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, (Hong Kong), pp. 5544–5549, 31 May–3 Jun 2014.

[47] R. Jiao, W. Chou, Y. Rong, and M. Dong, "Anti-disturbance attitude control for quadrotor unmanned aerial vehicle manipulator via fuzzy adaptive sigmoid generalized super-twisting sliding mode observer," *JVC/Journal of Vibration and Control*, vol. 28, no. 11-12, pp. 1251–1266, 2022.

[48] M. Jafarinasab and S. Sirouspour, "Adaptive Motion Control of Aerial Robotic Manipulators Based on Virtual Decomposition," in *IEEE International Conference on Intelligent Robots and Systems*, (Hamburg), pp. 1858–1863, 28 Sep–2 Oct 2015.

[49] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[50] C. A. C. Coello and M. S. Lechuga, "MOPSO: a proposal for multiple objective particle swarm optimization," in *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600)*, vol. 2, (Honolulu), pp. 1051–1056, 12–17 May 2002.

[51] X. Zhou and X. Zhang, "Multi-objective-optimization-based control parameters auto-tuning for aerial manipulators," *International Journal of Advanced Robotic Systems*, vol. 16, no. 1, pp. 1–13, 2019.

[52] Z. C. Qin, Y. Xin, and J. Q. Sun, "Multi-objective optimal motion control of a laboratory helicopter based on parallel simple cell mapping method," *Asian Journal of Control*, vol. 22, no. 4, pp. 1565–1578, 2019.

[53] Y. M. Zhang, A. Chamseddine, C. A. Rabbath, B. W. Gordon, C. Y. Su, S. Rakheja, C. Fulford, J. Apkarian, and P. Gosselin, "Development of advanced FDD and FTC techniques with application to an unmanned quadrotor helicopter testbed," *Journal of the Franklin Institute*, vol. 350, no. 9, pp. 2396–2422, 2013.

[54] Y. Zhang and J. Jiang, "Bibliographical review on reconfigurable fault-tolerant control systems," *Annual Reviews in Control*, vol. 32, no. 2, pp. 229–252, 2008.

[55] Y. Ding, Y. Wang, and B. Chen, "Fault-tolerant control of an aerial manipulator with guaranteed tracking performance," *International Journal of Robust and Nonlinear Control*, vol. 32, no. 2, pp. 960–986, 2022.

[56] C. Pose, J. Giribet, and I. Mas, "Adaptive Center-of-Mass Relocation for Aerial Manipulator Fault Tolerance," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5583–5590, 2022.

[57] A.-R. Merheb, H. Noura, and F. Bateman, "Passive Fault Tolerant Control of Quadrotor UAV Using Regular and Cascaded Sliding Mode Control," in *2013 Conference on Control and Fault-Tolerant Systems (SysTol)*, (Nice), pp. 330–335, 9–11 Oct 2013.

[58] A.-R. Merheb, H. Noura, and F. Bateman, "Active fault tolerant control of quadrotor UAV using Sliding Mode Control," in *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, (Orlando, FL), pp. 156–166, 27–30 May 2014.

[59] A.-R. Merheb, H. Noura, and F. Bateman, "Design of Passive Fault Tolerant Controllers of a Quadrotor Based on Sliding Mode Theory," *International Journal of Applied Mathematics and Computer Science*, vol. 25, no. 3, pp. 561–576, 2015.

[60] A.-R. Merheb, F. Bateman, and H. Noura, "Passive and active fault tolerant control of octorotor UAV using Second Order Sliding Mode control," *2015 IEEE Conference on Control Applications (CCA)*, pp. 1907–1912, 2015.

[61] A. R. Merheb, H. Noura, F. Bateman, and J. Al-Jaroodi, "Fault severity based Integrated Fault Tolerant Controller for quadrotor UAVs," *2015 International Conference on Unmanned Aircraft Systems, ICUAS 2015*, no. 1, pp. 660–668, 2015.

[62] A.-R. Merheb, H. Noura, F. Bateman, and J. Al-Jaroodi, "Fault severity based Integrated Fault Tolerant Controller for quadrotor UAVs," in *2015 International Conference on Unmanned Aircraft Systems, ICUAS 2015*, (Denver, Colorado), pp. 660–668, 9–12 Jun 2015.

[63] T. Li, Y. Zhang, and B. W. Gordon, "Passive and active nonlinear fault-tolerant control of a quadrotor unmanned aerial vehicle based on the sliding mode control technique," *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 227, no. 1, pp. 12–23, 2012.

[64] F. Sharifi, M. Mirzaei, B. W. Gordon, and Y. Zhang, "Fault Tolerant Control of A Quadrotor UAV Using Sliding Mode Control," in *2010 Conference on Control and Fault-Tolerant Systems (SysTol)*, (Nice), pp. 239–244, 6–8 Oct 2010.

[65] S. Mallavalli and A. Fekih, "A fault tolerant control design for actuator fault mitigation in quadrotor UAVS," *Proceedings of the American Control Conference*, vol. 2019-July, pp. 5111–5116, 2019.

[66] A. Mohammadi and A. Ramezani, "An Active Actuator Fault-Tolerant Control of a Quadrotor Based on Analytical Redundancy Relations," *Iranian Journal of Science and Technology - Transactions of Electrical Engineering*, vol. 6, 2019.

[67] Z. Hou, P. Lu, and Z. Tu, "Nonsingular terminal sliding mode control for a quadrotor UAV with a total rotor failure," *Aerospace Science and Technology*, vol. 98, p. 105716, 2020.

[68] B. Wang, X. Yu, L. Mu, and Y. Zhang, "A dual adaptive fault-tolerant control for a quadrotor helicopter against actuator faults and model uncertainties without overestimation," *Aerospace Science and Technology*, vol. 99, 2020.

[69] B. Wang, Y. Shen, and Y. M. Zhang, "Active fault-tolerant control for a quadrotor helicopter against actuator faults and model uncertainties," *Aerospace Science and Technology*, vol. 99, p. 105745, 2020.

[70] H. Alwi and C. Edwards, "Sliding mode fault-tolerant control of an octorotor using linear parameter varying-based schemes," *IET Control Theory & Applications*, vol. 9, no. 4, pp. 618–636, 2015.

[71] X. Wang, E. J. van Kampen, and Q. Chu, "Quadrotor fault-tolerant incremental nonsingular terminal sliding mode control," *Aerospace Science and Technology*, vol. 95, p. 105514, 2019.

[72] W. Gong, B. Li, Y. Yang, H. Ban, and B. Xiao, "Fixed-time integral-type sliding mode control for the quadrotor UAV attitude stabilization under actuator failures," *Aerospace Science and Technology*, vol. 95, 2019.

[73] S. Zeghlache, H. Mekki, A. Bouguerra, and A. Djerioui, "Actuator fault tolerant control using adaptive RBFNN fuzzy sliding mode controller for coaxial octorotor UAV," *ISA Transactions*, vol. 80, no. June, pp. 267–278, 2018.

[74] S. Barghandan, M. A. Badamchizadeh, and M. R. Jahed-motlagh, "Improved Adaptive Fuzzy Sliding Mode Controller for Robust Fault Tol- erant of a Quadrotor," vol. 15, no. 1, pp. 427–441, 2017.

[75] F. Chen, K. Zhang, Z. Wang, G. Tao, and B. Jiang, "Trajectory tracking of a quadrotor with unknown parameters and its fault-tolerant control via sliding mode fault observer," *Journal of Systems and Control Engineering*, vol. 229, no. 4, pp. 279–292, 2015.

[76] R. R. Benrezki, M. Tadjine, F. Yacef, and O. Kermia, "Passive Fault Tolerant Control of Quadrotor UAV Using a Nonlinear PID," in *IEEE Conference on Robotics and Biomimetics*, (Zhuhai), pp. 1285–1290, 6–9 Dec 2015.

[77] X. Zhang, Y. Zhang, C.-Y. Su, and Y. Feng, "Fault Tolerant Control for Quadrotor via Backstepping Approach," in *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, (Orlando, Florida), pp. 1–12, 4–7 Jan 2010.

[78] H. Khebbache, "Robust control algorithm considering the actuator faults for attitude tracking of an UAV quadrotor aircraft," *International Journal of Control and Automation*, vol. 5, no. 4, pp. 55–66, 2012.

[79] S. Zeghlache, A. Djerioui, L. Benyettou, T. Benslimane, H. Mekki, and A. Bouguerra, "Fault tolerant control for modified quadrotor via adaptive type-2 fuzzy backstepping subject to actuator faults," *ISA Transactions*, vol. 95, pp. 330–345, 2019.

[80] A. Bounemeur, M. Chemachema, and N. Essounbouli, "Indirect adaptive fuzzy fault-tolerant tracking control for MIMO nonlinear systems with actuator and sensor failures," *ISA Transactions*, vol. 79, no. April, pp. 45–61, 2018.

[81] J. Ghandour, S. Aberkane, and J.-C. Ponsart, "Feedback Linearization Approach for Standard and Fault Tolerant Control: Application to a Quadrotor UAV Testbed," *Journal of Physics: Conference Series*, vol. 570, no. 8, pp. 1–11, 2014.

[82] Q. L. Zhou, Y. Zhang, C. A. Rabbath, and D. Theilliol, "Design of feedback linearization control and reconfigurable control allocation with application to a quadrotor UAV," in *Conference on Control and Fault-Tolerant Systems*, no. 514, (Nice), pp. 371–376, 6–8 Oct 2010.

[83] A. Freddi, A. Lanzon, and S. Longhi, "A feedback linearization approach to fault tolerance in quadrotor vehicles," in *18th World Congress The International Federation of Automatic Control*, (Milano), pp. 5413–5418, IFAC, 28 Aug–2 Sep 2011.

[84] I. Sadeghzadeh and A. Mehta, "Fault-tolerant trajectory tracking control of a quadrotor helicopter using gain-scheduled PID and model reference adaptive control," in *Annual Conference of the Prognostics and Health Management Society*, (Montreal, Quebec), pp. 1–10, 25–29 Sep 2011.

[85] M. H. Amoozgar, A. Chamseddine, and Y. Zhang, "Fault-tolerant fuzzy gain-scheduled PID for a quadrotor helicopter testbed in the presence of actuator faults," in *2nd IFAC Conference on Advances in PID Control. IFAC Proceedings Volumes*, vol. 45, (Brescia), pp. 282–287, IFAC, 28–30 Mar 2012.

[86] L. Qin, X. He, R. Yan, and D. Zhou, "Active Fault-Tolerant Control for a Quadrotor with Sensor Faults," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 88, no. 2-4, pp. 449–467, 2017.

[87] D. Rotondo, F. Nejjari, and V. Puig, "Robust quasi-LPV model reference FTC of a quadrotor UAV subject to actuator faults," *International Journal of Applied Mathematics and Computer Science*, vol. 25, no. 1, pp. 7–22, 2015.

[88] B. Yu, Y. Zhang, and Y. Qu, "MPC-based FTC with FDD against Actuator Faults of UAVs," in *15th International Conference on Control, Automation and Systems (ICCAS 2015)*, (Busan), pp. 225–230, 13–16 Oct 2015.

[89] H. A. Izadi, B. W. Gordon, and Y. Zhang, "A data-driven fault tolerant model predictive control with fault identification," in *2010 Conference on Control and Fault-Tolerant Systems (SysTol)*, (Nice), pp. 732–737, 6–8 Oct 2010.

[90] K. Patan, *Artificial Neural Networks for Modelling and Fault Diagnosis of Technical Processes*. Berlin: Springer, 2008.

[91] C. Berbra, S. Lesecq, and J. J. Martinez, "A multi-observer switching strategy for fault-tolerant control of a quadrotor helicopter," *2008 Mediterranean Conference on Control and Automation - Conference Proceedings, MED'08*, pp. 1094–1099, 2008.

[92] N. P. Nguyen and S. K. Hong, "Sliding mode Thau observer for actuator fault diagnosis of quadcopter UAVs," *Applied Sciences (Switzerland)*, vol. 8, no. 10, 2018.

[93] D. Shi, Z. Wu, and W. Chou, "Super-Twisting Extended State Observer and Sliding Mode Controller for Quadrotor UAV Attitude System in Presence of Wind Gust and Actuator Faults," *Electronics*, vol. 7, no. 8, p. 128, 2018.

[94] Z. Cen, H. Noura, and Y. A. Younes, "Robust Fault Estimation on a real quadrotor UAV using optimized Adaptive Thau Observer," *2013 International Conference on Unmanned Aircraft Systems, ICUAS 2013 - Conference Proceedings*, no. Figure 1, pp. 550–556, 2013.

[95] H. Aguilar-Sierra, G. Flores, S. Salazar, and R. Lozano, "Fault Estimation for a Quad-Rotor MAV Using a Polynomial Observer," *Journal of Intelligent & Robotic Systems*, vol. 73, no. 1-4, pp. 455–468, 2014.

[96] L. Ma and Y. Zhang, "DUKF-based GTM UAV fault detection and diagnosis with nonlinear and LPV models," *Proceedings of 2010 IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications, MESA 2010*, pp. 375–380, 2010.

[97] A. Chamseddine, D. Theilliol, Y. M. Zhang, C. Join, and C. A. Rabbath, "Active fault-tolerant control system design with trajectory re-planning against actuator faults and saturation: Application to a quadrotor unmanned aerial vehicle," *International Journal of Adaptive Control and Signal Processing*, vol. 29, no. 1, pp. 1–23, 2015.

[98] H. A. Izadi, Y. Zhang, and B. W. Gordon, "Fault tolerant model predictive control of quad-rotor helicopters with actuator fault estimation," in *18th World Congress The International Federation of Automatic Control*, (Milano), pp. 6343–6348, IFAC, 28 Aug–2 Sep 2011.

[99] A. de Souza Cândido, R. Kawakami, H. Galvão, and T. Yoneyama, "Actuator Fault Diagnosis and Control of a Quadrotor," in *2014 12th IEEE International Conference on Industrial Informatics (INDIN)*, (Porto Alegre, RS), pp. 310–315, jul 2014.

[100] I. Castillo and L. B. Freidovich, "Describing-function-based analysis to tune parameters of chattering reducing approximations of Sliding Mode controllers," *Control Engineering Practice*, vol. 95, no. February 2020, p. 104230, 2020.

[101] U. Pérez-Ventura and L. Fridman, "Design of super-twisting control gains: A describing function based methodology," *Automatica*, vol. 99, pp. 175–180, 2019.

[102] I. Boiko, L. Fridman, A. Pisano, and E. Usai, "Analysis of chattering in systems with second-order sliding modes," *IEEE Transactions on Automatic Control*, vol. 52, no. 11, pp. 2085–2102, 2007.

[103] I. Boiko, L. Fridman, R. Iriarte, A. Pisano, and E. Usai, "Parameter tuning of second-order sliding mode controllers for linear plants with dynamic actuators," *Automatica*, vol. 42, no. 5, pp. 833–839, 2006.

[104] I. Boiko and L. Fridman, "Analysis of Chattering in Continuous Sliding-Mode Controllers," *IEEE Transactions on Automatic Control*, vol. 50, no. 9, pp. 1442–1446, 2005.

[105] A. Rosales, Y. Shtessel, L. Fridman, and C. B. Panathula, "Chattering Analysis of HOSM Controlled Systems: Frequency Domain Approach," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 4109–4115, 2017.

[106] H. Lee and V. I. Utkin, "Chattering suppression methods in sliding mode control systems," *Annual Reviews in Control*, vol. 31, no. 2, pp. 179–188, 2007.

[107] R. Featherstone and D. E. Orin, *Dynamics*, pp. 35–66. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008.

[108] M. J. Sidi, *Spacecraft Dynamics and Control: A Practical Engineering Approach*. Cambridge Aerospace Series, Cambridge University Press, 1997.

[109] K. Waldron and J. Schmiedeler, *Kinematics*, pp. 9–34. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008.

[110] L. Sciavicco and B. Siciliano, *Modelling and Control of Robot Manipulators*. London: Springer-Verlag, 2005.

[111] M. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modelling and Control*. Wiley and Sons, 2006.

[112] T. Antonsson, "Measuring propeller RPM: Part 3." `https://www.bitcraze.io/2015/02/measuring-propeller-rpm-part-3/`, 2015. Accessed; 13-August-2022.

[113] Bitcraze, "PWM to thrust." `https://www.bitcraze.io/documentation/repository/crazyflie-firmware/master/functional-areas/pwm-to-thrust/`. Accessed; 13-August-2022.

[114] S. Huştiu, M. Lupaşcu, Ş. Popescu, A. Burlacu, and M. Kloetzer, "Stable hovering architecture for nanoquadcopter applications in indoor environments," in *2018 22nd International Conference on System Theory, Control and Computing, (ICSTCC 2018)*, (Sinaia), pp. 659–663, 2018.

[115] A. R. Merheb, H. Noura, and F. Bateman, "Design of Passive Fault-Tolerant Controllers of a Quadrotor Based on Sliding Mode Theory," *International Journal of Applied Mathematics and Computer Science*, vol. 25, no. 3, pp. 561–576, 2015.

[116] H. J. Jayakrishnan, "Position and attitude control of a quadrotor UAV using super twisting sliding mode," *IFAC-PapersOnLine*, vol. 49, no. 1, pp. 284–289, 2016.

[117] L. Derafa, A. Benallegue, and L. Fridman, "Super twisting control algorithm for the attitude tracking of a four rotors UAV," *Journal of the Franklin Institute*, vol. 349, no. 2, pp. 685–699, 2012.

[118] M. Bouchoucha, S. Seghour, and M. Tadjine, "Classical and second order sliding mode control solution to an attitude stabilization of a four rotors helicopter: From theory to experiment," *2011 IEEE International Conference on Mechatronics*, pp. 162–169, 2011.

[119] S. Rajappa, C. Masone, H. H. Bulthoff, and P. Stegagno, "Adaptive Super Twisting Controller for a quadrotor UAV," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2016-June, no. 1, pp. 2971–2977, 2016.

[120] R. Akbar and N. Uchiyama, "Adaptive modified super-twisting control for a quadrotor helicopter with a nonlinear sliding surface," in *SICE International Symposium on Control Systems (SICE ISCS)*, (Okayama), pp. 93–98, 2017.

[121] H. Castañeda and J. Gordillo, "Spatial Modeling and Robust Flight Control Based on Adaptive Sliding Mode Approach for a Quadrotor MAV," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 93, no. 1, pp. 101–111, 2019.

[122] Y. Shtessel, M. Taleb, and F. Plestan, "A novel adaptive-gain supertwisting sliding mode controller: Methodology and application," *Automatica*, vol. 48, no. 5, pp. 759–769, 2012.

[123] S. Roy, S. Baldi, and L. M. Fridman, "On adaptive sliding mode control without a priori bounded uncertainty," *Automatica*, vol. 111, p. 108650, 2020.

[124] H. A. Talebi, K. Khorasani, and S. Tafazoli, "A recurrent neural-network-based sensor and actuator fault detection and isolation for nonlinear systems with application to the satellite's attitude control subsystem," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 45–60, 2009.

[125] K. Michail, K. M. Deliparaschos, S. G. Tzafestas, and A. C. Zolotas, "Ai-based actuator/sensor fault detection with low computational cost for industrial applications," *IEEE Transactions on Control Systems Technology*, vol. 24, no. 1, pp. 293–301, 2016.

[126] Y. Maki and K. A. Loparo, "A neural-network approach to fault detection and diagnosis in industrial processes," *IEEE Transactions on Control Systems Technology*, vol. 5, no. 6, pp. 529–541, 1997.

[127] M. R. Napolitano, Y. An, and B. A. Seanor, "A fault tolerant flight control system for sensor and actuator failures using neural networks," *Aircraft Design*, vol. 3, no. 2, pp. 103–128, 2000.

[128] Z. N. Sadough Vanini, K. Khorasani, and N. Meskin, "Fault detection and isolation of a dual spool gas turbine engine using dynamic neural networks and multiple model approach," *Information Sciences*, vol. 259, pp. 234–251, 2014.

[129] S. Sina Tayarani-Bathaie, Z. N. Sadough Vanini, and K. Khorasani, "Dynamic neural network-based fault diagnosis of gas turbine engines," *Neurocomputing*, vol. 125, pp. 153–165, 2014.

[130] J. Marti-Saumell, A. Santamaria-Navarro, C. Ocampo-Martinez, and J. Andrade-Cetto, "Multi-task closed-loop inverse kinematics stability through semidefinite programming," in *Proceedings - IEEE International Conference on Robotics and Automation*, (Paris), pp. 7108–7114, 2020.

[131] M. D. Fiore and C. Natale, "Discrete-time closed-loop inverse kinematics: A comparison between Euler and RK4 methods," in *2021 29th Mediterranean Conference on Control and Automation, MED 2021*, (Bari, Puglia), pp. 584–589, IEEE, 2021.

[132] A. Colome and C. Torras, "Closed-loop inverse kinematics for redundant robots: Comparative assessment and two enhancements," *IEEE/ASME Transactions on Mechatronics*, vol. 20, no. 2, pp. 944–955, 2015.

[133] S. Chiaverini, G. Oriolo, and I. D. Walker, *Kinematically Redundant Manipulators*, pp. 245–268. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008.

[134] P. Falco and C. Natale, "On the Stability of Closed-Loop Inverse Kinematics Algorithms for Redundant Robots," *IEEE Transactions on Robotics*, vol. 27, no. 4, pp. 780–784, 2011.

[135] B. Dariush, Y. Zhu, A. Arumbakkam, and K. Fujimura, "Constrained closed loop inverse kinematics," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pp. 2499 –2506, May 2010.

[136] B. Dariush, M. Gienger, A. Arumbakkam, Y. Zhu, B. Jian, K. Fujimura, and C. Goerick, "Online transfer of human motion to humanoids," *International Journal of Humanoid Robotics*, vol. 6, no. 2, pp. 265–289, 2009.

[137] B. Dariush, M. Gienger, A. Arumbakkam, C. Goerick, Y. Zhu, and K. Fujimura, "Online and markerless motion retargeting with kinematic constraints," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pp. 191 –198, Sept. 2008.

[138] M. I. Sánchez, J. Á. Acosta, and A. Ollero, "Integral Action in First-Order Closed-Loop Inverse Kinematics. Application to Aerial Manipulators.," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, (Seattle, Washington), pp. 5297–5302, 2015.

[139] M. T. Hagan, H. B. Demuth, M. H. Beale, and O. De Jesús, *Neural Network Design*. Oklahoma: Martin Hagan, 2014.

[140] B. Warsito, R. Santoso, Suparti, and H. Yasin, "Cascade Forward Neural Network for Time Series Prediction," *Journal of Physics: Conference Series*, vol. 1025, no. 012097, pp. 1 – 8, 2018.

[141] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics : Modelling, Planning and Control*. London, UNITED KINGDOM: Springer London, Limited, 2009.

[142] N. Kreciglowa, K. Karydis, and V. Kumar, "Energy efficiency of trajectory generation methods for stop-and-go aerial robot navigation," *2017 International Conference on Unmanned Aircraft Systems, ICUAS 2017*, pp. 656–662, 2017.

[143] J. Liu, *Sliding Mode Control Using MATLAB*. London: Academic Press, 2017.

[144] R. Vepa, *Nonlinear Control of Robots and Unmanned Aerial Vehicles: An Integrated Approach*. Boca Raton: CRC Press, 2016.

[145] F. Muñoz, I. González-Hernández, S. Salazar, E. S. Espinoza, and R. Lozano, "Second order sliding mode controllers for altitude control of a quadrotor UAS: Real-time implementation in outdoor environments," *Neurocomputing*, vol. 233, no. April 2017, pp. 61–71, 2017.

[146] S. Bouabdallah and R. Siegwart, "Backstepping and sliding-mode techniques applied to an indoor micro Quadrotor," in *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2005, pp. 2247–2252, 2005.

[147] S. Mallavalli and A. Fekih, "Sliding mode-based fault tolerant control designs for quadrotor UAVs-A comparative study," in *IEEE International Conference on Control and Automation, ICCA*, (Ohrid), pp. 154–159, 3–6 July 2017.

[148] M. O. Efe, "Integral sliding mode control of a quadrotor with fractional order reaching dynamics," *Transactions of the Institute of Measurement and Control*, vol. 33, no. 8, pp. 985–1003, 2011.

[149] F. Plestan, Y. Shtessel, V. Bregeault, and A. Poznyak, "New methodologies for adaptive sliding mode control," *International Journal of Control*, vol. 83, no. 9, pp. 1907–1919, 2010.

[150] J. A. Moreno and M. Osorio, "A Lyapunov approach to second-order sliding mode controllers and observers," in *Proceedings of the IEEE Conference on Decision and Control*, no. January, pp. 2856–2861, 2008.

[151] I. González-Hernández, S. Salazar, F. Muñoz, and R. Lozano, "Super-twisting control scheme for a miniature Quadrotor aircraft: Application to trajectory-tracking problem," in *International Conference on Unmanned Aircraft Systems, ICUAS*, pp. 1547–1554, 2017.

[152] M. Kahouadji, A. C. Braham, I. Meslouli, M. R. Mokhtari, and B. Cherki, "Robust attitude control based on continuous differentiator and adaptive super twisting for 3 DOF quadrotor," in *2018 6th International Conference on Control Engineering and Information Technology, CEIT 2018*, (Algiers), pp. 1–6, IEEE, 29–31 October 2018.

[153] H. Castañeda and J. Gordillo, "Spatial Modeling and Robust Flight Control Based on Adaptive Sliding Mode Approach for a Quadrotor MAV," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 93, no. 1, pp. 101–111, 2019.

[154] A. Levant, "Chattering analysis," *IEEE Transactions on Automatic Control*, vol. 55, no. 6, pp. 1380–1389, 2010.

[155] C. A. Martínez-Fuentes, U. Pérez-Ventura, and L. Fridman, "Chattering analysis for Lipschitz continuous sliding-mode controllers," *International Journal of Robust and Nonlinear Control*, vol. 31, no. 9, pp. 3779–3794, 2021.

[156] M. Asad, M. Ashraf, S. Iqbal, and A. I. Bhatti, "Chattering and stability analysis of the sliding mode control using inverse hyperbolic function," *International Journal of Control, Automation and Systems*, vol. 15, no. 6, pp. 2608–2618, 2017.

[157] M. Frigo and S. G. Johnson, "The design and implementation of FFTW3," *Proceedings of the IEEE*, vol. 93, no. 2, pp. 216–231, 2005.

[158] N. Kehtarnavaz, "CHAPTER 7 - Frequency Domain Processing," in *Digital Signal Processing System Design* (N. Kehtarnavaz, ed.), pp. 175–196, Burlington: Academic Press, second edition ed., 2008.

[159] J. B. Allen and L. R. Rabiner, "A Unified Approach to Short-Time Fourier Analysis and Synthesis," *Proceedings of the IEEE*, vol. 65, no. 11, pp. 1558–1564, 1977.

[160] L. Tan and J. Jiang, "Discrete Fourier Transform and Signal Spectrum," in *Digital Signal Processing*, pp. 91–142, 2019.

[161] B. L. G. Costa, C. L. Graciola, B. A. Angélico, A. Goedtel, and M. F. Castoldi, "Metaheuristics optimization applied to PI controllers tuning of a DTC-SVM drive for three-phase induction motors," *Applied Soft Computing Journal*, vol. 62, pp. 776–788, 2018.

[162] M. J. Blondin, J. Sanchis, P. Sicard, and J. M. Herrero, "New optimal controller tuning method for an AVR system using a simplified Ant Colony Optimization with a new constrained Nelder?Mead algorithm," *Applied Soft Computing Journal*, vol. 62, pp. 216–229, 2018.

[163] K. Hassani and W. S. Lee, "Multi-objective design of state feedback controllers using reinforced quantum-behaved particle swarm optimization," *Applied Soft Computing Journal*, vol. 41, pp. 66–76, 2016.

[164] T. T. Mac, C. Copot, T. T. Duc, and R. De Keyser, "AR.Drone UAV control parameters tuning based on particle swarm optimization algorithm," in *2016 20th IEEE International Conference on Automation, Quality and Testing, Robotics, AQTR 2016 - Proceedings*, pp. 1–6, IEEE, 2016.

[165] M. Dangor, O. A. Dahunsi, J. O. Pedro, and M. M. Ali, "Evolutionary algorithm-based PID controller tuning for nonlinear quarter-car electrohydraulic vehicle suspensions," *Nonlinear Dynamics*, vol. 78, no. 4, pp. 2795–2810, 2014.

[166] J. O. Pedro, M. Dangor, O. A. Dahunsi, and M. M. Ali, "Dynamic neural network-based feedback linearization control of full-car suspensions using PSO," *Applied Soft Computing Journal*, vol. 70, pp. 723–736, 2018.

[167] G. Reynoso-Meza, X. Blasco, J. Sanchis, and M. Martínez, "Controller tuning using evolutionary multi-objective optimisation: Current trends and applications," *Control Engineering Practice*, vol. 28, no. July 2014, pp. 58–73, 2014.

[168] A. Rodríguez-Molina, E. Mezura-Montes, M. G. Villarreal-Cervantes, and M. Aldape-Pérez, "Multi-objective meta-heuristic optimization in intelligent control: A survey on the controller tuning problem," *Applied Soft Computing Journal*, vol. 93, p. 106342, 2020.

[169] R. C. Gutiérrez-Urquídez, G. Valencia-Palomo, O. M. Rodríguez-Elias, and L. Trujillo, "Systematic selection of tuning parameters for efficient predictive controllers using a multiobjective evolutionary algorithm," *Applied Soft Computing Journal*, vol. 31, pp. 326–338, 2015.

[170] M. J. Mahmoodabadi, M. Taherkhorsandi, and A. Bagheri, "Optimal robust sliding mode tracking control of a biped robot based on ingenious multi-objective PSO," *Neurocomputing*, vol. 124, pp. 194–209, 2014.

[171] V. Martínez-Cagigal, "Multi-objective particle swarm optimization (MOPSO)." `https://www.mathworks.com/matlabcentral/fileexchange/62074-multi-objective-particle-swarm-optimization-mopso`, 2017. Accessed; 07-January-2019.

[172] K. Deb and T. Goel, "Controlled Elitist Non-dominated Sorting Genetic Algorithms for Better Convergence," in *Evolutionary Multi-Criterion Optimization. EMO 2001. Lecture Notes in Computer Science* (E. Zitzler, L. Thiele, K. Deb, C. Coello Coello, and D. Corne, eds.), vol. 1993, pp. 67–81, Berlin, Heidelberg: Springer, 2001.

[173] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 1917, pp. 849–858, 2000.

[174] K. Deb, "Multi-objective Genetic Algorithms: Problem Diffculties and Construction of Test Problems," *Evolutionary computation*, vol. 7, no. 3, pp. 205–230, 1998.

[175] W. K. G. Assunção, T. E. Colanzi, S. R. Vergilio, and A. Pozo, "A multi-objective optimization approach for the integration and test order problem," *Information Sciences*, vol. 267, pp. 119–139, 2014.

[176] T. M. Cabreira, L. B. Brisolara, and P. R. Ferreira, "Survey on coverage path planning with unmanned aerial vehicles," *Drones*, vol. 3, no. 4, pp. 1–38, 2019.

[177] T. M. Cabreira, C. D. Franco, P. R. Ferreira, and G. C. Buttazzo, "Energy-Aware spiral coverage path planning for UAV photogrammetric applications," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3662–3668, 2018.

[178] C. Di Franco and G. Buttazzo, "Coverage Path Planning for UAVs Photogrammetry with Energy and Resolution Constraints," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 83, no. 3-4, pp. 445–462, 2016.

[179] BitCraze, "Controllers in the Crazyflie." `https://www.bitcraze.io/documentation/repository/crazyflie-firmware/master/functional-areas/sensor-to-control/controllers/`, 2015. Accessed; 13-August-2022.

[180] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proceedings - IEEE International Conference on Robotics and Automation*, (Shanghai), pp. 2520–2525, 2011.

[181] E. J. Smeur, G. C. de Croon, and Q. Chu, "Cascaded incremental nonlinear dynamic inversion for MAV disturbance rejection," *Control Engineering Practice*, vol. 73, no. January, pp. 79–90, 2018.