

Politecnico di Torino

Politecnico di Torino

Master's Degree Course in Aerospace Engineering

Academic Year 2025/2026

Graduation Session September 2026

Design, Modelling and Control of a Fault-Tolerant Morphing Quadrotor

Sliding-mode and nonlinear predictive control of a variable-geometry airframe, from simulation to flight tests

Supervisors:

Dr. Stefano Primatesta

Dr. Kimon P. Valavanis

Candidate:

Lorenzo Bonelli

Abstract

This thesis presents the design, modelling, identification and control of a morphing quadrotor whose four arms rotate independently about vertical hinges, so that the airframe can adopt a continuum of planforms — the canonical X, T, H and O shapes among them — while airborne. The vehicle is an original prototype, weighing 3.964 kg, which has been fully designed, built, and tested in flight.

The control-oriented model analytically computes the centre of gravity, inertia tensor, and control-allocation matrix from the measured arm angles, ensuring that every geometric quantity used by the controllers is an explicit function of the vehicle's instantaneous shape. Based on this foundation, two distinct flight-control stacks are implemented:

First, a shape-scheduled sliding-mode controller operating at 250 Hz, equipped with an adaptive moment-bias estimator and configuration-dependent moment caps. Second, a morphing-aware nonlinear model-predictive controller (using `acados` real-time-iteration SQP, with $N = 30$ nodes over 10 ms at 100 Hz on the onboard companion computer), which features an offset-free disturbance observer and a supervisory guard. This guard enables a one-shot bumpless fallback to the sliding-mode stack when required.

Above these controllers, an energy-aware geometry optimiser dynamically reshapes the airframe online, driven solely by the requested wrench.

The parameters of the model were identified twice, on the bench and in flight, and every bench value that could be re-identified in flight was found to be wrong — the thrust coefficient by a factor of 1.78 — a discrepancy reported here, with its causes, as a methodological result. The system was validated in PX4 software-in-the-loop — on an idealised twin at design stage, on a deliberately perturbed one for the fault experiments, and finally with both control stacks on the plant identified from flight data — and in an outdoor campaign of nineteen instrumented flights over twelve field days. This campaign demonstrated fault-tolerant morphing on hardware. In one instance, where a propeller was destroyed in an earlier flight and the system refitted to a single rotor, the fault-blind optimiser converged three times out of three to the same reshape. This process successfully unloaded the degraded rotor and halved the static trim requirements.

The campaign also yielded the project's best tracking performance: a planar RMSE of 0.0732 m on a streamed figure-of-eight under predictive control, representing a 29 % improvement over the sliding-mode baseline on an identical trajectory. However, twenty seconds after that flight's first reconfiguration, a crash revealed the limit of the work: an arm-and-servo compliance mode near 6 Hz, which is not captured by rigid-arm models and triples the real loop gain at 3 Hz in the

T configuration. This dynamic effect consumed the predictive controller's entire stability margin. The thesis reports this event as a significant result rather than an accidental failure.

Acknowledgements

I would like to thank my supervisors for their guidance throughout this work, and for the freedom to build, fly and occasionally break an aircraft of my own design.

This thesis owes a great deal to the people who spent field days holding a transmitter, carrying batteries and calling out numbers while the vehicle was in the air. A flight campaign is not a solitary exercise, and this one was no exception.

Finally, I thank my family and my friends, for their patience and their support over the whole course of my studies.

Table of Contents

List of Tables VI

List of Figures VIII

Acronyms X

List of Symbols XIV

1 Introduction 1
- 1.1 Context and motivation . . . 1
- 1.2 Problem statement and objectives . . . 3
- 1.3 Contributions . . . 5
- 1.4 Scope and status of the evidence . . . 7
- 1.5 Thesis outline . . . 7

2 State of the art 9
- 2.1 Multirotor UAVs: fundamentals . . . 9
- 2.2 Variable-geometry and morphing multirotors . . . 10
- 2.3 Control of variable-geometry multirotors . . . 14
- 2.4 Disturbance rejection and payload transport . . . 17
- 2.5 Discussion and positioning of this work . . . 18

3 Design of the morphing quadrotor 21
- 3.1 Requirements and design drivers . . . 21
- 3.2 Concept and configuration space . . . 22
- 3.3 Airframe and mechanical design . . . 25
- 3.4 Actuation of the morphing degrees of freedom . . . 28
- 3.5 Payload provision and the 3 kg disturbance requirement . . . 31
- 3.6 Avionics, power and system integration . . . 31

4 Aerodynamic and propulsion characterization 33
4.1 Propulsion system overview . . . 33
4.2 The bench campaign . . . 34
4.3 Parameter identification . . . 38
4.4 Summary: the bench results and their status . . . 44

5 Variable-geometry modeling 46
5.1 Notation and reference frames . . . 46
5.2 Kinematics . . . 48
5.3 Mass and variable-geometry model . . . 49
5.4 Rotor model and PX4 interface . . . 53
5.5 Control allocation . . . 56
5.6 Limits of the model . . . 58

6 System identification 60
6.1 The two-source problem . . . 60
6.2 Bench identification . . . 61
6.3 Mass, inertia and geometry . . . 62
6.4 Thrust coefficient identified in flight . . . 62
6.5 Torque coefficient: an interval, not a measurement . . . 63
6.6 Actuator time constant and its amplitude dependence . . . 64
6.7 The arm and servo subsystem . . . 66
6.8 The arm-compliance mode . . . 66
6.9 Aerodynamic and energetic identification in flight . . . 68
6.10 What could not be identified . . . 69
6.11 Consolidated parameters . . . 70

7 Sliding-mode flight control 71
7.1 Attitude controller: sliding mode . . . 71
7.2 Moment-bias estimator . . . 76
7.3 Altitude controller: sliding mode with anti-windup . . . 78
7.4 Dynamic moment caps . . . 79
7.5 Actuator-lag lead compensation . . . 80
7.6 Arm-motion detector . . . 80
7.7 Discretisation and control loop . . . 81
7.8 Start-up gain invariants . . . 82

8 Energy-aware geometry optimization 84
8.1 Motivation: morphing as a degree of freedom . . . 84
8.2 Problem formulation . . . 85
8.3 Online optimisation algorithm . . . 89
8.4 Integration with the control stack . . . 91

8.5 The demonstrated case: fault-tolerant morphing 93
8.6 What the optimiser buys, and what it does not 95

9 Nonlinear model-predictive control with supervised fallback 99
9.1 Motivation and design goals 99
9.2 Prediction model 100
9.3 Optimal control problem 102
9.4 Offset-free disturbance observer 106
9.5 Software architecture 107
9.6 Deployed robustness layer 108
9.7 Supervised hot-standby fallback 110
9.8 Limits of the rigid-arm prediction model 113
9.9 Comparison with the sliding-mode baseline 115

10 Simulation framework and experimental design 117
10.1 Simulation pipeline 117
10.2 Software architecture of the control stack 124
10.3 Experimental design 128

11 Hardware realisation and the real-flight campaign 134
11.1 The as-built vehicle 134
11.2 Porting the control stack to hardware 135
11.3 The arm failsafe architecture 137
11.4 Identifying the model from flight data 138
11.5 The flight-test protocol as flown 138
11.6 What the campaign established 139
11.7 What the campaign did not establish 139

12 Results 141
12.1 Part I — Simulation results 141
12.2 Part II — Real-world flight tests 154

13 Conclusions and future work 165
13.1 Conclusions 165
13.2 Limitations 168
13.3 Future work 170

A Jacobians of the predictive model 173
A.1 Linearisation point and block structure 173
A.2 Translational and kinematic blocks 174
A.3 Rotational block: gyroscopic term and control effectiveness 175
A.4 Actuator lag and the discrete Jacobians 176

B Physical parameters and symbol-to-code map 177
B.1 Geometry, mass and inertia 177
B.2 Propulsion, actuation and failsafe thresholds 177
B.3 Symbol-to-code map 178
B.4 Known divergences between the code and what flies 180

C Flight-test protocol 181
C.1 Entry gates and standard setup 181
C.2 Override and emergency semantics 182
C.3 Test cards and abort criteria 183
C.4 Standard post-flight metrics 183

Bibliography 185

List of Tables

3.1 Design requirements 22
3.2 Configuration presets and their mechanical properties 25
3.3 System components 32

4.1 Bench propulsion results and their status 45

5.1 Parameters of the mass model 50

6.1 Bench and flight parameter estimates 61
6.2 Actuator time constant versus step amplitude 64
6.3 Servo loads measured in flight 66
6.4 Consolidated identified parameters 70

7.1 Nominal parameters of the attitude SMC controller. The three columns correspond to the axes (roll, pitch, yaw). 74
7.2 Gains derived from Table 7.1 in the X configuration, evaluated at the CAD-calibrated inertia of Table 3.2. K and C are the stiffness and damping of the linear loop inside the boundary layer. 75

8.1 Deployed optimiser weights and constants 89

9.1 Deployed cost weights 103
9.2 Deployed failover trip conditions 113
9.3 Comparison of the two control stacks 116

10.1 Validation of the disturbed twin against the flown aircraft: hover with the flight 5 constant trim moment injected, plus the velocity-antisymmetric term with coefficient c (in $\mathrm{N\,m\,s\,m^{-1}}$). The twin reproduces frequency and drift with a measured coefficient, but damps $\sim$25 % too much per half-cycle. 121

10.2 Fidelity map of the validation environments. Increasing hardware realism (top to bottom) is paid for by decreasing flight-dynamics fidelity. The nominal SITL row validates logic, not robustness: robustness claims require the disturbed-twin row. 124
10.3 Metrics used throughout the experimental evaluation. Roll and pitch are obtained from the odometry quaternion; "drift" is the planar distance from a reference position whose meaning (hover origin, phase start or path reference) depends on the experiment. 128
10.4 The executed campaign matrix (2026-07-28 to 2026-09-10). Every run carries a harness-adjudicated verdict; fault-phase runs that complete but fail the healthy-air tilt label are discussed in Sec. 10.3.1. 131

12.1 Register of the simulation experiments 143
12.2 Tracking error of the two stacks, final campaign 145
12.3 Fault-injection run matrix 149
12.4 Hover on hardware 154
12.5 Trajectory tracking on hardware 156
12.6 Helix tracking decomposed by channel 156
12.7 SMC versus NMPC on hardware 159
12.8 Fault-tolerant morphing on hardware, per trial 159
12.9 Measured cost of each geometry 161
12.10 Simulator against aircraft, matched quantities 162

13.1 The objectives, revisited 167

B.1 Airframe geometry, mass and inertia 178
B.2 Shape presets, levers and inertia 178
B.3 Propulsion and actuation parameters 179
B.4 Hardware ESC throttle-to-speed table 179
B.5 Known repository-versus-flight divergences 180

List of Figures

1.1 The morphing quadrotor of this thesis 2

2.1 Taxonomy of reconfigurable aerial vehicles 12

3.1 Arm-angle convention and the rotor stagger 23
3.2 The four configuration presets 24
3.3 Section through the hinge axis 27
3.4 From generative design to a printable part 27
3.5 Static structural analysis of the arm 29
3.6 Stress concentration at the web root 30

4.1 Propulsion test rig 35
4.2 Load-cell alignment and tare 37
4.3 Why the bench torque scale was rejected 39
4.4 Pointwise thrust coefficient 40
4.5 Thrust versus throttle against the manufacturer's table 42
4.6 Bench thrust law and throttle-to-speed map 42
4.7 Amplitude independence and torque saturation of the spin-up . . . 43
4.8 Actuator time constant across the operating envelope 44

5.1 Airframe reference geometry, top view 47
5.2 The four shape presets 48
5.3 Lever arms and inertia across the morph envelope 54
5.4 Attainable moment and lever arms by configuration 55

6.1 Actuator time constant and first-order fit quality 65
6.2 Doubling the profile velocity does not move the mode 67
6.3 Roll envelope through the flight phases 67
6.4 The arm-compliance mode identified on the accident log 68

7.1 Sliding-mode control architecture 72
7.2 Inertia-scheduled attitude gains 73

7.3 The boundary layer contains the whole commanded envelope 74

8.1 Fixed frame against reshape under the ramped fault 94
8.2 The optimiser on hardware with a physically damaged propeller . . 96
8.3 The spectral cost of staircase convergence 96

9.1 Predictive control architecture 109
9.2 Guard state machine and its gates 112
9.3 Failover timing budget and measured takeovers 113
9.4 The predictive controller on hardware 115

10.1 What each validation environment adds, and what it cannot claim . 123
10.2 The control stack as a node graph 127

12.1 The shape-O contrast between the stock autopilot and the custom stack 144
12.2 The two stacks on the same reference, overlaid 146
12.3 The predictive observer on the payload sequence 148
12.4 The geometry optimiser off against on 148
12.5 Fault-tolerant morphing under the replayed flight-9 rotor fault . . . 150
12.6 Per-lap figure-of-eight RMSE by configuration, Flight 15 155
12.7 Figure-of-eight under NMPC, three laps overlaid 157
12.8 Helix tour on hardware, three shapes 158
12.9 The reshape under the fault, flight against simulation 161
12.10 Simulation against flight on the same references 163

Acronyms

computer-aided design (CAD)

Parametric solid model from which mass, centre of gravity and inertia are derived

centre of gravity (CoG)

Centre of gravity; on a morphing airframe a function of the arm configuration, recomputed online

data distribution service (DDS)

Middleware standard used by ROS 2, here over a micro XRCE-DDS bridge to the autopilot

extended Kalman filter (EKF)

Recursive state estimator of the autopilot, fusing inertial, magnetic, barometric, satellite and range measurements

electronic speed controller (ESC)

Power stage converting the throttle command into the three-phase drive of a brushless motor

fused deposition modelling (FDM)

Additive-manufacturing process used for the printed structural parts

finite-element analysis (FEA)

Numerical structural analysis, used to size the printed arm links and connectors

front-right-down (FRD)

Body-fixed axis convention: x forward, y right, z down, with no axis swap anywhere in the stack

frequency response function (FRF)

Measured transfer function between excitation and response; used to identify the arm-compliance mode

fault-tolerant morphing (FTM)

Use of the morphing degree of freedom to redistribute authority and trim after the partial loss of a rotor

global navigation satellite system (GNSS)

Satellite positioning for outdoor navigation; the campaign flew on standard reception

global positioning system (GPS)

The satellite constellation and receiver used for outdoor position fixing

hardware-in-the-loop (HIL)

Validation setup in which the real flight hardware runs against a simulated plant

high-performance interior-point method (HPIPM)

Interior-point solver for the structured quadratic programs of the predictive controller

inertial measurement unit (IMU)

Triaxial accelerometer and gyroscope package, the primary source of the attitude estimate

model-predictive control (MPC)

Feedback law optimising a finite-horizon cost subject to explicit state and input constraints

north-east-down (NED)

Local navigation frame with x north, y east, z down

nonlinear model-predictive control (NMPC)

Predictive controller on the full nonlinear, morphing-aware model, solved at every tick

carbon-fibre-reinforced PETG (PET-CF)

Carbon-fibre-filled PETG, the printing material of the airframe

proportional-integral-derivative (PID)

Classical three-term feedback law; the servo inner loops and the native autopilot baseline

power spectral density (PSD)

Distribution of signal power over frequency, used to track the oscillatory modes observed in flight

pulse-width modulation (PWM)

Signalling format between the autopilot and the speed controllers

quadratic program (QP)

Optimisation problem with quadratic cost and linear constraints, solved at every predictive tick

radio control (RC)

The pilot's manual link: kill switch, override and takeover, always winning over the autonomous stack

root-mean-square error (RMSE)

Root-mean-square error, the primary tracking metric of this thesis

robot operating system (ROS)

Middleware and node framework of the companion-side control stack

real-time iteration (RTI)

One Newton-type iteration of the nonlinear program per control tick, warm-started

real-time kinematic (RTK)

Carrier-phase differential satellite positioning; no base station was operated in this campaign

simulation-in-hardware (SIH)

PX4 mode simulating the vehicle on the autopilot itself; used only for transport and failsafe validation

software-in-the-loop (SITL)

Unmodified firmware and companion stack running against a Gazebo plant on a workstation

sequential least-squares programming (SLSQP)

Gradient-based solver for the bounded nonlinear program of the geometry optimiser

sliding-mode control (SMC)

Robust feedback law driving the state onto a sliding surface despite bounded matched uncertainty

sequential quadratic programming (SQP)

Iterative method solving a quadratic approximation of a nonlinear program at each step

unmanned aerial vehicle (UAV)

Aircraft flown without an onboard pilot

List of Symbols

Symbol	Meaning	Units / value
Frames, pose and geometry		
$\boldsymbol{p}$	Position of the vehicle in the local NED frame	m
ϕ, θ, ψ	Roll, pitch and yaw (body FRD, $+X$ forward, $+Y$ right, $+Z$ down)	rad
$\boldsymbol{\omega}_b$	Body angular velocity	$\mathrm{rad\,s^{-1}}$
θ_i	Absolute heading of arm i, measured from $+X$ towards $+Y$	°
δ_i	Commanded joint offset of arm i from the X neutral, Eq. (3.1)	°, ± 135
$\boldsymbol{\theta}_{\mathrm{base}}$	X-neutral headings	[45,225,315,135]°
D	Hinge offset from the body centre, per axis	0.113072 m
L	Hinge-to-rotor length	0.235 m
R	Propeller radius (15×5.5)	0.1905 m
x_i, y_i	Rotor coordinates relative to the instantaneous CoG, Eq. (5.18)	m
$\bar{\ell}_\phi, \bar{\ell}_\theta$	Mean roll and pitch lever arms, Eq. (5.19)	m
Mass, inertia and propulsion		
m	Total rigid mass (no payload)	3.964 kg
$\mathbf{I}(\boldsymbol{\theta})$	Configuration-dependent diagonal inertia tensor	$\mathrm{kg\,m^2}$
J_{xx}, J_{yy}, J_{zz}	Its entries, CAD-calibrated by Eq. (5.17)	$\mathrm{kg\,m^2}$
k_T	Thrust coefficient, identified in flight	$4.1547 \times 10^{-5}\,\mathrm{N\,s^2}$
k_Q	Reaction-torque coefficient (an interval, not a measurement)	k_Q/k_T = 0.021 m to 0.027 m
ω_i	Speed of rotor i	$\mathrm{rad\,s^{-1}}$
ω_{hov}	Hover rotor speed	$483.7\,\mathrm{rad\,s^{-1}}$
T_{hov}	Hover thrust	38.887 N
τ_{act}	First-order actuator time constant, Eq. (5.22)	s

Symbol	Meaning	Units / value
Control: wrench and allocation		
$\boldsymbol{\tau}$	Commanded wrench $(T, M_\phi, M_\theta, M_\psi)$, Eq. (5.24)	N, N m
$\mathbf{A}(\boldsymbol{\theta})$	Allocation matrix, Eq. (5.25)	—
$\boldsymbol{M}$	Attitude moment produced by the sliding-mode law, Eq. (7.3)	N m
$\boldsymbol{b}$	Learned moment bias (adaptive trim)	N m
Sliding-mode controller		
$\boldsymbol{e}, \dot{\boldsymbol{e}}$	Attitude error and its filtered rate, Eq. (7.1)	rad, rad s^{-1}
$\boldsymbol{s}$	Sliding surface $\dot{\boldsymbol{e}} + \boldsymbol{\lambda} \odot \boldsymbol{e}$, Eq. (7.2)	rad s^{-1}
$\boldsymbol{\lambda}$	Surface slope	5.5 s^{-1} (roll, pitch)
$\boldsymbol{\delta}$	Boundary-layer width	2.5 rad s^{-1} (roll, pitch)
$\boldsymbol{k}_s, \boldsymbol{k}_d, \boldsymbol{k}_i$	Sliding, derivative and integral gains, Eqs. (7.4)–(7.6)	inertia-scheduled
$\boldsymbol{\omega}_n, \boldsymbol{\zeta}$	Target natural frequency and damping ratio	4.6 rad s^{-1}, 0.15
ρ	Ratio of integral bandwidth to $\boldsymbol{\omega}_n$	0.5
$\boldsymbol{I}_e$	Guarded error integral	rad s
Predictive controller		
N	Prediction horizon length	steps
$\boldsymbol{x}, \boldsymbol{u}$	Prediction state and input	—
$\boldsymbol{d}_v, \boldsymbol{d}_M$	Estimated force and moment disturbances (offset-free augmentation)	m s^{-2}, N m
$\mathbf{Q}, \mathbf{R}$	Stage weights on state and input	—
$\lvert L \rvert$	Open-loop gain magnitude at the compliance frequency	—

Chapter 1

Introduction

1.1 Context and motivation

Multirotor unmanned aerial vehicles (UAVs) have become a standard tool for inspection, mapping, transport, and aerial manipulation. The vast majority of these platforms are *rigid*, meaning their geometry—and consequently their inertia properties and distribution of control authority—is fixed at the design stage. A rigid airframe represents a single static compromise among footprint, agility, payload capacity, and disturbance rejection capabilities. As such, it lacks the ability to adapt once the mission requirements, payload configuration, or environmental conditions change.

Morphing (or variable-geometry) multirotors relax this constraint by allowing the airframe to reconfigure in flight. By rotating, folding or sliding their arms, these vehicles can squeeze through narrow gaps, change their footprint, redistribute control authority across the body axes and shift their centre of gravity to compensate for an off-centre or moving payload [1, 2, 3]. This flexibility comes at a price: the plant becomes time-varying, the mass and inertia properties depend on the instantaneous configuration, and the control allocation must be recomputed online [4]. Standard autopilots, which assume a fixed geometry, are not designed to handle these dynamic effects.

Three primary application drivers justify the associated complexity. The first is geometric in nature: inspection within industrial plants, tunnels, or collapsed structures requires a vehicle capable of shrinking its planform to navigate narrow openings and subsequently expanding it to recover stability margins. In this scenario, the operational footprint is dictated by the physical constraints of the environment rather than by the inherent flight envelope of the vehicle.

The second driver concerns aerial manipulation and the transport of off-centre or shifting loads. A displaced centre of gravity generates a persistent aerodynamic



Figure 1.1: The vehicle this thesis is about, in its neutral X configuration (propellers omitted for clarity). Four arms rotate independently about vertical hinges at the corners of the central body, so that the footprint, the inertia tensor and the control allocation change continuously *in flight*. It was designed and built from scratch, weighs 3.964 kg, and flew nineteen times; every result in Chapters 11 and 12 was obtained on this machine. The design is the subject of Chapter 3.

moment that the attitude control loop must counteract using static trim authority, which may exceed available actuator budgets under such conditions.

The third driver emerges directly from this study: *control authority is itself a resource that can be redistributed through geometric adaptation.* When a rotor degrades, whether due to a chipped propeller or a partially failed motor, the wrench deliverable by the allocator shrinks asymmetrically. The vehicle then faces two distinct options: it may attempt to compensate for the resulting trim error via feedback control until actuators saturate, or it may reconfigure its geometry to rebuild effective lever arms around the fault. This thesis designates the latter approach as *fault-tolerant morphing* and demonstrates its viability in flight, on physical hardware, with a mechanically damaged propeller.

The gap this thesis addresses is therefore one of *evidence* rather than of ideas. Morphing multirotors are most often demonstrated as mechanisms, or flown indoors with an external motion-capture system and a controller that treats each configuration as a separate operating point; comparatively few carry a single configuration-spanning closed-loop stack outdoors on their own airframe, fewer still compare two controller families on the same hardware under the same protocol, and the use of geometry as a fault-tolerance mechanism is, to the author's knowledge, not reported as a hardware demonstration with a physically damaged rotor — a

reading the survey of Chapter 2 supports, since none of the morphing platforms it reviews reports a propulsion-fault experiment at all.

The work reported here was carried out end to end on one machine: the airframe, the propulsion, the arm actuation, the two flight-control stacks, the simulation twin and the flight-test infrastructure were designed, built and then flown, nineteen times, until the campaign ended in the crash of Flight 19.

1.2 Problem statement and objectives

This thesis addresses the design and control of a fully custom morphing quadrotor whose four arms can rotate independently, letting the vehicle adopt a continuum of configurations, the canonical X, O, T and H shapes among them. The vehicle is designed to carry an off-centre payload of up to 3 kg, which the control stack must reject without ever sensing it; the sliding centre-of-gravity behaviour was emulated in simulation during the early development, and the physical sliding mechanism is left as future work (Chapter 3). The platform — airframe, propulsion, actuation and the complete control stack — was designed from scratch, built and flown.

The central problem is to keep such a vehicle stable and accurately controlled *across* its configuration space, while the geometry, the inertia and the control allocation change in flight and while persistent, unsensed disturbances — propulsion asymmetry, wind, rotor degradation and, by design budget, an off-centre payload — act on the vehicle. Four research questions organise the work.

1. **Can one control stack span the configuration space?** Is it possible to fly a single controller, with no per-shape retuning and no gain schedule, across a continuum of geometries whose inertia and allocation are recomputed online — and what, in practice, bounds the set of configurations that are actually flyable?

2. **What is the morphing degree of freedom worth?** While existing literature often assumes that reshaping the airframe yields improvements in efficiency and overall performance, the specific value of this geometric adaptability remains to be quantified under controlled conditions. Measured against a fixed frame in identical operating scenarios, what tangible benefits does morphing provide in terms of energy consumption, control authority, trim margin, and tracking accuracy? Conversely, which aspects of performance remain unchanged or are not improved by such geometric changes?

3. **Can geometry be used as a fault-tolerance mechanism?** This study investigates whether the airframe's geometry can serve as an active mechanism for fault tolerance. Specifically, if one rotor experiences a partial loss of thrust, is it possible for a supervisory control layer to reshape the vehicle without

direct knowledge of the fault state? By monitoring only the wrench demanded by the controller, this layer could potentially reconfigure the arm positions to unload the degraded rotor and restore stable trim conditions on real hardware in a repeatable manner.

4. **Does a predictive controller pay for itself?** On the same airframe, the same model and the same experiments, does a nonlinear model-predictive controller outperform a robust algebraic sliding-mode baseline, and what does that performance cost in terms of safety and of model fidelity?

The objectives that follow from these questions are:

- to design and physically realise an original morphing quadrotor and characterise its propulsion and actuation experimentally, on the bench and then in flight;
- to derive a control-oriented model of the variable-geometry dynamics in which mass, centre of gravity, inertia and allocation are explicit functions of the arm configuration, and to state where that model fails;
- to develop a robust flight-control stack that remains stable across the flyable configuration envelope (X, T and H in steady flight, O as a transit-only shape) and rejects persistent, unsensed disturbances — propulsion asymmetry, wind and rotor degradation — without any explicit disturbance model;
- to exploit the morphing capability as an additional degree of freedom, reshaping the geometry online to equalise rotor effort and to recover control authority and trim margin rather than to improve tracking accuracy — a distinction that is measured, not assumed, and that becomes decisive under rotor degradation;
- to use that same degree of freedom as a fault-tolerance mechanism, showing that a fault-blind geometry optimiser can recover control authority and trim after the partial loss of one rotor;
- to investigate an optimisation-based (model-predictive) controller as an alternative to the baseline, together with a supervisory guard that performs a one-shot, bumpless fallback to the sliding-mode stack, and to characterise both the benefit and the limits of that safety net;
- to validate the whole system in a high-fidelity simulation pipeline (PX4 software-in-the-loop with Gazebo) and on the real vehicle;
- to carry out an outdoor flight campaign on the physical prototype — nineteen numbered flights between 19 July and 6 August 2026 — covering hover, waypoint navigation, planar and three-dimensional trajectory tracking in the

X, T and H configurations, in-flight morphing, fault-tolerant reshaping with a physically damaged propeller, and the first hardware deployment of the predictive controller.

1.3 Contributions

The main contributions of this thesis are the following, ordered by the weight of the evidence that supports them.

- **An original morphing quadrotor, built and flown.** The platform was designed and realised from the ground up — rotating-arm mechanism, propulsion selection, servo actuation, avionics integration and the payload-carrying provision — and then flown in an outdoor campaign of nineteen numbered flights over twelve field days. The campaign includes the first outdoor hover, streamed planar and three-dimensional trajectory tracking in three configurations, in-flight reconfiguration under servo protection, and four losses of control, each of which is reported with its investigation.

- **The first hardware demonstration of fault-tolerant morphing.** With a physically damaged propeller — a 22 % to 28 % thrust deficit, measured in flight (Chapter 6) — mounted on rotor 2, the geometry optimiser, which never sees the fault and observes only the commanded moments, converged in three independent trials to the same arm-reshaping signature, reduced the learned roll trim from 0.46 N m to 0.62 N m to 0.27 N m to 0.30 N m and collapsed the pitch trim towards zero, with no actuator trip and three autonomous landings out of three (Chapter 8).

- **A morphing-aware nonlinear model-predictive controller (NMPC), deployed on hardware.** The controller is formulated on an augmented state that includes the rotor-speed lag, with the inertia tensor and lever arms recomputed from measured arm angles at every solve step. It incorporates an offset-free momentum-based disturbance observer and a supervisory guard that monitors solver health, attitude, body rates, and growing attitude oscillations. This last criterion was introduced following the incident described in the next contribution, which had not yet been observed in flight. The guard performs a one-shot, bumpless fallback to the sliding-mode controller when necessary. On hardware, this setup tracked the same figure-of-eight trajectory with a planar RMSE of 0.0732 m, compared to 0.103 m for the best sliding-mode flight on the identical path. This represents a 29 % improvement and constitutes the best tracking performance achieved in the entire campaign (Chapters 9 and 12).

- **A measured structural limit of rigid-arm morphing models.** The flight that produced this result ended in hover within the T configuration, before the T trajectory had commenced, due to a slowly growing roll oscillation at 3 Hz. Post-flight analysis identified the cause: the servo-actuated arms exhibit compliance at this mode frequency. In the T configuration, the real plant exhibits approximately three times the loop gain predicted by rigid-arm models. Consequently, the design's entire stability margin is consumed by a dynamic mode that no model in this study captures. The failure of the protection chain is also reported: every safety trip fired in the correct sequence, roughly twenty seconds after the instability began, because no threshold within the system specifically monitored for growing oscillations (Chapters 6 and 9).

- **A bench-versus-flight system identification process, with discrepancies resolved.** Model constants are identified on a calibrated six-axis bench and, where flight data permits, re-identified in the air. Every parameter identified through both methods proved to be inaccurate at the bench level. Specifically, the thrust coefficient was off by a factor of 1.78, the torque coefficient differed by more than its own confidence band width, and the transport delay varied by an order of magnitude. Furthermore, two servo tuning decisions promoted during bench testing—the integral gain and motion profile velocity—were both reversed after flight validation. Each discrepancy is traced to a specific coupling effect that the bench cannot reproduce, making the failure of the carefully executed bench campaign a methodological finding rather than a catalogue of errors (Chapter 6).

- **An online variable-geometry model and an associated sliding-mode control stack.** The centre of gravity, inertia tensor, and control-allocation matrix are defined as analytic functions of measured arm angles, refreshed at every control tick. Two structural assumptions—the planar rotor layout and the rigidity of the arms—are stated explicitly, noting that flight data later falsified these premises. Built upon this model is a centre-of-gravity-centric allocator, an inertia-scheduled attitude controller with configuration-dependent moment caps, and an adaptive moment-bias estimator. The saturation limit of this estimator is derived at run time to ensure it never overpowers the restoring action in any configuration, effectively absorbing persistent unsensed trims caused by propulsion asymmetry, wind, or rotor degradation (Chapters 5 and 7).

- **A reproducible validation framework.** This framework spans PX4 software-in-the-loop simulations with Gazebo against both nominal and deliberately perturbed plants, companion-in-the-loop executions of the flight

stack, and a documented flight-test infrastructure. The latter includes per-flight command cards with go/no-go criteria, versioned analysis scripts, and machine-readable metrics for every session (Chapters 10 and 11).

1.4 Scope and status of the evidence

Because a large part of this thesis rests on flight data gathered under field conditions, the standing of that evidence is stated once here and repeated wherever it matters.

The flight campaign is *not* a statistical study. Most experimental conditions were executed only once or twice, and all numerical results derived from single flights or pairs of flights are reported with their respective sample sizes explicitly stated. The three matched repetitions typically required for publication standards were not performed.

Most of the simulation evidence comes from one final campaign, flown in September 2026 with both control stacks on the same code and on the plant identified from flight data. Two earlier sets of runs are kept on the design-stage twin where nothing later replaced them, the sliding-payload sequence of the design brief and the baseline flown with the stock autopilot, together with the rotor-fault ladder of 31 July on the final plant; Chapter 12 says so wherever they appear.

Three limits of the realised system are declared from the outset. The O configuration is a *transit-only* shape: it does not sustain hover with the installed servos, which is a measured result and not an omission. The physical sliding-payload mechanism was never built — the 3 kg off-centre load survives as a sizing requirement, exercised as a simulated disturbance in the early design campaign and removed from the simulation model itself on 21 July 2026, not as a demonstrated capability. And reconfiguration under the predictive controller was never made safe on hardware: the remedies devised after the Flight 19 crash were checked only on a harness driven by the measured plant response, and, for the growing-oscillation detector, in simulation and in replay of the accident log, and the three flight cards written to test them (Flights 20–22) were not flown. They are future work, and Chapter 13 treats them as such.

1.5 Thesis outline

The remainder of this thesis is organised as follows. Chapter 2 reviews the state of the art on variable-geometry and morphing multirotors and on the relevant control techniques, and maps the gaps it identifies onto the contributions listed above. Chapter 3 presents the mechanical design, from the mission-driven requirements to the realised airframe, the rotating-arm mechanism, the servo actuation, the payload provision and the avionics integration. Chapter 4 reports the propulsion

characterisation carried out *before* the vehicle flew, together with the part of it that flight later overturned, and Chapter 5 derives the variable-geometry model: mass, centre of gravity and inertia as functions of the arm configuration, the rotor model with its actuator dynamics and PX4 interface, the configuration-dependent allocation, and the limits of the model. Chapter 6 quantifies the parameters of that model by identifying its constants from both bench and flight data. It reports discrepancies between these two sources along with their underlying causes, details which parameters could not be identified, and characterises the arm-compliance mode responsible for the outcome of Flight 19.

Chapter 7 develops the sliding-mode flight controller based on this model, while Chapter 8 presents the energy-aware geometry optimiser and its critical validation through fault-tolerant morphing in both simulation and hardware environments. Finally, Chapter 9 formulates the morphing-aware nonlinear model-predictive controller, its offset-free disturbance observer, its in-node robustness layer and the supervised hot-standby fallback, and reports the structural limit of its rigid-arm prediction model. Chapter 10 describes the simulation and validation pipeline, the fidelity map that assigns each environment the question it can answer, and the design of the experiments; Chapter 11 reports the hardware realisation and the real-flight campaign: the aircraft as built, the porting of the control stack to it, the arm-failsafe architecture, the procedure by which the model was identified from flight data, the test protocol, and an explicit statement of what the campaign did and did not establish; Chapter 12 reports the results in two parts, simulation and outdoor flight; and Chapter 13 returns a verdict on each objective, states the limitations and the negative results, and outlines future work. Three appendices support the main text: Appendix A derives the Jacobians of the predictive model, Appendix B lists every physical parameter with its source in the implementation, and Appendix C records the flight-test protocol as it was actually used.

Chapter 2

State of the art

This chapter develops the background against which the present work is placed. It starts from the conventional, fixed-geometry quadrotor and the control architecture inherited from it (Sec. 2.1), then turns to the body of work that relaxes the rigid-airframe assumption: a taxonomy of reconfiguration strategies (Sec. 2.2.1) and a review of the most relevant realised platforms (Sec. 2.2.2), surfacing what each leaves unaddressed. It then surveys the control side (Sec. 2.3) — robust and sliding-mode methods, optimisation-based and predictive methods, the control allocation that ties a desired wrench to the rotors when the geometry itself is a variable, and the fault-tolerant literature that reconfigures that allocation when a rotor degrades — and the rejection of persistent, never-sensed moment disturbances (Sec. 2.4), the operating regime this thesis singles out. It closes by synthesising the gaps and mapping them onto the contributions of this work (Sec. 2.5).

2.1 Multirotor UAVs: fundamentals

A conventional quadrotor is an *underactuated* rigid body: four upward rotors produce four scalar thrusts, yet the vehicle must track six degrees of freedom. The four rotor speeds map to a four-dimensional control wrench — a collective thrust along the body z-axis and three body moments — so horizontal translation is not directly actuated: the platform accelerates sideways only by first tilting. This couples the position loop to the attitude loop and fixes the now-standard *cascaded* architecture, an outer position loop computing a desired acceleration and turning it into a desired thrust and tilt, and a fast inner attitude loop driving the body orientation to that tilt. This thesis keeps that architecture for its baseline stack and works directly in the PX4 body frame, so that the controller speaks the same language as the autopilot it is compared against.

The last block in the chain is the *control allocation*: the map from the desired

wrench to the individual rotor commands. For a rigid quadrotor this is a single constant matrix, fixed at design time by the arm length and rotor layout and inverted once, with each rotor contributing a thrust $F_i = k_T\,\omega_i^2$, a reaction torque $Q_i = k_Q\,\omega_i^2$ and moments that follow from its lever arm to the centre of mass. The crucial observation for everything that follows is that this convenient structure rests on two assumptions — that the geometry is fixed and that the centre of mass sits at the centre of the rotor pattern — and that a morphing airframe violates both. The Newton–Euler rigid-body model, the cascaded position/attitude split and the constant mixer together form the fixed-geometry baseline, embodied by the native autopilot that serves as the comparison baseline in Chapter 12 — in simulation, since it was never flown on the real aircraft.

2.2 Variable-geometry and morphing multirotors

2.2.1 Taxonomy of reconfigurable aerial vehicles

Morphing multirotors break the fixed-geometry assumption by letting the airframe reconfigure. Four primary families recur, plus a more radical mode-changing class, distinguished by the kinematic primitive the actuator drives and by *when* the reconfiguration happens.

The first is *folding*: arms hinge about a crease or revolute joint to collapse the footprint, almost always between two discrete states, for transport and storage rather than flight performance. Mintchev et al. [3] take this to its limit with a pocket-sized origami quadrotor whose arms unfold passively under propeller torque, with no deployment actuator at all; Tuna et al. [5] maintain a single discrete fold/deploy state but power it with one servomotor and a four-bar crank-rocker mechanism shared across all four arms. What unites this family is that the geometry is *binary* and fixed during flight, so inertia and control allocation remain constant whenever the vehicle actually flies.

The second family, which encompasses the approach adopted in this study, is *rotating-arm* morphing, in which each arm rotates about a body-fixed axis to reshape the planform *in flight*, spanning a continuum rather than two discrete endpoints. Falanga et al. [1] established the canonical instance: four arms, each driven by its own servomotor, rotating to produce the X configuration plus H, O and T for narrow vertical gaps, narrow horizontal gaps and proximity inspection. Shiferaw et al. [6] build a four-arm revolute-joint airframe of the same topology that folds from X toward a compact H, and Cui et al. [2] adopt the same mechanism to morph among X, H, T and Y shapes to fit through restricted scenarios. Actuation can also be removed from the loop entirely: Bucki and Mueller [7] let the arms fold *passively* under aerodynamic and inertial loading, with no joint actuator at all, and Riviere et al. [8] exploit a morphing airframe for agility rather than for footprint.

A reduced-actuation variant collapses the four joints into one: Modi et al. [9] fold about a single central revolute joint driven by one servomotor, so a one-parameter family takes the vehicle from an X-quadcopter to a pair of stacked bicopters. The defining feature of the family is that the centre of gravity, the inertia tensor and the moment arms all change continuously while the vehicle flies.

The third family is *sliding* or *translational* reconfiguration, where mass moves along an axis rather than rotating about one: Liu et al. [4] formalise both modes side by side and treat the resulting time-varying centre of mass and inertia explicitly. A movable payload that slides along a body axis belongs to the same family. The 3 kg unmodelled mass analysed in this study is treated within this category, with the distinction that it acts as a *disturbance* to be rejected rather than a commanded degree of freedom.

The fourth family is *tilting* or thrust-vectoring reconfiguration, where the arms rotate to redirect the thrust vectors rather than to reshape the footprint, typically aiming to recover *full actuation*. The work by Ruggia [10] serves as a representative example, utilising twelve actuators for six controlled degrees of freedom. While this differs in objective from planform morphing, it shares the algebraic complexity that arises because the allocation map depends on the actuated configuration.

A fifth, more radical class changes flight *mode* entirely [11]. Recent surveys of aerial manipulation now recognise shape-changing, interconnected-actuated multi-body platforms as an emerging category in their own right [12]. Furthermore, similar uncertain and time-varying dynamic behaviours appear even without kinematic morphing when aerodynamics are reconfigured on demand [13].

Within this taxonomy the platform of this thesis is squarely a *rotating-arm* morphing quadrotor with four *independently* actuated revolute joints, spanning a continuum that includes the canonical X, O, T and H shapes. It sits at the fully-jointed end of the in-flight rotating-arm family — four independently driven joints where the reduced-actuation designs have one, and a continuum where the folding designs have two states — and, unlike the tilting class, it morphs to redistribute *control authority and footprint* rather than to become fully actuated. Figure 2.1 places the families and this platform among them.

2.2.2 Representative platforms

The closest prior art is the foldable drone of Falanga et al. [1]. It is built, like this work, from scratch around four independently rotating arms producing X/O/T/H shapes, it treats the morphing-induced centre-of-gravity shift and the morphology-dependent allocation as first-class problems, and it is the first to demonstrate stable flight without requiring any symmetry of the morphology. Its allocation re-centres on the recomputed centre of gravity, and its hover-endurance figures quantify the cost of morphing: relative to the X reference, endurance drops by roughly 17 %,

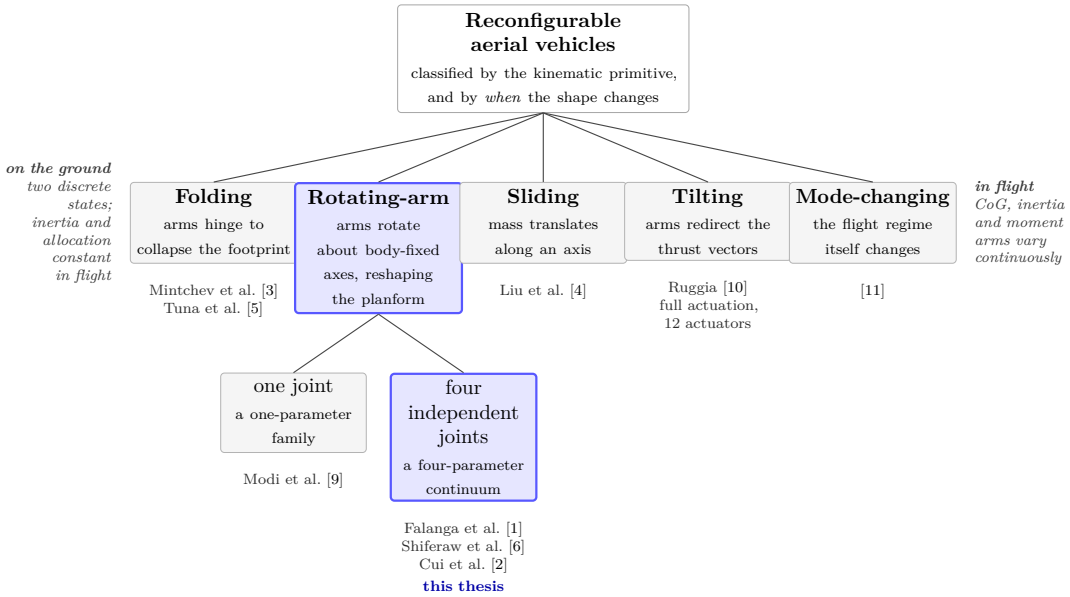


Figure 2.1: The families of reconfigurable aerial vehicle, organised by the kinematic primitive the actuator drives and by *when* the reconfiguration happens — the second axis being the one that matters for control, since it decides whether the centre of gravity, the inertia tensor and the moment arms are constants or signals. The platform of this thesis sits at the fully-jointed end of the in-flight rotating-arm family: four independently driven joints where the reduced-actuation designs have one, and a continuum where the folding designs have two states.

23 % and 63 % for H, T and O — precisely the kind of energetic penalty an online optimiser would seek to avoid. Three limitations are identified by the authors, framing the gap addressed in this study: morphologies are selected *manually* from a fixed task menu; the controller assumes a quasi-static transition ($\dot{\theta} \approx 0$); and automatic morphology selection together with high-speed exploitation of morphing remain open challenges. Endurance is not measured within the scope of this work, as the developed optimiser is evaluated based on rotor-load spread and the residual authority it recovers. Consequently, the comparison focuses on the underlying mechanism rather than on quantitative numerical values.

The platform of Cui et al. [2] adopts the same mechanism and shares the central difficulty, recomputing the allocation matrix online from the instantaneous centre-of-gravity offset, but directs its geometry selection toward *collision-free planning* rather than energy; its one-third battery saving from deforming through a gap is a useful independent confirmation that geometry choice has a real energetic lever. Shiferaw et al. [6], despite the strong topological similarity, stops at the kinematics and open-loop servo sequencing, explicitly deferring any closed-loop morphing-aware controller to future work. Modi et al. [9] is much simpler — one central joint, the allocation offloaded to the stock PX4 mixer and never re-centred

for a centre-of-gravity shift — but its fixed 25° motor tilt, added to retain roll authority at a cost of about 8 % of upward thrust, is a vivid reminder that control authority is itself a function of geometry. At the boundaries of the design space, Ruggia [10] resolve the redundant degrees of freedom of a fully-actuated rotating-arm hexacopter with an online optimising allocation subject to actuator limits — a direct precedent for posing geometry selection as an optimisation, candid about offering no computation-time guarantee — while Liu et al. [4] design a Lyapunov-based adaptive controller for a time-varying reconfigurable quadrotor, but validate only in simulation, assume no parametric uncertainty and report that stability diverges once the control-input delay exceeds 7 ms.

Two properties of the morphing mechanism itself are absent from all of these platforms, and both proved decisive on hardware. The first is that the joint actuator is a failure mode of the *airframe*, not merely of the mechanism: a servomotor that reaches its own overload protection releases its arm, and a released arm in a folded configuration is a free body inside the rotor discs. None of the reviewed works reports arm-actuator load, thermal state or a release-detection scheme, and the fault-diagnosis literature for aerial robots is written around propulsion actuators and sensors rather than articulated joints [14, 15]; this thesis therefore treats the arm driver as a *supervised subsystem*. The second is that the arm-and-servo chain is *compliant*: every model reviewed here, and the model used by both controllers in this thesis, treats the arms as rigid links, whereas in practice the chain has a lightly damped mode of its own in the 6 Hz to 7 Hz band that couples straight into the roll axis, and which the reviewed literature sidesteps rather than solves — Falanga et al. [1] step their transitions through quasi-static increments precisely so that the mechanism's own dynamics never enter the loop.

One further effect cuts across the family: as the planform closes, the rotor discs approach each other and, in the O-like shapes, partially overlap. The reviewed platforms register its *aerodynamic* consequence — Falanga et al. [1] as the endurance cost of the folded shapes, Cui et al. [2] compensating the overlap-induced thrust loss online — and the same penalty is measured in flight on the platform of this thesis (Sec. 3.3.4).

Read together, these platforms leave a consistent set of problems open. None combines, on a single physically-realised airframe: an *online analytical* variable-geometry model in which centre of gravity, inertia tensor *and* allocation matrix are closed-form functions of the live arm angles refreshed every control tick; the treatment of morphing as an *actuator*, with the geometry chosen continuously to *equalise rotor effort* rather than picked manually; the rejection of a persistent, *never-sensed* moment bias; the use of the geometry itself as a *fault-tolerance* resource; and a *fair, like-for-like* comparison between a robust and a predictive controller built on the same model. Where the closest prior art assumes the centre of gravity is known or static, schedules a fixed-form controller rather than rebuilding the geometry

online, demonstrates the mechanism without a configuration-spanning closed-loop controller, or studies a single controller in isolation, this thesis is positioned on the *integrated combination* of those elements, demonstrated on a physically-realised and flight-tested airframe rather than in simulation alone.

2.3 Control of variable-geometry multirotors

2.3.1 Robust and sliding-mode control

A morphing quadrotor is, from the controller's point of view, a plant whose parameters drift continuously, and on top of that an unmodelled, never-sensed moment bias — a degraded rotor, steady wind, the residual asymmetry of a folded shape, or the unmodelled 3 kg payload the design envelope is sized for — loads the attitude axes. This is exactly the structured, bounded, time-varying uncertainty for which *robust* nonlinear control is the natural answer: Michailidis et al. [13] treat a reconfiguring vehicle as a source of time-varying uncertainty for which conventional fixed-model design is limited or inapplicable, and provide a structured comparison of the candidate families. Among them, *sliding-mode control* (SMC) is attractive here for a specific reason: it is by construction insensitive to matched uncertainty and bounded disturbance once the sliding surface is reached, which is the desirable property when a matched, persistent, never-sensed moment disturbance is the dominant unknown. The recognised price is *chattering* from the discontinuous switching term, which the standard boundary-layer remedy trades for a bounded steady-state error.

SMC is well established in the aerial-manipulation and morphing literature as a robust choice [12], and the morphing airframe of Shiferaw et al. [6] is explicitly framed within that lineage even though the closed loop is left to future work. Its value for a *variable-inertia* plant, however, depends on a design detail that much of the prior art handles only implicitly: with fixed gains the closed-loop bandwidth and damping wander as the inertia changes with configuration, whereas *scheduling* the gains on the live inertia — as adopted here, and as motivated by the configuration-dependent moment-arm argument of Modi et al. [9] — makes the closed-loop natural frequency and damping configuration-invariant by construction. Compared with the adaptive-gain Lyapunov design of Liu et al. [4], an algebraic SMC law with a boundary layer is computationally trivial and tolerates a transport delay that the adaptive design does not: the round-trip delay of the deployed stack, measured in flight rather than inferred, is about 30 ms, four times the 7 ms beyond which their simulation diverges, and the platform flies with it because that delay is carried explicitly in a closed-form phase-margin condition checked before every flight. The full derivation is the subject of Chapter 7.

2.3.2 Optimisation-based and predictive control

The complementary answer to robustness is anticipation. *Model-predictive control* repeatedly solves a finite-horizon optimal-control problem over a model of the plant, which lets it respect actuator and state *constraints* explicitly and exploit *preview* of the reference — two things a reactive robust law cannot do natively. For a morphing quadrotor the constraint argument is sharp: the rotor speeds are box-bounded, the total thrust is limited and the feasible attitude envelope is configuration-dependent, so a controller that reasons about these bounds inside its own optimisation has a structural advantage over one that clips after the fact. The nonlinear variant keeps the full rigid-body model rather than a linearisation, which matters when the geometry is far from any single operating point, and surveys record (N)MPC alongside sliding-mode and geometric control as a state-of-the-art family [12, 13].

The practical objection has always been real-time feasibility on embedded hardware. Where prior morphing work either avoids prediction altogether — Falanga et al. [1] re-solve an algebraic Riccati equation online for a *linearised body-rate* controller, and Cui et al. [2] and Modi et al. [9] use differential-flatness or geometric cascades — the NMPC of Chapter 9 is made morphing-aware and real-time by construction: its optimisation variable is the four rotor speeds rather than the wrench, so the allocation and the rotor limits follow the morphing *automatically*; the inertia and lever arms are passed as parameters from the measured arm angles at every solve; and the problem is closed in real time by a real-time-iteration SQP scheme [16] with a Gauss–Newton Hessian and a partial-condensing interior-point solver via `acados` [17], warm-started from the previous solution. This addresses precisely the gap left by Ruggia [10], and the objection can now be answered with a measurement rather than an argument, consistent with what the embedded-NMPC literature reports on comparable hardware [18, 19]: on the flight companion the problem solves in 0.9 ms at the median and 4.2 ms at the worst case over the first complete autonomous flight, without a single quadratic-program failure.

Because both controllers here are built on the same rotor model, geometry, lever arms, allocation and PX4 actuator interface, they can be A/B-compared on identical experiments — a fair head-to-head that no single-controller study in the morphing literature provides — with the predictive controller's structural weaknesses stated plainly. The weakness usually cited, dependence on a solver that can stall, proved to be the lesser of the two issues on this platform. The binding constraint is *model fidelity*: the plant model assumes rigid arms, thereby excluding the 6 Hz compliance mode present in the arm-and-servo chain. When measured against flight data, the real roll loop exhibited approximately three times (10 dB) more gain than predicted by the model at 3 Hz in the T configuration, effectively consuming the entire design margin. Neither weakness is cured here: the supervised fallback catches the consequence of either, while the model gap itself is covered

operationally, by the actuator lag re-identified in flight and by an interim policy that confines predictive trajectory flight to the base configuration.

2.3.3 Control allocation under changing geometry

For a rigid quadrotor, the allocation is a constant matrix inverted once; for a morphing one, it is the single algebraic object through which the entire time-variation of the plant must pass. The literature converges on this necessity even while differing on method. Falanga et al. [1] recompute the moment allocation from each rotor's distance to the *recomputed* centre of gravity, Cui et al. [2] likewise rebuild the matrix online from the instantaneous offset, and Liu et al. [4] present the configuration-dependent control matrix as a stated contribution. The common thread is that re-centring the allocation on the *instantaneous* centre of gravity, rather than the geometric centre, is what a morphing or off-centre-loaded vehicle requires, and it is exactly what a fixed-geometry autopilot cannot do.

Method-wise, the field spans a spectrum from geometric square inversions to the constrained optimisation described by Ruggia [10]. The allocation strategy employed in this work sits at the well-conditioned, square end: a centre-of-gravity-centred direct 4×4 inversion built from live lever arms, with a least-squares fallback when the matrix becomes ill-conditioned. This choice is justified because the platform is conventionally actuated, so the difficulty lies not in redundancy but in the fact that the matrix changes every tick. The two controllers handle this variation differently, which forms part of their comparison: the sliding-mode stack inverts the allocation explicitly, whereas the NMPC sidesteps a separate allocation step by optimising rotor speeds directly, thereby absorbing the mapping into the dynamics.

2.3.4 Fault tolerance and control reconfiguration

A rotor that fails or degrades attacks the same algebraic object that morphing moves — the map from rotor speeds to wrench — and the fault-tolerant control literature has a mature answer for it: *reconfiguration of the allocation*, in which a detection and isolation stage estimates the loss of effectiveness and the controller redistributes the desired wrench over the healthy actuators. The taxonomy of that field is the subject of the review of Zhang and Jiang [20], and its aerial specialisation, including the diagnosis stage that must run first, is surveyed by Fourlas and Karras [14]. Two results bound the severity axis: under a *partial* loss of effectiveness an adaptive estimator can identify the fault and accommodate it inside the existing control structure [15], while under the *complete* loss of a rotor the yaw degree of freedom must be surrendered and the vehicle stabilised in a periodic spin about a reduced attitude [21] — a relaxed equilibrium that predictive formulations later

recovered inside an optimisation respecting the actuator limits explicitly [22], and that has since been flown autonomously with only onboard sensing [23].

What all of these schemes share is that the *geometry stays fixed*: reallocation redistributes demand over lever arms it cannot move, so the moment a degraded rotor can still contribute is bounded by an arm chosen long before the fault. The morphing platforms of Sec. 2.2.2 do not close the gap from the other side: they select their shape for footprint, for gap-crossing or to avoid an allocation singularity — never to redistribute authority around a degraded actuator, and none of them reports a propulsion-fault experiment at all. The theoretical position adopted follows from synthesising the two families: when the arms are actuated, the geometry becomes a reconfiguration variable in itself, and modifying it changes the lever arms, which is precisely what reallocation on a rigid frame cannot achieve. This mechanism is demonstrated in Chapter 12 using a physically damaged propeller with a measured thrust deficit ranging from 22 % to 28 %. The optimiser operates without receiving any fault information; it observes only the moments demanded by the controller, through which channel a degraded rotor appears indistinguishable from an off-centre payload. That demonstration belongs to the sliding-mode stack; repeating it under the predictive controller is a written but unflown experiment. The resulting division of labour is measured rather than assumed, and it recurs throughout the thesis: the *transient* of a fault belongs to the robust controller, and the *steady state* to geometric reconfiguration.

2.4 Disturbance rejection and payload transport

The disturbance regime this thesis singles out is one in which the moment bias is persistent, configuration-dependent and *never sensed*: there is no force-torque sensor, no payload estimator, no thrust-loss detector and no relative-position measurement. Three physically distinct sources produce that same signature on this platform, and all three were met in flight: a degraded rotor, the residual mass and aerodynamic asymmetry of a folded configuration, and steady wind. A movable payload is the fourth member of the family and sizes the design envelope, but it was not carried in any flight test. The survey of Barakou et al. [12] ties this regime directly to performance: when a payload is treated as a lumped disturbance rather than measured or modelled, tracking precision degrades to worse than 10 cm, and the same survey catalogues the standard countermeasures.

The morphing platforms reviewed above mostly stop short of this regime. Falanga et al. [1] grasp and carry an object but do not model it as a persistent control disturbance; Cui et al. [2] compensate only the overlap-induced thrust loss; Modi et al. [9] report no payload model and rely on the robustness of the underlying PID; and Ruggia [10] explicitly leave residual error to an oversimplified thrust-torque

term. Where adaptive or learning machinery appears it is aimed elsewhere: Liu et al. [4] adapt gains for the time-varying inertia but assume no parametric uncertainty, and the reinforcement-learning gain-tuning of Sönmez [24] compensates wind gusts on a fixed-geometry quadrotor while listing payload shifts as future work. The broader robust-control literature frames the same problem cleanly: Michailidis et al. [13] concede that fixed-weighting robust designs hinge on a pre-known uncertainty range and call for *adaptive* online estimation of that range.

This motivates the two-channel rejection strategy implemented, which is kept deliberately blind to the *source* of the disturbance. In the sliding-mode stack, the standing moment is cancelled by an *adaptive moment-bias estimator* driven by the sliding surface itself. This estimator leaks gently toward zero to prevent wind-up and is scaled per axis by the squared mean-lever ratio, ensuring that the weak axis of a collapsed configuration is not over-driven. In the NMPC, the same role is played by an *offset-free* momentum-based observer that augments the model with random-walk force and moment disturbances formed from the *applied* wrench rather than the commanded one. Both channels reject the disturbance without ever estimating the payload, and they are paired with a slow, decoupled, also payload-blind energy-aware optimiser. This optimiser acts on the disturbance's *cause* through the geometry itself: it re-centres the combined centre of gravity under the rotor pattern when the bias stems from an off-centre mass, or moves the lever arms to unload a degraded rotor when the source is different. The second mechanism is the one demonstrated in flight.

2.5 Discussion and positioning of this work

The survey reveals a coherent set of gaps, which the contributions presented in this study address directly. Most morphing-multirotor studies either demonstrate a mechanism or footprint change without implementing a full configuration-spanning closed-loop controller, handle morphing with fixed or gain-scheduled controllers rather than an online analytical geometry model, assume the payload or centre of gravity is known and static, or study a single controller in isolation. The primary contribution lies in the *integrated combination* of these elements, realised on a single airframe and validated through flight testing.

- **Online analytical variable geometry.** While [1] recomputes the centre of gravity and allocation but treats transitions quasi-statically, and [9] merely rescales fixed gains, this work defines the centre of gravity, the diagonal inertia tensor, and the 4×4 allocation matrix as closed-form functions of measured arm angles. These parameters are refreshed at every control tick, ensuring that the plant's time-variation is inherently integrated into both controllers rather than being approximated (Chapter 5).

- **Centre-of-gravity-centred allocation.** Unlike the fixed-geometry mixers in [9] and the native autopilot, and following the lineage of [1, 2, 4], the allocator utilises lever arms relative to the *instantaneous* combined centre of gravity. It employs a direct matrix inverse with a least-squares fallback mechanism (Chapter 7).

- **Clean two-time-scale separation of feedback from cause.** The fast loop nulls the standing moment in closed loop, while a slow, decoupled, and *fault- and payload-blind* multi-start SLSQP optimiser reshapes the geometry responsible for that moment. This process reduces the spread of rotor loads, serving as a monotone proxy for aerodynamic power, and recovers saturation margin on the most heavily loaded rotor. The optimiser is formulated against the control interface rather than any specific controller, allowing it to drive both stacks without modification. Chapter 12 shows the mechanism at work under a degraded rotor: for the sliding-mode stack the reshape is the difference between an authority-starved hover and a completed mission, and in the simulated twin of the hardware fault both stacks converge on the same reshaped geometry. This approach realises the morphing-as-actuator concept that Falanga et al. [1] identified as an open problem regarding automatic morphology selection (Chapter 8).

- **Geometry as a fault-tolerance resource.** The same optimiser, told nothing about the fault, reshapes the airframe around a degraded rotor: it unloads the damaged propeller and roughly halves the standing trim the attitude loop was holding — the learned roll bias falling from 0.46 N m to 0.62 N m to 0.27 N m to 0.30 N m — in three trials out of three on hardware with a physically broken propeller, under the sliding-mode stack; the same experiment under the predictive controller was run against the simulated fault at the flown severity, where both stacks converge on the same reshape, and is written and pending on hardware. Fixed-frame fault-tolerant control can only redistribute demand over lever arms it cannot move, and no reviewed morphing platform reports a propulsion-fault experiment at all.

- **A fair SMC-versus-NMPC head-to-head.** A robust algebraic sliding-mode baseline and a morphing-aware, rotor-speed-input, offset-free NMPC are built on the *same* model and interface, then A/B-compared on the same simulation matrix — four shapes, two references, clean air and gusts, and a twin of the hardware fault, with the sliding-mode stack additionally set against the native autopilot — and on hardware in the one configuration in which both controllers flew the same trajectory. Consequently, the like-for-like comparison that single-controller studies cannot offer is complete in simulation and partial on hardware.

- **The safety question, treated with rigour.** The predictive controller operates under a supervisory watchdog that maintains the sliding-mode stack as a muted hot standby, transferring control authority upon a single latched, irreversible event. This mechanism is not merely proposed; it is enabled by default on the flight hardware, triggers on seven independent conditions, and has been validated in closed-loop testing with a measured hand-over latency of 0.20 s to 0.52 s and a post-adoption position excursion below 0.08 m. It fired once in a real flight, correctly — and this thesis reports the uncomfortable part as well: the trip that fired was the hard body-rate threshold, roughly 20 s after the divergence began, because none of the original trips watched for a slowly growing limit cycle.

Two caveats regarding the scope and limitations of these claims are presented throughout the study. First, the controller comparison is complete in simulation but partial on hardware, as the native autopilot was evaluated only in simulation, while on the real aircraft, the two controllers share exactly one trajectory in a single configuration. Second, the results rely on a hardware campaign comprising nineteen flight sessions rather than simulation alone, with propulsion coefficients re-identified from flight data after bench values were found to overestimate the thrust coefficient by a factor of 1.78. The remaining open issues are explicitly stated in Chapter 13: a like-for-like hardware A/B test across multiple geometries, a hardware baseline against the native autopilot, and sufficient repetitions to convert single-flight measurements into statistical evidence.

The most significant gap addressed is one that no surveyed platform currently closes simultaneously: a single, end-to-end, fully custom variable-geometry quadrotor, flown outdoors on its own airframe, in which morphing is treated as a *control resource* rather than merely a footprint adaptation tool. This includes its use as a fault-tolerance mechanism, reshaping the airframe around a physically damaged rotor, under persistent disturbances that the controllers must reject without ever sensing their source.

Chapter 3

Design of the morphing quadrotor

This chapter describes the design of the morphing quadrotor, which was conceived and realised from scratch. It moves from the requirements to the mechanical design of the airframe and the rotating-arm mechanism, the actuation of the morphing degrees of freedom, and the system integration. The platform is a flying prototype: it was assembled and flown, nineteen flights having been carried out at the time of writing. The airframe was printed in PET-CF, and its arm geometry was fully designed with a generative tool in Fusion 360, the total mass is 3.964 kg. The final design this refers to is the machine on which every result of Chapters 11 and 12 was obtained.

The chapter describes the vehicle as it flew: every figure quoted refers to the final design. The control-oriented *model* of the geometry is deferred to Chapter 5, the propulsion characterisation to Chapter 4, the identification of the physical constants to Chapter 6, and the flight campaign to Chapter 11.

3.1 Requirements and design drivers

A primary external constraint governs the design: a maximum take-off weight below 25 lb (approximately 11.3 kg), established at the outset as the design ceiling for this class of platform. All other requirements are self-imposed engineering targets derived from the study's objectives. Specifically, the vehicle must realise distinct morphing configurations and the continuum between them, accommodate a meaningful payload capacity, retain control authority in every reachable configuration, and treat its own geometry as a degree of freedom *in flight*, rather than as a ground-configurable option. Table 3.1 summarises the resulting design space.

Table 3.1: Design drivers and quantitative targets. The achieved figures refer to the as-built airframe (CAD model of 31 July 2026); thrust and thrust-to-weight are the values identified in Chapter 6.

Driver	Target	Rationale / outcome
Take-off mass	$< 25\,\mathrm{lb}$ (11.3 kg)	Assumed design ceiling. Achieved: 3.96 kg empty, 6.96 kg with the full payload provision.
Morphing configurations	X, T, H and O presets plus the continuum; joint range $\pm 135°$	Footprint change and redistribution of control authority (Sec. 3.2).
Morphing in flight	Reconfiguration at $19.24\,°\,\mathrm{s}^{-1}$ under closed-loop control	Geometry as an actuation degree of freedom, not a ground option.
Payload provision	3 kg, off-centre and possibly moving, *unsensed*	Primary disturbance specification; the controllers are payload-blind by construction (Sec. 3.5).
Control authority	Full attitude and position authority in every reachable configuration	Achieved thrust-to-weight 2.60 (25.26 N per rotor).

3.2 Concept and configuration space

The vehicle is a quadrotor whose four arms rotate independently about vertical hinges located at the corners of the central body. This arrangement allows the rotor positions, along with the footprint, inertia tensor, and control allocation, to change continuously during flight. Each arm carries its rotor at a fixed radius from the hinge; consequently, a specific shape is defined simply by the set of arm angles.

The body frame follows the standard PX4 FRD triad ($+X$ forward, $+Y$ right, $+Z$ down), with no axis swaps throughout the control stack. The absolute arm angle θ_i is measured in this frame such that $\theta = 0°$ corresponds to the $+X$ direction and $\theta = 90°$ to the $+Y$ direction, meaning a positive increment represents a clockwise rotation when viewed from above. The neutral X configuration places the arms at $\theta_{\mathrm{base}} = [45, 225, 315, 135]°$, with each diagonal pair pointing away from one another. The joint angle actually commanded to servo i is defined as the offset from this neutral position:

$$\delta_i = \mathrm{wrap}_{180}(\theta_{\mathrm{base},i} - \theta_i)\,, \tag{3.1}$$

such that $\delta = 0$ on all four joints corresponds to the X shape. Every kinematic and allocation quantity in Chapter 5 is expressed using this convention, which Fig. 3.1 illustrates graphically.

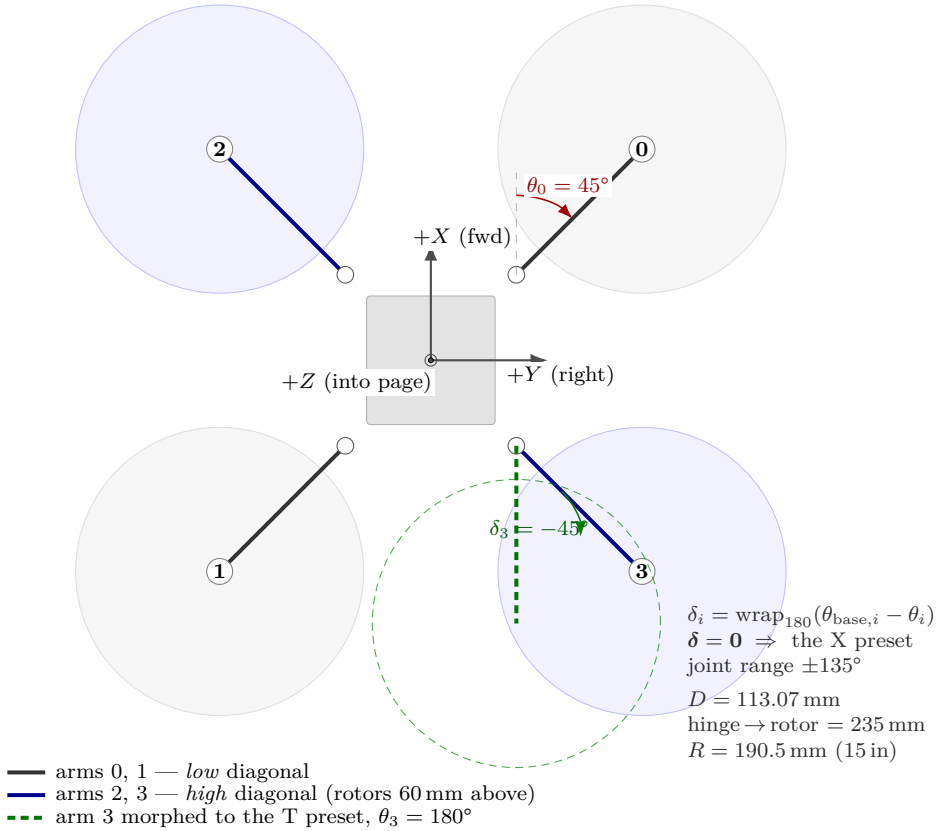


Figure 3.1: The arm-angle convention, drawn in the PX4 FRD body frame and viewed from above — so $+X$ points up the page and $+Y$ to the right, the same orientation as every trajectory plot in Chapter 12. The absolute angle θ_i is measured from $+X$ towards $+Y$; the commanded joint angle is the offset δ_i from the X neutral, Eq. (3.1), so $\boldsymbol{\delta} = \mathbf{0}$ is the X shape. Rotor discs are drawn to scale at the true 15 in diameter, which is what makes the 60 mm vertical stagger a requirement rather than a refinement: the arms of the high diagonal (2 and 3) carry their rotors above those of the low diagonal (0 and 1), so that discs which overlap in planform never share a plane. Arm 3 is also shown displaced to its T-preset angle.

Table 3.2 lists the configuration presets with their final angles. Every preset lies inside the $\pm 135°$ joint range that is enforced consistently at three independent points of the stack — the servo driver command clamp, the box constraint of the geometry optimiser and the joint limit of the simulation model — with at least $35°$ of margin, the worst case being the O at $-100°$. The O opening is quoted throughout by its magnitude, the commanded joint offset being negative; every O figure in this thesis refers to the $100°$ opening unless stated otherwise.

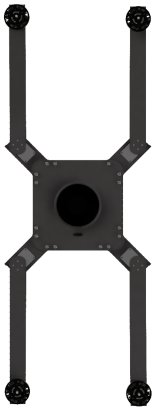
Each configuration redistributes footprint, inertia, and control authority differently; these properties are what make the catalogue useful, with the equations generating the lever arms and inertia tensor derived in Chapter 5. Two consequences



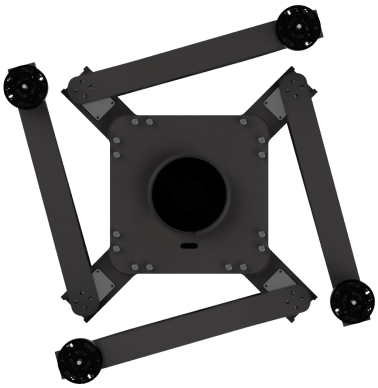
(a) X — $\boldsymbol{\delta} = \mathbf{0}$



(b) T — $\boldsymbol{\delta} = [-45, +45, +45, -45]^\circ$



(c) H — $\boldsymbol{\delta} = [+45, +45, -45, -45]^\circ$



(d) O — $\boldsymbol{\delta} = -100^\circ$ on all four

Figure 3.2: The four configuration presets of Table 3.2, rendered from the as-built CAD model and viewed from above, *all four at the same scale* so that the footprints can be compared directly. Propellers are omitted. The anisotropy that governs the flight campaign is visible without any measurement: X is symmetric; T aligns two rotors on the longitudinal axis and halves the roll lever; H is long in pitch and narrow in roll, the extreme case that makes it inadmissible as a take-off shape; and O closes the planform into a compact ring whose arms very nearly meet, which is what the 60 mm rotor stagger keeps apart.

Table 3.2: The configuration presets — absolute arm angles in the body FRD frame and joint offsets from the X neutral, Eq. (3.1) — with the mean control lever arms and the diagonal inertia tensor each of them produces, from the CAD-anchored mass model of the airframe. All presets respect the ±135° joint range with ≥35° margin.

Shape	θ [°]	δ [°]	Roll lever m	Pitch lever m	J_{xx}, J_{yy}, J_{zz} kg m^2
X	[45,225,315,135]	[0,0,0,0]	0.2792	0.2792	0.1217, 0.1198, 0.2030
T	[90,180,270,180]	$[-45, +45, +45, -45]$	0.2306	0.2306	0.1067, 0.0990, 0.1720
H	[0,180,0,180]	$[+45, +45, -45, -45]$	0.1131	0.3481	0.0354, 0.1752, 0.1773
O	[145,325,55,235]	-100 (all)	0.1636	0.1636	0.0611, 0.0611, 0.0999

highlighted by the table shape the entire flight campaign.

First, the H configuration is strongly anisotropic: its pitch inertia is five times greater than its roll inertia. Consequently, the equivalent pitch stiffness demanded by the attitude loop exceeds the moment ceiling admitted during take-off. As a result, H is excluded as a locked-shape take-off configuration, although it remains fully flyable when reached in flight.

Second, in the T configuration, two rotors align on the longitudinal axis, causing them to lose a significant portion of their roll lever arm. This reduces roll authority by half. Furthermore, because the two rotors that *retain* a roll lever spin in opposite directions, every roll command injects a yaw torque, and vice versa. The allocation imbalance this produces was measured in flight: the spread between motor commands in T is seven times the X value (Chapter 12). The roll–yaw coupling was not isolated in the campaign.

3.3 Airframe and mechanical design

3.3.1 Structural layout and materials

The airframe consists of a central body — two printed plates enclosing the battery, the flight controller, the companion computer and the power distribution — with the four arm hinges at its corners and the landing gear below. The structure is FDM 3D-printed in PET-CF, a carbon-fibre-filled PETG chosen over plain PETG for its higher stiffness and heat resistance: plain PETG softens under direct sun, which is disqualifying for an outdoor platform. A per-flight-day inspection of the PET-CF around the fastener holes for cracks and delamination is part of the pre-flight checklist.

The mass budget is derived from the CAD model dated 31 July 2026; these figures serve as the single source for the plant mass model used by both controllers and are adopted by the identification in Sec. 6.3. The total mass is 3.964 kg, comprising a central body with battery of 1.790 kg, four hinge connectors at 0.180 kg each, arm

links of 0.085 and 0.092 kg for the low and high diagonals respectively, and four rotor groups at 0.275 kg. The corresponding hover thrust is 38.89 N; an itemised breakdown is provided in Appendix B.

The centre of gravity is located 101.6 mm above the bottom plate and is effectively centred in planform ($x = -0.30$ mm, $y = 0.83$ mm), despite the vertical rotor stagger, because the raised arm pair forms a diagonal. An operational consequence of this layout is that the landing gear and its support polygon are sized specifically for the X configuration; consequently, flight procedures require reshaping the vehicle to X before every landing.

3.3.2 Rotating-arm mechanism

The arm is the core mechanical part of the vehicle. It was generatively designed in Fusion 360 to minimise mass under the rotor-thrust load, then reworked into a print-friendly part for FDM. The hinge-to-rotor length is 235 mm, and the hinge offset from the body centre is $D = 113.07$ mm. The two arms of the high diagonal additionally carry a riser ramp that sets their rotors 60 mm above those of the low diagonal, which is why they are oblique rather than straight and 7 g heavier. The stagger is paired *by diagonal* — arms 0 and 1 low, arms 2 and 3 high — so that the rotor discs that overlap in planform in the T, H and O configurations always lie on two different levels (Fig. 5.2). Each arm rotates about a vertical revolute joint supported by two rolling bearings on the hinge axis, which react the bending moment of the rotor thrust, and the rotation range is limited to $\pm 135°$ about the X-neutral position. That limit is enforced in software, consistently at the three points listed above, rather than by a mechanical stop: the servos run in extended (multi-turn) position mode, so a command outside the range would otherwise simply be executed, and the motor cables follow the arm without a slip ring, which is what makes a bounded range a requirement rather than a convenience. Figure 3.3 sections the joint; Fig. 3.4 shows the two stages the arm passed through.

3.3.3 Structural verification and its limits

The arm and the other critical printed parts were checked with static finite-element analysis in Fusion 360 at design time, and in service the printed structure held: across the campaign no printed part failed under flight loads, the structural damage suffered coming from the two crashes rather than from the material reaching a limit. The usual caveats of FEA on FDM parts apply — the analysis assumes an isotropic material, while the printed part is anisotropic across layer boundaries — but the more important point is what a static analysis cannot claim. The FEA is a strength check, not a stiffness check of the closed-loop system, and in flight the arm–servo–structure assembly behaves as a compliant mode near 6 Hz that the

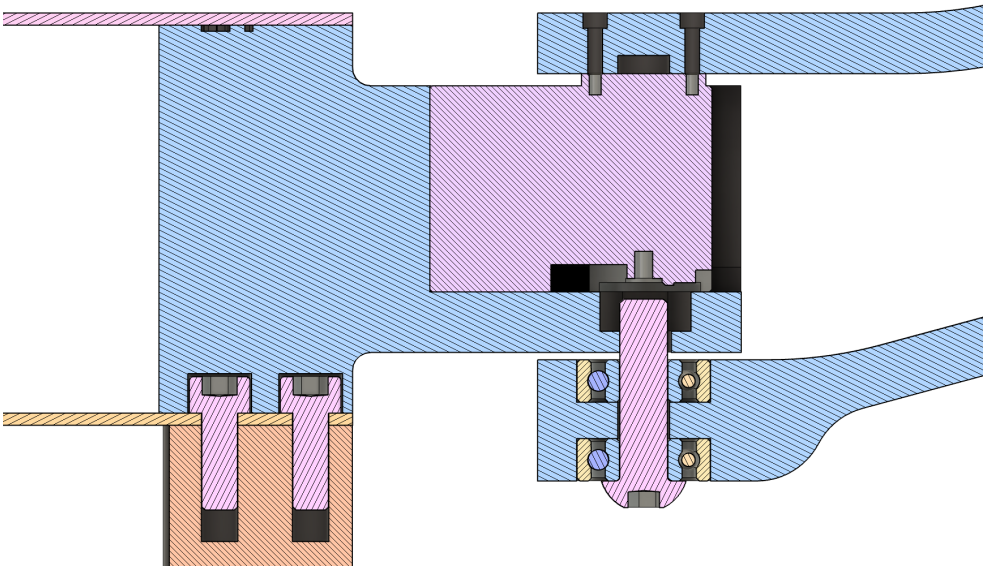
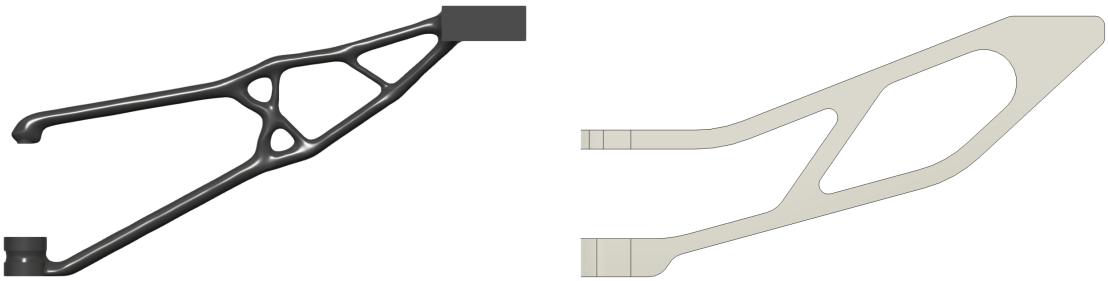


Figure 3.3: Section through the hinge axis of one arm, colour-coded by component. The two rolling bearings straddling the joint (right) are what react the bending moment of the rotor thrust into the body, leaving the servo to carry torque about the hinge axis alone; the servo body and its output shaft sit above them, coupled directly to the arm root with no external transmission. The motor cable follows the arm through the joint — there is no slip ring — which is why the ±135° range is a requirement enforced in software at three independent points of the stack rather than a mechanical stop.



(a) generative-design outcome (b) the part that was printed and flown

Figure 3.4: The arm as the generative solver produced it (a), minimising mass under the rotor-thrust load case, and as it was reworked for fused-deposition printing (b). The load paths survive the rework — the same upper and lower booms, the same diagonal web and the same relieved centre — but the organic variable-section geometry is replaced by constant-thickness prismatic sections that print without support and lay their layers across the load rather than along it. The rework is the reason the structural check of Sec. 3.3.3 is run on (b) and not on (a).

rigid-arm assumption of every controller in this thesis does not contain: it first appeared as a roll peak during morphing, was shown by a dedicated experiment to be structural, and was finally quantified on the flight in which it cost the vehicle (Sec. 6.8). No static structural analysis, however careful, would have surfaced this limitation of the design — Fig. 3.5 shows what it *did* establish, and what it did not.

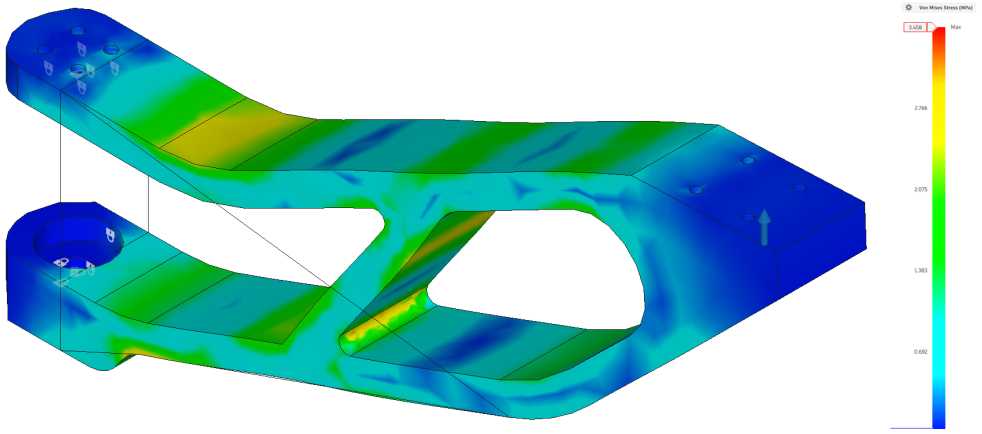
3.3.4 Aerodynamics across configurations

No aerodynamic coefficient enters the model of Chapter 5: the aerodynamic cost of the morphing configurations was quantified in flight, on the as-built vehicle. Hovering in T costs 911 W to 928 W against 743 W to 765 W in X — a 23 % penalty — while H costs 765 W to 784 W, only 3 % above X. The O configuration pays an additional overlap-specific penalty: the partial disc-disc overlap degrades rotor efficiency, raising the required collective and loading the arm servos asymmetrically — about 0.4 N m of extra aerodynamic torque on the most loaded arm and about 0.16 N m on the next — a load path that matters for Sec. 3.4. The energetic decomposition of the X-versus-O comparison, measured at the battery, is part of the flight identification of Sec. 6.9, and the per-configuration energetics are what turn configuration selection into the energy problem addressed by Chapter 8.

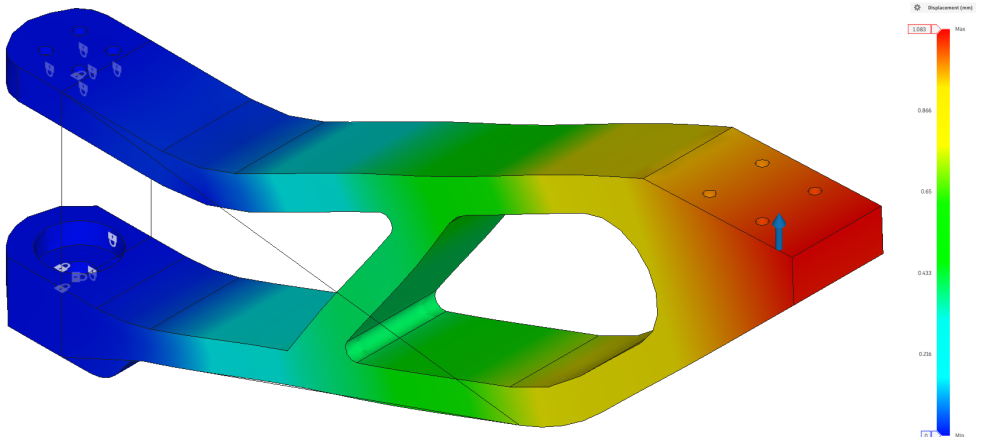
3.4 Actuation of the morphing degrees of freedom

Each arm is driven directly by a Dynamixel XC430-T150BB-T smart servo (model number 1070, the variant with a ball-bearing-supported horn), on a half-duplex TTL serial bus at 1 Mbaud through a U2D2 interface. The servos are powered from a dedicated 12 V rail (11.9 V measured under load), at which the stall torque is 1.6 N m; the reduction is the servo's internal gearbox, with no external transmission. The arm angle is read from the servo's own absolute encoder at 0.088° per tick; there is no external angle sensor. The servos operate in extended (multi-turn) position mode so that no joint works across the encoder wrap; their position gains ($P = 640$, $I = 0$, $D = 6000$ in native units — proportional and derivative only, the integral term being zero for reasons established in flight, Sec. 6.7) are rewritten at every initialisation because they live in volatile memory, and the zero of each servo was calibrated to a repeatability better than half an encoder tick. The commanded profile velocity is $19.24\,^\circ\,\mathrm{s}^{-1}$, which gives 2.3 s for a 45° transition (X↔T, X↔H), 4.7 s for 90° and 5.2 s for the 100° X↔O.

One property of this actuator dominates everything that follows: the XC430 has *no current sensor*. Its load register is an estimated PWM duty with 0.1 % resolution, and that estimate together with the angular droop under torque is the only load observable available. The servo's factory overload protection is correspondingly



(a) von Mises stress



(b) displacement

Figure 3.5: Static finite-element analysis of the printed arm, loaded at the motor mount by the rotor thrust (arrow) with the hinge bore constrained. The peak von Mises stress is 3.46 MPa at the root of the diagonal web, more than an order of magnitude below the strength of the printed material — consistent with the campaign, in which the structural damage suffered came from the two crashes rather than from the material reaching a limit. Panel (b) is the reason this is nonetheless *not* a stiffness verification of the closed loop: the arm tip deflects 1.08 mm under a static thrust, and that compliance, in series with the servo and the body, is the 6 Hz mode no controller in this thesis models (Sec. 6.8). The usual caveat on FDM parts applies: the analysis assumes an isotropic material, while the printed part is anisotropic across layer boundaries.

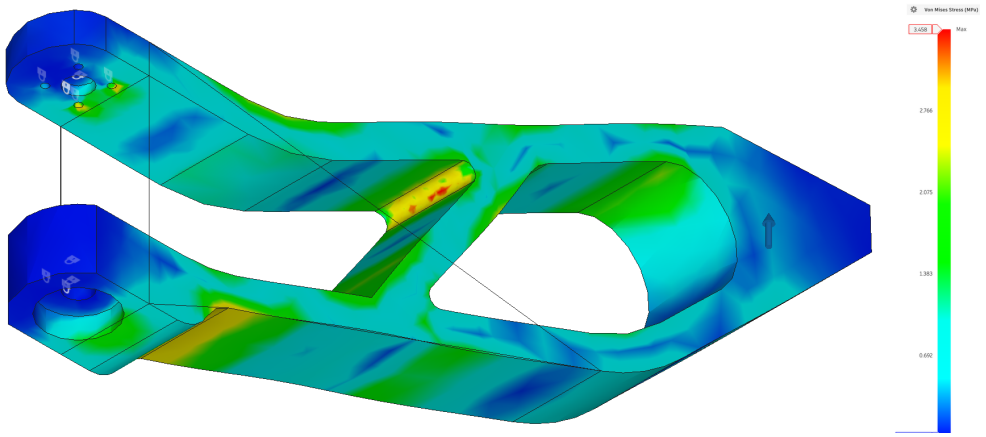


Figure 3.6: The same load case seen from the loaded side, which places the peak in view: the stress concentrates in a narrow band at the root of the diagonal web, where the two booms and the web meet, and falls away over the rest of the part. This is the feature the generative solver put there (Fig. 3.4) and the one the print-friendly rework had to preserve.

blunt: it latches on sustained load at roughly 45 % of stall torque,its threshold is fixed and not adjustable, and a latched servo turns torque off until rebooted.

Sizing, closed in flight. The torque sizing of this servo was closed in flight rather than on paper, and the outcome is partially negative. In X the four servos hold comfortably in the lower part of their duty range, and a morphing transition costs an additional 20 to 25 duty points — both regimes well clear of the protection threshold. In O, however, the most loaded arm settles essentially *at* the latch regime, with several degrees of angular droop: exactly the condition under which the factory protection released an arm in flight and cost the airframe (Table 6.3). The consequence for the design is stated plainly: with the XC430 class, O is a *transit-only* configuration — reachable, crossable, but never held under power — T is the reference morphing configuration for sustained flight, and a servo with true current control and roughly twice the stall torque (the XM430 class, 3.8 N m) is the identified upgrade path.

Protection of the arm subsystem. Because the actuator exposes no current measurement and its overload shutdown is fixed, non-adjustable and clearable only by a reboot, the actuation subsystem carries its own envelope monitor. Sustained load is tripped in two regimes: 40 % duty for 3 s while an arm is holding position, 85 % for 2 s while it is in transit, the distinction being necessary because a legitimate transition costs those 20 to 25 duty points and case temperature is tripped at 65 °C, below the 80 °C factory limit. In parallel, a 100 Hz per-servo watchdog compares the measured position against the position *expected from the motion profile* and detects

an uncommanded arm within 0.13 s to 0.16 s, against the 0.73 s that separated the release from the propeller strike in the one event of the campaign; on a factory latch the servo is automatically rebooted and re-initialised in 0.15 s end to end.

The monitor is wired into the mission logic as a design invariant rather than an advisory: take-off is refused while any arm fault is latched; a sustained-load trip in flight freezes further morphing, disables the geometry optimiser and commands a normal landing with the vehicle reshaped to X while any harder fault commands an immediate landing in the current shape. The operational detail of these reactions belongs to Chapter 11.

3.5 Payload provision and the 3 kg disturbance requirement

The 3 kg payload is a design driver and a disturbance specification, not a fitted component: the real airframe is designed to carry it but was never flown end-to-end. A two-axis prismatic slider with a 3 kg sphere was implemented in the Gazebo model and used during the controller development to emulate a sliding centre of gravity and test the robustness of the controllers

Both controllers are payload-blind by construction: no payload mass, position or motion is sensed or estimated, and an off-centre 3 kg mass must be absorbed as an unmodelled disturbance. The magnitude is significant: 3 kg at a $(0.2, 0.2)$ m diagonal offset applies about 5.9 N m of unmodelled moment on each horizontal axis, and the disturbance-rejection authority of both control designs is sized for that case. The constants dimensioned for this requirement are therefore not to be narrowed on the grounds that the flown vehicle carries no payload.

3.6 Avionics, power and system integration

The avionics follow the standard PX4 companion-computer architecture, with two additions specific to this vehicle: the servo bus of Sec. 3.4 and its dedicated power rail. Table 3.3 lists the components.

Power is split by function: the 6S battery feeds the four ESCs directly, while the dedicated 12 V converter feeds the companion computer and the servo bus, so that a propulsion-driven voltage sag cannot interfere with the arm actuation. The safety chain is deliberately simple. Its last resort is a hardware RC kill switch on a dedicated transmitter channel, verified on the ground at the start of every flight day. Above it sit the software layers described elsewhere: the arm-fault gates and reactions of Sec. 3.4, and the controller-level failsafes of Chapters 7 and 11. Manual RC override disengages the autonomous stack at all times.

Table 3.3: Avionics, propulsion and actuation components of the as-built vehicle.

Subsystem	Component	Notes
Flight controller	Pixhawk 6X	PX4 flight firmware, v1.17 line (Chapter 11)
Companion computer	LattePanda N100	runs the full control stack; uXRCE-DDS bridge to the autopilot
Motors (×4)	BrotherHobby Avenger 4110, 560 KV	12N14P outrunner; rotor group 0.275 kg
ESC (×4)	HGLRC 90 A	DShot600; the bidirectional speed-telemetry variant was disabled after proving unreliable on the flight vehicle (Sec. 6.10)
Propellers (×4)	15×5.5, two-blade, rigid	$R = 0.1905$ m
Arm servos (×4)	Dynamixel XC430-T150BB-T	TTL bus at 1 Mbaud via U2D2 (Sec. 3.4)
Battery	6S lithium-polymer	about 23.5 V under hover load
Servo / companion power	dedicated 6S→12 V converter	isolates the servo rail from propulsion sags
Altimeter	Garmin LIDAR-Lite v3	I2C (address 0x62); mounted 65 mm left of the body axis, 50 mm below the belly

Chapter 4

Aerodynamic and propulsion characterization

This chapter characterises the propulsion system of the morphing quadrotor as it was established *before the vehicle flew*: a six-axis bench campaign and the parameter identification built on it. Part of what the bench measured survived contact with the vehicle unchanged and is used throughout the thesis, specifically the rotor-speed envelope, the throttle-to-speed map and the small-signal actuator lag were all deemed reliable estimations. Its two headline results, the thrust and torque coefficients, did not: both were overturned by the in-flight identification of Chapter 6. They are reported here because the way a carefully executed bench campaign failed to predict the flying vehicle is a result in its own right.

4.1 Propulsion system overview

The propulsion unit pairs a BrotherHobby Avenger 4110 560 KV brushless motor with an HGLRC 90 A electronic speed controller (ESC) on a six-cell (6S) lithium-polymer battery (25.2 V at full charge). The propeller changed during the campaign: the bench characterisation was run with a folding 15.5×5.8 three-blade propeller (radius 0.1968 m), which the vehicle flew for half of the testing campaign, a 15×5.5 two-blade non-folding propeller (radius 0.1905 m was later mounted on the platform without bench testing it, the unreliablity of the coefficients estimation was the reason it was mounted on the drone directly. The thrust model was re-validated in flight after the change and required no constant update (Sec. 6.4).

The ESC is commanded over DShot. For the bench campaign it was run in bidirectional DShot, which returns the rotor speed on the signal wire itself [25]; that telemetry is the backbone of the identification below.

The 6S operating voltage was chosen so that the physical thrust law would

land close to the plant simulated throughout the development (Chapter 10). That expectation did not survive contact with the vehicle: the coefficient the aircraft actually delivers is a factor 1.78 below the bench value the simulation carried, this discrepancy was attenuated by the robustness of the SMC during the first flights and later fixed in the autopilot code.

The measured envelope frames the rest of the thesis; the figures below are computed with the flight-identified thrust coefficient, not with the bench value this chapter first produced. With the 3.964 kg rigid mass of the airframe, hover demands 9.72 N per rotor, reached at about 484 rad s^{-1}, roughly 38 % of the available per-rotor thrust and 62 % of the maximum rotor speed. The usable speed range is 120 rad s^{-1} to 779.8 rad s^{-1}, the ceiling being the full-throttle speed measured on a fresh battery under load; at that ceiling each rotor produces about 25.3 N, for a total of 101.1 N and a thrust-to-weight ratio of 2.60 on the bare platform, falling to 1.48 with the 3 kg design payload (Sec. 3.5). The aerodynamic cost of the morphing configurations themselves was not predicted from first principles but measured on the vehicle: hovering in the compact O configuration costs 25.1 % more battery power than hovering in the spread X (Sec. 6.9).

4.2 The bench campaign

Manufacturer thrust tables are sparsest exactly where this drone spends most of its life: hover demands about 38 % of the available thrust per rotor and, because thrust grows with the square of rotor speed, about 62 % of the maximum speed, so the throttle band around hover, roughly 0.39 to 0.42 normalised, falls squarely in the under-sampled region. Bench testing therefore measures the propulsion unit directly, its intended deliverables being the static thrust and reaction-torque curves from which k_T and k_Q are identified, the map from normalised ESC throttle to steady rotor speed, and the first-order actuator time constant τ_{act}, the single parameter to which the predictive controller of Chapter 9 is most sensitive.

4.2.1 Test rig and instrumentation

The experimental setup centres on an ATI Mini40 six-axis force/torque transducer (serial FT17838, US-20-40 calibration, ±10 V strain-gauge electronics). The motor is mounted on the sensor's tool side via a rigid 3D-printed adapter plate (Fig. 4.1), ensuring the propeller axis nominally coincides with the sensor z axis. This alignment allows the axial force channel F_z to measure thrust and the axial moment channel M_z to capture reaction torque. The primary advantage of employing a six-axis cell is that both rotor coefficients can be determined from a single run, whereas a conventional single-axis thrust stand typically requires estimating k_Q indirectly from electrical power. However, it must be noted that this campaign did

not yield the expected results: the torque channels proved unusable, the thrust scale was subsequently revised following flight data, and k_Q was ultimately derived from electrical power. This outcome represented a departure from the initial intent of utilising the six-axis rig to bypass indirect estimation methods.



Figure 4.1: The propulsion test rig: flight motor with the folding three-blade propeller, bolted through a 3D-printed adapter plate to the ATI Mini40 six-axis transducer on a rigid support column.

The transducer retains a NIST-traceable factory calibration dating back to November 2015. Given that this certificate significantly predates the current work, its bounds are interpreted as indicative of the instrument's general class rather than as guaranteed current performance. The calibrated ranges cover F_z up to $\pm 267\,\mathrm{N}$, lateral forces up to $\pm 89\,\mathrm{N}$, and all three torques up to $\pm 4.5\,\mathrm{N\,m}$. These limits accommodate full-throttle rotor loads with substantial margin, which also presents a challenge: the certified uncertainty at 95 % confidence is 1.25 % of full scale for F_z ($\approx 3.3\,\mathrm{N}$) and 1.50 % for M_z. This uncertainty level is appreciable relative to a per-rotor hover thrust of roughly 9.7 N. Consequently, the identification process does not rely on single-point readings; instead, the speed map is constructed from settled-window medians, and the coefficients are derived from regressions pooling aligned samples across the entire sweep.

The six strain-gauge voltages are sampled by a 12-bit NI USB-6008 unit on a dedicated acquisition computer via the manufacturer's software, which exports calibrated load components at a uniform rate of 62.5 Hz. However, this process provides no absolute time base and applies no tare, both of which must be reconstructed during analysis.

The ESC is driven directly by the flight controller over DShot on one of its direct outputs. This is deliberately the same digital path used in flight, ensuring that the identified throttle-to-speed map transfers without any change in the ESC interface. Throttle sequences are issued through the autopilot's actuator-test facility while the vehicle remains *disarmed*. These sequences are scripted by a bench node on the companion computer using the same middleware bridge employed by the control stack [26], with rotor speed data returned via bidirectional DShot telemetry [25] over the same link. The bench node enforces a conservative safety envelope, restricting operation to disarmed modes only, applying hard clamps to all commands, implementing a dead-man timeout that stops the motor if the node fails, and explicitly releasing actuator-test mode upon every exit path.

The speed log and force log operate on independent clocks. While the throttle-to-speed map and time-constant fits are computed exclusively from the speed log and remain unaffected by clock differences, the coefficient fits align the two records by cross-correlating the measured thrust history against a template $k_T\,\omega^2(t)$ derived from the speed log. A campaign on a single propulsion unit follows a fixed sequence: an initial check of spin direction and thrust sign at minimal throttle; static staircase sweeps ranging from just above the ESC's minimum sustained speed to full throttle, sampled densely in the hover band and repeated in independent captures; throttle steps at various operating points and amplitudes to fit τ_{act}; and a coarse vibration survey based on the per-level scatter of the force channels. The unit is powered by a 6S flight battery rather than a regulated supply, ensuring that the identified map incorporates the voltage sag experienced during flight.

4.2.2 What the campaign measured, and what it exposed

The flight-unit campaign, conducted in July 2026, comprised two independent full staircase sweeps with the load cell in the loop (2.5 s throttle plateaus, nineteen fit-usable levels pooled across the two captures). One sweep utilised a partially depleted battery, while the other employed a freshly charged unit, the latter also defining the upper limit of the speed envelope. These tests were complemented by a speed-only full sweep and a series of throttle steps at several operating points.

A dedicated follow-up session on the same day focused on measuring the actuator time constant over 156 step transitions logged at 100 Hz. The order of these steps was randomised within each run to mitigate the effects of battery depletion, which is monotonic in time; this randomisation effectively converts the gradual voltage drop into noise rather than a spurious amplitude trend. A fixed reference step bracketed every run to serve as a drift detector.

One planned measurement could not be performed: due to the lack of telemetry wiring for voltage and current on the ESC, and because pack-side current was not captured, no electrical power measurement exists for these ground runs. This

gap was subsequently closed using aerial data: the pack power logged during a later hover provides the only directly measured bound for the torque coefficient (Sec. 6.5).

The tare was *measured*, not estimated: every capture records idle stretches before and after the sweep, which agree on a baseline of about −0.6 N. The point deserves a methodological remark, because an earlier issue of this analysis did estimate the tare as a free offset in the thrust regression — and obtained a confident, self-consistent and wrong value of the opposite sign, because a free offset happily absorbs a residual synchronisation error. Quantities the data records directly should be read from the data, not fitted. Figure 4.2 shows both operations on the two captures.

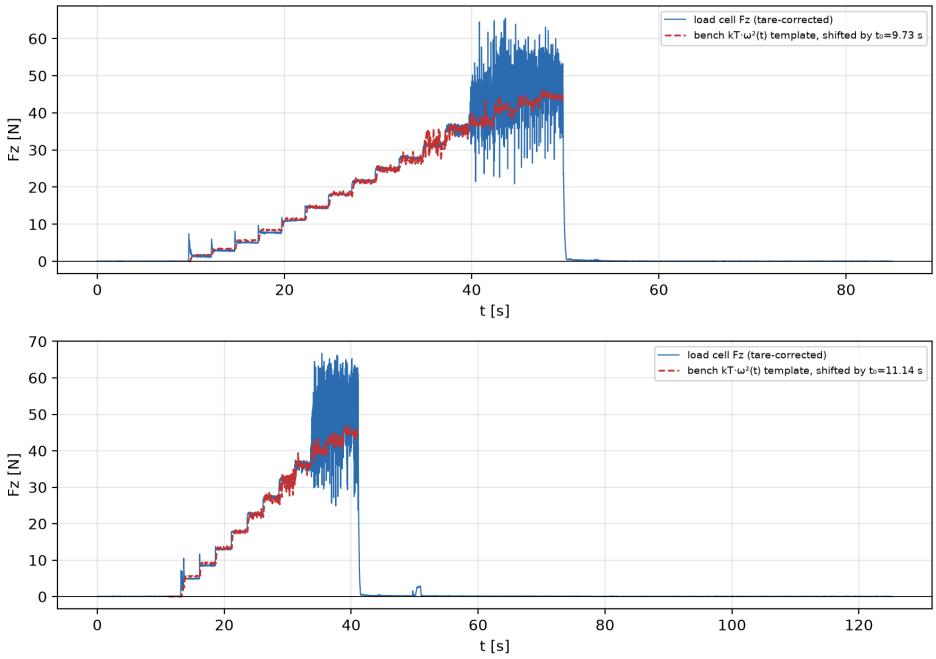


Figure 4.2: Time-base alignment and tare on the two staircase captures. The force and speed logs run on independent clocks, so the tare-corrected F_z of the cell (solid) is aligned against a template $k_T\,\omega^2(t)$ built from the speed log (dashed) by cross-correlation; the recovered offsets are 9.73 s and 11.14 s. The tare itself is read from the idle stretches that bracket every sweep, not fitted. The plateau staircase is clean up to about 710 rad s^{-1}, above which the scatter of the force channels explodes — the rig resonance that removes the top seven levels from every fit.

Two defects of the rig were exposed by the six-axis data itself. The first is a fixed tilt of the motor axis with respect to the load-cell z axis: across a sixteen-fold thrust range both captures show a *constant* lateral-to-axial force ratio, dominated by $F_y/F_z = -0.60(2)$, i.e. a resultant $|\boldsymbol{F}|/F_z = 1.17$. A constant ratio over that

range cannot be noise; only fixed geometry produces it. The implied misalignment averages 31.6°, a condition confirmed visually on the rig. The remedy is exact rather than approximate: thrust is defined as the vector magnitude $|\boldsymbol{F}|$, which remains invariant regardless of the motor's clocking within the fixture. Consequently, the thrust scale retains its validity as a genuine measurement without requiring any assumed correction factor. The primary cost of this tilt is increased noise, which likely contributes to the second defect identified in the data, since at full throttle the fixture sustains a lateral load exceeding 20 N: above roughly $710\,\mathrm{rad\,s^{-1}}$ (throttle ≥ 0.81) the per-plateau scatter of the force channels jumps from about 1 % to 24 % to 28 % of the mean, sharply and reproducibly in both captures. The seven affected plateaus are excluded from all coefficient fits, with the consequence that above about $700\,\mathrm{rad\,s^{-1}}$ the thrust and torque laws are physically-motivated ω^2 extrapolations, not measurements. Nineteen flights later no stand-like high-throttle resonance has ever appeared on the airframe; the structural mode that *did* matter in flight is of a different nature — an arm/servo compliance mode that this rig, mounting no arms, could not have seen (Sec. 6.8).

The torque channels, unlike the tilt, have no remedy. Two independent extractions were attempted: the axial moment channel alone, and the geometry-free projection $Q = \boldsymbol{T} \cdot \boldsymbol{n}$ of the full measured moment onto the true (tilted) propeller axis — exact for *any* mount offset $\boldsymbol{r}$, since $(\boldsymbol{r} \times \boldsymbol{F}) \cdot \boldsymbol{n} = 0$ when $\boldsymbol{F}$ is parallel to $\boldsymbol{n}$. Both land in a physically impossible region, implying rotor figures of merit of 1.27 and 1.52: a propeller producing its thrust for *less* shaft power than an ideal actuator disc, forbidden by momentum theory [27]. The contradiction is internal to the cell's own data — the force channels place the propeller axis 31.6° off the sensor axis, the moment channels place it on-axis ($T_y/T_z = -0.011$) — so no reprocessing recovers the torque. The *shape* of the torque data is excellent (the ω^2 law holds with $R^2 = 0.999$); it is the *scale* that is not credible. Figure 4.3 makes the contradiction graphic.

The mount was never squared: the static thrust bench was formally abandoned on 2026-08-02 in favour of in-flight identification, so these two captures remain the only load-cell data of the project.

4.3 Parameter identification

Thrust coefficient. Thrust against ω^2 is a clean line through the origin (Fig. 4.6): pooling the nineteen usable plateaus of the two captures,

$$k_T^{\mathrm{bench}} = 7.40 \times 10^{-5}\,\mathrm{N\,s^2} \qquad (R^2 = 0.9987), \tag{4.1}$$

with the two captures agreeing to 1.2 % and landing at 103.1 % of the manufacturer's independent bench line for the same motor–propeller pairing. Two by-products of the fit remain valid: pointwise, $|\boldsymbol{F}|/\omega^2$ rises across the speed range before flattening,

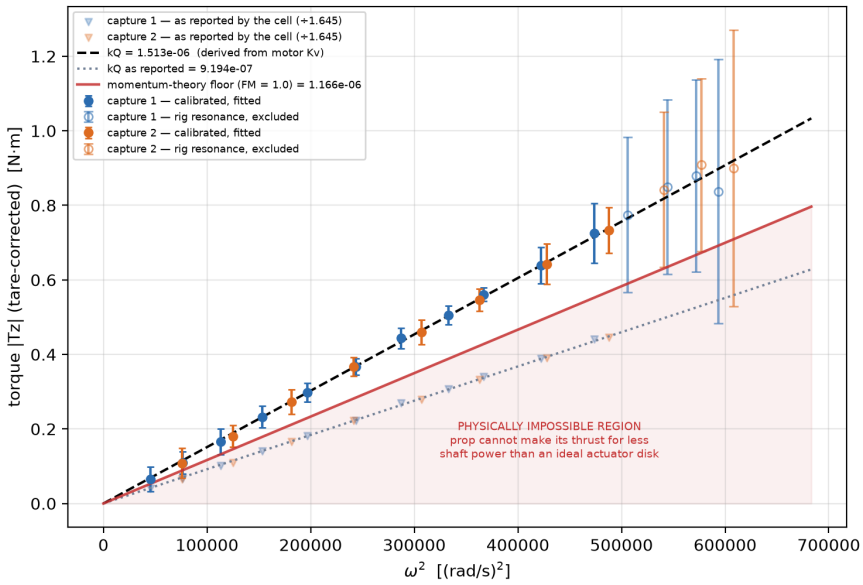


Figure 4.3: Measured reaction torque against ω^2. The ω^2 law itself is excellent ($R^2 = 0.999$), but the scale is not admissible: the shaded band below the momentum-theory floor (figure of merit = 1, computed against the corrected thrust scale) is forbidden — a propeller there would produce its thrust for less shaft power than an ideal actuator disc. The torque as reported by the cell falls inside it. The dashed line is the value derived from the motor speed constant, itself later retracted (Sec. 6.5); what the stack carries is an interval, not a measurement. Error bars are $\pm 1\sigma$ within each plateau; hollow markers are the excluded resonance levels.

a Reynolds-number effect that the manufacturer's own table reproduces, that the reference propeller datasets of Brandt and Selig [28] document across this size and speed class, and that no pure scale error could create or remove; and restricting the fit to the hover band ($350\,\mathrm{rad\,s^{-1}}$ to $500\,\mathrm{rad\,s^{-1}}$) moves the coefficient by only $-1.3\,\%$, so a single constant coefficient is the right model form. Both by-products are visible in Fig. 4.4.

The bench value did not survive validation. Equation (4.1) was overturned in flight by a factor 1.78. The constant adopted for the model throughout this thesis is $k_T = 4.1547 \times 10^{-5}\,\mathrm{N\,s^2}$, identified from the hover-thrust balance of a piloted flight (Sec. 6.4). Using the bench value, the take-off thrust ceiling mapped to only 0.68 times the vehicle weight, preventing lift-off in two flight sessions before the discrepancy was traced in the air. The 103.1 % agreement with the manufacturer data must therefore be reinterpreted: it is not a validation but rather the joint signature of bench and datasheet both deviating from the flying vehicle by the same factor.

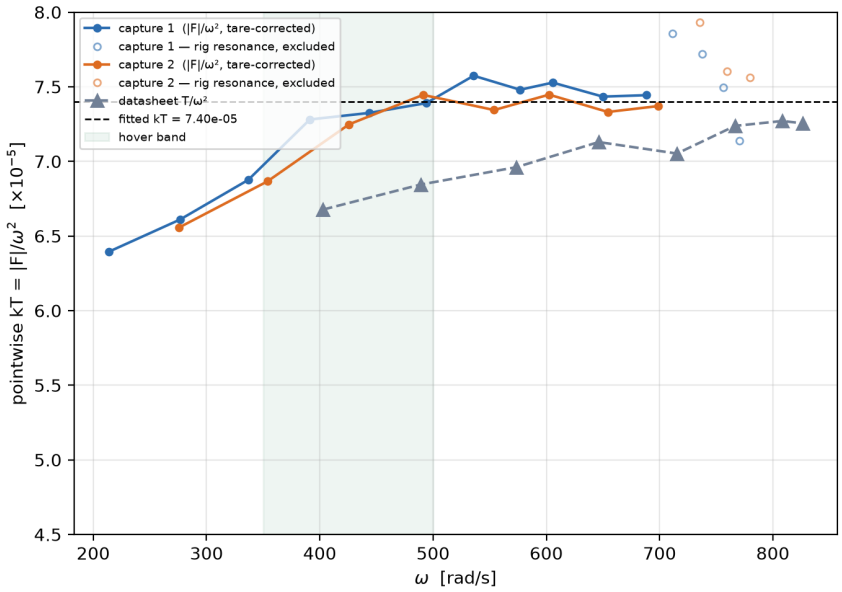


Figure 4.4: Pointwise $|\boldsymbol{F}|/\omega^2$ against rotor speed for the two captures, against the pooled fit and the manufacturer's table. The coefficient is not flat: it rises through the low- and mid-speed range and flattens beyond the hover band (shaded), the Reynolds-number trend that the manufacturer's own points reproduce — so it is a property of the propeller, not of the rig, and no pure scale error could create or remove it. The hollow markers above $710\,\mathrm{rad\,s^{-1}}$ are the resonance-affected plateaus, excluded from every fit.

Reaction-torque coefficient. With the load cell's torque scale rejected, k_Q was initially *derived* from external evidence: applying the motor's torque constant implied by its speed constant ($K_t = 9.5493/K_v$ N m A^{-1}) to the manufacturer's full-throttle current draw yielded $k_Q = 1.51 \times 10^{-6}\,\mathrm{N\,m\,s^2}$ ($k_Q/k_T = 0.0204\,\mathrm{m}$). This derivation was subsequently retracted alongside the thrust coefficient for three independent reasons: it requires a motor-plus-ESC efficiency of 112 % against the corrected thrust scale; it implies a rotor figure of merit of 0.375, which is implausibly low; and its admissibility bracket was computed using the superseded k_T. The model instead carries an interval rather than a single measurement: $k_Q/k_T = 0.024\,\mathrm{m}$, with a band of 0.021 m to 0.027 m, one edge of which was measured from the pack power of a later hover (Sec. 6.5).

The hover-point reaction torque implied by that interval, $k_Q\,\omega_{\mathrm{hov}}^2 \approx 0.23\,\mathrm{N\,m}$, deserves a comment of its own, because an earlier reading of it as comfortably inside the servo envelope was disproved by a crash. This is the torque that back-drives each arm servo through its rotor, about 15 % of the 1.6 N m stall torque of the XC430 arm actuators in static hover; in the compact configurations it grows until it breaks things. In the eleventh flight the most loaded arm sustained about 0.71 N m

— roughly 44 % of stall — long enough to trip the servo firmware's continuous-load protection, and the released arm swept the design overlap of the compact configuration and struck a propeller. The sizing of the arm actuation is therefore governed by k_Q, and closing k_Q is on the critical path of the arm design as much as of the yaw loop. One closure route proposed by the original bench analysis must be explicitly withdrawn: the spin-down study below pins the ratio J/k_Q, which suggested that an independent measurement of the rotor-plus-propeller inertia would determine k_Q outright. It does not, because the fitted $J = 1.108 \times 10^{-4}\,\mathrm{kg\,m^2}$ is not an independent quantity but a definition — the pipeline hard-codes $J/k_Q = 73.25$ from the retracted bench k_Q — and the geometric inertia of the propeller alone is three to five times larger. What would close the number is a bifilar-pendulum measurement of the rotor-plus-propeller inertia followed by a drive-free coast-down: about forty minutes of bench work that has not been carried out.

Speed envelope and throttle-to-speed map. The speed telemetry alone — untouched by any load-cell issue — fixes the envelope. On a freshly charged pack the full-throttle speed is $\omega_{\max} = 779.8\,\mathrm{rad\,s^{-1}}$ (7,446 RPM); the same command on a partially depleted pack reads $757.6\,\mathrm{rad\,s^{-1}}$, and the manufacturer's constant-voltage bench figure of $826.4\,\mathrm{rad\,s^{-1}}$ is not reachable at all on a real battery. Because thrust goes as ω^2, the vehicle delivers about 11 % less full-throttle thrust than the manufacturer's table promises — which does not contradict the 103.1 % agreement on k_T, a thrust-versus-speed law: the loss lives entirely in the throttle-to-speed map (Fig. 4.5). At the other end, the lowest sustained speed observed anywhere in the campaign is $117.5\,\mathrm{rad\,s^{-1}}$ and the model floor is set at $\omega_{\min} = 120\,\mathrm{rad\,s^{-1}}$. The full map (Fig. 4.6, right) is visibly nonlinear — compressive around hover, which is why large command steps near hover translate into modest speed changes — and it replaces the simulator's linear map on the physical vehicle (Sec. 5.4.3), interpolated over a 13-point table built from the fresh-pack sweep. The shape of this map turned out to matter more than its tabulated values: in ninety seconds of settled hover from a later flight the vehicle produced 10.4 % more thrust than the model predicts from the same commands, a residual that binning by collective attributes to the curvature of the map rather than to the thrust coefficient (Sec. 6.4). A battery explanation was tested and falsified — two flights at the same pack voltage show ratios six percent apart, and the ratio correlates with the collective ($R^2 = 0.42$) but not with the voltage ($R^2 = 0.03$).

Actuator time constant. The dedicated 100 Hz step study — 156 transitions spanning bases 0.10 to 0.70 and amplitudes up to 0.70 throttle — yields for the regime the controllers actually inhabit

$$\tau_{\mathrm{act}} = 0.079(17)\,\mathrm{s} \qquad (\Delta\omega < 200\,\mathrm{rad\,s^{-1}},\ n = 45), \tag{4.2}$$

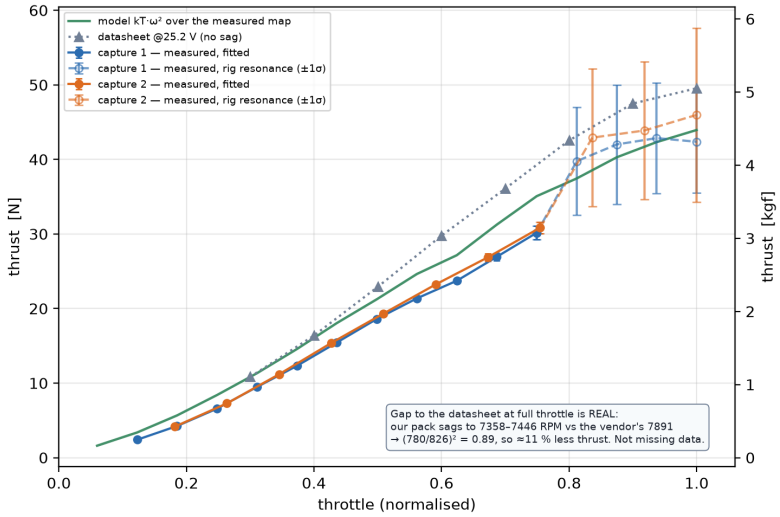


Figure 4.5: Per-motor thrust against normalised throttle, obtained by composing the identified thrust law with the measured throttle-to-speed map, against the manufacturer's constant-voltage table. The full-throttle shortfall is real and not a gap in the data: on a 6S flight pack the rotor reaches 7,358 rpm to 7,446 rpm against the vendor's 7,891, and $(780/826)^2 = 0.89$ accounts for the whole 11 %. The deficit therefore lives in the throttle-to-speed map, not in k_T.

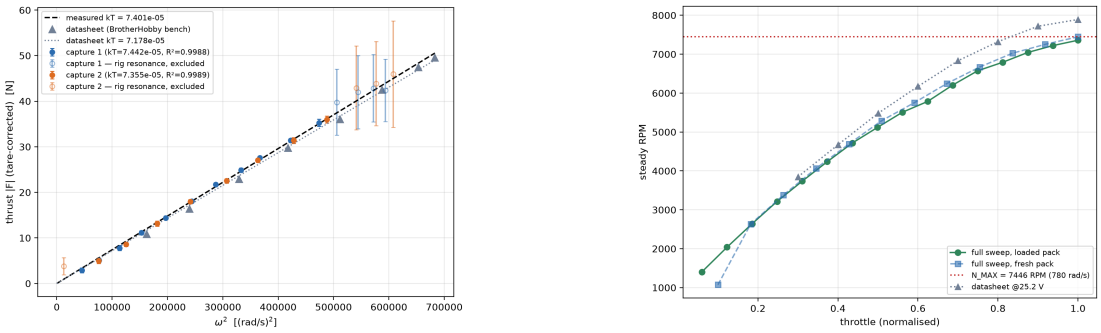


Figure 4.6: Left: measured thrust $|\boldsymbol{F}|$ versus ω^2 for the two captures and the pooled bench fit $k_T^{\text{bench}} = 7.40 \times 10^{-5}$, plotted with the manufacturer's bench points — an agreement the flight result later demotes from cross-check to the joint signature of two sources that both disagree with the vehicle (Sec. 6.4). Right: steady rotor speed versus normalised throttle for a loaded and a fresh pack: the ESC is not linear in speed, and the real ceiling sits below the manufacturer's constant-voltage figure by the pack's loaded sag. Plateaus above 0.81 throttle carry $\pm 1\sigma$ scatter bars and are excluded from the fits (rig resonance).

measured on the folding three-blade propeller of the bench era. Outside that regime the response is slower, and the study's design isolates *why*. At a fixed hover base, sweeping the command amplitude over a five-fold range moves the fitted τ by less than 0.03 s: the command delta itself is not the driver. Sorting all transitions by the *demanded speed change* instead reveals a monotonic growth, from 0.076 s below 100 rad s^{-1} to 0.153 s above 400 rad s^{-1} (Fig. 4.8, left), with near-perfect correlation from low bases ($r = +0.99$) and none from high bases, where the compressive throttle map keeps $\Delta\omega$ small no matter the command. A single rigid-body equation explains the whole pattern, $J\,\dot{\omega} = Q_{\text{em}} - k_Q\,\omega^2$: spin-*down* needs no motor torque and is dominated by aerodynamic drag — with k_Q fixed, one fitted inertia reproduces all 78 spin-downs to a median of 4.4 % of the step — while spin-*up* is limited by the finite motor torque, over-delivering on small steps and saturating on large ones (Fig. 4.7).

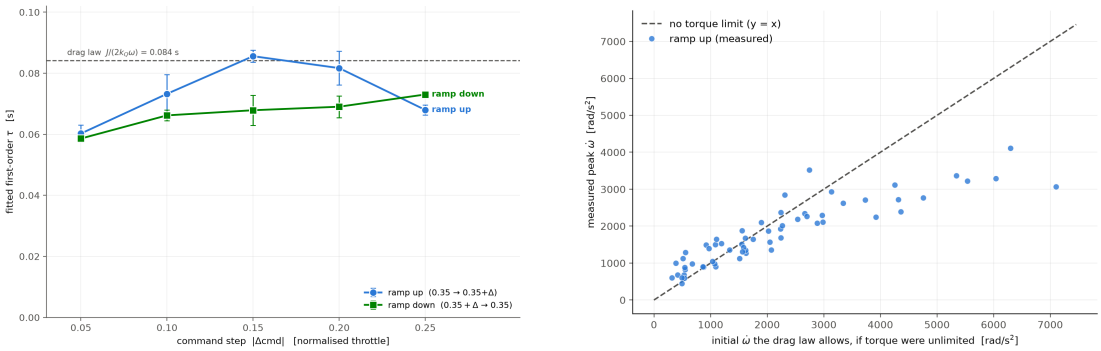


Figure 4.7: Left: at a fixed hover base, the fitted τ remains largely invariant as the command step is swept over a five-fold range. This indicates that the command delta does not dictate the time constant, validating the single constant in Eq. (4.2) for near-hover operations. Data points represent three repetitions in randomised order with error bars at $\pm 1\sigma$; the dashed line denotes the drag-law prediction $J/(2k_Q\omega)$. Right: measured peak $\dot{\omega}$ of every spin-up compared against the acceleration permitted by the drag law alone under unlimited torque. Small steps exceed the unit line, indicating that the motor exceeds its steady-state torque capability, while large steps fall below it due to finite motor torque saturation. This asymmetry explains why spin-up is slower than spin-down and causes τ to increase with the demanded speed change.

Three practical conclusions emerge from these results. First, at the hover operating point, the lag is first-order with $\tau_{\text{act}} \approx 0.08$ s. Second, the slowdown observed during large steps stems from torque and slew rate saturation, which is structurally distinct from a first-order lag; consequently, no speed-scheduled τ is fitted to this regime. Third, during take-off-scale transients the true lag reaches 0.15 s to 0.17 s, roughly twice the small-signal value; the subsequent life of this constant in the model is traced in Sec. 6.6.1. Three caveats travel with the fit: the fitted J is not a physical inertia, since it inherits the scale of the retracted

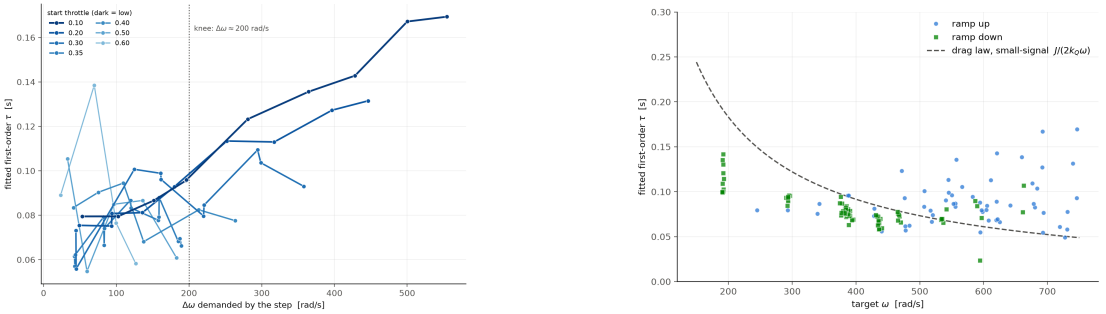


Figure 4.8: Left: fitted τ of spin-up transitions versus the demanded speed change, grouped by base throttle. From low bases (torque-limited regime), τ grows with $\Delta\omega$; from high bases, the compressive mapping keeps $\Delta\omega$, and consequently τ, small. Right: spin-down τ versus target speed, where aerodynamic drag alone decelerates the rotor. The linearised limit $\tau = J/(2k_Q\omega)$ (dashed line) is followed only loosely, whereas the full nonlinear model with a single fitted J tracks every transition to within a median of 4.4 % of the step.

k_Q; the drag-only reading of the spin-downs cannot exclude a non-aerodynamic ESC braking contribution; and the bench measures a restrained motor in static air, whereas in flight the inflow alters the aerodynamic term. A further limitation is specific to the propeller change: the amplitude dependence measured here belongs to the folding three-blade propeller and has never been re-measured on the 15×5.5 set, so the large-step regime — take-off, recovery — is unknown with the current hardware.

4.4 Summary: the bench results and their status

Table 4.1 consolidates what the bench campaign delivered and the status of each result after the in-flight identification. The constants actually adopted by the thesis are collected in Chapter 6 (Table 6.4); the symbol-to-implementation map is in Appendix B.

Three gaps of the campaign remain open on the vehicle that flies today. A single propulsion unit out of four was characterised, so motor-to-motor spread is unmeasured — the persistent hover yaw trim of about 0.096 N m is the candidate symptom — and the remaining units will not be bench-checked, the static thrust bench having been abandoned in favour of in-flight identification. Two vibration signatures observed in the fault-injection flights have never been compared against healthy propellers. And the 2015 calibration age of the transducer is no longer mitigated by the agreement with the manufacturer's bench, that agreement being now the joint signature of two sources which both disagree with the vehicle.

The constants derived in this chapter were propagated synchronously to all

Table 4.1: The propulsion parameters this chapter produced, and what became of each after the in-flight identification of Chapter 6.

Parameter	Bench result	Status
Thrust coefficient k_T	$7.40 \times 10^{-5}\,\mathrm{N\,s^2}$ ($R^2 = 0.9987$)	**Retracted.** Overturned in flight by a factor 1.78; the adopted $4.1547 \times 10^{-5}\,\mathrm{N\,s^2}$ is a composite of propeller and ESC map (Sec. 6.4).
Torque coefficient k_Q/k_T	0.0204 m, from vendor electrical data	**Retracted** (requires 112 % efficiency). Replaced by the interval 0.024 m, band 0.021 m to 0.027 m (Sec. 6.5).
Speed floor / ceiling	120 / 779.8 $\mathrm{rad\,s^{-1}}$	Stands; used unchanged on hardware.
Throttle-to-speed map	13-point table; nonlinear, compressive near hover	Stands, corroborated in flight; its *shape* is now the dominant thrust residual at the operating point. Hardware only.
Actuator time constant τ_{act}	0.079(17) s small-signal, ~ 0.15 s on large steps	Small-signal value confirmed in flight on the current propellers; large-step regime unmeasured on them.
Rotor+propeller inertia J	$1.108 \times 10^{-4}\,\mathrm{kg\,m^2}$ (78 spin-downs)	**Withdrawn.** Circular, and contradicted by the geometric propeller inertia; never used in the dynamics model.

system components, including the variable-geometry model, both controllers, the simulation plant, its autopilot-side airframe description, and the native allocation parameters of the autopilot. A re-validation of the simulation was performed following the bench campaign and passed, as it could not have failed to do: within the simulation environment, the ESC map is linear, and the simulator inverts the same linear map applied by the controller. Consequently, k_T cancels out during the round trip, ensuring that the map does not enter the control loop. The bench error was structurally invisible to the digital twin and surfaced only on the vehicle, in the two flight sessions that failed to leave the ground. The lesson is recorded as a design rule for every plant identity in this thesis: *a parameter that cancels in the simulator round trip cannot be validated by that simulator* — it must be checked against an observation the loop does not construct, which for the thrust law is the hover identity of Chapter 6.

Chapter 5

Variable-geometry modeling

This chapter develops the mathematical model of the morphing quadrotor on which the flight controllers of the following chapters are built. It fixes the reference frames and state variables (Sec. 5.1) and the attitude kinematics (Sec. 5.2); derives the mass and inertia of the airframe as analytical functions of the arm configuration (Sec. 5.3), the rotor thrust and reaction-torque model with its actuator dynamics and its mapping to the PX4 interface (Sec. 5.4), and the configuration-dependent control allocation (Sec. 5.5) that converts a desired wrench into rotor commands; and closes with an explicit statement of the limits of the model (Sec. 5.6), two of whose structural assumptions — the planar rotor layout and the rigidity of the arms — were later falsified by flight evidence. The numerical values of every constant introduced here are identified, and where applicable re-identified in flight, in Chapter 6.

5.1 Notation and reference frames

5.1.1 Reference frames

Two right-handed orthonormal frames coexist. The inertial frame $\mathcal{F}_W$ is Earth-fixed in PX4 NED convention — $\hat{\boldsymbol{x}}_W$ North, $\hat{\boldsymbol{y}}_W$ East, $\hat{\boldsymbol{z}}_W$ down — so that the altitude above the take-off plane is $h = -z$. The body frame $\mathcal{F}_B$ is PX4 *Front-Right-Down*: $\hat{\boldsymbol{x}}_B$ forward, $\hat{\boldsymbol{y}}_B$ right, $\hat{\boldsymbol{z}}_B$ down. All inertial quantities provided by the autopilot (attitude quaternion, body rates, IMU accelerations) are expressed in $\mathcal{F}_B$, and the controller operates *directly* in this frame: no intermediate controller frame is introduced and roll and pitch are not swapped. The in-plane direction of an arm at absolute angle θ is the unit vector $(\cos\theta, \sin\theta)$ in the $(\hat{\boldsymbol{x}}_B, \hat{\boldsymbol{y}}_B)$ plane, a positive increment of θ being a *clockwise* rotation viewed from above, consistent with $\hat{\boldsymbol{z}}_B$ pointing downwards. The frame, the angle convention and the physical layout of

the four arms are drawn in Fig. 5.1.

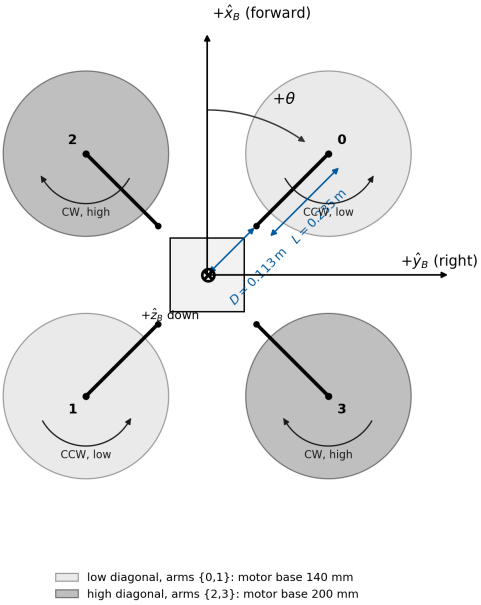


Figure 5.1: Top view of the airframe in the X configuration, in the body FRD frame ($\hat{\boldsymbol{z}}_B$ into the page). Arm indices, rotor spin directions and the two vertical stagger levels (Sec. 5.3.2) are marked at their physical positions; the hinge offset $D = 0.113\,\mathrm{m}$ and the arm length $L = 0.235\,\mathrm{m}$ set the scale, and the discs show the $0.1905\,\mathrm{m}$ propeller radius. A positive arm-angle increment is clockwise as seen from above.

5.1.2 State and setpoint variables

The regulated state comprises the altitude z (NED), the attitude quaternion $\boldsymbol{q} = (q_w, q_x, q_y, q_z)$ and the *intrinsic* Z-Y-X Euler angles (ϕ, θ, ψ) in $\mathcal{F}_B$; the vertical velocity $v_z = \dot{z}$ and the body angular rates (p, q, r), from which the Euler rates are recovered through the kinematic matrix of Sec. 5.2.1; and the morphing configuration, the vector of absolute arm angles $\boldsymbol{\theta} = (\theta_0, \theta_1, \theta_2, \theta_3)^\top$ measured in $\mathcal{F}_B$. Setpoints are tagged with subscripts ref or des, and the error of a signal x with respect to its setpoint is $e_x \triangleq x - x_{\mathrm{des}}$. Given the circular domain of ψ, the yaw error is always the shortest signed difference, mapped onto the principal interval:

$$\mathrm{wrap}_\pi(\alpha) \triangleq \Big((\alpha + \pi) \bmod 2\pi\Big) - \pi, \qquad e_\psi = \mathrm{wrap}_\pi(\psi - \psi_{\mathrm{ref}}). \tag{5.1}$$

5.1.3 Arm configuration and shape presets

The four arms are indexed by $i \in \{0,1,2,3\}$. Each arm is described by its *absolute* angle θ_i (the arm orientation in $\mathcal{F}_B$) and by the *relative* joint rotation $\delta_i = \theta_i - \theta_{i,\mathrm{base}}$,

where $\boldsymbol{\theta}_{\text{base}} = (45°, 225°, 315°, 135°)^\top$ is the X-shape reference configuration. By convention $\theta_i = 0°$ points along $+\hat{\boldsymbol{x}}_B$ and $\theta_i = 90°$ along $+\hat{\boldsymbol{y}}_B$, positive increments being clockwise from above; the morphing state vector is $\boldsymbol{\theta} = (\theta_0, \theta_1, \theta_2, \theta_3)$.

One sign convention must be flagged, because it is load-bearing for anyone comparing this document with the flight logs. The servo driver and the deployed geometry optimiser express joint rotations in the *opposite* sense, $\delta_i^{\text{drv}} = \theta_{i,\text{base}} - \theta_i = -\delta_i$, so every joint value recorded by the project carries the opposite sign of the δ_i used here and in Chapter 8. The mechanical joint limit $|\delta_i| \leq 135°$, enforced identically by the servo driver and by the geometry optimiser, is symmetric and therefore unaffected. Table 3.2 lists both conventions for the four named shape presets used throughout this thesis; the presets are drawn in Fig. 5.2. A requested shape that cannot be resolved to a preset or to four explicit angles aborts the launch by design — there is no silent fallback to X.

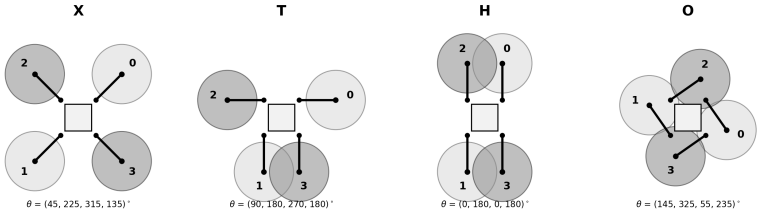


Figure 5.2: The four shape presets drawn in the body FRD frame with the propeller discs at their true 0.1905 m radius. Light and dark discs mark the low and high stagger levels (Sec. 5.3.2): every pair of overlapping discs in T, H and O straddles the two levels.

The rigid-body core of what follows is the standard multirotor model [29, 30]; what is not standard, and what the rest of this chapter develops, is that the mass distribution, the inertia tensor and the control effectiveness are functions of the arm angles rather than constants.

5.2 Kinematics

The attitude quaternion returned by PX4 in $\mathcal{F}_B$ is converted to the body Z-Y-X Euler angles by the standard formulae

$$\phi = \operatorname{atan2}\Big(2(q_w q_x + q_y q_z),\ 1 - 2(q_x^2 + q_y^2) \Big), \tag{5.2}$$

$$\theta = \arcsin\Big(\operatorname{clip}\Big(2(q_w q_y - q_z q_x),\ -1,\ +1 \Big) \Big), \tag{5.3}$$

$$\psi = \operatorname{atan2}\Big(2(q_w q_z + q_x q_y),\ 1 - 2(q_y^2 + q_z^2) \Big), \tag{5.4}$$

the clipping in (5.3) preventing numerical issues when the argument of arcsin marginally exceeds its domain because of a slightly non-normalised quaternion. The angles are used as-is in $\mathcal{F}_B$: no frame swap is applied.

5.2.1 Euler rates from body rates

The body angular rates are mapped to the Euler rates by the standard kinematic relation

$$\begin{bmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{bmatrix} = \begin{bmatrix}1 & \sin\phi\,\tan\theta & \cos\phi\,\tan\theta\\ 0 & \cos\phi & -\sin\phi\\ 0 & \sin\phi/\cos\theta & \cos\phi/\cos\theta\end{bmatrix}\begin{bmatrix}p\\ q\\ r\end{bmatrix}, \tag{5.5}$$

with the numerical floor $\cos\theta \leftarrow \operatorname{sgn}(\cos\theta)\max(|\cos\theta|, \varepsilon_\theta)$, $\varepsilon_\theta = 10^{-3}$, guarding the kinematic singularity at $\theta = \pm\pi/2$. The resulting rates are smoothed by a first-order discrete low-pass filter

$$\dot{\phi}_f^{(k)} \;=\; \alpha_r\,\dot{\phi}^{(k)} + (1-\alpha_r)\,\dot{\phi}_f^{(k-1)}, \qquad \alpha_r = 0.5, \tag{5.6}$$

and analogously on the other two axes. At the 100 Hz odometry callback $\alpha_r = 0.5$ corresponds to a filter time constant of about 10 ms. The value is itself a result of the project rather than an arbitrary choice: the filter originally ran at $\alpha_r = 0.163$, i.e. 51 ms of rate-feedback lag, which stacked on the roughly 80 ms of end-to-end actuation delay drove the attitude loop into a slow-onset 2 Hz coning instability with an e-folding time of about 20 s. The loop model shows full stabilisation for $\alpha_r \in [0.35, 0.8]$, and the change is worth about 15° of phase margin on the sliding-mode path (36.3 to 51.1°). Unless otherwise stated, the Euler rates in the following denote these filtered values.

5.3 Mass and variable-geometry model

5.3.1 Modular decomposition

The drone is modelled as the union of a central rigid body and four arm assemblies of *two types, paired by diagonal*. Each assembly is the concatenation of three point masses: a *connector* of mass m_c rigidly attached to the body at a fixed in-plane position determined by $\theta_{i,\text{base}}$, its mass including the two hinge bearings, which sit *on* the hinge axis and therefore do not move with the morphing; an *arm link* of mass $m_{a,i}$ rotating with the absolute angle θ_i, with centre of mass at distance $\ell_{a,i}$ from the joint; and a *rotor* of mass m_r at the free end, at distance L from the joint. The two arm types differ because the high diagonal (arms 2 and 3) carries the riser ramp that produces the 60 mm rotor z-stagger of Sec. 5.3.2: those arms are 7 g heavier and their centre of mass sits 9.2 mm farther from the hinge. The central body is a homogeneous box of mass m_b and dimensions (B_x, B_y, B_z) with centre of mass at the origin of $\mathcal{F}_B$.

One honesty note attaches to the table: the rotor mass $m_r = 0.275$ kg dates from the original weighing and was not re-weighed after the propellers were changed on 2026-08-03; it is the one entry not re-verified on the current build. The positions of

Table 5.1: Parameters of the mass model, from the CAD model of the airframe (2026-07-31). The hinge offset $D = 0.085 + 0.0397\cos 45°$ is carried as the exact expression rather than a rounded literal: the rounded value 0.1131 introduced a 0.061 mm drift between the analytical rotor positions and the simulation model.

Component	Symbol	Value	Unit
Body mass	m_b	1.790	kg
Body dimensions	(B_x, B_y, B_z)	0.170, 0.170, 0.1458	m
Connector mass (incl. hinge bearings)	m_c	0.180	kg
Arm-link mass, low diagonal ($i \in \{0,1\}$)	$m_{a,i}$	0.085	kg
Arm-link mass, high diagonal ($i \in \{2,3\}$)	$m_{a,i}$	0.092	kg
Rotor mass	m_r	0.275	kg
Connector CoG offset	ℓ_c	0.0232	m
Arm-link CoG offset, low diagonal	$\ell_{a,i}$	0.120186	m
Arm-link CoG offset, high diagonal	$\ell_{a,i}$	0.129426	m
Arm length (joint–rotor)	L	0.235	m
Joint–origin distance in $\mathcal{F}_B$	D	0.113072	m

the joints and of the connector anchor points in $\mathcal{F}_B$ are fixed by the constants

$$\boldsymbol{J}_x = D\,(+1,\,-1,\,+1,\,-1)^\top, \qquad \boldsymbol{J}_y = D\,(+1,\,-1,\,-1,\,+1)^\top, \tag{5.7}$$

$$\boldsymbol{C}_x = 0.085\,(+1,\,-1,\,+1,\,-1)^\top, \qquad \boldsymbol{C}_y = 0.085\,(+1,\,-1,\,-1,\,+1)^\top, \tag{5.8}$$

together with the base angles $\boldsymbol{\theta}_{\text{base}}$ defining the nominal X-shape geometry.

5.3.2 Mass-point positions and the rotor stagger

As a function of the morphing vector the planar positions of the three classes of mass points associated with each arm i are

$$x_i^c = C_{x,i} + \ell_c\,\cos\!\big(\theta_{i,\text{base}}\big), \qquad y_i^c = C_{y,i} + \ell_c\,\sin\!\big(\theta_{i,\text{base}}\big), \tag{5.9}$$

$$x_i^a = J_{x,i} + \ell_{a,i}\,\cos(\theta_i), \qquad y_i^a = J_{y,i} + \ell_{a,i}\,\sin(\theta_i), \tag{5.10}$$

$$x_i^r = J_{x,i} + L\,\cos(\theta_i), \qquad y_i^r = J_{y,i} + L\,\sin(\theta_i), \tag{5.11}$$

the connector position depending only on $\theta_{i,\text{base}}$ because it is rigidly attached to the frame, while arm link and rotor rotate rigidly with θ_i.

The four rotors of the physical aircraft are *not* coplanar: the airframe carries a 60 mm vertical stagger between the two diagonals, arms 0 and 1 low and arms 2 and 3 high, so that every pair of discs that overlaps in planform in the T, H and O presets straddles the two levels (Fig. 5.2); the spin-direction pairs coincide with the stagger levels.

The point-mass model is nevertheless kept *planar*: all mass points lie in $z = 0$ and the model carries no mz^2 term anywhere. This is a declared limit of the model, not a property of the vehicle. The stagger enters only through the CAD anchoring

of the inertia tensor (Sec. 5.3.5) and through the simulation plant; it does not affect the in-plane lever arms of Sec. 5.3.6, which are the only geometric quantity the allocator consumes.

5.3.3 Instantaneous centre of mass

The total rigid mass (excluding the payload provision, treated as an unobserved disturbance, Sec. 7.2) is

$$M \triangleq m_b + \sum_{i=0}^{3}\Big(m_c + m_{a,i} + m_r\Big) = 3.964\,\mathrm{kg}, \tag{5.12}$$

and the in-plane rigid centre of mass is

$$\boldsymbol{r}_{\mathrm{CoG}}(\boldsymbol{\theta}) = \frac{1}{M}\sum_{i=0}^{3}\Big[m_c\,(x_i^c, y_i^c)^\top + m_{a,i}\,(x_i^a, y_i^a)^\top + m_r\,(x_i^r, y_i^r)^\top\Big], \tag{5.13}$$

the central body, having its own CoG at the origin, contributing only to M. In the shapes whose angle sets are symmetric under the diagonal pairing (X, H and O) the CoG remains at the origin — not because the arms are identical, which the two arm types no longer are, but because the heavier pair {2,3} lies on a *single diagonal*, so its mass excess balances point-symmetrically about the origin. In the T shape, the only asymmetric preset, the CoG drifts by 3.8 cm along x (Table 3.2).

5.3.4 Inertia tensor

The inertia about the instantaneous CoG is obtained by adding the contribution of the central body, treated as a uniform box, and those of the arm point masses, all referred to the current CoG by the Huygens–Steiner parallel-axis theorem [1]:

$$I_{xx} = I_{xx}^b + m_b\,y_{\mathrm{CoG}}^2 + \sum_{i,k} m_{k,i}\left(y_i^{(k)} - y_{\mathrm{CoG}}\right)^2, \tag{5.14}$$

$$I_{yy} = I_{yy}^b + m_b\,x_{\mathrm{CoG}}^2 + \sum_{i,k} m_{k,i}\left(x_i^{(k)} - x_{\mathrm{CoG}}\right)^2, \tag{5.15}$$

$$I_{zz} = I_{zz}^b + m_b\left(x_{\mathrm{CoG}}^2 + y_{\mathrm{CoG}}^2\right) + \sum_{i,k} m_{k,i}\left[\left(x_i^{(k)} - x_{\mathrm{CoG}}\right)^2 + \left(y_i^{(k)} - y_{\mathrm{CoG}}\right)^2\right], \tag{5.16}$$

where $k \in \{c, a, r\}$ ranges over the mass-point classes and $I_{xx}^b = \frac{m_b}{12}(B_y^2 + B_z^2)$, $I_{yy}^b = \frac{m_b}{12}(B_x^2 + B_z^2)$, $I_{zz}^b = \frac{m_b}{12}(B_x^2 + B_y^2)$ are the principal inertias of the central box. Products of inertia are neglected: the tensor $\mathbf{I}(\boldsymbol{\theta}) = \mathrm{diag}(I_{xx}, I_{yy}, I_{zz})$ is treated as diagonal, an assumption supported by the near-symmetric frame layout and consistent with the absence of explicit cross-axis coupling in the control law, whose verification status is stated in Sec. 5.6.

5.3.5 CAD anchoring of the inertia tensor

The point-mass tensor is *not* what the controllers consume directly, and presenting it as such would make every inertia used by the control laws irreproducible: being planar the model carries no mz^2 term — it cannot see the rotor stagger — and it idealises the central body as a uniform box. It is therefore used only for its *dependence on* $\boldsymbol{\theta}$, while the absolute scale is anchored per-axis to the CAD tensor of the full assembly in X:

$$\mathbf{I}(\boldsymbol{\theta}) \;=\; \mathrm{diag}(\boldsymbol{s})\,\mathbf{I}_{\mathrm{model}}(\boldsymbol{\theta}), \qquad \boldsymbol{s} \;=\; \mathbf{J}_{\mathrm{CAD}} \oslash \mathbf{I}_{\mathrm{model}}(\boldsymbol{\theta}_{\mathrm{base}}) \;=\; (1.058,\; 1.041,\; 0.907)^{\top}, \tag{5.17}$$

with $\oslash$ element-wise division, so that $\mathbf{I}(\boldsymbol{\theta}_{\mathrm{base}}) = \mathbf{J}_{\mathrm{CAD}} = (0.1217,\, 0.1198,\, 0.2030)\,\mathrm{kg\,m^2}$ exactly and away from X the model contributes its trend in $\boldsymbol{\theta}$. This is a declared approximation — exact in X, extrapolated elsewhere — pending per-component CAD tensors. The CAD tensor is referred to the vehicle CoG, which the CAD places 101.6 mm above the bottom plate with a planar offset of $(-0.30,\, +0.83)$ mm (Sec. 6.3).

Two closure notes belong here. An earlier hypothesis suggesting that the roll/pitch inertia was underestimated by a factor of approximately two, inferred from a system-identification exercise on a crash record, was *retracted*. This error arose because the actuator had been modelled as a pure delay in that analysis; such a model shifts phase without attenuation, causing the magnitude attenuation characteristic of the true first-order lag to be incorrectly attributed to inertia. Furthermore, the composite inertia of the Gazebo model sits at $+5/+7/+12\,\%$ above $\mathbf{J}_{\mathrm{CAD}}$. This represents a deliberate and conservative plant-versus-controller mismatch that was never reconciled.

5.3.6 Rotor positions relative to the CoG and lever arms

The effective lever arms used by the allocator are the rotor coordinates relative to the current CoG [1, 2]:

$$x_i \;\triangleq\; x_i^r - x_{\mathrm{CoG}}, \qquad y_i \;\triangleq\; y_i^r - y_{\mathrm{CoG}}, \tag{5.18}$$

the pairs (y_i, x_i) entering the allocation matrix (5.25). In simulation a ground-truth geometry feedback published by the Gazebo twin can replace the analytical rotor positions and CoG, provided the measurement is refreshed within 0.2 s; no such measurement exists on the physical aircraft, where the corresponding flag is pinned off in every flight unit, so the real vehicle always closes the allocation on the analytical geometry evaluated at the servo-measured arm angles. The mean lever arms used by the axis scaling of the moment-bias estimator (Sec. 7.2) are

$$\bar{\ell}_{\phi}(\boldsymbol{\theta}) \;=\; \tfrac{1}{4}\sum_i |y_i|, \qquad \bar{\ell}_{\theta}(\boldsymbol{\theta}) \;=\; \tfrac{1}{4}\sum_i |x_i|. \tag{5.19}$$

The same quantities originally gain-scheduled the outer XY position loop, but that scheduling is deliberately disabled in the deployed configuration: the lower bound on the scale factor was raised to unity, so every morphing shape receives the X-shape position gains and tilt cap. The choice trades geometric adaptation for comparability — path-following performance can be compared across configurations on equal terms — and the residual authority limit of the short-lever shapes is enforced by the moment caps of Sec. 7.4 instead.

Table 3.2 evaluates the machinery of this section — CoG, mean levers and calibrated inertia — on the four shape presets. Two facts become visible. The only asymmetric preset is T, whose CoG shifts by 3.8 cm along x, a shift the lever arms of (5.18) absorb. H decouples the axes to an extreme degree — J_{xx} drops to less than a third of its X value while J_{yy} grows by 46 % — with the control consequences developed in Sec. 5.5.2. Figure 5.3 extends the same evaluation from the four presets to the whole continuum the mechanism can reach.

5.4 Rotor model and PX4 interface

5.4.1 Thrust and reaction torque

Each rotor is modelled as an aerodynamic actuator whose thrust and reaction torque about its rotation axis are proportional to the square of the angular velocity:

$$F_i \;=\; k_T\,\omega_i^2, \qquad Q_i \;=\; k_Q\,\omega_i^2, \tag{5.20}$$

with $k_T = 4.1547 \times 10^{-5}\,\mathrm{N\,s^2}$ and $k_Q = k_T \cdot 0.024\,\mathrm{m} = 9.971 \times 10^{-7}\,\mathrm{N\,m\,s^2}$. Neither coefficient is a direct bench measurement, yet their provenance is critical for all downstream applications: the initial bench value $k_T = 7.40 \times 10^{-5}$ was retracted after proving incompatible with flight, as a vehicle parameterised by this value could not achieve lift-off. Consequently, k_T was re-derived from the hover identity using a piloted position-hold log and subsequently confirmed independently on the current propellers. Because the allocation is homogeneous in k_T, the same factor of 1.78 would have mis-scaled every *moment* identically, which makes this the most load-bearing constant of the chapter. The ratio k_Q/k_T is not a measurement at all but the centre of the physically admissible band 0.021 m to 0.027 m, bounded above by an electrical-power argument and below by momentum theory; and k_T is a *composite* of the rotor and the ESC map rather than a property of the rotor alone, because no rotor speed is measured in flight. The full identification record is in Chapter 6 (Sections 6.4, 6.5 and 6.10).

The angular velocity of each rotor is constrained to $[\omega_{\min},\, \omega_{\max}] = [120,\, 779.8]\,\mathrm{rad\,s^{-1}}$ — a model floor set just above the lowest bench-verified sustained speed and the measured full-throttle speed on a fresh pack. The floor is a *controller-side* bound, not a physical one: on hardware the measured ESC table saturates flat below

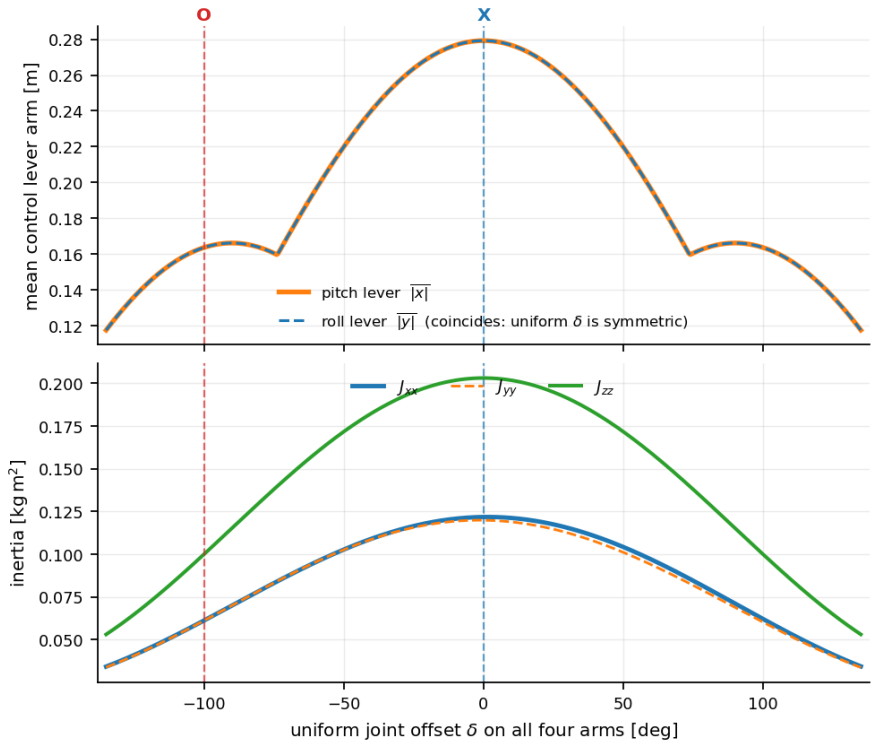


Figure 5.3: Mean control lever arms (5.19) and the diagonal inertia tensor (5.14)–(5.16), CAD-calibrated by (5.17), across the whole reachable envelope, evaluated for a uniform joint offset δ applied to all four arms. Roll and pitch levers coincide under a uniform offset, the configuration remaining symmetric; the asymmetric presets T and H are reached with per-arm offsets and are quantified separately in Fig. 5.4. Both curves are computed by the same functions the flight software calls, so the values at $\delta = 0$ and $\delta = -100°$ reproduce the X and O rows of Table 3.2 exactly. The lever arm falls monotonically away from X in either direction: every morphing configuration is bought with control authority.

its lowest entry, so a commanded $120\,\mathrm{rad\,s^{-1}}$ actually spins the rotor at about $148.3\,\mathrm{rad\,s^{-1}}$, and any control-authority budget at idle must be computed against the physical idle speed. The spin directions are + (CCW) for rotors 0,1 and − (CW) for rotors 2,3 (Fig. 5.1), which sets the signs in the yaw row of (5.25). The total thrust required to counteract the weight of the rigid drone, and the corresponding equilibrium speed assuming four identical rotors, are

$$T_{\mathrm{hov}} = Mg = 38.887\,\mathrm{N}, \qquad \omega_{\mathrm{hov}} = \sqrt{\frac{T_{\mathrm{hov}}}{4k_T}} = 483.73\,\mathrm{rad\,s^{-1}}. \tag{5.21}$$

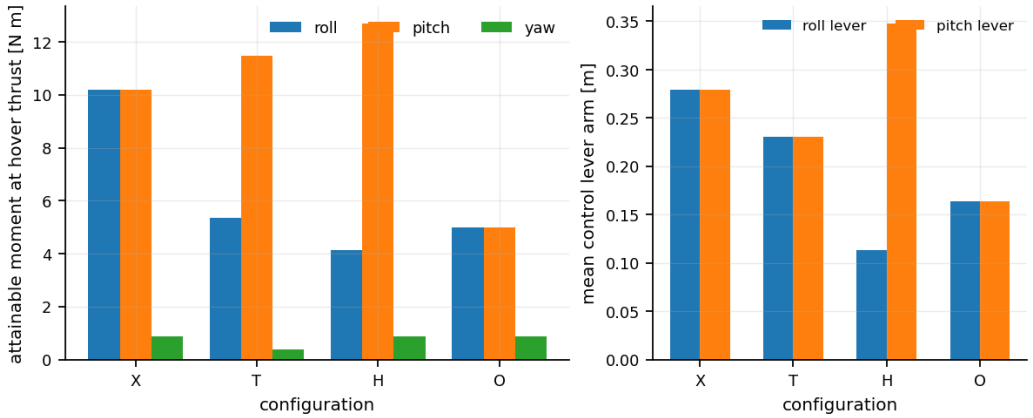


Figure 5.4: Left: the moment each configuration can actually raise about each body axis while holding hover thrust, obtained by maximising $|M|$ over the allocation (5.25) subject to the true per-rotor bounds $\omega_i^2 \in [\omega_{\min}^2, \omega_{\max}^2]$ and to the remaining two moments being zero — a linear program, not a scaling argument. Right: the mean lever arms that produce them. The two effects the flight campaign ran into are visible here before any flight: T halves the roll authority of X and collapses the yaw authority to 45 % of it, while H trades roll for pitch — the extreme anisotropy that makes it inadmissible as a take-off shape (Sec. 3.2).

5.4.2 Actuator dynamics and transport delay

The static map (5.20) does not describe how a rotor *reaches* a commanded speed. Each rotor is modelled as a first-order lag from the commanded angular velocity to the applied one,

$$\dot{\omega}_{a,i} \;=\; \frac{\omega_{c,i} - \omega_{a,i}}{\tau_{\text{act}}}, \tag{5.22}$$

which is the augmented state on which the prediction model of Chapter 9 and the Jacobians of Appendix A are built; the sliding-mode law does not consume it. The time constant is measured, but it is not a constant of the plant: the bench identification shows it to be *amplitude-dependent*, nearly doubling from the small-signal regime commanded near hover to take-off-scale steps, while the model can carry only one value (Secs. 6.6 and 6.6.1).

Distinct from the lag, the command path carries a *transport delay* through the companion-computer middleware, measured directly at about 30 ms round trip (Sec. 6.3.1) and injected explicitly in simulation, where the transport is otherwise instantaneous. The two effects must not be conflated: a first-order lag attenuates magnitude, whereas a pure delay adds phase without attenuation. Confusing them is precisely the error that once booked the lag's attenuation as missing inertia (Sec. 5.3.5).

5.4.3 Mapping to the PX4 interface

The PX4 *direct actuator* interface accepts normalised commands $u_i \in [0,1]$. The simulator plugin maps u_i linearly to a rotor speed in $[100,\ 779.8]\,\mathrm{rad\,s^{-1}}$, so for $F_i \propto \omega_i^2$ to be honoured the controller must publish a *normalised angular velocity*, not a normalised thrust:

$$u_i \;=\; \mathrm{clip}\!\left(\frac{\omega_i - \omega_{\mathrm{ESC,min}}}{\omega_{\mathrm{ESC,max}} - \omega_{\mathrm{ESC,min}}},\ 0,\ 1\right). \tag{5.23}$$

On the physical drone, the linear map is replaced by the bench-measured lookup table described in Sec. 4.3, as a real ESC is not linear in ω and varies with battery voltage. This table consists of thirteen measured points spanning $148.3\,\mathrm{rad\,s^{-1}}$ to $779.8\,\mathrm{rad\,s^{-1}}$, with interpolation applied between points. An import-time guard verifies strict monotonicity, containment within [0,1], and coverage up to $\omega_{\max}$; consequently, a corrupted table aborts start-up rather than allowing flight. The lookup table is active exclusively in the hardware configuration, whereas the simulator retains the linear map implemented by its motor plugin.

One structural consequence of this distinction warrants emphasis. In simulation, the round trip $\omega \to u \to \omega$ acts as an identity operation, since the controller and the motor plugin invert the same linear map. By construction, the simulated plant carries the same k_T as the controller. Therefore, an incorrect k_T remains *self-consistent* within the simulation environment and cannot be detected there; this structural symmetry explains why the bench error in k_T manifested only during flight.

5.5 Control allocation

5.5.1 Allocation matrix

Given the desired force/moment vector in $\mathcal{F}_B$,

$$\boldsymbol{\tau} \;=\; (T,\ M_\phi,\ M_\theta,\ M_\psi)^\top, \tag{5.24}$$

and the vector of squared angular velocities $\boldsymbol{w} \triangleq (\omega_0^2, \omega_1^2, \omega_2^2, \omega_3^2)^\top$, the allocation relation, obtained by summing the contributions of rotor thrust and reaction torque about the current CoG, is linear in $\boldsymbol{w}$:

$$\boldsymbol{\tau} \;=\; \mathbf{A}(\boldsymbol{y}, \boldsymbol{x})\,\boldsymbol{w}, \qquad \mathbf{A}(\boldsymbol{y}, \boldsymbol{x}) \;=\; \begin{bmatrix} k_T & k_T & k_T & k_T \\ -k_T y_0 & -k_T y_1 & -k_T y_2 & -k_T y_3 \\ k_T x_0 & k_T x_1 & k_T x_2 & k_T x_3 \\ k_Q & k_Q & -k_Q & -k_Q \end{bmatrix}. \tag{5.25}$$

The rows are total thrust; the roll moment $M_\phi = -\sum_i y_i F_i$, a right-wing-down moment requiring more thrust on the negative-y rotors; the pitch moment $M_\theta =$

$+\sum_i x_i F_i$; and the yaw moment, generated by the reaction torques with signs set by the rotor spin directions.

Configuration dependence of the allocation. $\mathbf{A}$ is invertible over the whole morphing envelope, but its conditioning is configuration-dependent, and the dependence is worth quantifying because it is part of the mechanism of the loss of control analysed in Sec. 5.6. In X, H and O the roll row and the yaw row are exactly orthogonal and $\mathrm{cond}(\mathbf{A}) = 41.7$. In T they are not: the roll levers split as $\boldsymbol{y} = (+0.349,\ -0.113,\ -0.348,\ +0.114)\,\mathrm{m}$ instead of the uniform $\pm 0.279\,\mathrm{m}$ of X, so only rotors 0 and 2 carry a large roll lever — and those two spin in *opposite* directions. The normalised roll and yaw rows then subtend $|\cos| = 0.454$ and $\mathrm{cond}(\mathbf{A})$ rises to 47.0: every commanded roll moment is bought with a yaw reaction that the remaining two rotors, holding only $D = 0.113\,\mathrm{m}$ of roll lever, must cancel.

5.5.2 Solution of the allocation problem

Since $\mathbf{A}$ is square and generically non-singular, the squared rotor speeds are recovered by direct inversion,

$$\boldsymbol{w} \;=\; \mathbf{A}^{-1}(\boldsymbol{y}, \boldsymbol{x})\,\boldsymbol{\tau}, \tag{5.26}$$

with the Moore–Penrose least-squares solution serving as a fallback whenever $\mathbf{A}$ becomes numerically rank-deficient. For an invertible square $\mathbf{A}$, any diagonal output weighting cancels out in (5.26); consequently, no weighted pseudo-inverse is employed. The loss of control authority associated with degenerate morphing shapes is managed downstream through the dynamic moment caps described in Sec. 7.4 and the lever-dependent axis scaling within the moment-bias estimator, rather than inside the allocation itself.

The H configuration represents an extreme case: with all four arms aligned along $\pm\hat{\boldsymbol{x}}_B$, the $L\sin\theta_i$ terms vanish, and the roll lever collapses exactly to the body half-width $D = 0.1131\,\mathrm{m}$. This corresponds to 40.5 % of its value in the X configuration, not zero, while the pitch lever extends to 0.3481 m. The resulting inertia split ($J_{xx} = 0.0354$ against $J_{yy} = 0.1752\,\mathrm{kg\,m^2}$) causes a *bolted* H configuration to violate the locked-shape take-off invariant in the start-up checks: the scheduled pitch stiffness $k_{s,\theta} = J_{yy}\,\omega_n^2 = 3.71\,\mathrm{N\,m}$ exceeds 0.8 of the hard take-off moment cap of $3.4\,\mathrm{N\,m}$. Therefore, a take-off locked in H is refused prior to arming, whereas an H configuration reached in the air after lift-off from X remains routinely operable. This refusal reflects an actuation-authority constraint rather than a controller defect.

Finally, the solution may contain negative components whenever the setpoint demands reverse thrust on a rotor, which is physically infeasible. A clip $\boldsymbol{w} \leftarrow$

$\max(\boldsymbol{w}, \mathbf{0})$ is therefore applied, followed by

$$\omega_i \;=\; \mathrm{clip}\Big(\sqrt{w_i},\; \omega_{\min},\; \omega_{\max}\Big) \tag{5.27}$$

and by u_i via (5.23). The saturation in (5.27) introduces actuator saturations that the attitude controller accounts for through its integral action with *anti-windup* (Sec. 7.1.2).

5.6 Limits of the model

Every model in this chapter was chosen for tractability, and each choice is an assumption that flight either upheld or falsified. Three limits must travel with the model, and the first was falsified quantitatively by the loss of the vehicle.

The arms are not rigid. The model treats each arm as a rigid link hinged to a rigid body. The loss-of-control flight showed that in T the true plant carries about three times — 10 dB — more gain than this model near 3 Hz: not a delay, but the resonant amplification of an arm/servo compliance mode, directly visible as an oscillation of the measured joint angles, whose excess gain equalled the entire design margin (Sec. 6.8). What matters here is that neither the control model nor the simulation plant carries any arm stiffness or damping, so no analysis built on this chapter can predict that mode.

The inertia tensor is assumed diagonal. Products of inertia are neglected throughout. The available CAD verification supports the assumption — $I_{xy}/I_{xz}/I_{yz}$ at 1.56, 1.60 and 0.30 % of the diagonal terms — but it was obtained on an earlier CAD revision and has not been repeated on the staggered, diagonally-asymmetric assembly described here.

The simulation plant is deliberately mismatched. The composite inertia tensor of the Gazebo model exceeds the CAD tensor consumed by the controllers by $+5/+7/+12\,\%$, and its actuator time constant differs from that of the flown vehicle. Both discrepancies are documented and conservative in direction; however, they imply that simulation results are generated against a plant that is not numerically identical to the controller's internal model.

The response to the falsified rigid-arm assumption was parametric rather than structural: the actuator-lag model was corrected to reflect the small-signal time constant of the current propellers, and Levenberg–Marquardt regularisation was applied to the predictive controller's QP. This adjustment restores the stability margin that had been consumed by the compliance mode in the T configuration. The current flight cards re-fly the full morphing trajectory tour, including T and

H configurations, under the predictive controller with this setup. A declared fallback exists should take-off validation fail: X-only trajectories remain under predictive control, while T/H transitions are handed over to the sliding-mode stack. Structural remedies, such as servos with closed-loop current control or a second-order compliant-joint state in the model, remain future work.

Chapter 6

System identification

Every physical constant used by the controllers of Chapters 7 and 9, by the geometry optimiser of Chapter 8 and by the simulation plant of Chapter 10 is identified in this chapter, together with the evidence that supports it and, where it exists, the evidence that later refuted it. The chapter is organised around a result that was not anticipated when the bench campaign was designed, and that turned out to be its main methodological outcome:

Every parameter that was identified on the bench and later re-identified in flight was found to be wrong. The thrust coefficient by a factor of 1.78, the torque coefficient by more than the width of its own confidence band, the transport delay by an order of magnitude, and both servo tuning decisions — the integral gain and the motion profile velocity — were reversed in flight after being promoted at the bench.

This is not a catalogue of errors. The bench measurements were, on their own terms, correct: performed with a calibrated six-axis load cell, internally consistent, and one of them agreeing with an independent vendor bench within 3 %. What the bench cannot replicate is the *coupling*: it measures a single rotor, rigidly restrained at constant speed with an unloaded arm, whereas the vehicle operates as a plant in which the rotor, airframe, servo-actuated arms, and estimator function as one integrated system.

6.1 The two-source problem

Two independent sources of parameter estimates are available. The *bench* offers control of the experiment — one variable at a time, direct force measurement, repeatable inputs — while the *flight* offers relevance at the cost of control. The pivotal observation is that hovering is a steady-state condition in which the total thrust *is* the weight, with no dependence on inertia, on any control gain, or on any

aerodynamic model:

$$T_{\text{true}} = m\|\mathbf{a}_{\text{ned}} - \mathbf{g}_{\text{ned}}\| \xrightarrow[\text{steady}]{} mg, \tag{6.1}$$

so any parameter that maps a command to thrust can be checked against Eq. (6.1) using nothing but a logged accelerometer and a logged collective. This single identity is what falsified the bench thrust coefficient (Sec. 6.4) and it is the reason the flight estimates are quoted with more confidence than the bench estimates they replaced, despite being obtained under far less controlled conditions. Table 6.1 summarises the outcome parameter by parameter; the remainder of the chapter derives each row.

Table 6.1: Parameters identified twice — once on the bench, once in flight. The last column names the observation that settled the disagreement.

Parameter	Bench	Flight	Falsified by
k_T	7.40×10^{-5}	4.1547×10^{-5}	Vehicle could not take off
k_Q/k_T	0.0204 m	0.024 m	Electrical power bound
τ_{act}	0.079 s	0.082 s	— (confirmed)
Transport delay	—	14.2 ms	Direct timestamping
Servo integral gain	$I = 400$	$I = 0$	Static load trips in hover
Profile velocity	PV = 14	inconclusive	Mode frequency unchanged

6.2 Bench identification

The bench campaign of 2026-07-15 is reported in full in Chapter 4: one flight motor on an ATI Mini40 six-axis load cell, driven through the autopilot's actuator-test interface with bidirectional-DShot speed telemetry, in two studies — thrust, torque, the throttle-to-speed map and the speed limits at 20 Hz, and the actuator time constant at 100 Hz. Only what that campaign hands to this chapter is repeated here.

Its final thrust figure is not statistical but geometric in origin. The first analysis fabricated a tare that does not exist and read the axial channel of a motor mounted 31.6° off the load cell's z axis; taking the tilt-invariant vector magnitude $T = \|\mathbf{F}\|$ removes the defect without any fitted correction and yields

$$k_T^{\text{bench}} = 7.40 \times 10^{-5}\,\text{N}\,\text{s}^2, \qquad R^2 = 0.9987, \tag{6.2}$$

the two captures agreeing to 1.2 % and landing at 103.1 % of the vendor's independent bench measurement — an agreement earned rather than imposed, and at the time taken as decisive corroboration. Section 6.4 shows that both benches are consistent with each other and both are inapplicable to the vehicle in flight, which is a different and more interesting situation than either being wrong.

The quantities that depend only on rotor speed are unaffected by the load-cell geometry and remain valid: the usable speed envelope is $120\,\mathrm{rad\,s^{-1}}$ to $779.8\,\mathrm{rad\,s^{-1}}$ and the thirteen-point throttle-to-speed map produced by the same sweep is still the map used on hardware (Sec. 6.4). The torque, by contrast, was never recoverable from this cell — both of its own torque readings imply a physically impossible figure of merit — so the value carried forward, $k_Q/k_T = 0.0204\,\mathrm{m}$, was *derived* from the motor velocity constant and the vendor's current figures rather than measured, a provenance correctly flagged at the time and finally replaced in Sec. 6.5.

6.3 Mass, inertia and geometry

Mass properties are taken from the CAD model of the as-built airframe rather than from a parameter fit, a choice made after an attempted identification produced a circular result (Sec. 6.3.1). The values — total mass 3.964 kg, inertia $[0.1217, 0.1198, 0.2030]\,\mathrm{kg\,m^2}$ in X, arm length 235.0 mm, hinge offset 113.072 mm, rotor z-stagger 60 mm — are tabulated with their sources in Appendix B. They are the CAD-anchored values included directly into the control allocation and into every gain scheduled on inertia.

6.3.1 A retracted identification, and the measurement that replaced it

An earlier version of this model included a transport delay of 180 ms, and one variant specified 595 ms. Neither value was measured directly; both were *inferred* from the frequency of an observed attitude resonance, assuming it was delay-driven. Using a delay derived in this manner to explain the very resonance that generated it constitutes a circular argument. The delay was subsequently measured directly by timestamping the middleware round trip and found to be smaller by roughly an order of magnitude: 14.2 ms mean one way, corresponding to about 30 ms for the round trip. This value is now explicitly injected into the simulation. The consequence extends beyond bookkeeping: with the retracted figure, arguments advocating to "soften the loop to buy back phase lost to transport" were quantitatively motivated; however, with a delay of 14.2 ms, such an argument holds no weight. As shown in the flight-19 analysis (Sec. 6.8), the phase deficit observed in flight is not attributable to transport delay at all.

6.4 Thrust coefficient identified in flight

The bench thrust coefficient of Eq. (6.2) was rejected not on statistical grounds but because a vehicle built on it does not fly: with $k_T = 7.40 \times 10^{-5}$ the model places

hover at a normalised collective of 0.273, two flight sessions on that basis produced ten take-off attempts and no flight, and since the allocation is homogeneous in k_T the same factor understated the available moments, leaving the attitude loop soft as well as the thrust command low. The correction applies Eq. (6.1) to the one log in which the airframe genuinely hovered under pilot control (95 s of manual position hold): the measured collective is 0.4152, the autopilot's own hover-thrust estimator [31] independently agrees at 0.4144, and inverting the measured throttle-to-speed map at that collective gives $\omega_{\text{hover}} = 482.87\,\text{rad}\,\text{s}^{-1}$, hence $k_T = (T_{\text{hover}}/4)/\omega_{\text{hover}}^2 = 4.1547 \times 10^{-5}\,\text{N}\,\text{s}^2$ — a factor of 1.78 below the bench value, and the value used throughout the thesis, so that the maximum per-rotor thrust is 25.26 N rather than 45 N and the thrust-to-weight ratio 2.60 rather than 4.2. The pattern across every estimate is unambiguous: the bench figure and the vendor datasheet (7.258×10^{-5}) agree within 2 %, the flight estimates (a second hover estimate through the ESC table gives 4.061×10^{-5}) agree within 2.3 %, and the two groups differ by 1.78, which rules out a simple error in our own bench procedure and points at a difference between the restrained single-rotor condition and the free-flying vehicle — an open question whose closing experiment is identified in Sec. 6.10. Two later measurements bound the adopted value without changing it. After the propeller change, three logs give ratios of measured to modelled thrust of 1.030, 0.994 and 0.964 in the hover collective band, so the roughly 12 % deficit predicted for the smaller propellers does not exist. And in 90 s of settled hover the measured thrust is 38.85 N against a modelled 35.19 N, an effective coefficient 10.4 % above the adopted one; binning the ratio by commanded collective across three flights (1,609 samples) gives 1.110, 1.036 and 0.983 in the 0.36 to 0.40, 0.40 to 0.44 and 0.44 to 0.55 bands, crossing unity at the collective of about 0.415 where k_T was anchored, so the residual is a defect in the *shape* of the throttle-to-speed map, not in the thrust coefficient, and rescaling k_T would only move it to a different operating point.

6.5 Torque coefficient: an interval, not a measurement

The torque coefficient governs yaw authority in its entirety, and it is the parameter this work knows least well; it is reported as an interval, with one edge measured and the other assumed. The measured edge is an upper bound obtained from electrical power alone: in the same hovering log the battery delivers 604.8 W at the four rotors net of the measured hotel load, and with $\sum \omega_i^3 = 4.514 \times 10^8$ from the ESC table the shaft power $P = k_Q \sum \omega_i^3$ gives

$$k_Q = \eta \cdot 1.339 \times 10^{-6}\,\text{N}\,\text{m}\,\text{s}^2 \;\leq\; 1.34 \times 10^{-6}\,\text{N}\,\text{m}\,\text{s}^2, \tag{6.3}$$

since the efficiency η cannot exceed unity. This bound uses no mass, no thrust coefficient, no air density and no yaw dynamics — only a power modulus and a rotor speed. The rotor speed entering Eq. (6.3) passes through the ESC table, which was itself under suspicion, so it was checked against a channel that does not: the accelerometer spectrum in the same window shows rotor tones at 73 Hz to 76 Hz against commanded speeds of 71.0, 75.3, 76.3 and 83.6 Hz, while the 57.6 Hz line that the superseded thrust coefficient would require lies in the noise floor. The ESC table is therefore corroborated independently, and with it the entire chain leading to the flight value of k_T (Sec. 6.4). The value carried in the model is

$$k_Q/k_T = 0.024\,\mathrm{m}, \qquad \text{honest band } 0.021\,\mathrm{m} \text{ to } 0.027\,\mathrm{m}, \tag{6.4}$$

the centre of the admissible interval, corresponding to $k_Q \in [8.7 \times 10^{-7}, 1.13 \times 10^{-6}]$, i.e. $\pm 15\,\%$. It replaces the bench figure of 0.0204 m, which was derived rather than measured. Presenting a control-relevant parameter as an interval is deliberate: every yaw authority margin quoted in this thesis inherits this $\pm 15\,\%$, and a bifilar pendulum measurement remains the outstanding item that would collapse it to a point.

6.6 Actuator time constant and its amplitude dependence

The rotor does not reach a commanded speed instantly, and the predictive controller of Chapter 9 carries a first-order lag to represent the delay. The identification of that lag is the one place where the bench and the flight agree — once the regimes are matched. The second bench study drove eight randomised step profiles, yielding 156 step transitions over 20,188 samples logged at 100 Hz, and the result is that the time constant is governed not by the commanded step in throttle but by $\Delta\omega$, the actual speed change demanded, and only once that change becomes large.

Table 6.2: Actuator time constant by demanded speed change. The small-signal regime is the one the predictive controller commands near hover.

Regime	τ	n
$\Delta\omega < 200\,\mathrm{rad\,s^{-1}}$ (hover, small signal)	0.079(17) s	45
$\Delta\omega \geq 200\,\mathrm{rad\,s^{-1}}$ (take-off scale)	0.116(29) s	16

The dependence is monotonic across the whole range, τ rising from 0.076 s below $100\,\mathrm{rad\,s^{-1}}$ to 0.153 s above $400\,\mathrm{rad\,s^{-1}}$ (Fig. 6.1), and the asymmetry between acceleration and deceleration explains the shape. Spin-down requires no motor torque — aerodynamic drag does the work — so fitting $J\dot{\omega} = Q_{\mathrm{em}} - k_Q\omega^2$ across all 78 spin-downs with k_Q fixed and the rotor inertia as the single free parameter

reproduces every deceleration to a median error of 4.4 % of the step. Spin-up, by contrast, is limited by the finite motor torque, over-delivering on small steps and saturating on large ones.

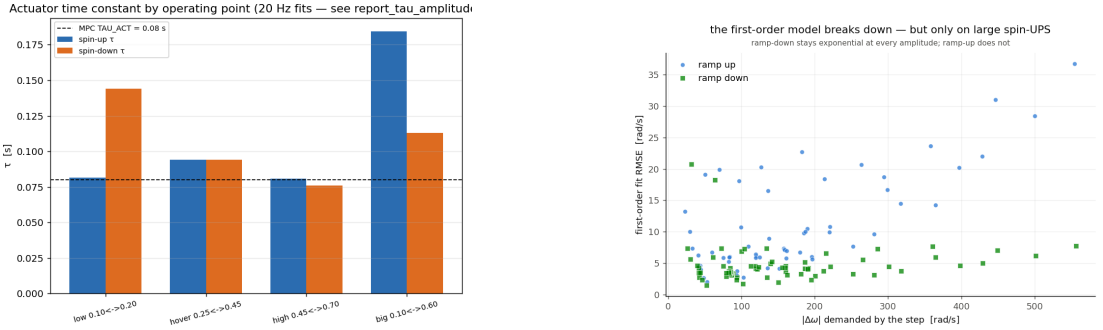


Figure 6.1: Left: identified actuator time constant across the tested operating points, the small-signal cluster near hover being the regime relevant to the predictive controller. Right: residuals of the first-order model fitted to all 78 spin-down transitions with a single free parameter — one inertia value reproduces every deceleration in the dataset.

6.6.1 Two retractions, and the in-flight confirmation

Two results from this study must be withdrawn, for the same circularity flagged in Sec. 6.3.1: the compact law $\tau = J/(2k_Q\omega)$ reproduces the small-signal limit of the drag model, but the inertia J that makes it work was fitted *through* that same model with k_Q held fixed, so the two scale together, the law cannot be cited as independent rotor physics, and the inertia value must not be quoted as a measured property of the rotor.

The operational history of this parameter is instructive. The initial bench value of 0.08 s was increased to 0.15 s for modelling purposes, as the predictive controller's quadratic program became ill-conditioned with the smaller lag. It was subsequently restored to 0.08 s for flight after an empirical frequency-response fit on flight data yielded values of 0.082 s in configuration X and 0.075 s in configuration T, thereby confirming the bench value within the regime it was intended to represent. One caveat relevant to reproducibility remains: the constant declared in the parameter file remains 0.15 s and the flown value is supplied at launch time, so any attempt to reproduce the flight results from the file alone will use the wrong lag. Finally, the amplitude dependence of Table 6.2 is a genuine limitation of the predictive model, which carries a single scalar lag: at take-off-scale steps the true time constant approaches twice the modelled one. It is equally inexpressible in the simulation plant, whose actuator model is first-order with a fixed constant.

6.7 The arm and servo subsystem

The morphing arms are driven by four Dynamixel XC430-T150BB-T smart servos on a dedicated 12 V rail, with a stall torque of 1.6 N m and, importantly for what follows, *no current sensor*. Load can therefore only be observed through the servo's own normalised load register and through the positional droop under torque. The loads measured in flight establish the operating envelope of the mechanism:

Table 6.3: Servo loads during flight, expressed as a percentage of stall torque. The O configuration represents the critical case.

Condition	Load	Note
Hover, X configuration	15 % to 24 %	nominal
Reconfiguration transit	35 % to 45 %	peak
Hover, O configuration (worst arm)	43.9 % (p95)	6.3° droop

The final row explains why the O configuration is flown exclusively as a transit shape and never held in steady state: the sustained load reaches the threshold at which the servo's factory overload protection triggers, and such a latch event during flight results in the release of the arm. This failure occurred once, and its dynamics were identified from the flight log — on release the arm back-drives at up to $336\,^\circ\,\mathrm{s}^{-1}$ and reaches the neighbouring rotor disc in 0.73 s, against which the protection layer built in response detects the runaway in 0.11 s to 0.16 s with a full detect-and-recover cycle of 0.15 s. The two tuning decisions taken on the bench for this subsystem were both reversed in flight: raising the servo integral gain improved static holding on an unloaded arm at the bench but in flight produced sustained loads of 49 % to 52 % and two static protection trips; doubling the motion profile velocity likewise reduced measured stiction on the bench but left the 6 Hz oscillation in place in flight (Fig. 6.2).

6.8 The arm-compliance mode

The mode that the profile-velocity experiment had failed to move was identified in full on the log of the accident that ended the campaign, and it is reported here as a measurement because that is what it is: the one dynamic property of this airframe that was found only by flying it.

The aircraft was hovering in the T configuration, some three seconds after reconfiguring into it, when a roll oscillation at 3 Hz appeared. It grew slowly, from about 2° to 5° peak to peak over sixteen seconds, and then broke out — 5.9°, 12°, 16° within three seconds — in hover, before any trajectory in T had started (Fig. 6.3). The obvious explanation was ruled out on the log itself: the joint

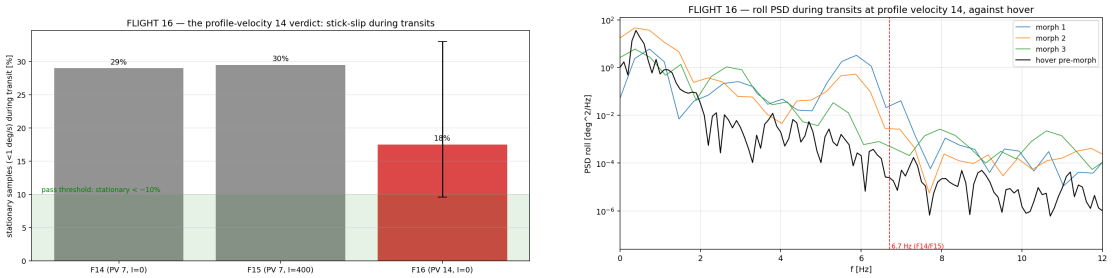


Figure 6.2: The profile-velocity experiment, flown on one flight. Left: the fraction of transit samples with a joint rate below $1\,^\circ\,\mathrm{s}^{-1}$ — the stick–slip indicator — with the commanded profile velocity doubled from $9.6\,^\circ\,\mathrm{s}^{-1}$ to $19.2\,^\circ\,\mathrm{s}^{-1}$. Right: the roll spectrum during the transits, against hover. Doubling the commanded rate cleans the holds, exactly as a low-velocity friction explanation predicts, and leaves the spectral peak where it was. The frequency does not follow the actuation, so it is not stick–slip but a property of the structure: the arm-compliance mode identified in Sec. 6.8.

feedback was alive at 100 Hz throughout, the arms had reached T to within 4.2° of target, and the servo serial port died only at impact. The arms were neither stuck nor unread.

What the spectra show instead is a single compliant mode shared by the arm, the servo and the structure (Fig. 6.4). The four arm-angle ripples and the airframe roll peak at the same frequency, which is the signature of one mode rather than of four independent servo disturbances, and that frequency is absent from every phase flown in the X configuration on both predictive flights. Fitted against the rigid-arm model the controllers carry, the real plant returns about three times the loop gain the model believes at 3 Hz in T: a resonant amplification, not a transport delay. Three times the gain is the entire stability margin the predictive design had, which is why the aircraft was lost in the configuration where the mode lives, and why nineteen flights of trajectory work in X had given no warning of it.

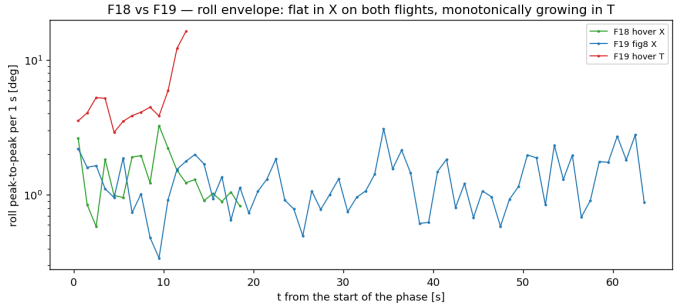


Figure 6.3: Roll envelope through the phases of the last flight of the campaign. The figure-of-eight in X is quiet; after the reconfiguration to T a 3 Hz oscillation grows through the settle window and then breaks out, in hover, before trajectory streaming in T had started.

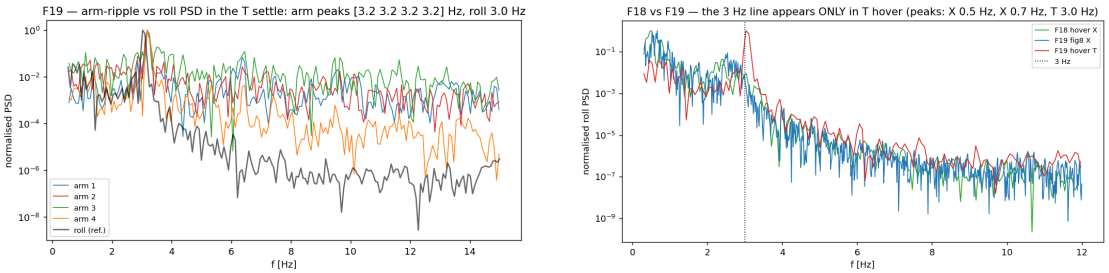


Figure 6.4: Left: the spectra of the four arm-angle ripples during the settle in T, against the roll spectrum of the same window. The arms ripple at the frequency at which the airframe rolls — one compliant mode shared by the arm, the servo and the structure, not four independent servo disturbances. Right: the roll spectrum by flight phase, for the two predictive flights. The line is absent from every phase flown in X and appears only in T hover.

6.9 Aerodynamic and energetic identification in flight

Three further quantities were identified from flight data and are used elsewhere in the thesis.

The velocity-moment coupling. A slow oscillation near 0.3 Hz observed in hover is not a constant disturbance but a plant mode, and the term that drives it is a moment proportional to translational velocity and *perpendicular* to it. The antisymmetric coefficient is approximately $-0.20\,\mathrm{N\,m\,s\,m^{-1}}$ and is consistent across three flights and two controllers, with individual estimates of -0.199, -0.252, -0.193 and -0.200. Its antisymmetric structure is what makes it identifiable and also what makes it irreproducible with the rotor-drag coefficient offered by the simulator, which can only generate a diagonal contribution; the term is therefore injected explicitly (Chapter 10).

The energetic cost of configuration. Comparing steady hover in X and O within a single flight gives 631.1 W against 789.4 W at the battery, an increase of 158.3 W or 25.1 %, against a modelled thrust increase of 19.6 % and a collective increase of 14.0 %; endurance in O is therefore approximately 0.80 of that in X. The gap between the 25.1 % energetic penalty and the 19.6 % thrust penalty is the aerodynamic loss from rotor overlap and wash, and it is the quantitative basis for treating configuration selection as an energy problem (Chapter 8). Two caveats attach: it was measured with the earlier propellers and at the earlier O opening angle, and the O window is 4.8 s against 12 s for X.

Fault severity. The rotor-fault profile used throughout the simulation campaigns models a 39 % thrust loss. When the physically damaged propeller recovered from Flight 9 was refitted and flown deliberately, the measured thrust deficit was 22 % to 28 % — roughly 0.65 of the modelled severity. The simulated fault is therefore *conservative* with respect to the physical one, which strengthens rather than weakens the fault-tolerance results of Chapter 12, but the discrepancy must be stated when the two are compared.

6.10 What could not be identified

The credibility of the preceding sections depends on being equally explicit about their limits.

Rotor speed is not measured in flight. This is the central limitation of the chapter. Bidirectional DShot, which made the bench identification possible, proved unreliable on the flight vehicle and was disabled permanently, and the alternative serial telemetry line is soldered but mute. Consequently every in-flight rotor speed here is *inferred* by inverting the bench throttle-to-speed map at the commanded collective, so what Sec. 6.4 identifies is not k_T in isolation but its composition with the ESC map at a fixed operating point — and the residual that varies with collective, reported in the same section, is the visible consequence, not being attributable to one factor or the other from these data. The single experiment that would separate them is a flight with working rotor-speed telemetry.

Three further gaps. The torque coefficient remains an interval, only its upper bound being measured; the planned re-measurement of the thrust curve with the current propellers was abandoned, the available fixture being inadequate for the loads, so the bench-versus-flight discrepancy is documented but not explained; and the amplitude dependence of the actuator lag is carried by neither the predictive model nor the simulation plant, both of which hold a single first-order constant against the factor 1.46 of Table 6.2.

The compliance mode is not in the simulator. The arm-compliance mode of Sec. 6.8 is the measured root cause of a real loss of control, and the simulation plant does not contain it: the simulated arms are rigid, so no amount of simulation on the current plant can reproduce or predict that failure.

6.11 Consolidated parameters

Table 6.4 lists the identified constants used throughout the remainder of the thesis, with the provenance of each. Appendix B maps them to the corresponding symbols in the implementation.

Table 6.4: Parameters adopted for the modelling, control and simulation work of the following chapters.

Parameter	Value	Provenance
Thrust coefficient k_T	$4.1547 \times 10^{-5}\,\mathrm{N\,s^2}$	flight, hover
Torque ratio k_Q/k_T	0.024 m (0.021 to 0.027)	power bound
Actuator lag τ_{act}	0.08 s (small signal)	bench + flight
Speed envelope	$120\,\mathrm{rad\,s^{-1}}$ to $779.8\,\mathrm{rad\,s^{-1}}$	bench
Hover speed	$483.7\,\mathrm{rad\,s^{-1}}$	derived
Hover collective	0.416	flight
Max thrust per rotor	25.26 N	derived
Thrust-to-weight ratio	2.60	derived
Mass m	3.964 kg	CAD
Inertia J (X) [$\mathrm{kg\,m^2}$]	0.1217, 0.1198, 0.2030	CAD
Transport delay	30 ms RTT	measured
Servo stall torque	1.6 N m	datasheet

Three entries carry qualifications that should travel with them: the torque ratio is an interval and not a point estimate; the actuator lag is valid in the small-signal regime only and is supplied at launch rather than by the parameter file; and the thrust coefficient is a composition with the ESC map, anchored at a hover collective of 0.415 and accurate to 10 % away from that operating point. One reconciliation belongs with the table as well: the hover speed is quoted as $\sqrt{T_{\mathrm{hover}}/4k_T} = 483.7\,\mathrm{rad\,s^{-1}}$, consistent by construction with the adopted k_T, whereas inverting the measured throttle-to-speed map at the same collective returns $482.87\,\mathrm{rad\,s^{-1}}$. The two differ by 0.18 %, the residual of the map fit at the hover operating point; the difference is immaterial at the accuracy claimed for k_T, and the derived entry is the one used downstream.

Chapter 7

Sliding-mode flight control

Building on the variable-geometry model of Chapter 5, this chapter presents the sliding-mode flight controller of the morphing quadrotor: the attitude law (Sec. 7.1), the adaptive moment-bias estimator (Sec. 7.2), the altitude law with anti-windup and take-off feed-forward (Sec. 7.3), the dynamic moment caps that reflect the authority available in each configuration (Sec. 7.4), the lead compensator that buys back the phase lost to the actuator pole (Sec. 7.5), the arm-motion detector (Sec. 7.6), the discretisation of the loop (Sec. 7.7) and the five start-up invariants that audit the tuning against the physics (Sec. 7.8).

Figure 7.1 shows how those pieces sit together.

A remark on provenance is due once, here. The gains and thresholds quoted are the deployed flight values, the outcome of a systematic re-tuning campaign against the simulation twin and of two amendments driven by the adoption of the CAD-anchored mass model and by flight evidence; several supersede earlier values that flew — and in one case crashed — the vehicle. Wherever a superseded value shaped the design, what it was and why it was abandoned is recorded.

7.1 Attitude controller: sliding mode

The inner loop regulates the Euler angles (ϕ, θ, ψ) with a first-order sliding-mode controller (SMC) [32, 33], a law with an established record on quadrotor attitude [34], with integral action, super-imposed on a pure derivative damping term. The output moment $\boldsymbol{M} = (M_\phi, M_\theta, M_\psi)^\top$ is combined with the thrust of Sec. 7.3 to form $\boldsymbol{\tau}$ of (5.24).

With the attitude error

$$\boldsymbol{e} \;=\; \Big(\phi - \phi_{\text{des}},\; \theta - \theta_{\text{des}},\; \text{wrap}_\pi(\psi - \psi_{\text{ref}})\Big)^\top, \tag{7.1}$$

and the error rate $\dot{\boldsymbol{e}} = (\dot\phi - \dot\phi_{\text{des}},\; \dot\theta - \dot\theta_{\text{des}},\; \dot\psi)^\top$, where the rates are the filtered

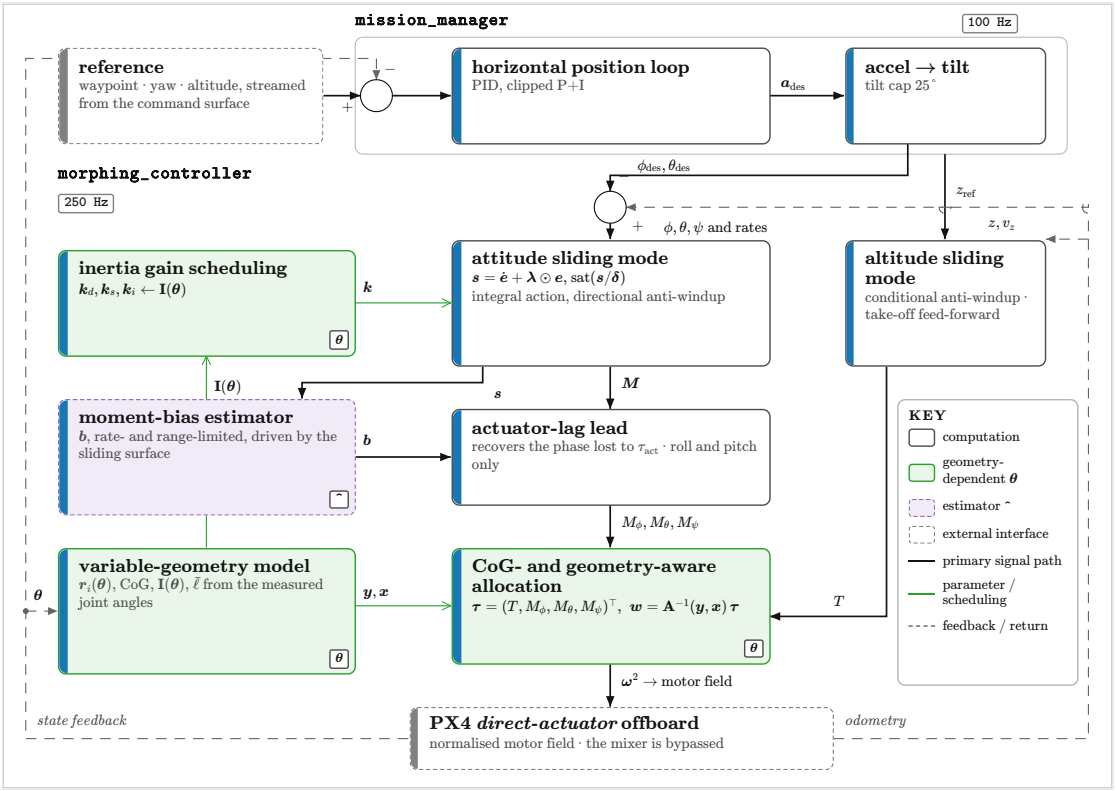


Figure 7.1: The sliding-mode stack, from the mission reference to the four rotor speeds. The outer loop runs in `mission_manager` at 100 Hz and produces a tilt and thrust reference; the inner loop runs in `morphing_controller` at 250 Hz and closes attitude and altitude, conditions the moment and allocates it. In the key, a $\boldsymbol{\theta}$ badge marks the blocks recomputed from the instantaneous geometry — the gain scheduling of Sec. 7.1.1, the variable-geometry model of Chapter 5 and the CoG-aware allocation — and a hat badge the adaptive moment-bias estimator of Sec. 7.2. Actuation bypasses the autopilot mixer entirely: the node publishes its own allocated motor commands in PX4's direct-actuator offboard mode.

derivatives of (5.6) and $\dot{\phi}_{\text{des}}, \dot{\theta}_{\text{des}}$ the optional rate feed-forwards (zero in hover, nonzero while tracking a streamed trajectory), the per-axis sliding surface is the canonical linear combination

$$\boldsymbol{s} \;=\; \dot{\boldsymbol{e}} + \boldsymbol{\lambda} \odot \boldsymbol{e}, \tag{7.2}$$

with $\boldsymbol{\lambda}$ the vector of surface slopes and $\odot$ the Hadamard product; in the sliding phase ($\boldsymbol{s} = \boldsymbol{0}$) the error obeys $\dot{\boldsymbol{e}} = -\boldsymbol{\lambda} \odot \boldsymbol{e}$. The control law, augmented with damping and integral action, is

$$\boldsymbol{M} \;=\; -\boldsymbol{k}_d \odot \dot{\boldsymbol{e}} \;-\; \boldsymbol{k}_s \odot \operatorname{sat}\!\left(\frac{\boldsymbol{s}}{\boldsymbol{\delta}}\right) \;-\; \boldsymbol{k}_i \odot \boldsymbol{I}_e \;+\; \boldsymbol{b}, \tag{7.3}$$

where $\boldsymbol{k}_d, \boldsymbol{k}_s, \boldsymbol{k}_i$ are the derivative, sliding and integral gains scheduled on the

instantaneous inertia (Sec. 7.1.1); $\boldsymbol{\delta}$ is the boundary-layer thickness, inside which the saturation is linear and chattering is suppressed; $\boldsymbol{I}_e$ is the guarded error integral (Sec. 7.1.2); and $\boldsymbol{b}$ is the moment bias of Sec. 7.2.

7.1.1 Inertia-based gain scheduling

At every iteration the gains are recomputed from the current diagonal inertia $\mathbf{I}(\boldsymbol{\theta}) = \mathrm{diag}(I_{xx}, I_{yy}, I_{zz})$ (Sec. 5.3.4):

$$\boldsymbol{k}_s = \left(I_{xx}, I_{yy}, I_{zz}\right) \odot \boldsymbol{\omega}_n^2, \tag{7.4}$$

$$\boldsymbol{k}_d = 2\,\boldsymbol{\zeta} \odot \left(I_{xx}, I_{yy}, I_{zz}\right) \odot \boldsymbol{\omega}_n, \tag{7.5}$$

$$\boldsymbol{k}_i = \rho \left(I_{xx}, I_{yy}, I_{zz}\right) \odot \boldsymbol{\omega}_n^2, \tag{7.6}$$

with $\boldsymbol{\omega}_n$ the per-axis target natural frequency, $\boldsymbol{\zeta}$ the damping ratio and $\rho = 0.5$ the ratio between integral bandwidth and $\boldsymbol{\omega}_n$. The ratio was raised from 0.25 during the anti-windup redesign: at constant declared trim authority in newton-metres (Sec. 7.1.2), doubling ρ doubles k_i and halves the state clamp, so the trim ceiling is unchanged while the charging time is halved, at a negligible cost in phase (the integral knee moves from 0.030 to 0.060 Hz against a crossover near 1 Hz).

Figure 7.2 evaluates the three schedules over the whole morph envelope.

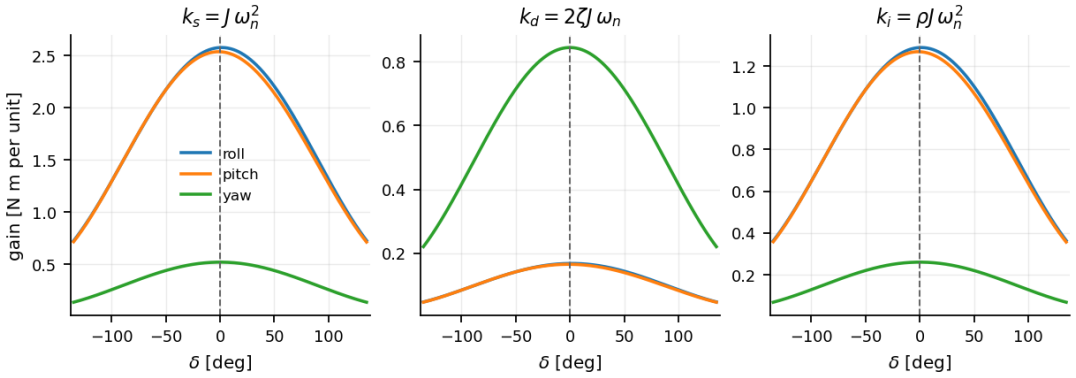


Figure 7.2: The scheduled gains (7.4)–(7.6) across the reachable envelope, for a uniform joint offset δ on all four arms, computed from the same CAD-calibrated inertia the flight software consumes. Every gain is proportional to the instantaneous inertia, so all three fall together as the airframe closes: at the O opening the roll and pitch gains are about half their X values. The multiplier is what makes the *ideal* sliding-phase dynamics configuration-independent; what it cannot make configuration-independent is the authority available to realise the demanded moment (Fig. 5.4), which motivates the dynamic caps described in Sec. 7.4.

The inertial multiplier renders the *ideal* sliding-phase dynamics independent of the configuration. However, with the deployed tuning, the boundary layer is

intentionally wider than the commanded envelope: the saturation knee δ/λ is set at 26° for roll and pitch against a 25° tilt cap, while the largest hover excursion measured on hardware was only 4.1°. Consequently, the vehicle operates within the boundary layer at all times, and the relevant closed loop is effectively linear, characterised by stiffness $K = k_s\lambda/\delta$ and damping $C = k_d + k_s/\delta$. In configuration X, this yields an effective natural frequency of $6.8\,\mathrm{rad\,s^{-1}}$ and a damping ratio of 0.72 on roll (Table 7.2). Thus, the low nominal value $\zeta = 0.15$ does not imply a lightly damped design, as most of the damping arises from the k_s/δ term. Figure 7.3 plots these three relevant angles on the same axis.

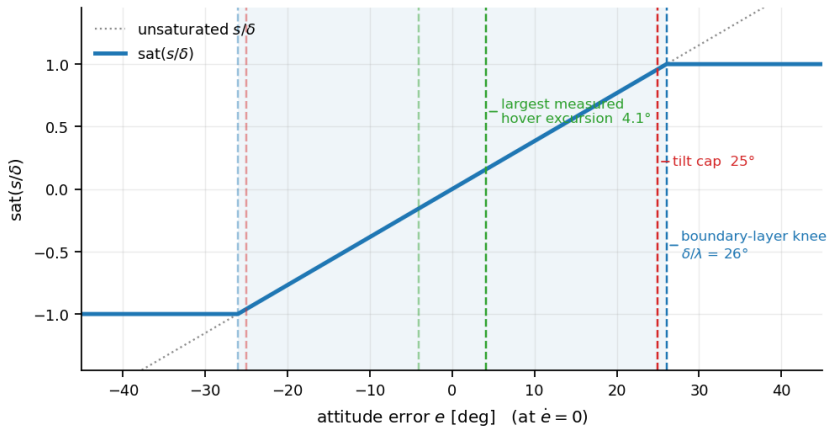


Figure 7.3: The saturation $\mathrm{sat}(s/\delta)$ of (7.3) on the roll axis, evaluated along $\dot{e} = 0$ so that $s = \lambda e$. The knee sits at $\delta/\lambda = 26°$, *outside* the 25° tilt cap the mission side is allowed to command, and the largest hover excursion ever measured on hardware is 4.1°. The vehicle therefore never leaves the boundary layer in normal operation: the relevant closed loop is the linear one, with stiffness $K = k_s\lambda/\delta$ and damping $C = k_d + k_s/\delta$, and the discontinuous switching that gives the method its name is never exercised.

The gains are constant in the error amplitude; an earlier error-driven boost schedule was removed because, at the already aggressive base gains, it pushed the loop past its stability limit and excited a self-sustaining oscillation as the error grew.

Table 7.1: Nominal parameters of the attitude SMC controller. The three columns correspond to the axes (roll, pitch, yaw).

Parameter	Roll	Pitch	Yaw
Natural frequency ω_n $[\mathrm{rad\,s^{-1}}]$	4.6	4.6	1.6
Damping ratio ζ	0.15	0.15	1.3
Surface slope λ $[\mathrm{s^{-1}}]$	5.5	5.5	4.0
Boundary-layer width δ $[\mathrm{rad\,s^{-1}}]$	2.5	2.5	1.2

Table 7.2: Gains derived from Table 7.1 in the X configuration, evaluated at the CAD-calibrated inertia of Table 3.2. K and C are the stiffness and damping of the linear loop inside the boundary layer.

Quantity	Roll	Pitch	Yaw
Sliding gain k_s [N m]	2.575	2.535	0.520
Damping gain k_d [N m s]	0.168	0.165	0.845
Integral gain k_i [N m rad^{-1} s^{-1}]	1.288	1.268	0.260
sat$(\cdot)$ knee δ/λ [°]	26.0	26.0	17.2
Stiffness $K = k_s\lambda/\delta$ [N m rad^{-1}]	5.67	5.58	1.73
Damping $C = k_d + k_s/\delta$ [N m s rad^{-1}]	1.20	1.18	1.28

Table 7.1 is the consolidated outcome of the 21-run re-tuning campaign of 2026-07-29, amended twice. The retuning of 2026-07-31, which accompanied the adoption of the CAD-anchored inertia tensor, lowered the roll/pitch ω_n from 4.8 to 4.6, so that k_s keeps tracking the *available moment authority* rather than the larger inertia; the yaw tightening of 2026-08-01 raised $\omega_{n,\psi}$ from 1.35 to 1.6 — lifting K_ψ from 1.23 to 1.73 N m rad^{-1} — after the take-off yaw cap, not the physics, was found to be what pinned the yaw bandwidth (Sec. 7.4.1), with the tracked yaw error dropping from 0.40 to 0.30° at the 95th percentile in the A/B validation. Earlier gain sets — $\boldsymbol{\omega}_n = (2.2, 2.2, 2.5)$ with $\delta = 0.35$ — are superseded, and not merely retuned: with $\lambda = 2$ that layer places the sat$(\cdot)$ knee at 10.0°, *inside* the commanded tilt envelope, so the restoring term saturates during normal manoeuvring and the law degenerates into a relay. That knee is the signature of the attitude-divergence loss of the second flight session (flight log 228), and it recurs as a static check in Sec. 7.8.

7.1.2 Integral action with directional anti-windup

The error integral is protected against windup and spurious accumulation by a set of guards, three of them ported from the PX4 rate controller [35]:

$$\boldsymbol{I}_e^{(k)} \;=\; \mathrm{clip}\Big(\boldsymbol{I}_e^{(k-1)} + \Delta t\, \boldsymbol{i}_f^{(k)} \odot \boldsymbol{e}^{(k)} \odot \boldsymbol{m}_{\mathrm{aw}}^{(k)} ,\; -\boldsymbol{L}_I(\boldsymbol{\theta}),\; +\boldsymbol{L}_I(\boldsymbol{\theta}) \Big). \tag{7.7}$$

The *directional anti-windup* mask $\boldsymbol{m}_{\mathrm{aw}}$ is built from $d_i = \mathrm{sign}(\tau_{\mathrm{req},i} - \tau_{\mathrm{ach},i})$, the direction in which the allocation failed to deliver the requested moment on axis i at the previous step (Sec. 7.7.1; a residual below 10^{-3} N m counts as delivered, $d_i = 0$): it is zero when d_i and e_i have opposite signs and one otherwise, so that accumulating error which would push the command further into an unreachable direction is frozen while the direction that unloads the saturation stays free. Only the futile direction is blocked — the one-sided property that the altitude anti-windup of Sec. 7.3.1 does not have. The *rate-error attenuation* $i_{f,\bullet} = \max(0,\, 1 - (\dot{e}_\bullet/\dot{e}_{\mathrm{ref}})^2)$,

with $\dot{e}_{\mathrm{ref}} = 400\,^{\circ}\,\mathrm{s}^{-1}$, is applied to the *rate* error exactly as in PX4 and not to the attitude error, where it would switch the integrator off precisely when the stationary trim deficit it exists to serve is large; at this vehicle's rate envelope it is deliberately near-inert. A *landed gate* stops integration on the ground, its primary source the PX4 land detector and its fallback the mission state — in practice the fallback, since the topic proved to update only on state changes. A *finiteness guard* discards a non-finite update candidate rather than zeroing it, so that a single NaN cannot poison the integral permanently. Finally, the *clamp* is not a constant in rad s but is derived per configuration from a declared DC authority in newton-metres,

$$L_{I,\bullet}(\boldsymbol{\theta}) \;=\; \frac{I_{\mathrm{auth},\bullet}}{k_{i,\bullet}(\boldsymbol{\theta})}, \qquad \boldsymbol{I}_{\mathrm{auth}} = (1.0,\, 1.0,\, 0.4)^{\top}\,\mathrm{N\,m}, \tag{7.8}$$

which evaluates to $(0.777,\, 0.789,\, 1.539)\,\mathrm{rad\,s}$ in X. The point of the derivation is that the trim authority $k_i L_I$ stays constant in newton-metres as the geometry changes: with the old fixed clamp it *collapsed* exactly where the asymmetry is largest — 0.124 N m in H against 0.352 N m in X.

This design supersedes an earlier integrator gated by an error deadband of $(0.15,\, 0.15,\, 1.50)^{\top}\,\mathrm{rad}$ and constrained by a fixed limit. The transition is driven by flight measurements rather than subjective preference. During the first two hovering flights (both characterised as degraded, with drifts of 21 and 14 m), the vehicle required 0.593 N m of constant roll trim. The total available DC authority was limited to 0.576 N m, yet the integral term contributed only 0.040 N m (11 % of its own ceiling) because the error remained outside the deadband for 70 % of the flight duration. Consequently, the integrator was inactive precisely when it was most needed. An optional freeze function, active while the arms are in motion (Sec. 7.6), is retained in the code but disabled in the current flight configuration.

7.2 Moment-bias estimator

To compensate quasi-static disturbances not captured by the model — primarily an unknown mobile payload and residual gravity moments due to CoG mismatches — a moment-bias vector $\boldsymbol{b}$ is maintained and summed into (7.3). It is updated adaptively from the sliding surface through a mechanism that combines a lever-dependent rate, a quiet-zone release, a low-passed drive and a leak toward zero. With the normalised drive $\boldsymbol{v} = \mathrm{sat}(\boldsymbol{s}/\boldsymbol{\delta}_b)$, $\boldsymbol{\delta}_b = (1.07,\, 1.07,\, 0.43)^{\top}$, and the *quiet-zone* mask $\boldsymbol{m}_q = [\,|\boldsymbol{e}| < \boldsymbol{d}_{b,e}\,] \odot [\,|\dot{\boldsymbol{e}}| < \boldsymbol{d}_{b,\dot{e}}\,]$ — thresholds $\boldsymbol{d}_{b,e} = (0.6,\, 0.6,\, 0.5)^{\top\,\circ}$ and $\boldsymbol{d}_{b,\dot{e}} = (1.5,\, 1.5,\, 10)^{\top\,\circ}\,\mathrm{s}^{-1}$ — which zeroes the drive where the system is effectively at rest, the masked drive is low-passed,

$$\boldsymbol{v}_f^{(k)} \;=\; \boldsymbol{v}_f^{(k-1)} + \alpha_b \left(\boldsymbol{v}^{(k)} \odot (1 - \boldsymbol{m}_q^{(k)}) - \boldsymbol{v}_f^{(k-1)} \right), \qquad \alpha_b = \Delta t/\tau_b, \quad \tau_b = 2\,\mathrm{s}, \tag{7.9}$$

and integrated with a leak and a run-time saturation,

$$\boldsymbol{b}^{(k)} \;=\; \mathrm{clip}\Big(\big(1-\lambda_b\,\Delta t\big)\,\boldsymbol{b}^{(k-1)} \;-\; \big(\boldsymbol{r}_b \odot \boldsymbol{\sigma}\big)\,\Delta t\;\boldsymbol{v}_f^{(k)},\; -\boldsymbol{L}_{\mathrm{eff}}(\boldsymbol{\theta}),\; +\boldsymbol{L}_{\mathrm{eff}}(\boldsymbol{\theta})\Big), \tag{7.10}$$

$$\boldsymbol{L}_{\mathrm{eff}}(\boldsymbol{\theta}) \;=\; \min\Big(\boldsymbol{L}_b \odot \boldsymbol{\sigma}(\boldsymbol{\theta}),\; 0.5\,\boldsymbol{k}_s(\boldsymbol{\theta})\Big), \qquad \sigma_\bullet = \max\Big(\sigma_{\min},\; \min\big(1,\; (\bar{\ell}_\bullet/\ell_{\mathrm{ref}})^2\big)\Big), \tag{7.11}$$

the lever-axis factor $\boldsymbol{\sigma}$ acting on roll and pitch only (unity on yaw), with $\ell_{\mathrm{ref}} = 0.25\,\mathrm{m}$ and $\sigma_{\min} = 0.1$. The parameters are $\lambda_b = 0.02$ (leak), $\boldsymbol{r}_b = (0.8,\, 0.8,\, 0.25)^\top$ (adaptation rate, the yaw axis slow in keeping with its low control priority) and $\boldsymbol{L}_b = (1.0,\, 1.0,\, 0.18)^\top\,\mathrm{N\,m}$ (declared limit). The roll/pitch rate is a flight-driven choice: at the earlier 0.1 the estimator needed about 7 s to learn a static trim, while the take-off transient lives or dies in the first 2 to 3 seconds.

The declared limits are set just below the design invariant $\boldsymbol{L}_b \le 0.5\,\boldsymbol{k}_s$ ($1.29\,\mathrm{N\,m}$ on roll in configuration X). The roll and pitch limit accommodates the roll trim range of 0.593 to $0.698\,\mathrm{N\,m}$ measured during initial hovering flights with 43 % margin, while in-flight estimates remain well within bounds; for instance, during the first clean hover, the roll bias varied from -0.360 to $0.280\,\mathrm{N\,m}$, representing only 36 % of the limit.

Two structural safeguards, inherent to the design rather than tuning parameters, ensure estimator safety across all configurations. First, the drive signal is low-pass filtered with $\tau_b = 2\,\mathrm{s}$ prior to integration. While the bias estimates a DC moment, integrating the raw surface during manoeuvres lags the disturbance by approximately 90°, introducing negative damping that excites a bistable position-attitude limit cycle near 0.3 Hz. This filter maintains full-rate DC learning while reducing the pump gain in the 0.3 Hz to 1.2 Hz band by roughly a factor of four; without it, the limit cycle was reproduced in 2/2 simulation runs under CPU-jitter stress. Second, the saturation (7.11) is the run-time minimum between $\boldsymbol{L}_b \odot \boldsymbol{\sigma}$ and $0.5\,\boldsymbol{k}_s(\boldsymbol{\theta})$, with k_s scheduled on the inertia of the current shape: folding the arms lowers k_s and the bias ceiling follows it, so a bias able to cancel the restoring term of (7.3) is impossible *by construction* in every configuration and not only in the base one. The ceiling evaluates to $(1.29,\, 1.27,\, 0.26)\,\mathrm{N\,m}$ in X, $(1.13,\, 1.05,\, 0.22)$ in T, $(0.37,\, 1.85,\, 0.23)$ in H and $(0.65,\, 0.65,\, 0.13)$ in configuration O at 100°. The squared dependence of $\boldsymbol{\sigma}$ on the mean levers also reduces both rate and limit along the weak axis, preventing the slow position-attitude oscillation observed in configuration H, where $\bar{\ell}_\phi \approx 0.113\,\mathrm{m}$ contrasts with $\bar{\ell}_\theta \approx 0.348\,\mathrm{m}$ (Table 3.2). While tracking a streamed trajectory, the update is suspended entirely, including both accumulation and leak. Driving the bias from a manoeuvre-dominated surface would lag the disturbance and excite the same limit cycle; consequently, the trim learned during hover is simply held until the trajectory concludes.

Interpreted physically, the bias is a wide-band estimate of the low-frequency and constant components of the unmodelled moment disturbances that would otherwise

drive the integrator into saturation: the surface acts as the error signal, and when it is persistently non-zero in one direction the bias shifts in the opposite one, while the leak guarantees relaxation to zero in the absence of a persistent disturbance.

7.3 Altitude controller: sliding mode with anti-windup

The altitude loop produces the total thrust T (the first component of $\boldsymbol{\tau}$ in (5.24)) by an SMC formulation analogous to the attitude one, augmented with a *reaching law* and an anti-windup integral. All quantities are in NED: z is negative in flight and v_z negative when climbing. With $e_z = z_{\text{ref}} - z$ and $\dot{e}_z = \dot{z}_{\text{ref}} - v_z$, the surface and the law are

$$s_z \;=\; \dot{e}_z + \lambda_z\, e_z, \qquad \lambda_z = 2.0, \tag{7.12}$$

$$T \;=\; T_{\text{hov},b} \;-\; k_{d,z}\,\dot{e}_z \;-\; k_{s,z}\,\text{sat}\Big(s_z/\delta_z\Big) \;-\; k_{r,z}\,s_z \;-\; k_{i,z}\,I_{e_z}, \tag{7.13}$$

with $k_{s,z} = k_{d,z} = 8.0$, $k_{r,z} = 3.0$, $k_{i,z} = 4.0$, $\delta_z = 0.35$ and $T_{\text{hov},b} = T_{\text{hov}}/(\cos\phi\cos\theta + \varepsilon_T)$, where $\varepsilon_T = 10^{-6}$. This term corrects the nominal hover thrust for the tilt of the thrust-disc plane. The $-k_{r,z}s_z$ component injects a contribution proportional to the distance from the surface, thereby shortening the reaching time.

The vertical-velocity feed-forward signal $\dot{z}_{\text{ref}}$ is published by the mission manager alongside the altitude reference. Take-off and waypoint paths generate z_{ref} via a trapezoidal profile, limited to an acceleration of $0.5\,\text{m}\,\text{s}^{-2}$ and a slew rate of $0.5\,\text{m}\,\text{s}^{-1}$, with a braking envelope defined by $\sqrt{2ad}$. This ensures that $\dot{z}_{\text{ref}}$ remains continuous. This continuity is critical: an earlier configuration set $\dot{z}_{\text{ref}} = 0$, causing the loop to track the climb ramp with a steady-state lag predicted by the surface, specifically $\dot{z}_{\text{slew}}/\lambda_z = 0.25\,\text{m}$. The reference profile is held until PX4 reports the vehicle as armed, so that the reference cannot run away while the vehicle is still on the ground; if the arming feed is degraded, the hold is released after a 3 s timer.

7.3.1 Conditional anti-windup integrator

The integral I_{e_z} is updated only when (7.13), evaluated in its unsaturated form T_{u} with the previous integral state, falls inside the physical actuator range $[T_{\min}, T_{\max}]$, with $T_{\min} = 0.3\,T_{\text{hov}}$ and $T_{\max} = 200\,\text{N}$; otherwise it is held. When it is updated it is clamped at $L_{I_z} = 10.0$; T_{u} is then recomputed with the new state and clipped to the same range. Two properties distinguish this guard from the attitude integrator's (Sec. 7.1.2): it freezes both directions rather than only the futile one, and it tests against the nominal actuator range rather than the limit that actually bites first —

a range wide enough that in practice the guard rarely engages, leaving the clamp as the effective protection.

During take-off, T is additionally lower-bounded by $T_{\min}^{\mathrm{TO}} = T_{\min}$ to guarantee sufficient lift-off thrust, and upper-bounded by a climb-acceleration soft-start limit:

$$T \;\le\; \max\!\Big(\hat{T}_{\mathrm{hov}} + M\,a_{\mathrm{climb}},\; T_{\min}^{\mathrm{TO}}\Big), \qquad a_{\mathrm{climb}} = 2\,\mathrm{m\,s^{-2}}, \tag{7.14}$$

where $\hat{T}_{\mathrm{hov}} = T_{\mathrm{hov}} - k_{i,z}\,I_{e_z}$ denotes the integrator's real-time estimate of hover thrust. This cap restricts the climb acceleration to prevent the airframe from experiencing a sudden jerk due to the large altitude error present at arming, while remaining agnostic to payload mass: as only the integrator's internal estimate is used in the calculation, the thrust ceiling automatically adjusts upward to accommodate unmodelled payloads, rather than pinning thrust below the required hover level for a loaded vehicle.

7.4 Dynamic moment caps

Before $\boldsymbol{\tau}$ is passed to the allocator (Sec. 5.5) its moment components are clipped by limits that reflect the authority physically available in the current configuration. On roll and pitch that authority is the differential thrust the four rotors can deliver times the mean lever on the axis,

$$M_{\bullet}^{\mathrm{cap}} \;=\; \min\!\Big(\,\sigma_M \cdot \Delta T_{\max} \cdot \bar{\ell}_{\bullet},\; M_{\bullet}^{\mathrm{hard}}\Big), \qquad \bullet \in \{\phi, \theta\}, \tag{7.15}$$

with $\sigma_M = 1.0$, $\Delta T_{\max} = 8\,k_T\,\omega_{\max}^2$ and the geometry-independent hard cap $M^{\mathrm{hard}} = 60\,\mathrm{N\,m}$; yaw, governed by reaction torque rather than a lever arm, carries the constant cap $M_{\psi}^{\mathrm{cap}} = 1.0\,\mathrm{N\,m}$.

7.4.1 Take-off caps

During take-off *only* — the `TAKING_OFF` mission state — the caps are tightened to $(M_{\phi}^{\mathrm{TO}},\, M_{\theta}^{\mathrm{TO}},\, M_{\psi}^{\mathrm{TO}}) = (3.4,\, 3.4,\, 0.7)\,\mathrm{N\,m}$, the effective per-axis cap becoming $M^{\mathrm{cap},*} = \min(M^{\mathrm{cap}}, M^{\mathrm{TO}})$ and the commanded moment being clipped symmetrically to it. The restriction of the regime is deliberate and has a history: an earlier version applied the conservative set also in steady hover with no active waypoint, which leaked the take-off yaw ceiling into essentially the whole flight — on the log-228 loss the yaw channel sat pinned at the then 0.2 N m cap for 72 % of the flight and, with $k_{s,\psi}$ at 1.8 times the cap, the yaw law was a bang-bang relay. The current values keep the take-off roll/pitch ceiling at about a third of the achievable moment ($M_{\mathrm{avail}} = 9.84\,\mathrm{N\,m}$ per axis in X) and the yaw ceiling at 83 % of the achievable yaw authority (0.846 N m) — the margin that allows $\omega_{n,\psi} = 1.6$ to clear the relay guard of Sec. 7.8.

7.5 Actuator-lag lead compensation

The rotor-speed loop behaves as a first-order lag, and the margin analysis of this stack carries its pole conservatively at $\tau = 0.15\,\mathrm{s}$, the upper end of the bench-identified amplitude dependence (Sec. 6.6). At the $6\,\mathrm{rad\,s^{-1}}$ to $8\,\mathrm{rad\,s^{-1}}$ bandwidth of the stiffness loop that pole costs 42° to 50° of phase, and the closed-form margin map of Sec. 7.7.2 put the tuning that rang at 1.2 Hz in simulation at only 3.4° of phase margin. A first-order lead

$$H_{\mathrm{lead}}(s) \;=\; \frac{1 + t_c\, s}{1 + t_f\, s}, \qquad t_c = 0.12\,\mathrm{s}\ \text{(roll, pitch)}, \quad t_f = 0.025\,\mathrm{s}, \tag{7.16}$$

is therefore applied to the SMC moment *before* the caps of Sec. 7.4, ensuring that physical ceilings remain downstream of the compensation. The filter inverts the low-frequency portion of the motor pole, providing approximately 30° of phase lead at $7\,\mathrm{rad\,s^{-1}}$ at the cost of a high-frequency gain ratio $t_c/t_f = 4.8$. For yaw, $t_c = t_f$ is enforced such that $H_{\mathrm{lead},\psi} = 1$ exactly, as non-neutral yaw lead characteristics have not been validated. Implemented in state-space form via $\dot{x} = (M - x)/t_f$ and $y = x + t_c(M - x)/t_f$, the filter operates with a normalised step size $\Delta t/t_f = 0.16$. This configuration remains stable under forward Euler integration and maintains unit DC gain, thereby leaving the static allocation unchanged.

The compensator is active on hardware as well as in simulation — an explicit deployment decision, the risk that the real ESC is not first order having been declared and accepted — and the first clean hover confirmed the choice: the feared 1.2 Hz ring was absent, with the 0.9 Hz to 1.5 Hz band carrying 4.9 % of the roll power at 0.34° rms (one flight; Chapter 11). It is the one block of the SMC chain promoted to hardware on the strength of a flight measurement.

7.6 Arm-motion detector

An arms_moving flag, utilised to gate the integrator (Sec. 7.1.2) and bias adaptation, is derived by low-pass filtering the measured joint angular velocities:

$$\boldsymbol{\omega}_{a,f}^{(k)} = \alpha_a\, \boldsymbol{\omega}_a^{(k)} + (1 - \alpha_a)\, \boldsymbol{\omega}_{a,f}^{(k-1)}, \quad \text{with } \alpha_a = 0.225.$$

The flag is asserted when $\max_i |\omega_{a,f,i}|$ exceeds the threshold $\omega_a^{\mathrm{th}} = 0.15\,\mathrm{rad\,s^{-1}}$, and remains active for a hold time of $\tau_{\mathrm{hold}} = 1\,\mathrm{s}$ after the last detected variation, allowing transients to settle before re-enabling gated estimators. The threshold was increased from an initial value of 0.005 because single encoder tick dither between joint-state messages can produce instantaneous rates in the range 0.096 to $0.15\,\mathrm{rad\,s^{-1}}$, which would otherwise keep the flag permanently true; genuine morphing slews, typically on the order of $1\,\mathrm{rad\,s^{-1}}$, reliably exceed this threshold. In the current flight configuration, this gate is globally disabled, allowing continuous

adaptation of both the integrator and bias. The detector remains implemented to permit re-enabling the original gating behaviour during morphing-intensive testing phases.

7.7 Discretisation and control loop

The controller runs in discrete time at $f_{\text{ctrl}} = 250\,\text{Hz}$, $\Delta t = 0.004\,\text{s}$, all integrators and low-pass filters implemented with explicit Euler, which is adequate for the dynamics and gains used. One rate deserves care: the Euler-rate low-pass of (5.6) does *not* run in this loop but in the PX4 odometry callback at 100 Hz, so its time constant for a given coefficient is 2.5 times what a 250 Hz reading would suggest — the distinction that matters when sizing the rate-feedback lag (Sec. 5.2.1). The loop rate is a runtime parameter, so it can be lowered when the transport to PX4 cannot sustain 250 Hz, every Δt-scaled term adapting automatically.

7.7.1 Per-iteration sequence

At every step the controller reads the PX4 and joint sensors; converts the quaternion to Euler angles (5.2)–(5.4) and computes the filtered Euler rates (5.5)–(5.6); reconstructs the instantaneous geometry — mass points, CoG, inertia, lever arms and mean levers (5.9)–(5.19); forms the attitude error (7.1) and surface (7.2); schedules the gains (7.4)–(7.6) and updates integrator (7.7) and moment bias (7.10); evaluates the attitude moments (7.3) and lead-compensates them (7.16) upstream of the caps; evaluates the altitude thrust (7.13)–(7.14); applies the dynamic caps (7.15); allocates (5.26) and converts to rotor speeds and to the PX4 command (5.27)–(5.23); computes the allocation residual $\boldsymbol{\tau}_{\text{req}} - \boldsymbol{\tau}_{\text{ach}}$ on the *unclipped* moments, whose per-axis sign is the saturation direction consumed by the directional anti-windup at the next step; and publishes the `ActuatorMotors` message in *direct actuator* semantics together with the diagnostic telemetry.

Constructing the residual from already-clipped moments would render it blind to the limiting constraint: in log-228, the yaw channel reached its cap for 72 % of the flight duration, meaning a downstream residual computation would have been identically zero. This would mask saturation while the integrator continued to charge. A safety supervisor therefore wraps the control loop: it aborts operations (muting motors and disarming) upon detection of a tumble ($|\phi| + |\theta| > 75°$ while airborne) or non-finite state estimates, and stands down during detected pilot takeovers or failsafe events. Stale odometry is managed in two stages: if data age exceeds 0.15 s, the last command is held; if staleness persists beyond 0.7 s while engaged, both the offboard heartbeat and motor stream are muted. This prevents a healthy but stale heartbeat from indefinitely masking the condition, allowing the native PX4 offboard-loss failsafe to engage.

7.7.2 Sliding-mode stability considerations

In the sliding phase, combining (7.2) with (7.3) yields the first-order asymptotic error dynamics $\dot{e}_\bullet = -\lambda_\bullet e_\bullet$. This ideal argument, however, describes a regime the deployed controller deliberately never enters: because the boundary layer covers the whole commanded envelope (Sec. 7.1.1), the operating regime is the *linear* one inside the layer, where the law reduces to a second-order stiffness loop with $K = k_s\lambda/\delta$ and $C = k_d + k_s/\delta$. Stability is therefore argued not from a damping-ratio condition — with $\zeta = 0.15$ most of the damping comes from k_s/δ — but from the phase margin of that loop, evaluated in closed form at start-up on

$$L(j\omega) \;=\; \frac{C\,j\omega + K}{J\,(j\omega)^2}\; H_{\text{lead}}(j\omega)\; \frac{e^{-j\omega T_d}}{1 + j\omega\,\tau}, \tag{7.17}$$

with $\tau = 0.15\,\text{s}$ the actuator pole and $T_d = 30\,\text{ms}$ the measured round-trip transport delay (Secs. 6.6 and 6.3.1). The check is fatal below 10° of margin and a warning below 15°; in X, with the K and C of Table 7.2, all three axes clear it. The historical criterion $(k_d + k_s/\delta)\,\tau/J \le 1.5$ constrains only the rate loop, and trusting it alone proved costly: a simulated configuration that passed it ($\delta = 3$, $\lambda = 6$) diverged at about 1 Hz because its stiffness loop stood at −3.8° of phase margin. The boundary layer itself replaces the ideal sliding mode with a *quasi-sliding* one [33]: finite-time reachability is traded for chattering elimination, accepting a steady-state residual of order $\delta_\bullet$, compensated in turn by the integral action and the adaptive bias.

7.8 Start-up gain invariants

The tuning of this chapter is checked at node start-up against five invariants, a fatal violation preventing the controller from arming. The set is the methodological residue of three losses — the log-228 flight, a wind divergence of the simulation twin, and the simulated $\delta 3/\lambda 6$ crash of Sec. 7.7.2 — each translated into a static check so that its class of error cannot recur silently. The *relay guard* requires $k_s \le 0.8\,M^{\text{cap}}$ in every cap regime, steady and take-off; above it the restoring term saturates inside its own authority and the law degenerates into a relay, as it did on the log-228 set where $k_{s,\psi}$ was 1.8 times the take-off yaw cap. The *authority guard* requires $k_s \le 0.8\,M_{\text{avail}}$ on the idle-derated achievable moment ((9.84, 9.84, 0.85) N m in X), the yaw check being only a warning while k_Q remains unmeasured (Sec. 6.5). The *envelope guard* requires the sat(·) knee δ/λ to lie outside the commanded envelope — 25° on roll and pitch, 15° on yaw — against the 10.0° of the log-228 set. The *bias guard* requires $L_b \le 0.5\,k_s$, fatal at the base shape and a warning elsewhere, since the run-time clamp (7.11) enforces this bound across all configurations regardless. The fifth check is a closed-form *phase-margin* verification applied to (7.17).

The correct start-up signature on hardware consists of zero invariant violations,

whether fatal or warning-level. The deployed gains yield zero fatal errors and zero warnings in configuration X; folded shapes trigger only non-fatal warnings (one in H, three in O at 100°). Furthermore, the start-up path rejects all environment-variable tuning overrides on hardware: hooks intended for simulation A/B studies are restricted to the simulator, ensuring that the flight configuration exactly matches the source constants and remains fully audited by the invariants.

Flight-validation status. The stack described in this chapter is the one flown on the physical vehicle, and its central claims are hardware-backed rather than simulated (Chapters 11 and 12). The consolidated tuning and the lead compensator were promoted to hardware on the first clean hover flight: 101 s of continuous hovering at 2.02 m with a maximum drift of 0.62 m, a tilt median of 1.17° and 95th percentile of 2.72°, and no trace of the feared 1.2 Hz ring. The same controller subsequently flew planar trajectory streaming (figure-of-eight RMSE 0.103/0.129/0.137 m in X, T and H; one flight per configuration), three-dimensional helical streaming (0.118/0.221/0.230 m, vertical error below 0.08 m), lap-level repeatability within a single flight (0.107 ± 0.017, 0.189 ± 0.032, 0.195 ± 0.009 m over three laps per shape) and, over three helix runs per shape on two flights, 0.139 ± 0.028, 0.163 ± 0.050 and 0.193 ± 0.035 m; finally, with a physically damaged propeller, it carried the fault-tolerant morphing demonstration of Chapter 12. Two design targets are *not* met on hardware and are reported as open. The 1° tilt gate fails: the quietest 10 s window of the clean hover has a median tilt of 0.89°, four to six times the 0.15° to 0.21° of the simulation twin in still air. And the slow attitude mode at 0.25 Hz to 0.55 Hz is bounded but not eliminated — 49.7 % of the roll power, about 1.1° rms; the mode is spectrally broad and its frequency is not resolved better than ±0.05 Hz to 0.1 Hz.

Chapter 8

Energy-aware geometry optimization

The morphing capability is not only a means to change footprint: the arm configuration is an additional degree of freedom that can be used to improve flight performance. This chapter formulates the online optimisation of the arm geometry that reduces control effort while bounding the loss of control authority, and describes how it integrates, as a decoupled supervisory layer, with either flight-control stack. The treatment starts from the physical motivation (Sec. 8.1), states the optimisation problem (Sec. 8.2), develops the online algorithm that solves it in real time (Sec. 8.3) and details its integration with the control stack (Sec. 8.4). It then closes on what the optimiser has been *measured* to buy — in simulation under the flight-9 rotor fault and on hardware in the three-trial fault campaign of flight 17 (Sec. 8.5) — and on what it does not (Sec. 8.6).

8.1 Motivation: morphing as a degree of freedom

Rotating the four arms moves the rotors in the body plane and therefore changes, simultaneously, the three geometric quantities derived in Chapter 5: the instantaneous centre of mass, the inertia tensor and the centre-of-mass-relative lever arms that populate the allocation matrix [1, 2]. The arm configuration is thus a slow actuator that reshapes how a commanded wrench is distributed over the four rotors.

The opportunity this presents is best described for a *class* of disturbances rather than specific objects. An off-centre payload and a degraded rotor appear identically to the stabilising controller: as persistent, quasi-static moments that require permanent differential thrust across the rotors, causing some motors to run persistently harder than others. This configuration wastes energy, since for a fixed

collective thrust, the most uneven distribution is also the most power-intensive; it simultaneously erodes saturation margins, as the busiest rotor operates closest to its ceiling with minimal residual authority for attitude control. The optimiser developed here is deliberately agnostic to the specific source of the disturbance: it does not observe payload position or individual rotor efficiency, but responds solely to the wrench requested by the controller.

Morphing offers a physical, rather than purely feedback, remedy, and it acts through two distinct mechanisms. The first is *relocation of the rigid centre of mass*, so that it counteracts the inferred offset of the combined one — the mechanism the demand model of Eq. (8.3) encodes. The second, and in practice the dominant one, is the *reshaping of the lever arms* in the allocation matrix: moving the rotors redistributes the same commanded moment over different motors, dissolving the rigid quadrotor's constraint that a roll or pitch moment must be carried by a fixed diagonal pair. The hardware fault campaign of Sec. 8.5 separates the two: the converged geometries move the rigid centre of mass by only 5 mm to 9 mm — worth roughly 0.2 N m through the demand model, less than half of the moment actually returned to the actuation — while the load rebalancing delivered by the new lever arms accounts for the rest.

Two use cases anchor the layer. The 3 kg mobile payload, whose position the controller never observes, remains the *sizing* case of the design; it has, however, never been exercised on the current vehicle — the payload rig was removed from the simulation model during the mass-model consolidation, and payload rejection has not been demonstrated in flight. The *demonstrated* case, in simulation and on hardware, is the degraded rotor of Sec. 8.5. In both cases the optimiser is a *decoupled* supervisory layer sitting on top of the stabilising controller: it observes only the wrench being requested and reshapes the geometry to make that request cheaper to satisfy, which keeps the fast attitude loop and the slow geometry loop cleanly separated and lets the same optimiser serve either control stack unchanged.

8.2 Problem formulation

8.2.1 Decision variables and candidate geometry

The optimiser acts on the four *relative* arm angles $\boldsymbol{\delta} = (\delta_0, \delta_1, \delta_2, \delta_3)^\top$ (in degrees), the joint rotations away from the X-shape reference of Sec. 5.1.3, so that the absolute configuration evaluated by the model is $\boldsymbol{\theta}(\boldsymbol{\delta}) = \boldsymbol{\theta}_{\text{base}} + \boldsymbol{\delta}$, and each joint is bounded by its mechanical workspace,

$$-\delta_{\max} \le \delta_i \le \delta_{\max}, \qquad \delta_{\max} = 135^\circ, \quad i = 0, \ldots, 3. \tag{8.1}$$

The box (8.1) is the mechanical limit of the servo joint, mirrored by the command clamp of the servo driver; it is a per-joint, independent bound and says nothing

about the depth of a reshape as a whole, a point taken up in Sec. 8.2.4. The decision variables are always referred to the X-shape reference and the node publishes an *absolute* target to the arm drivers, so the optimiser is blind to the commanded shape: engaging it from a T, H or O preset overwrites that preset rather than optimising around it, which is why every fault-tolerance experiment in this work is flown from X. Two further scope limits follow from the engagement gate of Sec. 8.4: the optimiser runs only while the mission is HOVERING, and the moment-bias estimator whose output it feeds on is frozen during trajectory tracking, so geometry adaptation and path following have never been exercised simultaneously.

8.2.2 Predicted rotor loads for a candidate geometry

Given a candidate $\boldsymbol{\delta}$ and the quasi-static wrench demand $\boldsymbol{\tau} = (T, M_\phi, M_\theta, M_\psi)^\top$ currently requested by the controller, the optimiser predicts the rotor loads that the candidate geometry would require. The demand is the wrench the controller has *actually commanded* — taken after the moment caps and *including* the learned moment bias of Sec. 7.2 — not the unclipped request. The optimiser evaluates the configuration-dependent motor positions relative to the candidate centre of mass and the resulting allocation matrix $\mathbf{A}(\boldsymbol{\theta})$, and solves the allocation problem for the squared rotor speeds

$$\boldsymbol{w}(\boldsymbol{\delta}) = \left(\omega_0^2,\, \omega_1^2,\, \omega_2^2,\, \omega_3^2\right)^\top = \mathbf{A}\Big(\boldsymbol{\theta}(\boldsymbol{\delta})\Big)^{-1}\, \tilde{\boldsymbol{\tau}}(\boldsymbol{\delta}), \tag{8.2}$$

each component being proportional to a rotor thrust through $F_i = k_T\, \omega_i^2$. The demand $\tilde{\boldsymbol{\tau}}$ is not the raw request but a value corrected for the moment induced by the candidate centre-of-mass shift itself.

Centre-of-mass-shift demand model. Because the optimiser does not identify the cause of the demand, it interprets the requested roll and pitch moments as signatures of an unknown offset between the combined centre of mass and the geometric centre. Shifting the rigid body's centre of mass from its position in the *measured* geometry, $(x_{\mathrm{ref}}, y_{\mathrm{ref}})$, to that of the candidate configuration, $(x_{\mathrm{cog}}, y_{\mathrm{cog}})$, alters the moment generated by collective thrust about the new centre. The lever arms are measured from the centre of mass, so a shift $+\Delta y$ reduces every y_i and, through the roll row $-y_i k_T$, *raises* the roll demand by $T\,\Delta y$, while a shift $+\Delta x$ reduces every x_i and, through the pitch row $+x_i k_T$, *lowers* the pitch demand by $T\,\Delta x$. To first order,

$$\tilde{\boldsymbol{\tau}}(\boldsymbol{\delta}) = \boldsymbol{\tau} + \kappa\, T \begin{bmatrix} 0 \\ +\big(y_{\mathrm{cog}} - y_{\mathrm{ref}}\big) \\ -\big(x_{\mathrm{cog}} - x_{\mathrm{ref}}\big) \\ 0 \end{bmatrix}, \qquad \kappa = 0.595, \tag{8.3}$$

thrust and yaw being unaffected. The response is *antisymmetric* in the two axes, the asymmetry being carried by the term signs and not by κ, which is a strictly positive lumped *magnitude* gain: the geometrically exact coefficient is unity, and κ detunes this first-order estimate to absorb the lever-arm linearisation. One caveat attaches to the deployed value: $\kappa = 0.595$ was tuned *before* the axis antisymmetry was corrected — the pitch sign was inverted in the loop at the time — and has been carried over unchanged, so it should be re-identified against the corrected model; since its role is an under-correcting magnitude detune, the inherited value errs on the side of conservatism rather than instability. Equation (8.3) is the mechanism through which the optimiser «knows» that relocating the rigid centre of mass can cancel a quasi-static moment: a candidate whose centre of mass moves toward the inferred offset reduces $\tilde{\boldsymbol{\tau}}$, and hence the differential thrust in Eq. (8.2).

8.2.3 Objective: motor-thrust spread as an energy proxy

The primary objective is to equalise the rotor loads. With $w_i^+ = \max(w_i, 0)$ the physically realisable squared speeds and $\bar{w}^+$ their mean, the optimiser minimises the normalised load *spread*

$$S(\boldsymbol{\delta}) = \frac{1}{\left(\bar{w}^+\right)^2} \frac{1}{4} \sum_{i=0}^{3} \left(w_i^+ - \bar{w}^+\right)^2, \tag{8.4}$$

the squared coefficient of variation of the rotor thrusts. The choice of S as an *energy proxy* is justified by momentum theory [27]: the aerodynamic power of a rotor scales as $P_i \propto F_i^{3/2}$, a strictly convex function of its thrust, so for a fixed collective thrust — the first row of the allocation is fixed by the demand — Jensen's inequality makes $\sum_i P_i$ minimal precisely when all thrusts are equal, that is when $S = 0$. Driving S down therefore reduces the total aerodynamic power for the same lift, and simultaneously lowers the peak rotor load $\rho = \max_i w_i^+ / \bar{w}^+$, which is reported as a saturation-margin diagnostic.

8.2.4 Constraints, authority and regularisation

The spread alone is not enough: minimising it could drive the arms into folded configurations that are infeasible or that consume too much control authority. Four further terms shape the cost.

Feasibility barrier. A negative w_i in Eq. (8.2) means a rotor would have to produce impossible negative thrust. Rather than rejecting such candidates with a hard cliff that would stall a gradient solver, the violation enters as a smooth,

normalised quadratic barrier

$$B(\boldsymbol{\delta}) = \frac{1}{4}\sum_{i=0}^{3}\left(\frac{\max(-w_i,\,0)}{\overline{|\boldsymbol{w}|}}\right)^2, \qquad \overline{|\boldsymbol{w}|} = \tfrac{1}{4}\textstyle\sum_i |w_i|, \tag{8.5}$$

while the balance metrics are evaluated on the clipped, realisable thrusts so the candidate remains differentiable.

Authority floor. To prevent rotors from folding too close to the centre, the cost function penalises the loss of lever arm. Defining $\ell(\boldsymbol{\delta}) = \min(\overline{|\boldsymbol{y}|},\ \overline{|\boldsymbol{x}|})$ as the smaller of the mean y- and x-lever arms, a one-sided quadratic penalty is applied only when ℓ falls below a specified floor:

$$L(\boldsymbol{\delta}) = \Big[\max(0,\,\ell_{\min} - \ell(\boldsymbol{\delta}))\Big]^2, \qquad \ell_{\min} = 0.24\,\mathrm{m}. \tag{8.6}$$

This limit acts as a *soft* floor rather than a hard constraint: the optimiser is permitted to cross it if the reduction in spread provides sufficient benefit, which occurs in practice. Indeed, every deployed solution from the fault campaign lies just below this threshold. Nor does the floor preserve authority symmetrically: the reshaped geometry *trades* axes rather than keeping both, the idle-derated available moment going from [9.84, 9.84, 0.85] to [10.95, 8.17, 0.85] N m at the flight-17 tail shape. Finally, the value $\ell_{\min} = 0.24\,\mathrm{m}$ sits 14 % below the X-shape mean lever of 0.2792 m, so the floor is far less binding than its name suggests and should be re-derived against the current geometry.

No excursion bound. The cost contains no term that bounds the depth of a reshape. The box (8.1) is per-joint and independent, so the only quantities that restrain how far the optimiser moves the arms away from X are the two regularisers of the next paragraph, which are deliberately light. In the June payload runs the depth of the arm excursion, not the direction of the payload, was the controlling variable of the sliding-mode loss of control (Sec. 12.1.4). Adding an explicit excursion penalty to the cost is a stated prerequisite before the optimiser is allowed to explore outside the hover envelope.

Regularisers and composite cost. Two quadratic regularisers keep the solution well-behaved, one anchoring the geometry to the X-shape and one keeping it close to the current configuration:

$$R_0(\boldsymbol{\delta}) = \frac{1}{4}\sum_i\Big(\frac{\delta_i}{\delta_{\max}}\Big)^2, \qquad R_c(\boldsymbol{\delta}) = \frac{1}{4}\sum_i\Big(\frac{\delta_i - \delta_i^{\mathrm{cur}}}{\delta_{\mathrm{c}}}\Big)^2, \quad \delta_{\mathrm{c}} = 45^\circ, \tag{8.7}$$

where $\boldsymbol{\delta}^{\mathrm{cur}}$ are the relative angles of the *measured* geometry — so that the static arm droop observed in flight (3.3° to 4.8° at the 95th percentile) enters the current-geometry regulariser. The optimiser minimises the weighted sum, subject to the

Table 8.1: Cost weights for Eq. (8.8), shaping parameters, and demand/output conditioning constants defined in Secs. 8.3.2 and 8.3.3. †Tuned prior to the pitch-sign correction in Eq. (8.3) and retained unchanged (see text).

Symbol	Term	Value
w_S	spread (objective)	3.0
w_B	feasibility barrier	100.0
w_L	authority floor	20.0
w_0	X-shape regulariser	0.04
w_c	current-geometry regulariser	0.03
$\ell_{\min}$	mean-lever floor	0.24 m
$\delta_{\max}$	joint limit	135°
δ_{c}	current-norm scale	45°
κ	CoG-shift moment scale	0.595†
Δt_{opt}	optimiser period	1 s
τ_{LP}	demand low-pass	4 s
M_{db}	input deadband	0.6 N m
h_M	input Schmitt hysteresis	0.1 N m
$\dot{\delta}_{\max}$	command rate limit	$8\,^\circ\,\mathrm{s}^{-1}$
d_{out}	output deadband (per joint)	1.5°
h_{rev}	reversal hysteresis	1.0°

box constraint (8.1),

$$\boxed{J(\boldsymbol{\delta}) = w_S\, S + w_B\, B + w_L\, L + w_0\, R_0 + w_c\, R_c} \tag{8.8}$$

with the weights listed in Table 8.1. The spread and feasibility barrier dominate the cost; the latter carries a large weight to strongly penalise negative-thrust candidates. The authority floor acts as a stiff soft-constraint, while the two regularisers are intentionally light to bias the solution without overriding physical constraints. If the allocation yields non-finite values, the candidate is assigned a large constant cost, ensuring its exclusion from the search space.

8.3 Online optimisation algorithm

The problem (8.8) is non-convex in $\boldsymbol{\delta}$, through the allocation inverse and the centre-of-mass map, so it is solved at each step by a bounded sequential-least-squares program (SLSQP) launched from several starting points, one of which is an analytic warm start derived from the demand; the result is then rate-limited and deadband-gated before it is commanded.

8.3.1 Analytic centre-of-mass-shift seed

A good initial guess comes directly from the cancellation idea: to oppose the inferred offset the rigid centre of mass should move so as to null the requested roll/pitch moment, a conservative displacement in that direction being

$$\boldsymbol{s}^{\star} = g_s \left(+\frac{M_\theta}{T},\ -\frac{M_\phi}{T} \right)^{\top}, \qquad g_s = 0.7, \tag{8.9}$$

where the seed is omitted if $|T|$ is negligible. The sign convention mirrors the antisymmetry in Eq. (8.3): the centre of mass must shift forward to counteract a nose-up pitch demand and leftward to cancel a right-roll demand. The gain g_s intentionally under-corrects relative to the full-cancellation value $1/\kappa \approx 1.68$, ensuring the seed serves as a robust, partial warm start that subsequent search iterations refine. This desired shift is mapped to joint steps via the centre-of-mass Jacobian $\mathbf{J}_{\text{cog}} \in \mathbb{R}^{2\times 4}$, constructed using central differences ($\pm 0.5°$) around the current configuration. The under-determined system is solved in a least-squares sense, $\boldsymbol{\delta}^{\text{step}} = \mathbf{J}_{\text{cog}}^{+}\, \boldsymbol{s}^{\star}$; the resulting step is clamped to 45° in the ∞-norm and added to the current configuration to yield the seed $\boldsymbol{\delta}^{\text{seed}}$.

8.3.2 Multi-start SLSQP search, rate limiting and output deadband

The search is initiated from the warm start $\boldsymbol{x}_0$ (the previous command clipped to bounds) and, when a seed exists, from three additional points: the seed itself, the midpoint of the $\boldsymbol{x}_0$–seed segment, and the previous command advanced by one-quarter of the seed step. Each candidate is refined via bounded SLSQP (maximum 60 iterations, tolerance 10^{-7}); the lowest-cost feasible result across all starts is retained, provided it does not exceed the warm-start cost. The selected optimum is first rate-limited to ensure the commanded step never exceeds $\dot{\delta}_{\max} = 8\,^{\circ}\,\text{s}^{-1}$ per optimiser period:

$$\boldsymbol{\delta}^{\text{cmd}} = \boldsymbol{x}_0 + \text{clip}\Big(\boldsymbol{\delta}^{\star} - \boldsymbol{x}_0,\ -\Delta_{\max},\ \Delta_{\max}\Big), \qquad \Delta_{\max} = \dot{\delta}_{\max}\, \Delta t_{\text{opt}}, \tag{8.10}$$

followed by a final clipping to the joint bounds, with $\Delta t_{\text{opt}} = 1\,\text{s}$.

The rate-limited command is not forwarded unconditionally. Residual noise in the filtered demand causes the SLSQP optimum to jitter by fractions of a degree; forwarding this directly would subject servo gears to continuous micro-motion. Acceptance is therefore applied per joint: joint i adopts the new command only if it deviates from the last accepted value by at least $d_{\text{out}} = 1.5°$. An additional margin $h_{\text{rev}} = 1.0°$ is required when the move reverses the previous direction to account for backlash crossing; joints failing these thresholds retain their prior values. Consequently, the arms converge via a staircase of discrete steps of at least 1.5° rather than continuously. This behaviour protects servo gears during prolonged

hover and, as detailed in Sec. 8.5, is the primary source of the layer's measured vibration cost.

8.3.3 Demand conditioning

The wrench demand the optimiser reacts to must be the slow, quasi-static component, not the fast attitude-regulation transients, and two filters achieve this. First, every demand message received between two optimiser steps — the sliding-mode controller publishes at 250 Hz, the predictive controller at 100 Hz, so the window mean makes the decimation rate-agnostic — is accumulated and reduced to its window mean, an anti-alias decimation to the optimiser rate. Second, the window mean feeds a first-order low-pass $\boldsymbol{\tau}_f \leftarrow a\,\boldsymbol{\tau}_{\text{win}} + (1-a)\,\boldsymbol{\tau}_f$ with $a = \Delta t_{\text{opt}}/(\tau_{\text{LP}} + \Delta t_{\text{opt}}) = 0.2$, $\tau_{\text{LP}} = 4\,\text{s}$, and $\boldsymbol{\tau}_f$ is the demand used in Eqs. (8.2)–(8.3). Finally, a deadband stops the arms from chasing trivial disturbances: a re-optimisation is triggered only when

$$\sqrt{M_{\phi,f}^2 + M_{\theta,f}^2} \;\geq\; M_{\text{db}} = 0.6\,\text{N m}, \tag{8.11}$$

otherwise the previous command is held; the deadband is a Schmitt trigger, once tripped releasing only below 0.5 N m, so a demand hovering at the threshold cannot chatter the optimiser on and off.

Because the learned moment bias is part of the demand, the disturbance estimator and the optimiser operate *in series*: as the geometry absorbs the disturbance the bias decays (0.70→0.38 N m in simulation, 0.62→0.30 N m in flight), the demand shrinks with it, and once the filtered roll/pitch demand falls back inside the deadband the geometry is simply held. The migration of trim from actuation to geometry is thus the observable signature of the layer, and its own stopping criterion. In closed loop the conditioning chain also leaves a slow residual mode of its own: in the fault campaign the thrust spread was observed to *breathe* with a period of roughly 6 s — the beat of the 1 Hz reshape cadence against the rate limiter and the output deadband, visible in Fig. 8.1. The oscillation is bounded and does not couple into the attitude loop, but it is the price of quantising the geometry command. Algorithm 1 collects the per-step procedure.

8.4 Integration with the control stack

The optimiser runs as a separate ROS 2 node at 1 Hz, deliberately far slower than the attitude loop, which enforces the time-scale separation between the fast stabilising controller and the slow geometry adaptation. It consumes the commanded wrench published by whichever stack is flying — both export the same $(T, M_\phi, M_\theta, M_\psi)$ demand on a common topic — together with the measured joint angles, and publishes the absolute arm targets to the arm drivers, which apply their own rate limiting. The interface is stack-agnostic, but the outcome is not: in simulation the

Algorithm 1 One optimiser step (period $\Delta t_{\mathrm{opt}} = 1\,\mathrm{s}$).

if not (enabled $\wedge$ mission active $\wedge$ state = HOVERING $\wedge$ demand and joints fresh) **then** reset window, **return**
$\boldsymbol{\delta}^{\mathrm{cur}} \leftarrow \mathrm{wrap}_{180}(\boldsymbol{\theta}_{\mathrm{meas}} - \boldsymbol{\theta}_{\mathrm{base}})$; $(x_{\mathrm{ref}}, y_{\mathrm{ref}}) \leftarrow \boldsymbol{r}_{\mathrm{CoG}}(\boldsymbol{\theta}_{\mathrm{meas}})$
on first enable: $\boldsymbol{\delta}^{\mathrm{cmd}} \leftarrow \boldsymbol{\delta}^{\mathrm{cur}}$ ▷ bumpless start
update $\boldsymbol{\tau}_f$ by window-mean decimation and the low-pass
if $\sqrt{M_{\phi,f}^2 + M_{\theta,f}^2}$ exceeds the Schmitt-gated deadband of Eq. (8.11) **then**
    build the seed of Eq. (8.9)
    $\boldsymbol{\delta}^{\star} \leftarrow$ best of the multi-start bounded SLSQP on J (Eq. (8.8))
    rate-limit $\boldsymbol{\delta}^{\star}$ (Eq. (8.10))
    for each joint i: accept the new δ_i^{cmd} only if it moves by $\geq d_{\mathrm{out}}$ from the last accepted value ($+\,h_{\mathrm{rev}}$ on a direction reversal), **else** hold ▷ staircase
end if
publish $\boldsymbol{\theta}^{\mathrm{cmd}} = \boldsymbol{\theta}_{\mathrm{base}} + \boldsymbol{\delta}^{\mathrm{cmd}}$ to the arm drivers

optimiser rescues the sliding-mode stack under a rotor fault, whereas the predictive stack has no completed run at that severity at all: with the fault injected it does not leave the ground, and its only completed fault runs are the three twins of flight 17 at the severity the aircraft actually had (Sec. 12.2.4). Fault-tolerant morphing under the predictive stack is therefore restricted to hover in X and has not been flown; it is planned future work.

Engagement and authority. The optimiser retains control of arm commands only when explicitly enabled, in HOVERING mode, and with both demand and joint feedback considered fresh (within 0.5 s); otherwise it stands down and resets its averaging window. Upon transitioning to this active state, the command is seeded with the measured geometry to ensure a bumpless engagement. When disabled, authority reverts to the manual and preset shape commands published by the mission manager, specifically the X, O, T, and H presets from Table 3.2, with configuration O at its final 100° opening. Authority is not relinquished between active steps falling within the deadband: provided the optimiser remains enabled and in HOVERING mode, the mission manager remains silent, and the optimiser holds and re-publishes its previous command until a significant roll or pitch demand recurs.

Supervision of the arm channel. The optimiser never owns the arms unconditionally. The arm driver runs its own watchdog — runaway detection at $35\,^{\circ}\,\mathrm{s}^{-1}$, sustained-load trips at 40 % for 3 s (85 % for 2 s during transits) and droop trips at 6° for 2 s — the mission manager consumes its status to gate take-off and to trigger an automatic landing, and landing always reshapes back to X. In the hardware

fault campaign the landing sequence ran three times out of three with the optimiser still enabled, and the servo of the faulty arm raised an overload warning during that reshape, which is the expected place for it to appear.

Coexistence with the attitude loop and moment-bias estimator. The optimiser and the controller's own disturbance rejection are complementary: the fast loop continues to null the residual moment in closed loop, while the optimiser slowly removes the *cause* of that moment. The separation is one of time scale, not of effect. The rate limit bounds the *command* to 8° per step, but the step is executed by the servos and gated by the output deadband, causing arms to move in bursts of at least 1.5° once per second. On hardware, this manifests as a hump in the 8 Hz to 15 Hz band of the roll-rate spectrum, approximately a decade above the fixed-arm hover level, while the 95th-percentile tilt remains within 3.5° to 4.2°. The second coupling is parametric: reshaping alters the inertia tensor, and since sliding-mode gains are scheduled on $J(\boldsymbol{\theta})$, the geometry loop effectively retunes the attitude loop during convergence. Both effects were bounded in flight, though neither is zero.

8.5 The demonstrated case: fault-tolerant morphing

The optimiser was formulated as an energy layer, but its decisive test came as fault tolerance. To a layer that observes only the requested wrench, a degraded rotor is indistinguishable from an off-centre payload: both appear as a persistent, quasi-static roll/pitch demand on the same topic, and the machinery of the preceding sections applies unchanged. The fault model is the profile that reproduces the propeller destroyed in flight 9: rotor 2 loses 39 % of its thrust coefficient and carries an additional torque deficit of 0.283 beyond the thrust path, calibrated on the 0.157 N m yaw-bias floor measured in that flight. A severity scale multiplies the deficit: 1× is the fault as flown, 1.5× the declared test envelope.

8.5.1 Simulation campaign

Ten runs were flown in simulation on the twin with the final parameter set, one run per scenario ($n = 1$; no seed repetition), with the fault engaged in the air either as a step or as a 20 s ramp. At 1× engaged as a step, the fixed frame survives on feedback alone: the sliding-mode controller recovers in about 15 s from a 22.8° tilt peak, but settles into a visibly asymmetric hover with a thrust spread of 130 rad s^{-1}, leaving the faulty rotor pinned at 597 rad s^{-1}. With the optimiser enabled, the same scenario converges to a spread of 34 rad s^{-1}; the faulty rotor is unloaded to

526 rad s^{-1}, the arms settle at [57, 249, 279, 158]°, and the learned roll bias decays from 0.71 to 0.41 N m. This indicates a migration of trim effort from actuation to geometry.

At 1.5× severity, engaged as a ramp, the comparison shifts from comfort to feasibility (Fig. 8.1). The fixed frame is *authority-dead*: both bias axes pin at their ±1.0 N m clamps, the vehicle drifts 97 m and never reaches its waypoint — alive, but no longer controllable in position. With reshape the same mission *completes*, holding position within 0.21 m to 0.40 m, with a thrust spread of about 50 against 261 rad s^{-1}. The step at 1.5×, by contrast, kills the fixed and the morphing vehicle *identically* — in the reshape run the arms had not moved at the moment of breakup. The division of labour is measured, not assumed: the transient of a fault belongs to the feedback controller, the steady state to the optimiser, and fault-tolerant morphing does not protect against the breakout itself.

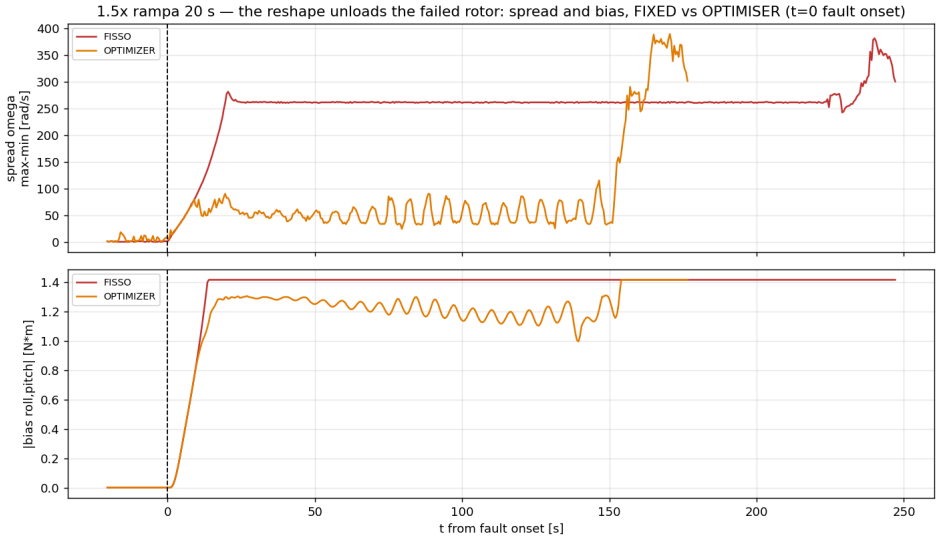


Figure 8.1: Simulated 1.5× rotor fault ramped over 20 s (onset at $t = 0$): fixed frame against optimiser-enabled reshape, single run per condition. Top: motor-speed spread, plateauing near 260 rad s^{-1} for the fixed frame against roughly 50 rad s^{-1} with reshape; the slow ∼ 6 s breathing of the reshaped trace is the residual mode of the conditioning chain (Sec. 8.3.3). Bottom: norm of the learned roll/pitch trim; the fixed frame pins both axes at the ±1.0 N m clamp (norm $\sqrt{2} \approx 1.41$) and goes authority-dead, while the reshaped vehicle settles near 1.2 — the roll axis still at its clamp — and completes the mission. The excursions at the far right of each trace belong to the end-of-run descent.

8.5.2 Hardware demonstration: flight 17

The hardware experiment removes the simulation from the loop entirely: the very propeller destroyed in flight 9 was preserved and refitted on motor 2 — the same

rotor the simulated profile degrades — turning the fault model into a physical object. Three trials were conducted, one battery per trial ($n = 3$), each incorporating an internal baseline: hovering with the fault and fixed geometry for 31 s to 58 s, followed by optimiser activation for approximately 70 s, concluding with landing while the optimiser remained enabled. The measured severity of the damaged propeller corresponds to a thrust deficit of 22 % to 28 %. This represents approximately 65 % of the simulated fault magnitude, rendering the simulation conservative relative to the physical failure. The per-trial metrics are reported in Sec. 12.2.4; what matters here is the mechanism they confirm. In every trial the optimiser unloads the damaged rotor and returns trim authority to the controller — the roll bias drops from 0.46 N m to 0.62 N m to 0.27 N m to 0.30 N m and the pitch bias collapses toward zero — while the vehicle continues to hold position, no servo watchdog trips and the landing sequence runs autonomously three times out of three.

The reshaping constitutes the most significant result, as its signature proves repeatable across all three trials. The arm carrying the damaged propeller (arm 1) deflects by 20° to 24°, arm 2 counter-deflects by −25° to −31°, while arms 0 and 3 move by 6° to 14° and 3° to 5°, respectively. This solution was identified consistently from three distinct transients, exhibiting the same sign pattern as the converged simulation solution (offsets [+12, +24, −36, +23]° from X), with comparable displacements on the two most-affected arms and a minimal shift on arm 3. This behaviour is consistent with the milder physical fault severity. Meanwhile, the rigid centre of mass shifts by only $(-5, -9)$, $(-3, -9)$, and $(-7, -9)$ mm, indicating that the benefit is delivered by the relocated lever arms rather than by centre-of-mass displacement. This confirms the mechanism ranking established in Sec. 8.1. Figure 8.2 displays data from the three trials.

The staircase convergence is not free. With the optimiser enabled a hump appears in the 8 Hz to 15 Hz band of the roll-rate power spectrum, about a decade above the fixed-arm fault hover, in all three trials — the mini-transits of the $\geq 1.5°$ steps executed once per second. The cost is bounded: the 95th-percentile tilt stays at 3.5° to 4.2° and does not grow over the optimiser-on windows. A separate spectral peak at 23 Hz to 26 Hz, present in *both* phases, is not attributable to the optimiser and remains unexplained (Fig. 8.3).

8.6 What the optimiser buys, and what it does not

The campaigns above allow the closing claims of this chapter to be stated as measurements rather than expectations — including the negative ones.

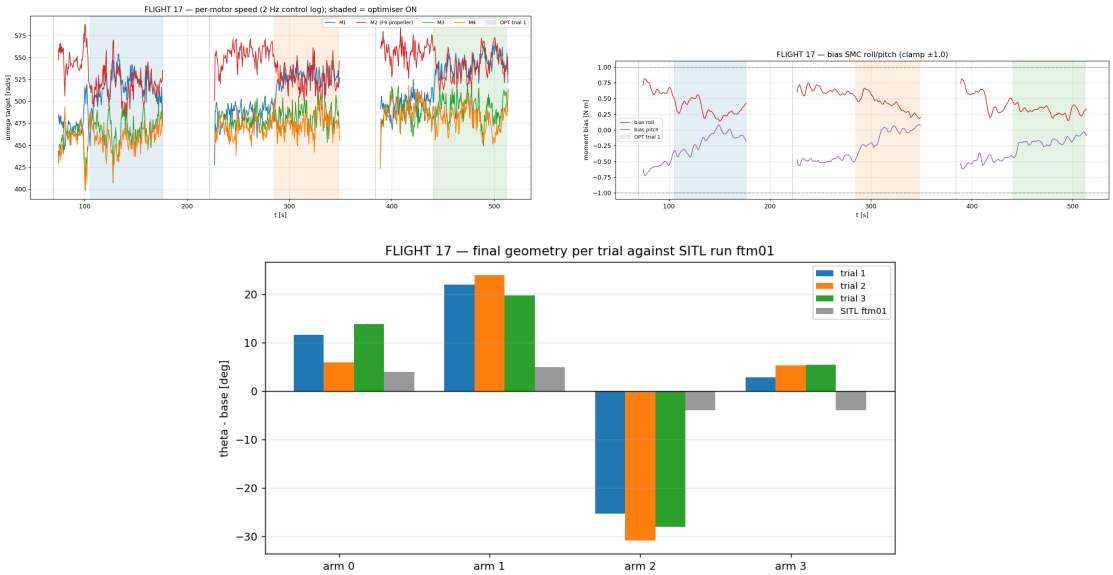


Figure 8.2: Flight 17, three trials on one battery each with the propeller destroyed in Flight 9 refitted on rotor 2. Top left: per-motor speed, with the optimiser-enabled windows shaded — the damaged rotor is unloaded and the four speeds converge. Top right: the learned roll and pitch moment bias, which falls from 0.46 N m to 0.62 N m to 0.27 N m to 0.30 N m as the arms take up the asymmetry: the trim migrates from actuation to geometry. Bottom: the converged arm geometry of the three trials against the simulation solution. The same solution is found three times from three different transients, and its sign pattern matches the simulation campaign of Sec. 8.5.1; the milder physical fault (about 0.65 of the simulated severity) gives correspondingly smaller swings.

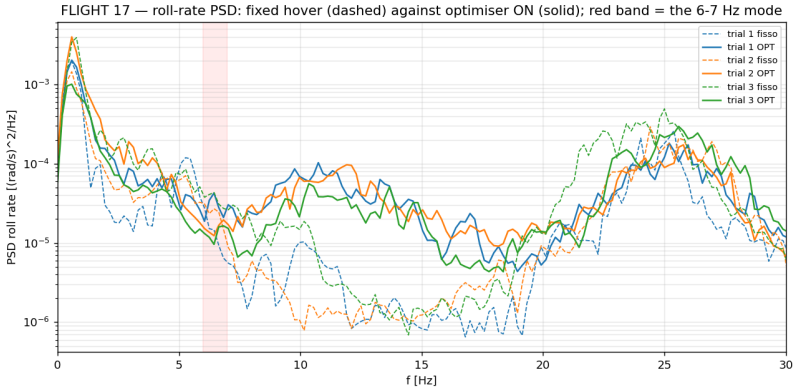


Figure 8.3: Roll-rate power spectrum in fault hover on Flight 17, fixed arms against the optimiser enabled. The optimiser adds a hump about a decade high in the 8 Hz to 15 Hz band — the once-per-second mini-transits of the $\geq 1.5°$ steps through which the search converges. The cost is bounded and does not grow: the 95th-percentile tilt stays at 3.5° to 4.2° across the optimiser-on windows. The peak at 23 Hz to 26 Hz is present in *both* phases and is therefore not attributable to the optimiser; it remains unexplained.

It redistributes load; it does not improve tracking. On the payload sequence the worst per-phase drift moves from 0.061 to 0.074 m when the optimiser is enabled, while the rotor-speed spread falls by 63 % to 81 %; under the 1× rotor fault the spread collapses from 130 to 34 $\mathrm{rad\,s^{-1}}$ and the learned roll trim from 0.71 to 0.41 N m while the vehicle keeps flying (Chapter 12). The optimiser is not a disturbance rejector; this claim was tested and found false, as disturbance rejection is handled by the feedback loops regardless of configuration. Instead, the optimiser modifies how this rejection is sustained: it distributes the load evenly across all four rotors at healthy load levels rather than pinning one rotor near its saturation limit.

The benefit becomes decisive when authority, not accuracy, binds. At 1.5× fault severity, this mechanism determines whether the vehicle remains operational or fails due to lack of control authority. Without it, trim limits are pinned, resulting in 97 m of drift and failure to reach the waypoint; with it, the mission completes successfully. The flight-17 campaign confirms this behaviour on hardware at the severity level provided by the physical propeller damage.

Authority is bounded, not preserved. The authority floor of Eq. (8.6) bounds, but does not forbid, the loss of lever arm: the deployed solutions cross the soft floor, and the reshaped geometry trades pitch authority for roll authority rather than preserving both. The claim this chapter can support is that the loss is bounded and was never the binding constraint in any flown experiment.

The energy claim remains a proxy. While the spread objective is grounded in momentum theory, no direct electrical-power measurement comparing optimiser-on versus optimiser-off states exists for the real vehicle. The sole power comparison presented in this thesis (Chapter 12: 993.6 W to 924.2 W, about 7 %, on the June payload sequence) was made on the design-stage twin. This setup exhibits zero model error by construction, and the data are derived from commanded allocations rather than logged ESC telemetry; thus, it serves as a consistency check for the proxy metric rather than evidence of measured energy savings. The only battery-side power measurements on the actual vehicle compare X and O *presets* (Sec. 6.9), not optimiser states.

The payload case is a design requirement, not a demonstrated one. The 3 kg unobserved payload remains the sizing criterion for the design, and associated constants are current. However, since the payload rig has been removed from the simulation model and payload rejection has never been tested in flight, early campaign experiments involving payloads constitute historical data based on a

superseded plant. The scenario demonstrated in this chapter, both in simulation and on hardware, is that of a degraded rotor (Sec. 8.5).

Chapter 9

Nonlinear model-predictive control with supervised fallback

As an alternative to the sliding-mode baseline of Chapter 7, this chapter develops a nonlinear model-predictive controller (NMPC) for the morphing quadrotor. The predictive formulation handles state and input constraints explicitly, exploits the future reference and rejects persistent disturbances in an offset-free manner, while remaining *morphing-aware*: the configuration-dependent inertia and lever arms enter every solve as parameters, so the controller follows the airframe through a shape change instead of assuming a frozen geometry. The treatment covers the design goals (Sec. 9.1), the prediction model with its actuator-lag and disturbance augmentations (Sec. 9.2), the optimal control problem (Sec. 9.3) and the offset-free observer that closes the loop on the unmodelled effects (Sec. 9.4), the software realisation (Sec. 9.5), the two-layer safety net actually flown — the in-node robustness layer (Sec. 9.6) and the supervised hot-standby fallback to the sliding-mode baseline (Sec. 9.7) — the structural limit of the rigid-arm prediction model exposed in flight (Sec. 9.8), and a comparison against the baseline (Sec. 9.9).

9.1 Motivation and design goals

The cascaded sliding-mode stack is robust and computationally light, but its structure imposes three recurring limitations: the outer position loop clips its tilt setpoint at the geometric limit, so an aggressive entry *saturates* the inner loop and the transient has to be tamed by hand; the loop has no preview, reacting to the instantaneous error and discovering the braking phase of an arrival only one step

at a time; and the steady-state rejection of a persistent disturbance is delegated to a moment-bias integrator with a deadband and an arm-motion freeze, a mechanism that trades responsiveness for the risk of an integrator deadlock.

The predictive controller inherently resolves these limitations by design. Tilt, velocity, rotor-speed, and total-thrust limits are incorporated into the optimisation as explicit, *anticipated* constraints across the entire horizon. Consequently, the control law never plans trajectories that exceed the vehicle's capabilities. By consuming the future reference window, the controller initiates deceleration or turning manoeuvres before significant tracking errors accumulate. The prediction model is further augmented with a slowly-varying disturbance term. This term is estimated online and applied as feedforward compensation, ensuring steady-state error elimination without relying on an ad-hoc integrator. Moreover, inertia values and centre-of-mass-relative lever arms are passed to each solve as parameters derived from *measured* arm angles. Thus, the model accurately tracks the current configuration rather than assuming a nominal X-shape geometry. The controller interfaces with identical command topics and actuator outputs as the baseline system, enabling direct A/B comparison on identical experiments. It is noted that, in this version, arm configuration remains outside the decision vector. An energy-aware geometry optimiser, detailed in Chapter 8, operates alongside the controller in a decoupled manner.

9.2 Prediction model

The predictive model reuses the physical sub-models established for the baseline — the rotor thrust and reaction torque, the variable-geometry inertia and the centre-of-mass-relative lever arms — and assembles them into a single continuous-time vector field driven directly by the rotor speeds. Working in the PX4 frames of Sec. 5.1, the body state is

$$\boldsymbol{x} = \big(\underbrace{p_x, p_y, p_z}_{\boldsymbol{p}},\ \underbrace{v_x, v_y, v_z}_{\boldsymbol{v}},\ \underbrace{\phi, \theta, \psi}_{\boldsymbol{\Theta}},\ \underbrace{w_x, w_y, w_z}_{\boldsymbol{\omega}_b}\big)^{\top} \in \mathbb{R}^{12}, \tag{9.1}$$

with position and velocity in the NED inertial frame, intrinsic Z-Y-X Euler angles and *body* angular rates $(w_x, w_y, w_z) = (p, q, r)$, the indexed notation being adopted for symmetry with the components of the vector field.

9.2.1 Rotor-speed input and the actuator-lag augmentation

A distinctive choice of this formulation is that the optimisation variable is the vector of four rotor speeds $\boldsymbol{\omega}_c$ rather than the collective wrench. Building the wrench *inside* the model from $\boldsymbol{\omega}^2$ and the live geometry has two consequences: the allocation is no longer a fixed inverse evaluated at the nominal shape — it

follows the morphing automatically — and the rotor limits become a simple box on $\boldsymbol{\omega}_c$, valid for any arm angle. Because the brushless drive does not follow a speed command instantaneously, the state is augmented with the *actually applied* rotor speeds $\boldsymbol{\omega}_a$, which track the command through a first-order lag

$$\dot{\boldsymbol{\omega}}_a = \frac{\boldsymbol{\omega}_c - \boldsymbol{\omega}_a}{\tau_{\text{act}}}, \qquad \tau_{\text{act}} = 80\,\text{ms}, \tag{9.2}$$

so that the body responds to $\boldsymbol{\omega}_a$ while the controller regularises the command; the augmented state is $\boldsymbol{z} = (\boldsymbol{x}^\top, \boldsymbol{\omega}_a^\top)^\top \in \mathbb{R}^{16}$.

The lag constant is the single most consequential number in the model, and its identification history — a bench study of 156 step transitions, a deliberate detour through 150 ms, and the in-flight frequency-response fit that restored the bench value — is Sec. 6.6. Two facts from it shape the design. The lag is not a constant but amplitude-dependent, about 80 ms near hover and rising towards twice that at take-off-scale steps (Table 6.2), while the model can hold only one value, chosen for the hover regime. And the choice matters asymmetrically: biased low, the optimiser believes in authority the actuators do not have; biased high, the horizon under-reacts — and it was the low-biased error that proved dangerous. The restoration to 80 ms is part of the remedial package for the flight-19 accident, and the large-step regime with the current propellers has never been measured (Sec. 9.8).

9.2.2 Continuous nonlinear dynamics

The wrench generated by the applied rotor speeds, for the centre-of-mass-relative motor positions (x_i, y_i) and the thrust/torque coefficients of Eq. (5.20), is

$$\begin{aligned} T &= k_T \textstyle\sum_{i=0}^{3} \omega_{a,i}^2, & M_\phi &= -k_T \textstyle\sum_{i=0}^{3} y_i\, \omega_{a,i}^2, \\ M_\theta &= k_T \textstyle\sum_{i=0}^{3} x_i\, \omega_{a,i}^2, & M_\psi &= k_Q\,(\omega_{a,0}^2 + \omega_{a,1}^2 - \omega_{a,2}^2 - \omega_{a,3}^2). \end{aligned} \tag{9.3}$$

The thrust accelerates the airframe along the body up-axis, i.e. along *minus* the third column of the Z-Y-X rotation matrix (that column points down in the FRD/NED convention — hence the leading minus sign); writing $c_\bullet = \cos(\bullet)$, $s_\bullet = \sin(\bullet)$, the translational dynamics in NED are

$$\dot{\boldsymbol{p}} = \boldsymbol{v}, \qquad \dot{\boldsymbol{v}} = \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} - \frac{T}{M} \begin{bmatrix} c_\psi s_\theta c_\phi + s_\psi s_\phi \\ s_\psi s_\theta c_\phi - c_\psi s_\phi \\ c_\theta c_\phi \end{bmatrix} + \boldsymbol{d}_v, \tag{9.4}$$

with M the total rigid mass and $\boldsymbol{d}_v$ the unmodelled translational acceleration. The attitude kinematics map the body rates to the Euler-angle rates,

$$\dot{\phi} = w_x + s_\phi t_\theta\, w_y + c_\phi t_\theta\, w_z, \quad \dot{\theta} = c_\phi\, w_y - s_\phi\, w_z, \quad \dot{\psi} = \frac{s_\phi}{c_\theta}\, w_y + \frac{c_\phi}{c_\theta}\, w_z, \tag{9.5}$$

well conditioned because the tilt stays small, and the rotational dynamics use the diagonal, configuration-dependent inertia of Eqs. (5.14)–(5.16), including the gyroscopic coupling and the unmodelled body moment $\boldsymbol{d}_M$:

$$\begin{aligned} \dot{w}_x &= \frac{M_\phi - (I_{zz} - I_{yy})\, w_y w_z + d_{M,x}}{I_{xx}}, \\ \dot{w}_y &= \frac{M_\theta - (I_{xx} - I_{zz})\, w_x w_z + d_{M,y}}{I_{yy}}, \\ \dot{w}_z &= \frac{M_\psi - (I_{yy} - I_{xx})\, w_x w_y + d_{M,z}}{I_{zz}}. \end{aligned} \tag{9.6}$$

Equations (9.2)–(9.6) define the augmented vector field $\dot{\boldsymbol{z}} = \boldsymbol{f}(\boldsymbol{z}, \boldsymbol{\omega}_c; \mathbf{I}, \boldsymbol{x}_m, \boldsymbol{y}_m, \boldsymbol{d}_v, \boldsymbol{d}_M)$, where $\boldsymbol{x}_m, \boldsymbol{y}_m$ collect the motor positions. The geometry triple and the disturbance pair are supplied to every solve as parameters: since the arms move slowly compared with the prediction horizon they are accurately treated as constant over it, which lets the geometry be recomputed from the measured arm angles each tick without a symbolic re-derivation.

9.2.3 Offset-free disturbance augmentation

The additive terms model the slowly-varying effects that the rigid model omits. Three matter in practice: the lateral aerodynamic trim and the wind; the residual error of the thrust map, measured at about 10 % on hardware, which alone puts $0.93\,\mathrm{m\,s^{-2}}$ on the vertical channel with nothing attached to the airframe; and the thrust deficit of a damaged rotor, measured at $-22\,\%$ to $-28\,\%$ in the fault-tolerance flights. A standing design requirement adds a fourth: the vehicle must remain flyable with an unmodelled payload of up to 3 kg, and although the payload rig was removed from the simulation model and was never carried in flight, the requirement is what sizes the observer clamps of Sec. 9.4. Each disturbance is modelled as a random walk, estimated online and fed forward in Eqs. (9.4) and (9.6); because the controller plans against the estimated disturbance, the steady-state position and attitude error vanish *by construction* [36, 37] — the principled replacement for the moment-bias integrator of the baseline, and one that remains active during manoeuvres rather than only in hover.

9.3 Optimal control problem

9.3.1 Cost and reference preview

Over a horizon of N shooting nodes with spacing Δt, and with the discrete dynamics obtained by an explicit Runge–Kutta integration of $\boldsymbol{f}$, the controller minimises a

Table 9.1: Deployed cost weights (diagonal entries). All entries were flown on 2026-08-06 except the yaw pair (Q_ψ, Q_{w_z}), tripled after those flights and not yet flown.

Block	Channels	Diagonal
$\boldsymbol{Q}_p$	p_x, p_y, p_z	(8, 8, 12)
$\boldsymbol{Q}_v$	v_x, v_y, v_z	(4, 4, 2)
$\boldsymbol{Q}_\Theta$	ϕ, θ, ψ	(1, 1, 6)
$\boldsymbol{Q}_\omega$	w_x, w_y, w_z	(8, 8, 0.9)
$\boldsymbol{R}$	$\omega_{c,0\ldots3}$	10^{-4} (each)
$\boldsymbol{R}_d$	$\omega_{c,j} - \omega_{a,j}$	$5 \cdot 10^{-3}$ (each)
w_e	terminal multiplier	10

nonlinear least-squares cost on the body state and the command,

$$J = \sum_{k=0}^{N-1} \left(\left\| \boldsymbol{x}_k - \boldsymbol{x}_k^{\mathrm{ref}} \right\|_{\boldsymbol{Q}}^2 + \left\| \boldsymbol{\omega}_{c,k} - \omega_{\mathrm{hov}} \, \mathbf{1}_4 \right\|_{\boldsymbol{R}}^2 + \left\| \boldsymbol{\omega}_{c,k} - \boldsymbol{\omega}_{a,k} \right\|_{\boldsymbol{R}_d}^2 \right) + \left\| \boldsymbol{x}_N - \boldsymbol{x}_N^{\mathrm{ref}} \right\|_{\boldsymbol{P}}^2, \tag{9.7}$$

where ω_{hov} is the per-rotor hover speed of Eq. (5.21) replicated over the four rotors, $\boldsymbol{Q}$ weights the tracking error, $\boldsymbol{R}$ regularises the command towards hover, and the terminal weight is $\boldsymbol{P} = w_e \, \boldsymbol{Q}$.

The command term is regularised towards hover rather than towards its own previous value, as a standard rate-of-change penalty was previously tested and reverted due to under-damping of the control loop. The third cost component instead penalises the *demanded actuator slew*. Along the model trajectories defined by $\boldsymbol{\omega}_c - \boldsymbol{\omega}_a = \tau_{\mathrm{act}} \dot{\boldsymbol{\omega}}_a$, this term acts as a lead penalty anchored to the lag state. It features zero equilibrium bias and remains invariant with respect to node spacing. This formulation closes the remaining curvature gap in the input profile. With $\boldsymbol{R}$ alone, a single reference-reversal tick could rail a rotor command by approximately $400\,\mathrm{rad\,s^{-1}}$. The deployed dosage reduces this peak to roughly $80\,\mathrm{rad\,s^{-1}}$. A doubled penalty coefficient was evaluated but rejected, as it consumed excessive tracking authority, resulting in the vehicle settling metres off the origin.

Two entries carry their history. The lateral velocity weight was consolidated at 4 by the simulation parity campaign: raising it from 1 brought the clean-air hover tilt from 1.20 to 0.407° at unchanged position hold, and pushing it beyond 4 makes that trade worse. The yaw entries were tripled on 2026-08-06 after the heading hunting measured in flight (Sec. 12.2.3). Roll and pitch carry a small but non-zero weight: an earlier tuning left them at zero, on the argument that the required tilt should emerge from the position and velocity cost alone, and on the honest actuator plant that tuning left the attitude in a bounded 1.5 Hz limit cycle at ±17° to 18°. Pricing attitude and body rate took the 99th-percentile tilt to 1.4° to 1.7° and, far from trading position for attitude, improved the hover hold from 9

to 1.5 cm: the limit cycle was costing position accuracy too.

Analysis of the deployed solver clarifies why adjusting yaw weights resolves the heading issue while changes to roll and pitch weights have minimal effect. In this formulation, roll/pitch attitude stiffness is not dictated by the attitude weights themselves. Scaling these weights leaves the effective stiffness essentially constant at approximately $10\,\mathrm{N\,m\,rad^{-1}}$. This gain arises from position and velocity costs propagating through the model dynamics, and with $\boldsymbol{R} \sim 10^{-4}$, the cost function remains nearly scale-invariant. Yaw exhibits opposite behaviour; it receives no stiffness contribution from the position cost and relies entirely on its specific weight term. The originally deployed weights resulted in a yaw stiffness of only about $0.06\,\mathrm{N\,m\,rad^{-1}}$, corresponding to a bandwidth near 0.1 Hz. This low responsiveness matches the slow heading hunting observed during flight, characterised by 44° peak-to-peak oscillations at approximately 0.2 Hz. Tripling the yaw weights triples the effective stiffness and is predicted to reduce the wander magnitude to about 15° peak-to-peak. Although this improvement has been verified via probe analysis and simulation, flight validation remains pending. This asymmetry also explains why softening attitude weights fails to detune the roll/pitch loop when such detuning is desired (Sec. 9.8).

The controller utilises the *preview* window as its reference input, which is supplied by the reference generator across the entire horizon. For regulation tasks, this corresponds to the ramped setpoint from the governor (Sec. 9.6). In the case of streamed trajectories, the reference adopts a quadratic extrapolation of the latest sample:

$$\boldsymbol{p}_k^{\mathrm{ref}} = \boldsymbol{p} + \boldsymbol{v}\,k\Delta t + \tfrac{1}{2}\boldsymbol{a}\,(k\Delta t)^2.$$

This extrapolation is paired with velocity feedforward, and its acceleration term curves the preview towards the intended path: by accounting for centripetal components, such as those of a helix, it eliminates the corner-cutting lag typically associated with straight-tangent references.

9.3.2 Constraints

The optimisation enforces, at every node of the horizon, the physical envelope of the platform:

$$|\phi_k|,\ |\theta_k| \le \phi_{\max} = 0.436\,\mathrm{rad}\ (25°), \tag{9.8}$$

$$|v_{x,k}|,\ |v_{y,k}|,\ |v_{z,k}| \le v_{\max} = 3.5\,\mathrm{m\,s^{-1}}, \tag{9.9}$$

$$\omega_{\min} \le \omega_{c,j,k} \le \omega_{\max}, \quad j = 0,\ldots,3, \tag{9.10}$$

$$\underline{c}\,T_{\mathrm{hov}} \le k_T \textstyle\sum_j \omega_{c,j,k}^2 \le \overline{c}\,T_{\mathrm{hov}}, \tag{9.11}$$

with the rotor-speed range of Eq. (5.20) and total-thrust factors $(\underline{c}, \overline{c}) = (0.3,\ 1.8)$ about the hover thrust of Eq. (5.21). The tilt and velocity bounds are box

constraints on the predicted state; the rotor bound is a box on the input, geometry-independent and hence valid for any arm angle, unlike a fixed-shape allocation inverse; and the total-thrust limit is a nonlinear constraint that caps the climb and forbids a near-zero-thrust free-fall that the per-rotor box alone would allow.

The tilt bound is not a local constant of the predictive controller but the airframe's declared envelope, read from the same shared constant as the baseline; since a quadrotor accelerates by tilting, it also sets the horizontal acceleration ceiling $a_{\max} = g \tan \phi_{\max} = 4.57\,\mathrm{m\,s^{-2}}$. An earlier local copy of 14.3° had silently diverged from the envelope, leaving the predictive controller with 55 % of the baseline's horizontal authority on the same airframe, and the discrepancy went unnoticed until the two stacks were compared on identical manoeuvres. The hard tilt bound can optionally be replaced by a slack-penalised soft constraint so that the quadratic program can never be infeasible at the tilt corner; this was implemented, tried and left disabled — in the high-severity campaign the soft variant did exactly its job, zero QP failures against 3.4 % of the ticks in the matching hard-bound run, and the vehicle stayed on the ground all the same while the open attitude limit cycle reached ±25°: infeasibility at the tilt corner was not the binding problem.

9.3.3 Horizon, discretisation and numerical solution

The deployed configuration employs $N = 30$ shooting nodes with a step size of $\Delta t = 10\,\mathrm{ms}$, providing a 0.3 s preview window, while the control loop operates at 100 Hz. The nonlinear program is resolved using a real-time-iteration (RTI) sequential quadratic programming approach. This strategy executes a single SQP iteration per tick, employing a Gauss–Newton Hessian and the structure-exploiting partial-condensing HPIPM interior-point solver for the underlying QP. The solver is warm-started from the previous tick's primal-dual solution so that the single Newton step stays close to the optimum. The problem definition and solver code are generated and compiled via the `acados` toolchain [17], which produces C code optimised for onboard deployment. In the flight configuration, a small Levenberg–Marquardt regularisation term (10^{-2}) is added to the Gauss–Newton Hessian alongside an 80 ms lag constant. At the shorter lag, the unregularised QP tends to collapse into minimum-step failures; this regularisation eliminates that failure mode without incurring measurable tracking costs.

This implementation supersedes the originally envisioned linear-time-varying prototype. While relinearising about the reference would yield a convex QP near hover, the full nonlinear solve at 100 Hz accurately captures large-tilt and gyroscopic terms, provided it fits within the computational budget, which it does. Benchmarks on the flight companion with the full deployed configuration show that the solver costs 3.37 ms at the median and 3.99 ms at the 99th percentile over 3,000 solves, comfortably within the 10 ms budget. Rare outliers in the range of

21 ms to 27 ms were traced to second-generation Python garbage collection rather than solver performance and disappear when the collector is frozen. A reduced fallback configuration remains available for weaker companions (50 Hz, $N = 15$, full-condensing qpOASES, p99 1.75 ms) but was never required in practice. In flight conditions, the solver ran even faster, achieving sub-millisecond medians on the same companion (Sec. 12.2.3).

A separate latency matters independently of the solve time: the *transport* delay between issuing the command and PX4 applying it. The controller proved regime-dependent in this delay — an aggressive long cruise that flew repeatably in clean simulation tumbled when several milliseconds of unmodelled command staleness were injected — but on hardware the question was settled directly: the flight-19 analysis returns an extra lag of at most 12 ms, so the transport path is not where the stability margin was lost. What the same analysis does show is a threefold gain excess at 3 Hz outside the nominal shape — a modelling problem, not a latency problem (Sec. 9.8).

9.4 Offset-free disturbance observer

The estimator follows the momentum-based line of nonlinear disturbance observers [38], in the form specialised to flying robots by Tomic et al. [39]: the external wrench is recovered from the discrepancy between the measured momentum rate and the one the model predicts from the applied actuation, without differentiating a noisy velocity.

The disturbances that close the offset-free loop are estimated by a momentum-based observer, chosen over a naive acceleration-residual estimator because it does not differentiate the noisy velocity and body-rate signals and, crucially, does not feed back positively through the actuator lag. The residual is computed from the *applied* wrench, reconstructed using the lagged rotor-speed estimate and live geometry (Eq. (9.3)), rather than from the raw command. An earlier observer design that relied on the command signal wound up through the lag dynamics, resulting in a growing hover tilt instability.

On the translational channel, the nominal model acceleration combines gravity with the applied thrust along the body z-axis:

$$\boldsymbol{a}_{\mathrm{nom}} = (0,0,g)^{\top} - (T_a/M)\,\boldsymbol{r}_3(\boldsymbol{\Theta}).$$

The observer maintains an auxiliary state $\boldsymbol{\zeta}_v$ with bandwidth L_v:

$$\dot{\boldsymbol{\zeta}}_v = -L_v\left(\boldsymbol{a}_{\mathrm{nom}} + \hat{\boldsymbol{d}}_v\right), \qquad \hat{\boldsymbol{d}}_v = \mathrm{clip}\big(\boldsymbol{\zeta}_v + L_v\,\boldsymbol{v},\, -\boldsymbol{d}_v^{\max},\, \boldsymbol{d}_v^{\max}\big), \tag{9.12}$$

incorporating back-calculation anti-windup after the clipping stage. A similar construction is applied to the angular momentum $\boldsymbol{p}_\omega = \mathbf{I}\boldsymbol{\omega}_b$, including the gyroscopic

correction $\boldsymbol{c} = \boldsymbol{\omega}_b \times (\mathbf{I}\boldsymbol{\omega}_b)$ and the applied moment $\boldsymbol{M}_a$:

$$\dot{\boldsymbol{\zeta}}_M = -L_M \left(\boldsymbol{M}_a - \boldsymbol{c} + \hat{\boldsymbol{d}}_M \right), \qquad \hat{\boldsymbol{d}}_M = \mathrm{clip}\big(\boldsymbol{\zeta}_M + L_M \, \mathbf{I}\boldsymbol{\omega}_b, \, -\boldsymbol{d}_M^{\max}, \, \boldsymbol{d}_M^{\max} \big), \quad (9.13)$$

applied independently per axis. Both bandwidths are set to $L_v = L_M = 2\,\mathrm{s}^{-1}$, yielding a settling time of approximately 1.5 s to 2 s.

The disturbance bounds are anisotropic and derived analytically rather than selected empirically. The lateral acceleration components are constrained by the tilt envelope itself:

$$\pm g \tan \phi_{\max} = \pm 4.57\,\mathrm{m\,s}^{-2},$$

ensuring that the bound stays consistent by construction after an envelope change — an earlier hand-written copy embedded the obsolete 14.3° limit — and a lateral estimate wound beyond it would place the tracking stationary point outside the feasible set. The vertical component is given $\pm 12\,\mathrm{m\,s}^{-2}$: the 3 kg unmodelled payload of the design requirement needs $7.5\,\mathrm{m\,s}^{-2}$ at steady hover, but the on-ground wind-up before lift-off transiently overshoots to about $11.5\,\mathrm{m\,s}^{-2}$, which railed the earlier $8\,\mathrm{m\,s}^{-2}$ clamp; on real hardware the same channel also absorbs the measured thrust-map error of about 10 %. The moment clamp is $\boldsymbol{d}_M^{\max} = (8,\, 8,\, 0.42)^\top\,\mathrm{N\,m}$: roll and pitch are sized on the payload case — 5.9 N m per axis, which 8 N m covers while staying under the 8.65 N m the airframe can produce — while yaw is sized differently and for a different reason, gravity being vertical so that a hanging payload makes no yaw moment at all, and the achievable yaw authority being only 0.842 N m; the clamp is set at half of that, mirroring the bias rule of the baseline. The isotropic 8 N m it replaced let the observer demand ten times the torque that exists, and a disturbance estimate the vehicle cannot counter is worse than no estimate.

The observer is gated on the post-warmup flag rather than on the airborne state: after the pre-arm warmup it must keep integrating even while the vehicle is still on the ground, because that is exactly what winds the vertical estimate up enough to lift an unmodelled load, and gating it on lift-off would deadlock the takeoff. While disarmed the auxiliary states are kept synchronised with the live velocity and rate, so integration resumes bumplessly the instant PX4 arms.

9.5 Software architecture

The controller is implemented as a single ROS 2 node that serves as a drop-in replacement for the two nodes comprising the baseline stack. It manages the takeoff/landing state machine, the PX4 offboard and arming handshake, and actuator output, while driving the NMPC instead of the cascade sliding-mode controller. The node is launched independently so that identical command topics can drive either stack over the same experimental setup. It subscribes to vehicle

odometry and status from PX4, joint states carrying measured arm angles, and the status channel of the arm driver. The latter acts as a gate for takeoff and triggers automatic landing upon specific servo-fault codes. User command topics are also monitored.

The node publishes normalised motor commands on the PX4 actuator interface, along with offboard heartbeats, vehicle commands, mission state, wrench data, and enable signals consumed by the energy optimiser. A health record is published at every tick, *including while idle*; its freshness serves as the liveness signal for the supervisor described in Sec. 9.7. This record contains the state code, solver status, consecutive QP-failure count, solve time, tracking error, mute and pilot-override flags, altitude datum, and a flag raised when any arm deviates more than 6° from its target. This last flag allows the supervisor's oscillation detector to be suppressed during shape changes.

The node reuses unchanged the rotor mapping, motor positions, and inertia model from Chapter 5, ensuring that both stacks share identical actuator plumbing. A four-state machine—IDLE, TAKING_OFF, HOVERING, LANDING—governs the mission logic. Control engagement is timed by a post-warmup counter rather than relying on best-effort PX4 arming feedback. Every landing, whether commanded or automatic, begins with reshaping the vehicle to the X-configuration.

Figure 9.1 shows the resulting stack as one picture: the same span as the sliding-mode architecture of Fig. 7.1, with the two cascaded loops and the separate allocation replaced by a single constrained solve.

9.6 Deployed robustness layer

The single-iteration real-time iteration scheme [40] is computationally efficient but conditional. From an infeasible or distant initial state, a single Newton step may fail, and an occasional suboptimal solve can cause abrupt jumps in rotor commands. This highlights that the nominal robustness of constrained predictive schemes is not automatic [41]. Consequently, the safety net integrated within the node is not a separate controller but a layer designed to keep the problem well-conditioned, thereby reducing failure frequency. It bounds commands so that no single bad solve can compromise airframe integrity and enables graceful recovery when failures occur. This layer acts as the first line of defence; the supervised hot-standby fallback described in Sec. 9.7 operates above it and is armed for every real flight.

Reference conditioning. The reference trajectory is never stepped directly to the raw waypoint, as large jumps in the setpoint can cause the real-time iteration quadratic program to fail. Instead, a trapezoidal governor ramps the tracked setpoint toward the waypoint with bounded speed and acceleration. The speed

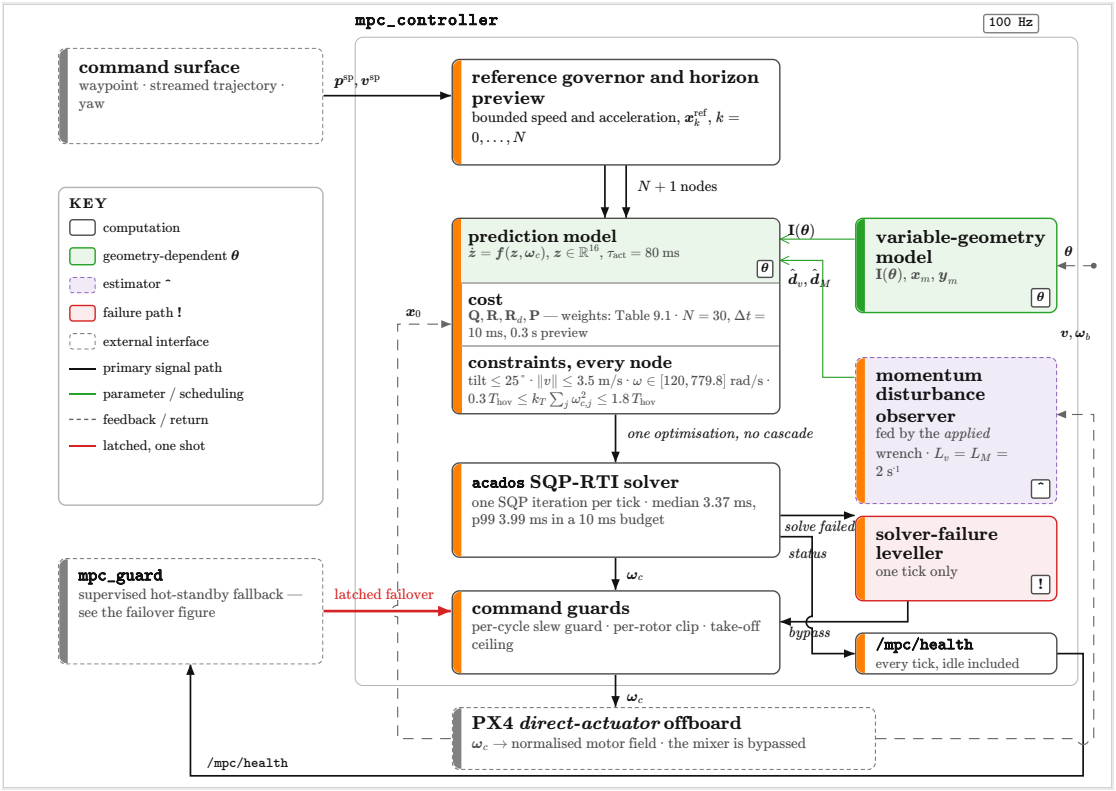


Figure 9.1: The predictive stack from the mission reference to the four rotor speeds, and the sibling of Fig. 7.1. What the baseline splits into cascaded loops and a separate allocation is here a single object — prediction model, cost and constraints — solved by one `acados` SQP real-time iteration per 10 ms tick, with the governor of Sec. 9.6 supplying the 0.3 s preview and the momentum observer of Sec. 9.4 closing the offset-free loop. Two paths leave the node: rotor commands, slew-guarded and replaced by the leveller whenever a solve fails, straight into PX4's direct-actuator interface; and the health record whose freshness alone arms the supervisor of Sec. 9.7.

limit is capped at the maximum value from which the airframe can still brake to a stop at the target location. This motion law is propagated over the entire horizon so that the preview window includes the upcoming deceleration phase. During large descents, the downward rate is separately limited to preserve vertical thrust margin. While a trajectory stream is active, each sample is dead-reckoned to the current time, ensuring the reference evolves continuously rather than in discrete steps. Upon termination of the stream, velocity ramps to rest along a continuous braking reference that is recomputed at every control cycle.

During takeoff, two additional mechanisms maintain solver conditioning while the heavy airframe accelerates slowly from the pad. A rotor-speed ceiling caps total thrust at the estimated hover thrust plus a gentle climb margin. Additionally, a lead clamp restricts how far ahead of the grounded airframe the climbing reference

may move.

Command guards and abort. A per-cycle slew guard limits the rate of change in rotor commands between ticks, preventing any single spurious solve from flooring a motor or flipping the vehicle. Velocity and body-rate odometry inputs are low-pass filtered before entering the model. In the event of a solver failure, the node falls back for that tick to a hover-thrust command augmented with a small proportional–derivative attitude leveller, allocated through the live geometry, until the solver recovers. If the airframe is lost—defined as a tilt sum exceeding 75° or non-finite state values—the node stops the motors, commands a disarm sequence, and returns to IDLE. This ensures that a crash never leaves the motors spinning.

This abort logic is strictly a last resort, as disarming in flight means losing the airframe. The supervised fallback described earlier is armed to intercept such divergences much sooner, handing the flight over to a controller that still flies. This is exactly what happened during the flight-19 accident. Finally, if PX4 exits offboard mode or disarms while the node believes it is flying, this condition is interpreted as a safety-pilot takeover. The node then stands down to IDLE and mutes all outputs, never attempting to re-arm a vehicle now under pilot control.

On the companion computer, the controller node is pinned to a dedicated core with real-time FIFO priority (`chrt -f 50`, `taskset` to one core), while the supervisor operates under its own real-time schedule (`chrt -f 40`). The latter configuration was implemented as a direct response to spurious failovers observed in simulation when the supervisor suffered from CPU starvation.

Two hardware flights were conducted on 2026-08-06 using this stack; their quantitative results are in Chapter 12. It is relevant to note here that throughout both flights, the solver never approached its computational budget. Median solve times ranged from 0.90 to 0.97 ms, while 99th percentile values stayed between 1.03 and 2.07 ms. The maximum observed time of 9.7 ms occurred entirely in the final seconds, *after* attitude divergence had already begun. Consequently, the numerical machinery was not a factor in the accident; the quadratic program failed only once the vehicle exceeded 20° of tilt.

9.7 Supervised hot-standby fallback

The per-tick leveller addresses transient solver failures but does not serve as a certified backstop. It lacks position tracking capability, and a sustained period of ill-health would leave the airframe drifting under a bare attitude hold. The remedy retains the sliding-mode controller from Chapter 7, which is already characterised and trusted, as a hot-standby fallback managed by a supervisor. This architecture is not merely proposed; it is deployed and armed on every real flight, validated

through 46 unit checks and a closed-loop simulation suite. It has been triggered once in actual operation during the flight-19 accident.

As-built architecture. A third node, designated as the guard, holds no authority over autopilot inputs and monitors health status exclusively. The launch sequence starts the sliding-mode controller and its mission manager with `standby:=true`: from that moment they subscribe to odometry, vehicle status, the land detector and the joint states, with their filters warm, and publish nothing. This is necessary because the baseline's mission-inactive path emits zero setpoints and a disarm request when in offboard mode.

The liveness contract relies on the health record defined in Sec. 9.5. Published at every tick even while idle, its *freshness* serves as the primary signal. Thus, a hung control loop is indistinguishable from a dead one, which is precisely the intended behaviour. The switchover mechanism uses a single latched message on a reliable, transient-local topic and is *irreversible*. Upon activation, the predictive controller mutes itself via the same stand-down machine used for pilot takeovers. The sliding-mode controller then adopts the flight with zero integrator states and a hold-here reference. No handshake is required because the vehicle is already armed and in offboard mode.

This design deliberately differs from classical supervised multiplexers that employ bumpless transfer in both directions. A supervisor that hands authority back to a controller it has just declared unhealthy risks re-importing the failure during handover. Conversely, a single latched transition to a trusted controller is simple enough to allow exhaustive testing. The branch that races the autopilot's own failsafe is the dead-stream one, and it is sized by a design budget rather than by a measurement: 0.20 s of tolerated staleness, one 0.01 s guard tick and 0.004 s of latch transport, 0.214 s in all, against the 0.5 s of PX4's `COM_OF_LOSS_T`. This topic also functions as a manual lever for the ground operator, with an adoption latency validated at 0.251 s. Figure 9.2 is the machine itself — five states, the gates each trip has to cross and the single latched arc out — Table 9.2 lists the trips, and Figure 9.3 puts the three measured takeovers against that budget.

Trip conditions. Table 9.2 lists the seven deployed trip conditions. The seventh was added after the flight-19 accident and is the only one that watches a *trend* rather than a level: the peak-to-peak of the linearly detrended roll over a 1.0 s sliding window, evaluated every 0.5 s. As first flown it tripped on three consecutive evaluations above 6°, each at least 1.2 times the previous one, or on two consecutive evaluations above 14°. Replayed on the accident log, only its level branch fires; the growth branch does not, and in the final simulation campaign the level branch tripped three times on decaying post-reconfiguration transients. The detector was therefore rewritten on 10 September to judge the five evaluations of a 2.5 s horizon

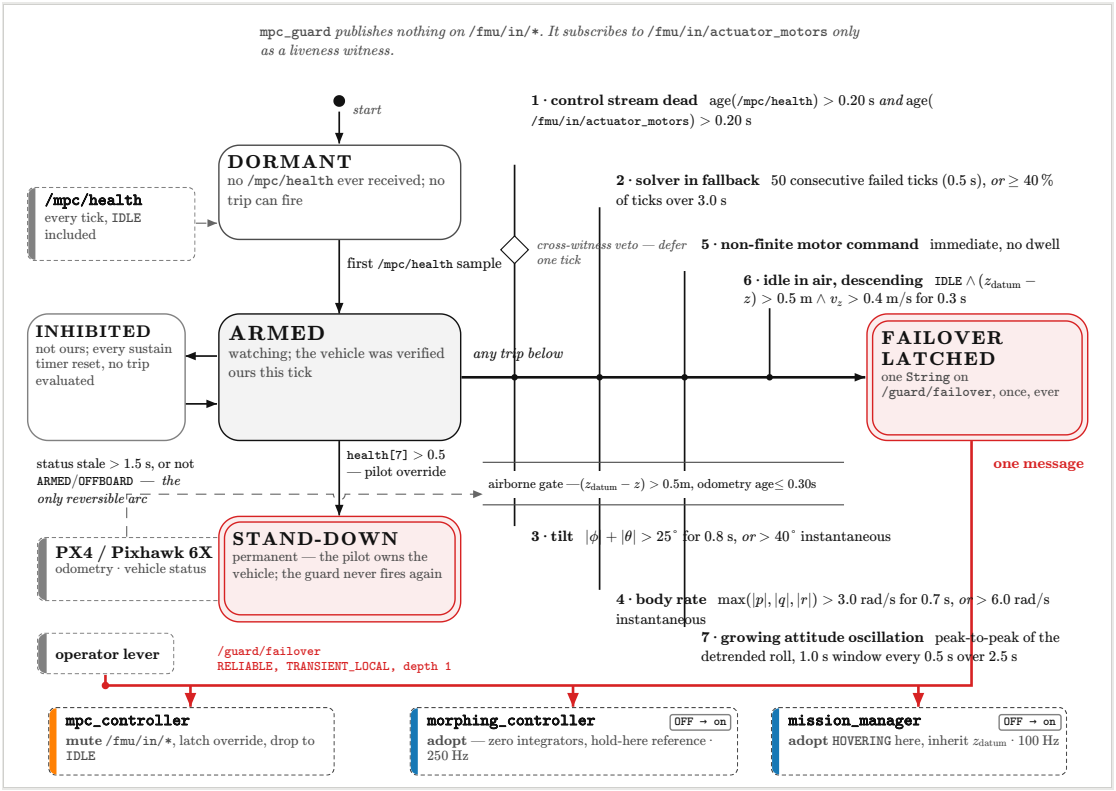


Figure 9.2: The supervisor as a state machine. Table 9.2 holds the thresholds and Fig. 9.3 the latencies; this figure shows the machine and its gating. The airborne gate drawn on the trip arcs fails *open* when the altitude datum is unset, and it is not the only gate: trips 1 to 4 and 7 also require a flight-state code, and every trip from the second on a `/mpc/health` sample younger than 0.20 s; trip 5 fires in any state, trip 6 only in IDLE. `morphing_controller` and `mission_manager` have run in standby since launch — every subscription live, no publisher — so the latch un-mutes them rather than starting them. The arc into FAILOVER LATCHED is one-way by design: a supervisor that hands authority back to a controller it has just declared unhealthy re-imports the failure with the handover.

together: all five above 3° and the last above 6°, a total growth of at least 1.8 with no single step above 2 and at least three increases out of the four, plus a hard backstop at 30° peak-to-peak on two consecutive evaluations. On the same accident log the revision fires 1.8 s before the hard body-rate trip and leaves the decaying transients alone (Sec. 12.1.6); it has not flown either. Three classes of false positive were found in simulation and are gated out: shape transitions, suppressed through the morphing flag of the health record and for 2.0 s after it clears; the take-off transient, the detector arming only in HOVERING or LANDING and only after 4.0 s in that state; and translation manoeuvres, armed only below 0.4 m s^{-1} of horizontal speed. The detector has never fired in a real flight, and its false-positive surface is

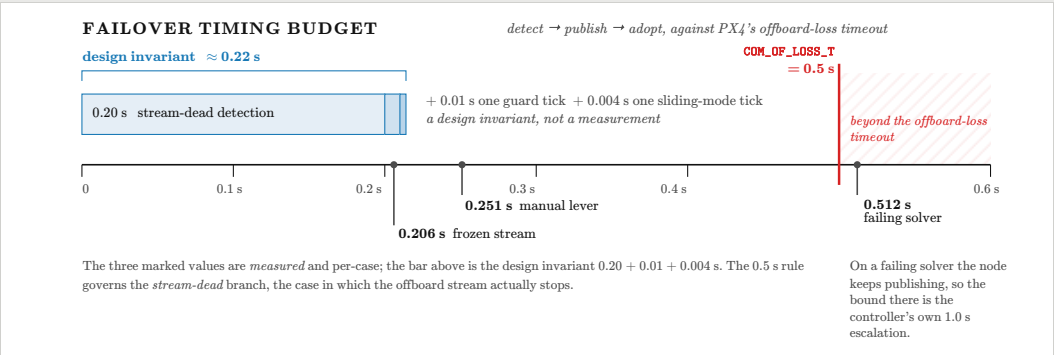


Figure 9.3: The failover timing budget and the three takeovers measured against it. The 0.22 s invariant is a design sum, not a measurement — 0.20 s of tolerated staleness (Table 9.2), one 0.01 s guard tick and 0.004 s of latch transport, 0.214 s in all — and it sizes one branch only. That branch, the dead stream, is the one racing PX4's `COM_OF_LOSS_T` of 0.5 s, and it was measured at 0.206 s. The other two marks are separate cases: the manual lever at 0.251 s, and the persistently failing solver at 0.512 s, which spends 0.5 s of that proving the failure persistent while the offboard stream stays alive, so the autopilot's timeout never starts (Sec. 12.1.6).

Table 9.2: Deployed failover trip conditions. Physical trips are gated on the airborne state; no trip fires on stale odometry, as the fallback controller utilises the same odometry source, and a pilot override permanently stands down the supervisor.

Condition	Threshold
Control stream dead	Health *and* actuator streams stale for 0.20 s
Solver in fallback	Continuous QP failure for 0.5 s, or $> 40\,\%$ of ticks over 3 s
Tilt	25° sustained for 0.8 s, or 40° instantaneous
Body rate	$3.0\,\mathrm{rad\,s^{-1}}$ sustained for 0.7 s, or $6.0\,\mathrm{rad\,s^{-1}}$ instantaneous
Non-finite motor command	Immediate
Idle-in-air descent	> 0.5 m above datum with $v_z > 0.4\,\mathrm{m\,s^{-1}}$ for 0.3 s
Growing attitude oscillation	See text (added after flight 19)

not proven exhausted: it is a hover-only instrument by construction.

9.8 Limits of the rigid-arm prediction model

The prediction model is morphing-aware in its *geometry*: the inertia and the lever arms follow the measured arm angles into every solve. It is rigid in its *structure*: the arms are modelled as rigid bodies rigidly attached to the airframe, so the model contains no structural mode — in particular not the arm/servo compliance mode

near 6 Hz that had already announced itself as stick–slip in the sliding-mode flights and was later identified as a mode of the mechanism (Sec. 6.8). Flight 19 is the measurement of what that omission costs.

The accident flight (Sec. 6.8) shows that in the T configuration the real plant carries about three times the modelled gain from commanded moment to roll rate near 3 Hz, while at 1 Hz the model is accurate; the discrepancy is not a delay but a resonant amplification. Closing the measured plant response with the deployed controller reproduces the accident quantitatively: at the observed 3 Hz limit cycle the loop gain is $|L| = 1.33$ at a phase of $-171°$ — the Barkhausen condition, met because the compliance mode had consumed the entire design margin. The model did not merely miss the oscillation; it certified as stable a loop that was not.

Two remedial paths were tried in the day after the accident, and only one survived measurement. Detuning the attitude authority through the cost fails for the structural reason established in Sec. 9.3.1 — the roll/pitch stiffness does not come from the attitude weights, and the lever that does reduce it prices slow manoeuvres and fast disturbance rejection alike, so the softened controller either tripped in gusts or parked metres from the commanded origin, stable yet ineffective. The viable solution lies within the model itself: restoring the actuator lag to its measured small-signal value of 80 ms prevents the controller from over-compensating in the 2 Hz to 4 Hz band. Against the measured plant response, this adjustment reduces the loop gain at full authority in T from 1.33 to 0.78, providing a margin of 2 dB to 3 dB. This is notably lower than the 6 dB margin assumed by the rigid-arm design, representing a calculated risk that must be covered operationally rather than structurally.

Pending a dedicated gate flight, an interim policy provides operational cover: trajectory execution under the predictive controller is restricted to the X configuration. Reconfiguration tasks are delegated to the sliding-mode controller, whose lower attitude bandwidth imposes less demand on the resonant band. Three deployed remedies frame the planned re-entry into the other shapes — the corrected lag constant, the growing-oscillation detector of Sec. 9.7, and a hard attitude gate that starts a trajectory stream only after the roll peak-to-peak has stayed below 4° for two seconds — but the gate flight itself exists as three written cards, none of which has been flown. Until that gate is green, the claim this chapter can make is bounded honestly: tracking under the predictive controller on hardware is demonstrated in the X configuration only, on a single flight, and outside X it is not demonstrated at all. The structural remedies remain open and are detailed in Sec. 11.7. These include implementing a servo with closed-loop current control and higher stall torque to raise the mechanical impedance of the joint, or adopting a second-order attitude model that encompasses the 6 Hz mode, albeit at the cost of a stiffer and larger optimal control problem. A residual model risk is also documented: actuator lag exhibits amplitude dependence (Table 6.2), and the large-step take-off

regime with the current propellers has never been measured. Consequently, the gate protocol prioritises testing the take-off phase first, with a legacy-lag fallback prepared for contingency.

Two properties of the deployed solver are worth showing before the comparison, because they bound what the hardware evidence can and cannot say (Fig. 9.4): the solver never ran out of time, and the one complete trajectory it flew is the best of the campaign.

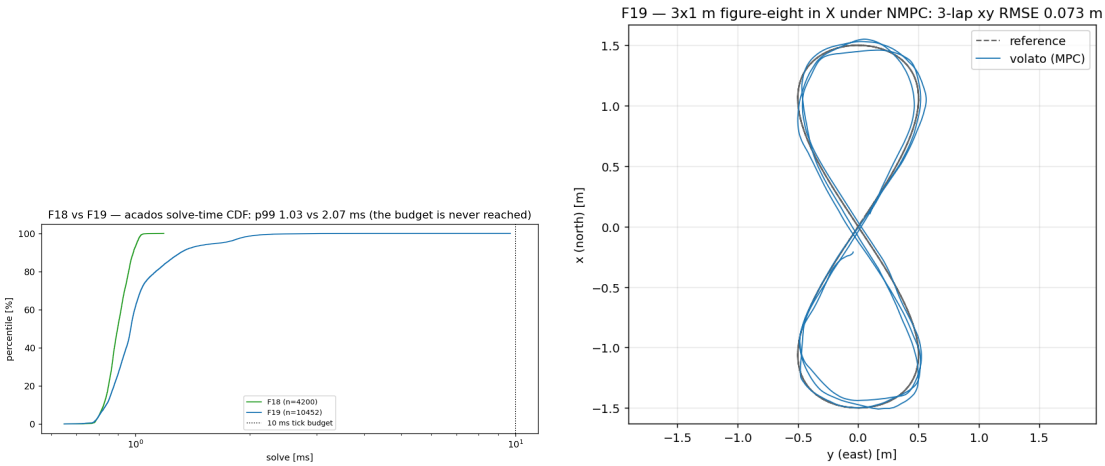


Figure 9.4: Left: the cumulative distribution of the acados solve time on the companion computer, over the two flights that ran the predictive stack. The single-iteration real-time iteration keeps the 99 % point far inside the 10 ms budget of the 100 Hz loop: computation is not what limits this controller on this hardware. Right: the 3×1 m figure-of-eight flown in the X configuration on Flight 19 — three laps, and the best tracking result of the whole campaign. Both panels come from real flight, not simulation; the solver failures that appear in the accident log are a *consequence* of the attitude divergence of Sec. 6.8, not its cause, and they begin after the tilt, not before it.

Predictive control of multirotors is by now an established practice — with ROS implementations [42], perception-aware variants [43] and learned residual dynamics [44] — and the value of the present chapter is not the method but what it is asked to do: solve against a plant whose inertia and control effectiveness are themselves signals.

9.9 Comparison with the sliding-mode baseline

Table 9.3 summarises the differences between the two control stacks, both of which share the rotor model, the variable-geometry inertia, the lever arms and the actuator interface of Chapter 7. The systematic A/B comparison in simulation is the subject of Chapter 12; the hardware evidence is that of Chapter 11 and Chapter 12, and

Table 9.3: Comparison of the two control stacks. The measured-tracking row reports the same figure-of-eight trajectory on hardware; each entry is a single flight per configuration.

Aspect	Sliding-mode baseline	Predictive controller
Structure	two cascaded loops plus allocation	a single optimisation plus allocation
Feedforward, preview	piecewise, no torque feedforward; no preview	implicit and optimal from the model; full reference window
Constraints	enforced by clipping	explicit and anticipated over the horizon
Steady error / payload	moment-bias integrator with deadband and freeze	offset-free by construction (observer-fed)
Morphing	decoupled energy optimiser; allocation follows the CoG	geometry parameters fed to every solve
Model fidelity	no prediction model to be wrong	margin depends on the model: an unmodelled structural mode cost the airframe outside the nominal shape
Tuning, compute	gains; algebra per tick	weights and horizon; a nonlinear program per tick
Failure mode	always produces a command	solver can fail; recovered per-tick by the leveller and, on sustained ill-health, by the supervised fallback

the last row of the table condenses it.

The benefits address the specific limitations of the baseline: entry transients are automatically planned within tilt limits, steady-state hold offsets are eliminated via an offset-free design rather than through an integrator prone to deadlock, and payload moments are managed through an explicit, continuously estimated disturbance. On hardware, these gains are quantified — in the nominal shape the predictive controller flew the most accurate trajectory of the project — and so is the price, of which flight 19 is the unit: every payoff is conditional on the fidelity of the prediction model, and where the model was wrong the baseline kept the vehicle and the predictive controller lost it. The comparison is therefore not a ranking but a trade.

Chapter 10

Simulation framework and experimental design

This chapter describes the simulation and validation pipeline used to develop and test the control stack, and the design of the experiments whose results are reported in Chapter 12. The pipeline spans software-in-the-loop testing against a nominal and against a deliberately disturbed plant, hardware-in-the-loop and software-in-hardware testing, each validating a different aspect of the system. The guiding principle is a clean separation of concerns: each environment is trusted only for the question it can actually answer, and the fidelity map of Sec. 10.1.5 states explicitly which question that is. The chapter then documents the software architecture of the control stack (Sec. 10.2), common to both controllers, and closes with the experimental design (Sec. 10.3). The framework was not only designed but exercised: between 2026-07-28 and 2026-08-06 it ran six full campaigns — 21 retuning runs against the disturbed twin, 10 fault-tolerant-morphing runs, 22 shape-matrix runs, 29 predictive-stack runs, 14 pre-flight suite runs and 11 post-crash diagnostic runs — every one adjudicated by the harness verdict of Sec. 10.3.3.

10.1 Simulation pipeline

The platform is exercised through three nested validation venues of increasing hardware realism but, importantly, *decreasing* flight-dynamics fidelity. Software-in-the-loop (SITL) runs the autopilot, the physics and the control stack entirely on the development PC and is the primary venue — flight-faithful, however, only against the disturbed twin of Sec. 10.1.2, since in the nominal configuration plant and controller share one parameter module and the model error is zero by construction. Hardware-in-the-loop (HIL) moves the real control stack onto a separate companion computer while keeping the same simulated plant, exposing real loop scheduling

and a real-machine transport. Software-in-hardware (SIH) runs the autopilot's own internal simulator inside the real flight controller, exercising the exact transport, firmware and failsafe chain of the finished drone. Real-flight fidelity is not delivered by any single environment: it is assembled from the closed-loop dynamics validated in SITL, the parameters identified in Chapter 6 and the incremental flight cards of Chapter 11, which alone introduce real sensors and propeller-induced vibration.

10.1.1 Software-in-the-loop

SITL runs on the development PC across three concurrent processes. PX4 SITL is launched together with Gazebo (Harmonic, `gz-sim` 8.11.0) through the PX4 build tree [45], selecting the custom `morphing_drone` airframe; the control stack runs on ROS 2 Jazzy and the autopilot is built from a local fork of PX4 at `v1.17.0-alpha1`. Gazebo loads the bespoke morphing-drone description — four revolute arm joints, four rotor joints and a downward `gpu_lidar` standing in for the rangefinder — together with the physics, the motors and the simulated sensors; the two prismatic payload joints and the 3 kg slider of the original description were removed on 2026-07-21, when the real airframe was confirmed to fly with no payload. The simulated motor plant applies the same quadratic thrust law as the real propellers, but the coefficients used during the early campaign were wrong by a factor 1.78: the bench thrust coefficient was falsified and re-derived from flight on 2026-07-27, which moved the simulated hover rotor speed from 361.8 to 482.9 rad s^{-1}; under the old constants the aircraft could not take off, by construction.

A Micro XRCE-DDS agent carries the PX4 topics over DDS on the local host [26]. Because everything runs on `localhost` the transport is exactly zero — there is no delay block anywhere in the SITL chain, and Gazebo's motor plugin can only express a first-order *lag*, which attenuates magnitude, where a pure delay adds phase and attenuates nothing. The real companion-to-autopilot path was measured directly from a flight bag at about 30 ms round trip (Sec. 6.3.1), and the framework therefore injects exactly that value controller-side, in SITL only: on the real vehicle the delay is physically present, and injecting it there would count it twice.

A point of architecture recurs throughout the results: the morphing control is *not* PX4's internal flight controller but a custom companion-side node that reads the autopilot's odometry and publishes direct motor commands, performing its own geometry-aware allocation, PX4 retaining only state estimation, sensor input/output and actuator output. One claim must be dismantled before any SITL result can be trusted: the quadratic thrust law is *not* what makes SITL faithful. Plant and controller read the same parameter module (Sec. 10.2.1), so in the nominal configuration the model error is zero by construction and any controller appears to work — while flight 5 measured 0.593 N m of roll trim in the simplest configuration that exists, arms bolted in X, where the model says exactly zero. The

disturbed twin exists precisely to restore a non-zero, *measured* model error, and only results obtained against it should be read as robustness claims. One deliberate residual mismatch is kept even in the nominal plant: the composite inertia of the Gazebo links sits 5 % to 12 % above the CAD tensor the controller consumes, in the conservative direction.

Wind deserves its own honest history, because for part of the project it did not exist: until 2026-07-28 the wind-effects plugin was loaded nowhere, so the windy-world "spot checks" of that period exercised no wind on the airframe at all. Since then the plugin lives in the simulator's server configuration — deliberately not in the world file, because Gazebo replaces the server plugins with any declared inside a world, silently removing physics and sensors — and the windy world seeds a mean horizontal wind of $3.0\,\mathrm{m\,s^{-1}}$, matching the field rule for the real aircraft, with sinusoidal horizontal gusts of $\pm 50\,\%$ amplitude at a 3.3 s period, Gaussian noise and a slow $\pm 15°$ wander of the wind direction over a 17 s period. The force scaling was calibrated against the translational drag measured in flight, taking the top of the measured band so that the twin is worse than the truth. One property must be declared plainly: the 3.3 s gust period falls inside the measured phugoid band, so the windy world is a deliberate *resonance test*, not generic turbulence, and the prescribed sweep over gust periods has not been executed.

10.1.2 The disturbed twin

The nominal SITL plant cannot measure robustness: validation against it is structurally tautological. The disturbed twin, introduced on 2026-07-28, restores the missing error. An injector process, launched alongside the *unmodified* flight stack, perturbs the plant with a seeded, per-rotor, per-axis model error whose magnitudes are anchored to flight measurements: the injected error is not adversarial noise but the measured distance between the model and the aircraft. Every simulation result dated before 2026-07-28 was necessarily obtained against the nominal plant and is labelled as such wherever it is cited in Chapter 12.

The injection is split across two channels, assigned by physics. Thrust-side errors, including per-rotor thrust-coefficient scatter and the propeller damage described in Sec. 10.1.3, are applied by intercepting the motor-command topic and republishing it scaled, so that the error passes through the true actuation chain of motor time constants, saturations and command map. Genuinely external disturbances, such as the moment from a centre-of-gravity error, the velocity-antisymmetric aerodynamic moment, and the torque-coefficient split between rotor directions, are applied as an instantaneous wrench on the base link at the 250 Hz physics step. The error distribution is: per-rotor thrust scatter $\sigma = 0.07$ (flight 5's chipped propeller sat 12 % to 14 % low, which 2σ covers); torque scatter $\sigma = 0.10$; a centre-of-gravity offset $\sigma = 0.014\,\mathrm{m}$; arm-angle play $\sigma = 1.5°$; and the velocity-antisymmetric

moment at 0.25 N m per m s^{-1}, the top of its measured band. Of these five, the centre-of-gravity sigma is the *only fitted parameter*: it was chosen so that the median injected planar moment in X matches the 0.698 N m that flight 5 measured, and it carries a declared falsifier: if flight 5 turns out to have been flown in wind, the same wind is counted twice and the value must be re-anchored on flight 6. The falsifier has never been executed. One detail was paid for in runs before it was understood: the centre-of-gravity trim moment is gated on being airborne, since applied on the ground it capsized the vehicle during take-off with the integrators still empty, and two runs whose disturbance authority differed by a factor 3.4 then produced identical trajectories; the A/B was measuring the take-off artefact.

Three layers guarantee that the injector cannot reach the real vehicle: its directory is not a ROS package, so the build system never installs it on a flight machine; the injector process refuses to start whenever the real-hardware flag, the locked-arms mode or the hardware ESC table is present; and a dedicated test asserts that the flight launch file contains neither the topic remap nor any mention of the injector. One cost is declared rather than hidden: the motor channel adds a middleware hop the real aircraft does not have, measured at 1 ms to 4 ms (about 1.5° of phase at the attitude crossover, against a 23.9° margin), and a wrench-only channel selection exists for experiments where that matters.

One experiment decides whether the twin is credible: reproducing, with the exact controller build that flew, the slow position oscillation the real aircraft exhibits in hover (Table 10.1). With only the constant flight 5 trim moment injected the twin does *not* oscillate — it decays and slides away 141 m where the aircraft drifted 21 m: the DC deficit and the oscillation are two independent defects, and an integrator cannot cure a plant mode. Adding the velocity-antisymmetric term produces the mode, and its frequency was predicted *before* the runs from the phugoid model $\tau\ddot{v} + \dot{v} + (g\,c/K)\,v = 0$, with K and τ fixed by the attitude-loop tuning rather than fitted: 0.271 Hz predicted against 0.29 Hz measured at $c = 0.20$. What does not close is the damping: the twin decays at $\times 0.72$ to 0.79 per half-cycle in every condition tried, against $\times 0.98$ to 1.07 in flight — about 25 % of energy per half-cycle unaccounted for. Two hypotheses were refuted (it is not the aerodynamic coefficient, and it is not the bias-estimator pump: switching the estimator off *reduces* the damping, so the bias path is net stabilising); the surviving suspect is gusts, which the twin could not represent at the time of the comparison.

Two silent-failure modes of the injector are reported because they nullified runs before being caught, and both generalise beyond this project. A quality-of-service mismatch (a transient-local subscriber does not match a volatile publisher in DDS, and the middleware reports neither error nor warning) meant that for a period the twin *never injected*; the remedy is not only the corrected profile but a heartbeat that declares the run FATAL when the message counters are still zero. And the injector, spawned outside the launch process tree, survived every shutdown: on one

Table 10.1: Validation of the disturbed twin against the flown aircraft: hover with the flight 5 constant trim moment injected, plus the velocity-antisymmetric term with coefficient c (in $\mathrm{N\,m\,s\,m^{-1}}$). The twin reproduces frequency and drift with a measured coefficient, but damps ∼25 % too much per half-cycle.

Run	c	Drift at 21 s	Frequency	Growth/half-cycle
Constant trim only	—	141 m	0.09 Hz	×0.33
Velocity term	0.20	23.0 m	0.29 Hz	×0.73
Velocity term	0.25	19.8 m	0.31 Hz	×0.72
Bias estimator off	0.20	41.5 m	0.28 Hz	×0.79
Flight 5 (real)		21.1 m	0.29 Hz	×0.98
Flight 6 (real)		14 m	0.32 Hz	×1.07

occasion seven live instances were found publishing wrenches onto the same link, and wrenches *sum*, producing about 0.5 N m of parasitic yaw on a vehicle that was not yet armed. The harness now hunts it down at teardown.

10.1.3 The fault injector and the severity scale

The fault-tolerance experiments require a rotor fault, and the framework injects a deterministic one anchored to the propeller damage that flight 9 actually suffered. The `f9` profile removes 39 % of the thrust of rotor 2 (the measured 3.78 N deficit) and a further 28.3 % of its torque coefficient beyond what the thrust path already carries. This figure is derived rather than assumed: the yaw bias in the flight 9 log sat pinned at its 0.157 N m clamp, of which the thrust deficit alone explains 0.091 N m, and the remainder divided by the healthy rotor torque gives the extra torque deficit. This is physically consistent, since the lost blade tips dominate torque even more than thrust.

The severity scale is deliberate and asymmetric. Scale 1.0× is the *official* injection for every run, representing the flight as it was flown; it is conservative with respect to the physical re-fly of the damaged propeller, which measured about 0.65 of the modelled severity. Scale 1.5× is the declared envelope, legal only for in-air engagement: a 1.5× step from the ground remains a failed take-off even with the trapezoidal take-off ramp introduced later, which identifies it as an *authority* limit of the fixed frame, not a reference-shaping problem. Scale 2.0× is never flown on the fixed frame: the healthy-rotor thrust ceiling (5.6 N) is below the hover share it would need to carry (9.7 N), so the experiment would measure arithmetic, not control. Two handles shape the engagement (a step T seconds after the first lift-off, replicating the real event, and a linear ramp over R seconds), separating death-by-transient from death-by-steady-state, a distinction that Chapter 12 shows to be the entire story of the fixed-versus-morphing comparison.

10.1.4 Hardware-in-the-loop and software-in-hardware

In the poor-man's HIL arrangement, PX4 SITL and Gazebo keep running on the PC while the real control stack runs on a separate companion computer connected over a LAN link through the same uXRCE-DDS transport, adding a real separate-machine transport and real loop scheduling. Command staleness on the motor-command topic was measured at about 28 ms on a Raspberry Pi 4 and about 8 ms on the LattePanda N100 that became the flight companion, bracketing the 30 ms round trip later measured on the real aircraft. The sliding-mode controller had to be degraded to 100 Hz in HIL on the Pi 4 and was subsequently flown at the full 250 Hz on the N100. A pointed demonstration of staleness sensitivity came from the predictive stack: on the N100 hover was rock solid, yet a long aggressive cruise oscillated into a flip that tripped the tilt abort, while re-flying the identical cruise in clean SITL landed cleanly on all five attempts. The HIL sessions distilled operating rules carried unchanged into real flight: respawn a fresh simulator for every flight and cross-check its ground truth rather than the autopilot's estimate; run no middleware CLI commands on the companion while it flies; publish every high-level command with retries rather than once; and start a fresh DDS agent for every autopilot boot.

In the SIH arrangement, PX4's internal simulator runs *on* the real flight controller with its outputs inhibited, while the companion runs the real control stack over the finished drone's actual Ethernet link (the exact transport chain of the completed aircraft). It confirmed a real closed loop at 1 ms to 2 ms, the versioned status feedback used by the handshake, the pilot-override, disarm and kill latches on the real firmware, and the offboard-loss failsafe. It does *not* validate closed-loop flight quality: both controllers fail to hover on SIH while both fly in Gazebo, and that two independent, very differently structured controllers fail on the same plant yet fly on another is the decisive cross-check that the cause lies in the SIH *plant*. The investigation refuted with data every competing hypothesis (motor geometry, inertia and yaw coupling, the thrust curve, chattering, estimator windup, rate-feedback delay and communications), leaving a residual effective-authority gap of order 20 % in the internal simulator's dynamics. The flight campaign then resolved the concern; the sliding-mode stack flew the real aircraft across thirteen flights without any trace of the SIH instability. The SIH module is deliberately not built into the firmware the aircraft flies, so these results are a one-off bring-up artefact.

10.1.5 What each environment validates

Figure 10.1 and Table 10.2 consolidate the discussion into a single fidelity map.

It is the interpretive key for the whole of Chapter 12: every figure of Part I of that chapter comes from SITL and therefore characterises control logic and

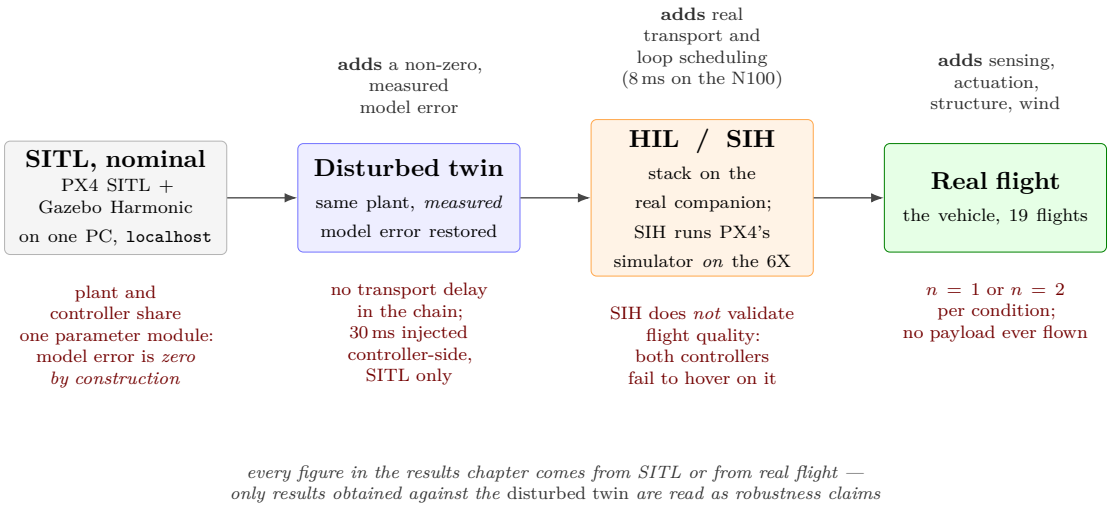


Figure 10.1: The four environments, ordered by what each one adds to the loop. Reading them in this order is what keeps the results honest: the nominal SITL plant and the controller share one parameter module, so its model error is zero by construction and *any* controller works against it; the disturbed twin restores a non-zero, measured error and is the only environment against which a robustness claim is made; HIL adds real transport and scheduling, while SIH is a bring-up instrument that does not validate flight quality; and only real flight adds sensing, actuation, structure and wind — which is where the arm-compliance mode of Sec. 6.8 lived, invisible to everything to its left.

closed-loop dynamics, not transport, sensing or actuation. Three real risks that none of these venues surfaces were later paid for in flight and belong in the map explicitly: servo stick-slip resonance near 6 Hz coupling into roll, shown across flights 14–16 to be a structural mode and not a profile artefact; arm and servo compliance, which multiplies the plant gain in the T configuration and was the root cause of the flight 19 crash (Sec. 6.8); and propeller-induced IMU vibration, handled at bring-up by gyroscope notch filtering and mechanical isolation of the flight controller.

Table 10.2: Fidelity map of the validation environments. Increasing hardware realism (top to bottom) is paid for by decreasing flight-dynamics fidelity. The nominal SITL row validates logic, not robustness: robustness claims require the disturbed-twin row.

Environment	Validates	Does *not* validate
SITL (nominal)	Control logic, mission sequencing and closed-loop flight dynamics (hover, attitude, tracking, the X/O/T/H morphs, the optimiser benefit), with plant and controller sharing one parameter module.	Robustness of any kind (zero model error by construction); real transport, firmware or failsafe timing; real sensor noise and vibration; real ESC/actuator behaviour; the compliance of the real arms.
SITL + disturbed twin	Robustness against a *measured* model error (rotor and torque scatter, CoG offset, arm play, velocity-antisymmetric moment); gusty wind; the deterministic rotor fault at 1.0× and 1.5×; a 30 ms transport delay.	Arm/servo compliance — the root cause of the flight 19 crash; the speed-dependent actuator lag; the live gust as seen by the airframe; and it damps the phugoid ×0.72 to 0.79 per half-cycle against ×0.98 to 1.07 in flight.
HIL	The full real stack on a real companion against a live autopilot; loop scheduling, jitter and solver timing under load; controller sensitivity to command staleness.	The exact flight transport; real flight-controller sensors, firmware or failsafe.
SIH	The real transport, handshake, failsafe and latch chain on the real flight controller: Ethernet closed loop at 1 ms to 2 ms, versioned status feedback, override and kill latches, offboard-loss failsafe.	Closed-loop flight quality: both controllers fail on SIH yet fly in Gazebo, isolating the cause to the simplified SIH plant. Not re-runnable on the flight firmware, and later contradicted by thirteen clean sliding-mode flights.

10.2 Software architecture of the control stack

The control stack is a graph of middleware nodes that, from the user's viewpoint, remains identical across the sliding-mode and predictive variants: the same command surface drives either controller, enabling direct A/B comparisons. The sliding-mode stack splits the controller into an outer reference/mission node and an inner attitude/altitude node, whereas the predictive stack runs a single controller node while keeping the two sliding-mode nodes alive but muted as a hot standby behind a failover watchdog. Both stacks share the geometry optimiser and the arm driver.

Six nodes make up the catalogue of the two stacks (Fig. 10.2): the sliding-mode flight stack runs four of them, the guarded predictive stack all six. `mission_manager`

(100 Hz) acts as the sliding-mode outer loop: it manages the mission state machine, horizontal position control, acceleration-to-tilt conversion, reference generation, and arm-target dispatch. `morphing_controller` (250 Hz) serves as its inner loop: handling inertia-scheduled attitude and altitude control, CoG-aware allocation to rotor ω^2, moment-bias estimation, offboard/arm handshake, and the safety supervisor, while publishing motor commands directly. The predictive variant is centred on `mpc_controller` (100 Hz), a standalone NMPC node with its own take-off/land state machine, momentum-based disturbance observer, and arm-aware predictive solve. Supporting this is `mpc_guard` (100 Hz), a one-shot latched failover watchdog that hands flight control to the muted sliding-mode nodes in case of failure; SITL trip latencies in the August campaign were 0.202 s on a frozen stream, 0.515 s on QP failure, and 0.251 s on the manual lever, with zero false positives in null cases; the final-campaign values are those of Fig. 9.3. Finally, `arm_optimizer` (1 Hz) is the geometry optimiser, acting only while hovering and enabled, and blind to the fault it compensates; `gz_arm_driver` (100 Hz) rate-limits absolute arm commands, forwards per-joint targets to the simulator, and republishes measured joint angles from ground truth. On hardware, the equivalent `dynamixel_driver` drives the smart servos over a serial bus.

One operating rule attached to the simulation arm driver was established after four lost airframes and generalises to every simulator consumer: Gazebo topics are *world-scoped*, so the world name must be resolved (from the environment, then by discovery on the live topic list) rather than hard-coded. A hard-coded feedback path rendered the driver blind in the windy world while the joint *commands* (which are model-scoped) remained active: the plant morphed while the stack still allocated for the starting shape, causing the airframe to be lost about 9 s after the first shape command in four out of four runs. This failure is asymmetric and silent; the half of the interface that continues working precisely obscures the half that has failed.

Both stacks expose an identical user-facing command surface on a small set of `/cmd/*` topics, allowing experiment scripts to issue the same high-level commands regardless of the running controller: an action (take-off, land, reset, idle), a waypoint, an absolute arm-angle set, a per-joint angle set, a discrete shape, an optimiser enable, an arm-mode selection and a streamed trajectory setpoint. One further operator input sits outside that namespace: the manual failover lever, a single message published reliably and transient-locally (so that it reaches even a subscriber that connects late) to hand the flight from the predictive controller to the sliding-mode standby from the ground station. An operational rule, learned on hardware in flight 17, applies to every command topic: commands must be published repeatedly rather than once, since middleware discovery races can silently swallow single messages.

Actuation bypasses the autopilot's mixer and control allocation entirely. Both controllers request a direct-actuator offboard mode [46] and publish their own

allocated motor commands; each performs its own CoG- and arm-aware allocation of total thrust and the three moments into rotor ω^2, then maps ω to the normalised motor field through the centralised parameter source. Because the autopilot's DDS bridge republishes motor commands with last-writer-wins semantics, both nodes *mute* all outbound traffic under pilot override or failsafe, rather than merely streaming zeros, ensuring that a stale command cannot leak through. Autopilot feedback arrives on out-topics: vehicle odometry serves as the primary state source, while the versioned vehicle status drives the handshake and the override/kill watchdog. Separately, a geometry-feedback path carries measured joint angles and, in simulation only, measured centre-of-gravity and rotor positions taken from ground truth; the sliding-mode controller prefers this measured geometry when fresh, falling back to model-based geometry on real hardware where the measured topic does not exist.

10.2.1 Launch configurations and the parameter source

Four launch files select the control architecture, and exactly one runs at a time: the *sliding-mode stack* (mission manager, inner controller at 250 Hz, optimiser and arm driver); the *native-PX4 baseline*, which deliberately omits the inner controller and the optimiser, leaving the autopilot's own attitude and position controllers in charge of the actuators and providing the baseline of Chapter 12; the *predictive stack*, in which the predictive node takes the actuators while the two sliding-mode nodes start *muted as hot standby* alongside the failover watchdog, and which also owns the solver's runtime environment and the solver overrides — including the actuator time constant and the Levenberg regularisation, which since 2026-08-06 carry the fix for the flight 19 oscillation ($\tau_{\text{act}} = 0.08$ s, Levenberg 0.01) — overridden back to 0.15 s for the final SITL campaign, in which the model is matched to the simulated plant (Sec. 12.1.7); and the *disturbed-twin wrapper*, which *includes* one of the flight launches unmodified — the flight path is literally the same file, not a copy that can diverge — and adds the model-error and fault injector alongside it, parameterised by profile, scale (with `scale:=0` giving a clean twin *through the same launch*, so that an A/B differs by exactly one variable), seed (recorded with the run: the seed *is* the model error), engagement mode, channel selection and stack. This vanilla-versus-custom distinction is what makes the baseline comparison fair: airframe, simulator and command stream are held fixed, and only the node that owns the actuators changes.

A single module is the source of truth for the plant: every physical and propulsion constant, the rigid mass model and the derived inertia, the centre-of-gravity, inertia and allocation model and the ESC command mapping. Both controllers and the optimiser import it, so that simulator, controllers and optimiser share one physical-parameter source. The payoff is real for the *controller* side, but it is not true of

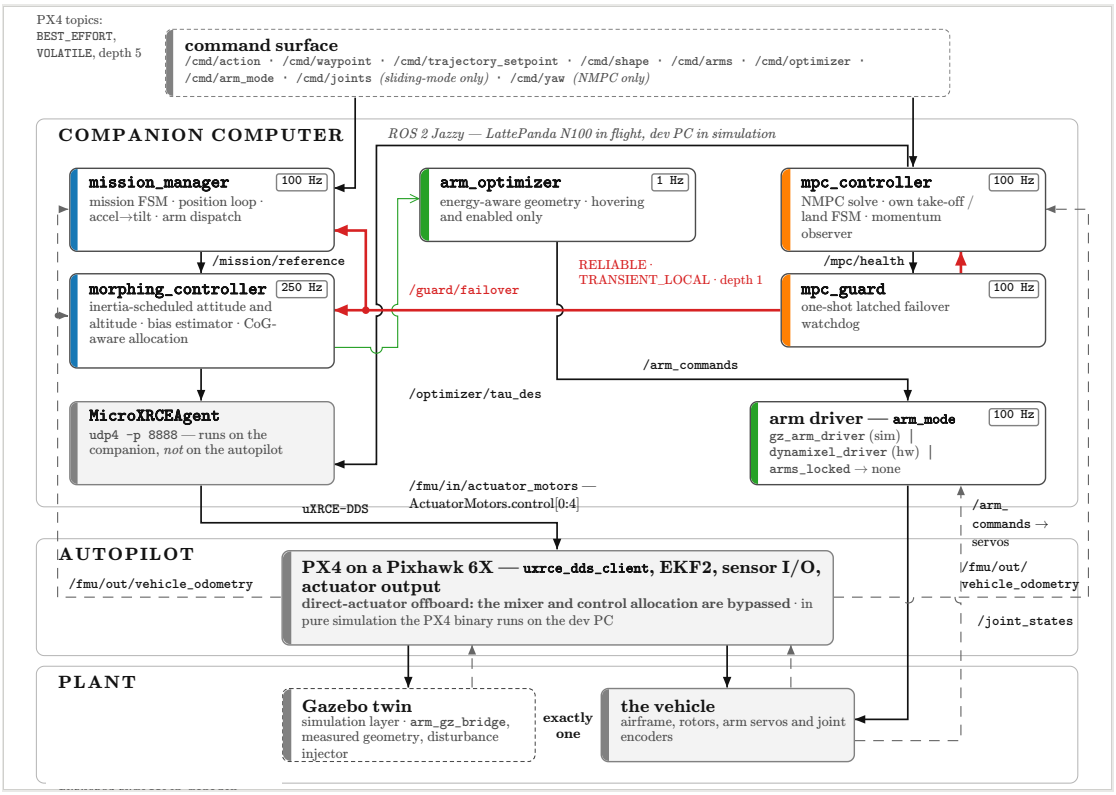


Figure 10.2: The nodes of both stacks, their rates and the topics between them. The sliding-mode flight stack runs four ROS nodes — `mission_manager` at 100 Hz, `morphing_controller` at 250 Hz, `arm_optimizer` at 1 Hz and the arm driver — and the guarded predictive stack six, adding `mpc_controller` and `mpc_guard` at 100 Hz while the two sliding-mode nodes stay alive but *muted* as a hot standby, un-muted by the red `/guard/failover` latch. The two launch files are mutually exclusive, so the stacks never own `/fmu/in/actuator_motors` at once, and one command surface drives either of them — which is what makes the A/B comparison of Chapter 12 a controlled one. The morphing path never touches the autopilot.

the plant as a whole: geometry is a *lockstep* quantity across four artefacts that must change in one commit — the kinematic layer, the parameter module, the Gazebo description and the PX4 airframe file. The adoption of the CAD-anchored geometry on 2026-07-31 is the worked example, with arm length, rotor stagger, the CAD inertia tensor and the airframe rotor arm all moved together. Worse, the Gazebo description lives inside two nested upstream submodules with no push rights, so its edits do not travel with the repository: they are carried as cumulative patches and must be re-applied on a fresh clone. This is a declared fragility of the evidence base, not a solved problem — a lidar mounting offset was silently lost exactly this way.

10.3 Experimental design

The experiments evaluate the control stack along six axes: stability across configurations; dynamic tracking; the geometry optimiser *as an authority and energy allocator* rather than as a disturbance rejector — a reading the campaign measured to be false, since at 1.0× fault severity the optimiser does not improve tracking but does unload the damaged rotor and halve the actuation trim; robustness to a non-zero model error and to gusty wind; fault-tolerant morphing under the deterministic rotor fault of Sec. 10.1.3; and controller failover. A native-PX4 baseline provides context, and the sliding-mode-versus-predictive A/B runs through all of the axes. Each experiment is automated end-to-end, records a labelled data set with full provenance, and is reduced by a dedicated analysis script to a fixed set of metrics.

10.3.1 Metrics

All metrics are computed by the analysis scripts directly from the recorded logs, using a single common time origin so that cross-topic quantities stay aligned. Table 10.3 defines them; these are exactly the quantities reported in Chapter 12.

Table 10.3: Metrics used throughout the experimental evaluation. Roll and pitch are obtained from the odometry quaternion; "drift" is the planar distance from a reference position whose meaning (hover origin, phase start or path reference) depends on the experiment.

Metric	Unit	Definition
Peak tilt	°	Maximum over the trial of |roll| + |pitch|, the unambiguous loss-of-control signature; the custom stacks latch airframe-lost at 75°, which the harness counts as a failed run.
Windowed tilt p_{99}	°	The 99th percentile of the tilt sum within each consecutive 10 s window of a 150 s hover, interior empty windows materialised so that an odometry dropout fails the run instead of passing silently. Pass threshold 1.0°.
Maximum drift, settling time	m, s	Peak planar distance from the phase-start position or the time-aligned reference, and the time from that peak until the drift first falls back below 0.5 m.
Motor saturation	%	Fraction of samples at which the maximum normalised motor command exceeds 0.95.
Planar / 3D RMSE	m	Horizontal RMS tracking error against the time-aligned analytical reference; for the helix, the three-dimensional error with the reference altitude shifted onto the flight's own datum.

continues on the next page

Table 10.3 – continued from the previous page

Metric	Unit	Definition
Motor-speed spread	$\mathrm{rad\,s^{-1}}$	The optimiser's direct target, $\max_i \omega_i - \min_i \omega_i$ per sample, as a time mean and as a converged tail mean.
Mean propeller power	W	Per sample $\sum_i k_Q \omega_i^3$; the optimiser's secondary energy metric.
Moment-bias magnitude	N m	The disturbance estimator's steady output as an early-window versus tail-window pair — the signature that separates *compensating* for a fault from *reconfiguring* around it.
Failover latency, solve time	s, ms	Time from fault injection to the latched hand-over; and the per-cycle predictive solve, p_{50}/p_{99}.

The windowed tilt p_{99} is the gate that adjudicates every hover-bearing SITL run, and its 150 s duration is motivated by a measured blindness: an 8 s smoke hover is blind to a slow instability, a coning mode with an e-folding time of about 20 s having passed every short smoke test while diverging. Two caveats make the gate readable. It is a *clean-air, healthy-airframe* gate — in the windy world the same stacks measure 3.18° and 3.90° while flying correctly, so windy runs are adjudicated against 5° — and a fault run that completes its mission still fails the label, because not even the real flight 9, at $p_{99} = 4.8°$, would pass it.

Three methodological points apply throughout. First, the rotor speeds, and therefore the spread and the power, are not measured ESC telemetry but commanded quantities, taken from the commanded actuator stream by inverting the ESC normalisation (Sec. 6.10). Second, time alignment is by minimisation of the interior tracking error for *both* stacks, refined to 0.01 s; the obvious alternative, detecting departure from the origin, is biased about 1 s late because the eased-in blend starts the reference velocity from zero. Third, the normalised motor field is *not* comparable between simulation and hardware: in SITL the ESC map is linear by construction, while on the aircraft it is a 13-point bench table, so the same $361.8\,\mathrm{rad\,s^{-1}}$ reads 0.385 in simulation and 0.273 through the bench table. Saturation percentages and collective values must be compared within a venue, never across it.

A reference bug is worth reporting for its methodological lesson. The figure-of-eight generator gained a lobe-height parameter on 2026-08-01, and the analysis script was not updated the same day: it kept rebuilding the reference with ±0.75 m lobes where the aircraft had flown ±0.5 m, which inflated *every* figure-of-eight RMSE computed that day, in simulation and in flight alike. The error was found by inspection of a path plot and the whole campaign re-analysed — the flown trajectory had always been the correct box, and the published flight 10 tour numbers moved from 0.199/0.199/0.209 to 0.103 (X), 0.129 (T) and 0.137 m (H). The rule recorded

from it: when a trajectory generator gains a parameter, every reconstructor of that reference must be updated the same day, and any RMSE computed before that date is void.

10.3.2 Experiment catalogue and campaign matrix

The executable suite is a set of seventeen runners, each a short declarative script over the command surface of Sec. 10.2. The official trajectories, held fixed across both stacks and all campaigns, are the 3×1 m figure-of-eight box (period 20 s, three loops, streamed at 50 Hz with full feed-forward) and the ascending helix of 3 m diameter and 5 m climb. Both are analytic and smooth to the derivative order the feed-forward consumes, which is the property that matters for a differentially flat multirotor [47]; no optimisation-based generator [48] is used, deliberately, so that the reference is identical across stacks, campaigns and configurations and every difference measured belongs to the controller. The runners fall into five roles: *gates*, the 150 s tilt-gated hover smoke and the maiden-flight profile; *tracking*, the official references per shape or as a single-battery tour; *baseline*, the same helix on the native PX4 controller with a 60° tilt abort, which is what lets it record the autopilot's failure on asymmetric shapes as a bounded event; *robustness*, covering in-flight shape switches, the failover cases including a null case that must *not* trip, and the arm-servo fault chain; and *diagnosis and historical*, including the payload runners preserved but no longer runnable. Three honest limits should be recorded so that Chapter 12 can be read accurately: payload-rejection results obtained against the nominal, zero-model-error plant prior to the rig's removal are labelled historical; no dedicated per-shape-stability or optimiser-benefit runner exists anymore, as both roles have been absorbed into the campaign matrix; and the helix streams its approach under the predictive stack, where the stepped ramp collapses into the waypoint acceptance gate, while the sliding-mode helices keep the stepped ramp of the mission manager.

Table 10.4 states which grids were actually executed, with which stack and with what outcome; every run passed through the harness of Sec. 10.3.3 and carries an adjudicated verdict. Beyond these six campaigns the results tree carries smaller named series — the flight 5 reproduction and phugoid-validation runs, the tuning searches, the coefficient re-validation gates, the O-shape gate at its post-crash opening angle, the arm-fault reaction validation and three HITL benches — for 92 adjudicated verdicts in that tree alone.

Just as important is what was designed and remains unexecuted, because the verdicts above must not be read as covering it: the gust-period sweep that the wind plugin's own note prescribes; the 2× fault with the optimiser; the falsifier of the centre-of-gravity sigma, the twin's only fitted parameter; the re-fit of the torque-deficit injection from 250 Hz logs; the dedicated fault-run gate; and the SITL

Table 10.4: The executed campaign matrix (2026-07-28 to 2026-09-10). Every run carries a harness-adjudicated verdict; fault-phase runs that complete but fail the healthy-air tilt label are discussed in Sec. 10.3.1.

Campaign	Stack	Design and outcome
Retuning against the disturbed twin	SMC	21 runs: severity cascade 3.0× → 1.5× → 1.0× → 0.5× → clean over three seeds. Outcome: the consolidated controller constants of Chapter 7.
Fault-tolerant morphing	SMC, X	10 runs: the `f9` fault at 1.0×/1.5×, from the ground, as an in-air step and as a ramp, fixed frame versus optimiser. Outcome: the fault-tolerance thesis demonstrated at 1.5×.
Shape matrix v2	SMC	22 cells: four shapes × two trajectories × phases clean / wind / fault+wind / fault+wind+optimiser. 21/22 complete; the single red a touchdown tip-over after an already-completed mission.
Shape-matrix mirror	NMPC	29 runs on the same cells plus a three-run failover phase. 25/29; the guarded phases ran 8/8 clean and 8/8 in wind with zero failovers, while the naked-fault phase of that August campaign (4/8, on the superseded plant and not reported in Chapter 12) produced the operating rule that the predictive stack never flies a rotor fault without its guard.
Pre-flight suite	NMPC	14 runs on the exact head that would fly the first predictive flights, 13/14. Gate for flights 18 and onward.
Post-crash diagnostic suite	NMPC	11 runs sweeping the actuator time constant and the Levenberg regularisation after the flight 19 crash. Outcome: the crash loop condition reproduced at full gain; the corrected constant was adopted at launch on the strength of the measured frequency response (Sec. 6.6.1).
Final campaign	SMC, NMPC	16 + 16 tracking cells (four shapes, two references, clean air and gusts, on the final plant with no injected model error), three failover cases and 3 + 3 twins of flight 17 at the flown fault severity. Of the predictive runs attempted, 22 were accepted and 13 were not (Sec. 12.1.6). It is the source of Part I of Chapter 12.

re-validation of the flight 19 fix, which would require injecting arm compliance into the plant until the measured frequency response of Sec. 6.8 is reproduced — the twin does not model that compliance, and no run on the current plant can stand in for it.

10.3.3 Reproducibility and data handling

Reproducibility is built into the framework at three levels: a uniform run-time harness, a fixed on-disk layout with full provenance and an adjudicated verdict, and a separate offline analysis stage.

Every run goes through a single harness, which brings up Gazebo, the DDS agent, the autopilot, and the requested control stack, executes the experiment, archives the controllers' per-cycle logs, and tears down the environment. Four behaviours are load-bearing, each added after a specific failure: the autopilot's persisted parameter files are wiped before every flight because stale estimator calibrations once caused a fly-away; readiness is probed on the odometry topic using the sensor-data quality-of-service profile, since the default profile never matches and renders a healthy autopilot invisible; the launch file and arguments can be overridden via environment variables, allowing the disturbed twin to wrap unmodified flight launches; and the injector is explicitly terminated at teardown. A common client library provides blocking helpers (take-off, land, go-to-waypoint, trajectory streaming, set-shape, set-optimiser, and an attitude-quiet gate introduced after the flight 19 crash that refuses to start streaming while roll oscillates), enabling experiments to be defined as short declarative sequences.

Each trial is recorded with standard ROS 2 bag tooling over a fixed topic list, accompanied by a metadata file documenting the experiment, parameters, timestamps, status, and Git commit hash. Version control is deliberately limited to per-trial metadata, harness verdicts, plots, and summary tables; raw bags are excluded to ensure that verdicts and metadata serve as the durable record while bags remain local artefacts. A run is classified as `FLIGHT OK` only if it produces at least one fresh trial metadata file with status `ok`, contains no take-off-failure marker, and shows no airframe-lost abort in the launch log; the verdict file additionally records the provenance details (controller, launch file, arguments, world, and control-law overrides) that the trial metadata lacks. These criteria were added after specific failures; until 2026-07-23, runs failing waypoint, landing, or tilt gates still reported `FLIGHT OK` with a zero exit code. Conversely, a non-zero exit on a *fault* run does not indicate a crash, as the tilt gate is calibrated for healthy aircraft dynamics. The offline stage then flattens each bag to scalar columns on a global time origin, pairs it with its metadata, and computes the metrics defined in Sec. 10.3.1, ensuring that every figure in Chapter 12 can be traced back to the exact code and data that generated it.

Simulation is not only a measuring venue; it is the gate on the flight campaign. Three artefacts implement the gate: a nine-row smoke matrix re-run on every head that will fly; a per-flight gate suite that reproduces the exact tour a flight card will fly, in the world that card expects, before the card is approved; and the maiden-flight profile, required of any stack flying for the first time — the predictive

stack's first flight was gated this way, with a hover tilt p_{99} of 0.486° against the 1.0° threshold.

Chapter 11

Hardware realisation and the real-flight campaign

This chapter reports the hardware realisation of the morphing quadrotor and the real-flight campaign that followed it: nineteen numbered flights over twelve field days between 19 July and 6 August 2026. It describes the aircraft as built, how the control stack was ported to it and made to run in real time, how the failsafe layer around the morphing actuation works, how the model constants were identified from flight data, and the protocol under which all of this was flown. It closes with a statement of what the campaign did and did not establish. The per-flight results are in Chapter 12, the constants identified from the flight data in Chapter 6, and the full operational protocol in Appendix C.

11.1 The as-built vehicle

The vehicle that flew the campaign is the one described in Chapter 3; Appendix B lists every parameter with its source in the implementation.

The hinge-to-rotor arm length is 235.0 mm, with a body half-corner offset of $D = 113.072\,\mathrm{mm}$. The rotors on the two diagonals are mounted at different levels, separated by 60 mm. With a total rigid mass of 3.964 kg, the hover thrust is calculated to be 38.9 N. The inertia tensor in the X configuration is $J = [0.1217, 0.1198, 0.2030]\,\mathrm{kg\,m^2}$.

Propulsion is provided by four BrotherHobby Avenger 4110 560 KV motors driving 15×5.5 inch two-blade rigid propellers ($R = 0.1905\,\mathrm{m}$). These propellers were mounted on 3–4 August, after the original folding three-blade set had been damaged in flight. The propulsion system utilises HGLRC 90A speed controllers operating on DShot600, powered by a 6S pack. One limitation shapes the entire identification chapter and must be stated here: *no rotor speed was ever measured*

in flight, so every in-flight rotor speed in this thesis is inferred through the bench throttle-to-speed map (Sec. 6.10). The arms are rotated by four Dynamixel XC430-T150BB-T smart servos on a dedicated 12 V rail (stall torque 1.6 N m), in extended position mode at 0.088° per tick, with a position PID of 640/0/6000 and a profile velocity of 19.24 $^\circ\,s^{-1}$. The servo has *no current sensor*: its load register reports an estimated PWM duty, and the positional droop under torque is the only independent load proxy — both facts are load-bearing for the failsafe architecture of Sec. 11.3. The flight controller is a Pixhawk 6X running the project's firmware fork, the companion computer a LattePanda N100 running the ROS 2 stack over Ethernet through uXRCE-DDS, with the sliding-mode stack at 250 Hz and the NMPC stack at 100 Hz, and a lidar providing the altitude reference. Of the bench characterisation of Chapter 4, the two headline coefficients did not survive contact with the aircraft; what did survive is the throttle-to-speed map, the rotor-speed envelope and the small-signal actuator time constant.

11.2 Porting the control stack to hardware

11.2.1 Software preparation

Two pre-flight audits (2 July, 41 findings; 5 August, MPC-specific) shaped the software that flew. Every physical constant lives in one module, with an environment switch selecting between the thirteen-point hardware ESC table and the linear SITL map — a switch that encodes a structural lesson, since in simulation the plant inverts the same linear map the controller applies, so the thrust coefficient *cancels* and simulation could never have detected the factor-of-1.78 error of Sec. 6.4. The PX4 failsafes were set with readback and recorded (Appendix C), together with an altitude-divergence watchdog of the project's own, which force-lands the vehicle when the height the estimator reports and the height the lidar measures drift apart, and which has tripped correctly in flight on a sudden shift of the satellite frame. The arm protections of Sec. 11.3 were ported to *both* stacks before the MPC flights, together with the hot-standby guard as a default of the MPC launch. No card was flown without a green SITL suite on the same commit; that gate earned its keep, since a false-positive guard failover caused by a starved guard process, and three false positives of the oscillation detector added after Flight 19, were discovered *only* in simulation.

11.2.2 Companion-computer deployment and timing

Both stacks run on the same companion: the SMC at 250 Hz and the NMPC at 100 Hz. Real-time configuration is applied *per process after launch* — `chrt -f 50` and a dedicated core for the controller, `chrt -f 40` for the guard — and never by

wrapping the whole launch: a bench test that wrapped the launch starved the servo serial driver and dropped the joint feedback from 100 to 61.8 Hz with read errors. The guard is pinned for the same reason in reverse: a starved guard produces spurious failovers. The first start of the MPC stack regenerates the acados code (two to five minutes) and is done at the bench, not at the field.

The predictive solver was timed on the bench before it was trusted in the air, and the bench run carries one diagnostic worth recording: its 21 ms to 27 ms outliers came from the Python garbage collector, not from the solver, and with real-time priority and the collector disabled the worst case fell to 4.46 ms with no long ticks at all. In flight the solver ran with a median of about 0.9 ms and a worst case of 4.2 ms on the first predictive flight, roughly four times faster than on the bench. That is not a paradox but a different experiment: in flight the solver runs warm on a smooth trajectory, while the bench replays cold, randomised problems. Both flights closed every solve inside the 10 ms budget of the 100 Hz loop (Chapter 12).

The transport latency between companion and autopilot was measured in flight at 14.2 ms one way, about 30 ms round trip; the 180 ms and 595 ms figures quoted earlier in the project were inferred, circular and retracted. The autopilot CPU load in flight averaged 49.2 % to 50.0 %.

11.2.3 Data recording and the bench-versus-flight lesson

Each flight records the PX4 ulog, a session rosbag plus one bag per trial, the controller node log, a parameter dump and written field notes. The campaign also lost data, for ordinary reasons: log files opened on a tmpfs were truncated by a stack restart or vanished at power cut, the MPC controller log on *both* MPC flights. The recoveries are equally reproducible — the session bag of Flight 19 was re-indexed and yielded 182.5 s covering the morph, the settle and the crash — and two rules entered the protocol as a result: every log copy is verified after the copy, and the logger writes to disk, never to tmpfs. A third lesson matters for anyone reanalysing the data: the per-trial bag starts at the *start of the trajectory*, so the settle window in which the Flight 19 instability was born exists only in the session bag.

The servo subsystem produced the most generalisable process lesson of the campaign, written as a comment in the driver source so that it cannot be lost: the unloaded-arm bench does not predict in-flight servo behaviour, in either direction. The integral gain was promoted at the bench, where it cut the stationary fraction of a transit from 75 % to 3 %, and rejected in flight by a direct A/B *within the same flight* — two runs at $I = 400$ ended in static load trips, while the run flown at $I = 0$ completed both trajectory sets with no trip — because under the rotor-torque load the integrator nulls the position error by pushing *through* the stiction. The profile velocity went the other way: rejected at the bench, kept in flight for its clean holds,

and then shown *not* to be the fix by Flight 16, in which the 6 Hz oscillation kept its frequency and was thereby marked as a structural mode rather than a motion artefact (Sec. 6.7). The operating rule that survives is the in-source comment: the next change to the servo gains or profile is validated in flight, not at the bench.

11.3 The arm failsafe architecture

The servos carry no current sensor and their factory overload shutdown is fixed and clearable only by a reboot, so the morphing actuation needs a protection layer of its own. Chapter 3 gives the thresholds and the design argument behind them; what follows is how the layer is built and how it behaves in the air.

Four latched fault codes are published on a status topic — serial port dead, servo unreadable, arm runaway, overload risk — and three sentinels watch for them at decreasing rates: a 100 Hz per-servo watchdog, a 2 Hz poll of the servo hardware-status register, and the mission manager, which reacts to every status publication as it arrives.

The watchdog compares the measured position against the position *expected from the motion profile*, never against the goal, because mid-transit the goal is legitimately a hundred degrees away. A directional gate separates a runaway from the fast snap-back of an arm recovering towards its target, and the branch trips at $35\,^\circ\,\mathrm{s}^{-1}$, or at 18° of error sustained for 0.10 s, or on 6.0° of droop held for 2.0 s, or at 65 °C of case temperature. The load trip runs in two regimes, because a transit costs twenty to twenty-five points of duty on top of the static load: the static threshold alone, tuned on stationary holds, tripped on every morph of the first morphing flight — three trips and three automatic landings in one flight — so a stall-catching ceiling applies during transits instead. Both regimes were checked against sixteen synthetic scenarios and against replays of the recorded flight traces, on which the protection stays quiet through legitimate transits and still fires on the one in-flight arm release the campaign suffered (Sec. 6.7).

Recovery is automatic where it can be. A factory latch clears only by rebooting the servo, so the driver reboots a latched servo once per session and re-initialises it, the whole detect-and-recover cycle closing in 0.15 s against the 0.73 s an unheld arm needs to back-drive into its neighbour's rotor disc (Sec. 6.7). The reaction is in the mission logic rather than in an advisory: an overload-risk code disables the optimiser, blocks further morphing and commands a normal landing reshaped to X, the long-lever configuration that unloads the critical servo, all within a millisecond of the status message; any harder code commands an immediate landing in the current shape.

The runaway sentinel fired six times with a clean hardware error byte, always with at least two servos involved inside 100 ms and always on the ground: mechanical

jolts at touchdown or during handling, not faults. The on-ground gate that answers them suppresses the runaway branch alone, and only while the autopilot's land detector reports fresh ground contact, so the sentinel is untouched in flight. One residual is left open: a jolt in the instant of touchdown can arrive before the land detector does.

11.4 Identifying the model from flight data

Several of the constants the controllers use could not be measured on the bench, or were measured there and turned out to be wrong. The procedure that replaced the bench is the same in every case. The safety pilot flies the aircraft by hand in the autopilot's stabilised mode, holding a steady hover or driving one axis at a time with short doublets, while the autopilot logs the full state at its own rate and the companion computer records the commands it issued. After the flight the logs are dumped and analysed offline: hover segments give the collective and the trim the aircraft actually needs, the doublets give the attitude response, and the commanded allocation recorded alongside gives what the controller believed it was asking for. The difference between what the aircraft needed and what the controller commanded is the correction applied to the constant under test.

This is how the thrust coefficient was falsified and re-derived. The same offline route — commanded against measured, on the logs the aircraft brought back — also served for quantities no hand-flown segment could isolate: the servo loads and the arm behaviour under real aerodynamic load rather than on an unloaded bench, and the actuator time constant, fitted from the frequency response of the last flight of the campaign (Sec. 6.6.1). The session of 25 July was devoted to identification outright, and Flight 12 repeated the exercise after the propellers were changed. Chapter 6 reports the values, the two occasions on which a bench number had to be retracted, and the one constant that is still an interval rather than a measurement.

11.5 The flight-test protocol as flown

The protocol written before the campaign — a general staged plan with numbered test cards — survived contact with the field in structure but not in form. What was actually flown, from Flight 3 onward, is a *card per flight*: a self-contained operational document with explicit entry gates, a parameter fingerprint executed on the ground before launch, a declared envelope, abort criteria, a three-level quick-stop procedure and a post-flight routine. Twenty-one such cards exist in the repository, bound together by two rules: one change per flight, and a cascade gate. Appendix C reproduces the full protocol; three elements are worth stating here because they are non-obvious. The abort criteria are *positional, not attitudinal*

— an XY excursion beyond 1 m sustained for more than 2 s in hover, an altitude error beyond 0.5 m, or an attitude oscillation whose amplitude *grows*, with three to four degrees of steady tilt in wind explicitly not an abort — and the accident that ended the campaign (Sec. 6.8) is the proof that the growth criterion is the right one and that no automatic threshold was watching it. The *override semantics* are equally blunt: an RC takeover mutes the entire stack within about 200 ms, after which the node never reclaims control; the kill switch stops the motors without autonomous re-arm; and in non-X shapes, an RC takeover is *degraded* because the native controller assumes a centred vehicle that no longer exists. One *process lesson cost a trial*: commands on mission topics must never be published with `--once`, as the publisher can terminate before discovery matches all subscribers; therefore, the required protocol form is `--times 5 --rate 2`.

11.6 What the campaign established

Stated without inflation, the campaign established the following on hardware: the platform hovers cleanly (Flight 7: 101.1 s, maximum drift 0.62 m); it tracks streamed planar and three-dimensional trajectories in three configurations (Flights 10 and 14); it reconfigures in flight under protection (Flight 13); the fault-tolerant morphing thesis holds with a real, physically damaged propeller in three out of three trials (Flight 17); and the NMPC both runs and outperforms the SMC on hardware in X (Flights 18 and 19, −29 % tracking error). It also *refuted*, by flight, a list of prior beliefs; these refutations are themselves results: the bench thrust coefficient (Flights 3–4), the O configuration as a hover shape (Flight 13), the servo integral gain and profile velocity as fixes for the 6 Hz oscillation (Flights 15–16), and the rigid-arm assumption of the predictive model, falsified by the mode identified on the log of the last flight (Sec. 6.8). Where a number rests on a single flight or a pair of flights, the sample size is stated at the point of citation; the three repetitions under matched protocol required for publication constitute part of the declared gap.

11.7 What the campaign did not establish

The credibility of the preceding sections requires an equally blunt assessment of what remains open. **Morphing under NMPC has not been demonstrated to be safe on hardware**: the corrected actuator lag has been checked only against the measured frequency response, never in flight and never in simulation — against the simulated plant a model at 0.08 s collapses the solver (Sec. 12.1.7); Flights 20–22 were never flown, and no hardware evidence exists that reconfiguration under the

predictive controller is safe. Furthermore, **the actuator lag itself is amplitude-dependent and has not been measured for the current propellers**. **The compliance mode remains unresolved** (Sec. 6.8): the declared structural remedies, namely servos with closed-loop current control or explicit modelling of the mode, exist only as intentions. **The O configuration remains unviable for hover** with the present servo hardware. **Yaw under NMPC exhibits wander** (44° peak-to-peak on Flight 18 and 71° on Flight 19 at approximately 0.2 Hz); while the cause was identified and a threefold fix implemented, it has never been flown. **Two estimator anomalies are mitigated but not understood**: the $\sim$2 m disagreement between the estimator's height and the lidar, absorbed by taking the altitude datum at take-off, and the heading resets of $\pm$12°–21° at every spin-up, absorbed by holding the aircraft a few seconds at idle before launch. **The thrust coefficient remains a composite value** coupled with the ESC map; without rotor-speed telemetry, the two are inseparable, so the torque coefficient is the centre of an admissible interval, not a measurement. **No direct SMC-versus-NMPC comparison exists for fault tolerance**: only the SMC side was flown (Flight 17), while the NMPC counterpart remains in the unflown Flight 22; every claim in this thesis about morphing or fault tolerance *under the predictive controller* is therefore a simulation claim. **Statistical power is limited to n=1 or n=2 per condition**, and cross-shape comparisons are confounded by protocol differences, session drift, and wind. **The geometry optimiser lacks an explicit excursion bound in its cost function**: although each joint is constrained to $\pm$135°, the depth of reshaping is limited solely by regularisers. Finally, **the NMPC consumes joint feedback without a staleness check**; a driver that silently ceases publishing would freeze the model geometry at the last sample without triggering an alarm — a defect identified and exonerated in the Flight 19 analysis but never formally closed.

Chapter 12

Results

This chapter reports the experimental evaluation of the morphing quadrotor and its control stack, in two parts. Part I (Sec. 12.1) is the simulation evidence. Part II (Sec. 12.2) is the outdoor campaign described in Chapter 11: nineteen instrumented flights between 19 July and 6 August 2026, from the first hover to the first hardware flights of the predictive stack. Flights 1 to 17 flew the sliding-mode controller (SMC), flights 18 and 19 the nonlinear model-predictive controller (NMPC); flight 18 completed its mission and flight 19 was lost after reconfiguring to T. Three further flight cards (20–22) were written but not flown. The experiments exercise the SMC of Chapter 7, the NMPC of Chapter 9 and the energy-aware geometry optimiser of Chapter 8; the operational protocol is in Appendix C.

12.1 Part I — Simulation results

Everything in this part was obtained in PX4 software-in-the-loop with Gazebo (Chapter 10). Most of it comes from a single campaign flown on 9 and 10 September 2026, once the code had stopped moving: both control stacks on the same code, the same plant, the same references and the same test matrix, so that any cell of the sliding-mode table can be set against the same cell of the predictive table. Two older sets of runs are kept because nothing later replaced them — the sliding payload of the design brief, which only the design-stage twin could carry, and the baseline flown with the stock autopilot — and the rotor-fault experiment of 31 July is kept because it is the one with a hardware counterpart, flight 17. Section 12.1.1 says which run belongs where. Almost every condition is a single run: the numbers below are individual realisations. The only repeated simulation experiment is the twin of flight 17, flown three times per stack; every other standard deviation in this thesis comes from the repeated hardware trials of Part II.

12.1.1 The experiments and where they come from

The final campaign flies the four arm configurations X, O, T and H through the two streamed references the aircraft flew outdoors: the 3×1 m figure-of-eight with a 20 s period, and the ascending helix of 1.5 m radius, 12 s period and 5 m climb. Each reference is flown twice, once in clean air and once in the gusty world of Chapter 10, whose $3\,\mathrm{m\,s^{-1}}$ mean wind carries sinusoidal gusts of $\pm 50\,\%$ at a 3.3 s period, Gaussian noise and a slow wander of direction. Each stack therefore has sixteen tracking cells. The predictive stack adds the three standing failover cases of its supervisor (a frozen command stream, a solver that keeps failing, and a null case that must not trip), and both stacks fly a simulated twin of the hardware fault-tolerance experiment of flight 17, three trials each. Every tracking and failover cell is a single run; the twin of flight 17 is the one repeated experiment of the campaign. The sliding-mode helices reach the helix entry through the stepped ramp of the mission manager, the predictive ones through a streamed approach: under prediction the stepped waypoints are accepted as soon as they are issued and the ramp is skipped. The helix error is computed from the aligned start of the helix onward either way, so the approach does not enter the metric.

The plant is the final parameter set: the CAD-anchored geometry adopted on 31 July 2026, the thrust coefficient identified in flight, the 15×5.5 propellers and the O preset at $+100°$ (Appendix B). In clean air and in gusts no model error is injected: the aim of those cells is to compare the two stacks on the same plant, not to measure robustness. The flight software moved through four commits, from the morning of 9 September to the small hours of 10 September, and the two controllers are identical in all four: one commit revises the oscillation detector of the supervisor (Sec. 12.1.6), and the other three touch only the injector launch file, the figure-of-eight analyser — which was then re-run over every log of the campaign — and the archived logs of an earlier campaign. The predictive stack runs with the guard armed, as it does on the aircraft, and with the actuator lag of its model set equal to the rise-time constant of the simulated plant, 0.15 s (the plant itself falls with 0.12 s); the value it flew on the aircraft is discussed in Sec. 12.1.7. A run counts as completed when the mission finishes, the vehicle lands and nothing aborts, and a predictive run counts only if the standby never took over — except in the three failover cases, where the hand-over is the experiment — which is checked in the launch log of every run rather than assumed.

12.1.2 Why not the stock autopilot

A fair question to ask of any custom flight controller is whether the stock one would have done. It was asked early, on the design-stage twin, by flying the ascending helix in each configuration with the native PX4 position and attitude loops, which know nothing about the arms, and then with the custom sliding-mode stack. In X,

Table 12.1: The simulation experiments reported in Part I, with the plant they were flown on and the hardware experiment each one has for a counterpart. "Design-stage twin" is the June simulation model, which still carried the payload rig and shared its parameters with the controller; "final plant" is the flight-identified parameter set of Appendix B.

Experiment	Plant, date	Stack	Runs	Hardware counterpart
Helix per shape, stock autopilot against the custom stack (Sec. 12.1.2)	design-stage twin, June	PX4, SMC	4 + 4	none: the stock autopilot never flew the aircraft
Sliding payload, 0.2 m sequence, optimiser off and on (Sec. 12.1.4)	design-stage twin, June	SMC, NMPC	2 + 2 reported	none: no payload was ever flown; flight 17 exercises the same estimators with a damaged rotor
Rotor-fault severity ladder, hover in X (Sec. 12.1.5)	final plant, 31 July	SMC	10	flight 17
Figure-of-eight and helix, four shapes, clean air and gusts (Sec. 12.1.3)	final plant, 9–10 Sept	SMC, NMPC	16 + 16	flights 10 and 15 (figure-of-eight) and 14 and 16 (helix) for the SMC, flight 19 for the NMPC
Failover cases of the supervisor (Sec. 12.1.6)	final plant, 9–10 Sept	NMPC	3	the single activation of flight 19
Twin of flight 17, fault at the flown severity (Sec. 12.2.4)	final plant, 9–10 Sept	SMC, NMPC	3 + 3	flight 17

T and H the stock autopilot copes: it overshoots once at the helix entry, settles, and then holds the reference laterally about as well as the custom stack does, while using about two and a half times as much tilt to do it (peak tilt 51.5, 48.1 and 51.3° against 19.9, 18.6 and 20.3°). Its rate-loop integrator reads the centre-of-gravity offset and the asymmetric thrust map of a folded shape as a slowly varying torque bias and trims it away; the hover-thrust estimator adapts the thrust scaling without being told the mass or the geometry; and desaturation gives up yaw first when the mixer runs out of room. None of that needs to know that the airframe has changed shape, which is why three of the four configurations fly.

The fourth does not. In O the thrust map is not a perturbed version of the X map the autopilot was configured with but a different one, and an integrator that trims a parametric error cannot trim a structural one: the native loops wind up chasing an error they cannot remove and the vehicle tumbles, reaching 264° of tilt and leaving the commanded helix towards the ground, while the custom stack completes the same helix with a mid-turn wobble and a peak tilt of 20.2° (Fig. 12.1). That is the limit of a fixed effectiveness matrix: it tolerates the mismatch morphing creates while that mismatch still looks like a disturbance, and

fails once the mismatch is structural — the case the geometry-aware allocation of Chapter 5 handles directly. The comparison was never repeated on the final plant or on the aircraft, and Part II adds that O is not a hover shape for the real vehicle either, for a reason the control law has nothing to do with.

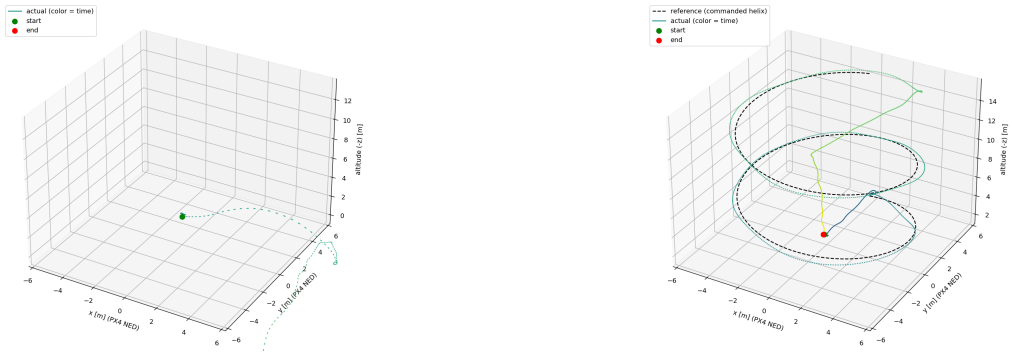


Figure 12.1: The ascending helix in the O configuration, design-stage twin, single runs. Left: the stock PX4 controllers lose the vehicle — the trajectory leaves the commanded helix and descends out of the plotted volume. Right: the custom sliding-mode stack with centre-of-gravity-centred allocation completes the same helix with a visible mid-turn wobble and a peak tilt of 20.2°.

12.1.3 Trajectory tracking across configurations

Table 12.2 gives the tracking error of the two stacks in every cell of the final campaign. In clean air the two controllers land in the same band. On the figure-of-eight the sliding-mode stack closes between 0.024 and 0.031 m of planar RMSE across the four shapes and the predictive stack between 0.033 and 0.063 m, with T its weakest shape; on the helix the three-dimensional RMSE is 0.046 m to 0.055 m for the sliding-mode stack and 0.060 m to 0.087 m for the predictive one. Gusts roughly double the figure-of-eight error of both stacks, to 0.050 m to 0.053 m for the sliding-mode stack and 0.058 m to 0.077 m for the predictive one, and cost far less on the helix: 0.058 m to 0.062 m for the sliding-mode stack against 0.051 m to 0.112 m for the predictive one, which is the better of the two in X (0.051 against 0.059 m) and clearly the worse in T (0.112 against 0.058 m). The sliding-mode stack is the tighter controller in fifteen cells out of sixteen, but the margin is small, between two millimetres and five centimetres. What the table supports is that the two stacks track to the same order of magnitude on this plant, that the sliding-mode stack is consistently a little tighter, and not much more; in particular, no dependence of the gap on the wind is claimed.

The two stacks reach those numbers in different ways, and the overlays of Fig. 12.2 show how. The sliding-mode stack follows the reference with a small, steady lag and rounds the lobes of the figure-of-eight slightly; the predictive stack

Table 12.2: Tracking error in the final simulation campaign, one run per cell: planar RMSE of the figure-of-eight and three-dimensional RMSE of the helix, in metres, for the sliding-mode (SMC) and predictive (NMPC) stacks in clean air and in gusts.

Reference, condition	Stack	X	O	T	H
figure-of-eight, clean air	SMC	0.031	0.027	0.025	0.024
	NMPC	0.033	0.035	0.063	0.034
figure-of-eight, gusts	SMC	0.052	0.051	0.050	0.053
	NMPC	0.067	0.061	0.077	0.058
helix, clean air	SMC	0.051	0.046	0.055	0.048
	NMPC	0.067	0.060	0.087	0.066
helix, gusts	SMC	0.059	0.060	0.058	0.062
	NMPC	0.051	0.076	0.112	0.082

sits on the reference for most of a lap and then leaves it in short excursions, which is what its higher RMSE in T is made of. Attitude activity tells the same story from the other side. Over the hover that precedes each trajectory, the sliding-mode stack keeps its tilt at the 95th percentile between 1.4 and 2.7° in clean air and between 4.3 and 4.7° in gusts, across all eight cells of each condition. The predictive log has no separable hover phase, so its figures run over the whole airborne time and are an upper bound on the same quantity: 1.5° to 3.1° and 4.2° to 5.8°. Under gusts both are dominated by the forced response at the gust period, not by a loop mode. What the predictive stack does that the sliding-mode stack does not is to produce, in the gusty world and only after a reconfiguration, a roll transient of 13° to 24° peak to peak at about 0.7 Hz that dies away on its own — from 24 to 4° in 18 s after the morph to H, from 13 to 3° in three seconds after the morph to T. The transient matters because it is the kind of motion the supervisor watches for after flight 19, and it decays where the flight-19 oscillation grew; Sec. 12.1.6 returns to it. The predictive solver, finally, was never the bottleneck: the 99th percentile of the solve time lies between 0.74 and 1.07 ms on the simulation host, and no quadratic program failed in any completed run.

12.1.4 Payload disturbance rejection and the geometry optimiser

The sliding payload of the design brief — a 3 kg mass on a two-axis rig, moved without the controllers being told — is the disturbance the moment-bias estimator of Chapter 7 and the observer of Chapter 9 were sized for, and the one experiment in which the optimiser can be watched doing its everyday job rather than its emergency one. It was flown in June on the design-stage twin, the only simulation model that

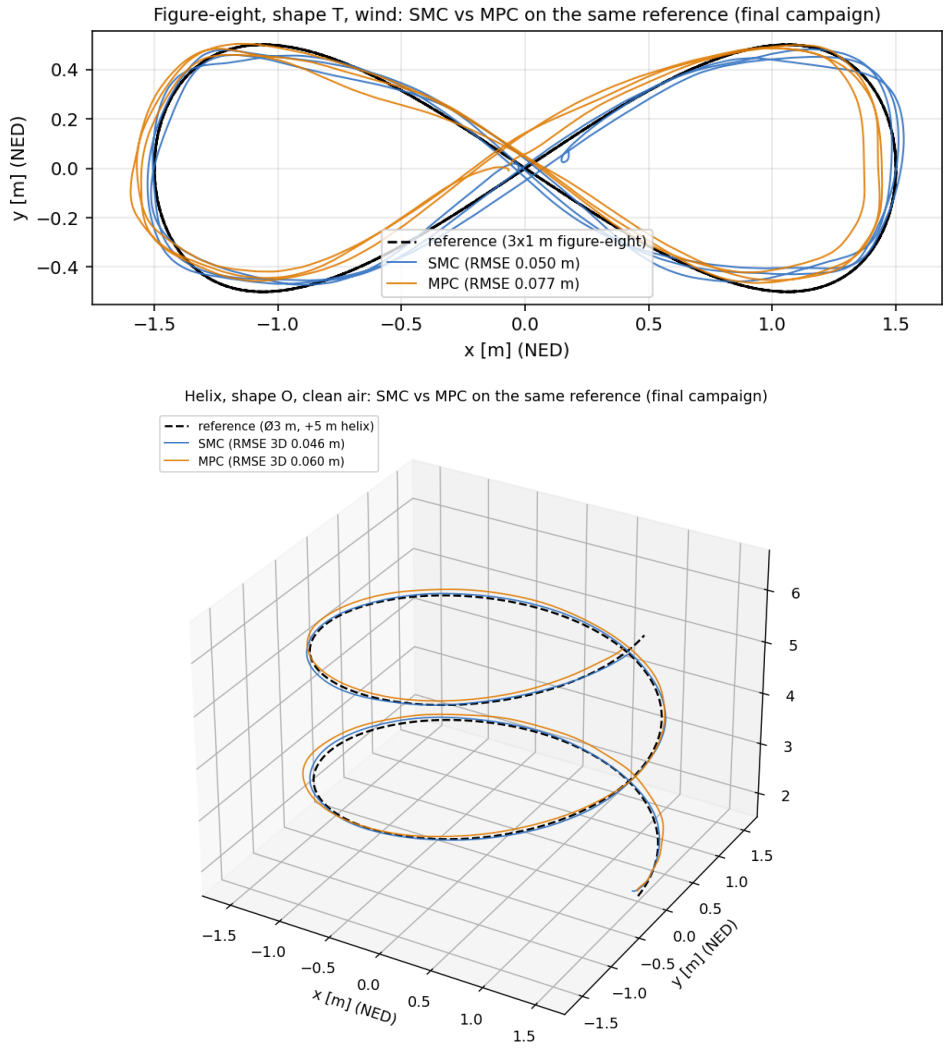


Figure 12.2: Top: the figure-of-eight in T under gusts, sliding-mode (0.050 m) and predictive (0.077 m) stacks overlaid on the reference — the sliding-mode stack rounds the lobes, the predictive one leaves the path in short excursions. Bottom: the helix in O in clean air, 0.046 against 0.060 m, where both sit on the reference for the whole climb. Final campaign, final plant, one run per stack and cell.

ever carried the rig. The payload is commanded through a fixed sequence of offsets (baseline, x, centre, y, diagonal, recovery) of 0.2 m per axis, and for each phase the analysis records the largest planar drift from the position at the start of the phase and the peak tilt. The offsets are small on purpose: a drift of the order of a metre inside a phase would be the vehicle wandering, several times further than the mass that set it off.

The sliding-mode stack rejects the whole sequence to a few centimetres, with a worst per-phase drift of 0.061 m with the optimiser off and 0.074 m with it on, and it does so the way the design intended: the bias estimator learns the standing moment of the offset mass and the allocator holds it. The predictive stack shows what the disturbance observer is for. Without it the offset is never estimated and the vehicle drifts by up to 1.972 m in a phase; with it the worst drift falls to 0.132 m, about fifteen times less, and the estimate steps with the commanded offsets and holds between them (Fig. 12.3). The offset is estimated, not absorbed by an integrator. The estimator that flew this sequence has since been replaced by the momentum-based observer of Sec. 9.4, so the figure records the mechanism rather than the shipped estimator; the mechanism is the one that reappears in the simulated twin of flight 17, where the same kind of estimate tracks the trim of a damaged rotor (Sec. 12.2.4).

With the payload sitting off-centre the optimiser has a static asymmetry to work on, which is the case it was designed for, and the sequence isolates it cleanly: the same sequence flown twice under the sliding-mode stack, once with the optimiser off and once with it on. Its direct target, the spread of rotor speeds, falls by about 63 % in the diagonal phase and 81 % in the x phase; the mean propeller power that follows from the rebalancing falls from 993.6 to 924.2 W in the diagonal phase, about 7 %; and the worst drift moves only from 0.061 to 0.074 m (Fig. 12.4). An earlier run of the same sequence with softer gains shows the same reductions, which is some reassurance that the effect is not a property of one tuning. The power figure is a proxy: rotor speeds are not measured in the simulator but reconstructed from the commanded allocation through the thrust and torque constants of the day, so the relative change is the number to keep and the watts are indicative. So the optimiser buys actuation margin and, on this proxy, a little power, and nothing in tracking — the division of labour Chapter 8 assumes.

One of the two 0.2 m sequences flown on 8 June was lost, and the loss taught something about the optimiser rather than about the payload. The bias estimator had learned the static moment of the mass correctly, the allocation and the state estimate were sound, and the predictive stack flew the identical condition at 1.7° of tilt. What broke the sliding-mode run was the depth of the reshape: with the arms 55° to 70° away from the X preset the products of inertia the sliding-mode model leaves out are no longer small, and the attitude loop diverged in an anti-phase roll–pitch mode (correlation −0.89), the tilt growing from 10 to 35° in about a

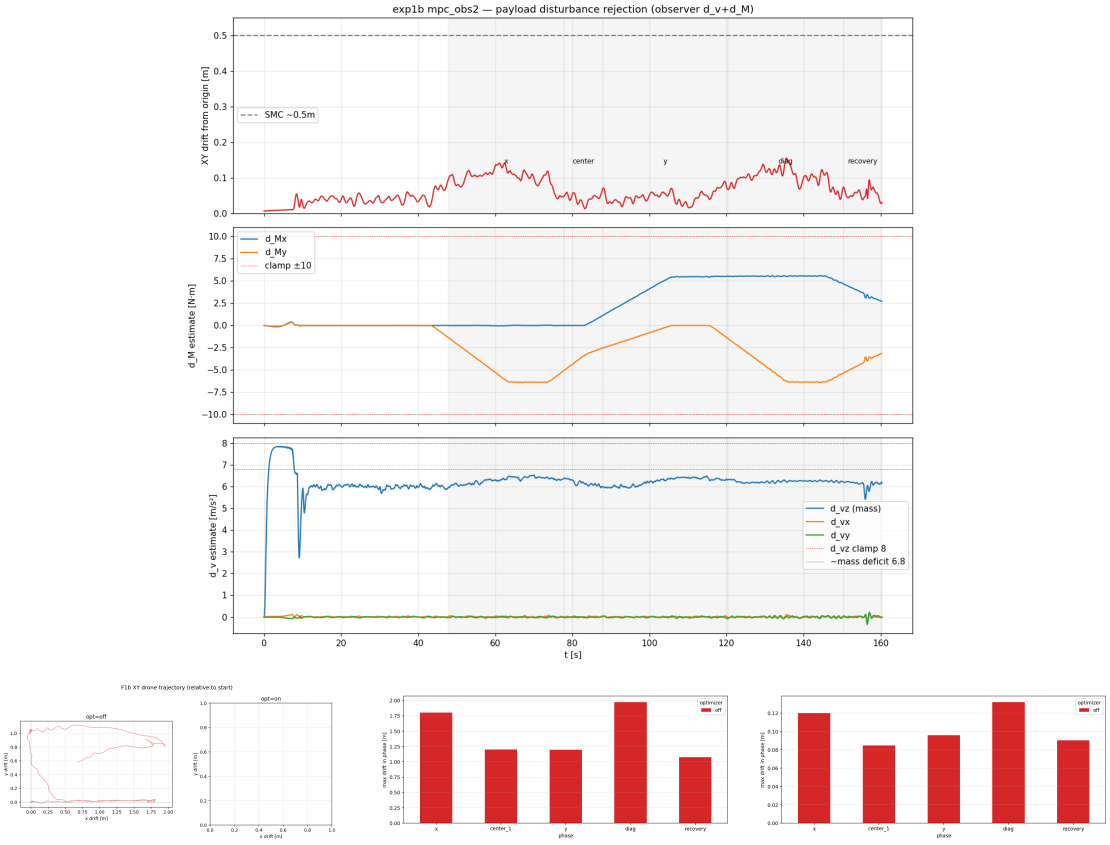


Figure 12.3: Top: the disturbance estimate of the predictive stack through the payload sequence. The moment channels step with the commanded offsets and hold, and the force channel settles on the mass deficit. Bottom, left to right: the planar trajectory with the observer off, and the worst drift per phase with the observer off and on — 1.972 m down to 0.132 m. Design-stage twin, single runs.



Figure 12.4: The optimiser on the payload sequence, off against on, under the sliding-mode stack. Design-stage twin, June, one run per condition. Left: rotor-speed spread, its direct target, down by 63 % to 81 % depending on the phase. Centre: the mean propeller power that follows, about 7 % lower in the diagonal phase — a proxy computed from the commanded allocation, not a battery measurement. Right: the worst drift per phase, which barely moves.

second. The controlling variable was the excursion, not the direction of the payload, and the optimiser has no term that limits it: the box constraint bounds each joint on its own and says nothing about how far the whole geometry has moved. An explicit excursion penalty in the cost is the open item that follows (Chapter 8).

12.1.5 Fault-tolerant morphing under a measured rotor fault

The central claim of this thesis is that geometric reconfiguration is a usable fault-tolerance mechanism. The experiment that carries the claim was run on 31 July 2026, on the consolidated sliding-mode stack and the final airframe parameters, against a rotor fault that is not hypothetical: it replays, deterministically, the propeller damage sustained in flight 9 (Sec. 10.1.3) at two severities: 1×, the damage as estimated from the flight-9 log, and 1.5×, the declared test envelope. The damaged propeller itself, refitted and flown in flight 17, measured at roughly 0.65 of the 1× model, so the simulated fault is conservative with respect to the real one. Table 12.3 lists the ten runs, each a single trial; the fault can engage from the ground, as a step in the air, or as a 20 s ramp — the ramp reproducing both the growth of the real damage and the charge time of the trim integrators.

Table 12.3: The ten runs of the 31 July fault campaign. "Fixed" locks the X geometry; "morphing" enables the geometry optimiser with no other change. The harness tilt gate (1°, calibrated for a healthy vehicle — even the real flight 9, at p99 4.8°, would fail it) marks every completed fault run as failed; the outcomes below are the physical outcomes.

Fault	Engagement	Geometry	Outcome
none	—	fixed X	baseline: tilt p99 0.242°
1×	from ground	fixed X	mission complete, tilt p99 1.43°
1.5×	from ground	fixed X	take-off failed (authority)
1×	step in air	fixed X	recovers in ∼15 s, peak 22.8°
1.5×	step in air	fixed X	airframe lost in < 1 s
1×	step in air	morphing	complete; spread 130 → 34 rad s^{-1}
1.5×	step in air	morphing	airframe lost, identically to fixed
1.5×	ramp 20 s	fixed X	authority-dead: drifts 97 m
1.5×	ramp 20 s	morphing	mission complete, holds position within 0.21 m to 0.40 m
1.5×	ramp, bias clamp 1.25	morphing	holds within 0.155 m to 0.296 m, zero saturation

Two results come out of the ten runs, and together they divide the labour between the two mechanisms. The first is that *the step belongs to the sliding-mode law and the steady state to the optimiser*: when the 1.5× fault engages as a step in the air, fixed geometry and morphing die identically, because the optimiser

acts on a timescale of seconds while the divergence lasts half a second. When the same severity ramps in over 20 s the engage transient all but disappears and the reconfiguration wins: the fixed airframe ends authority-dead — moment bias pinned on all three axes, 97 m of drift without ever reaching its waypoint — while the optimiser-enabled vehicle completes the mission, holding position inside 0.21 m to 0.40 m with the trim clamp at its flown value, and inside 0.155 m to 0.296 m with no motor saturation at all in the tenth run, where the clamp is raised to the 1.25 N m that the trim-authority analysis of Chapter 7 allows. Throughout, the optimiser is never told that a propeller is broken. The second result is that *the trim migrates from actuation to geometry*: over the 1× step recovery the learned roll bias decays from 0.70 to 0.38 N m as the arms take up the asymmetry — the signature that distinguishes reconfiguring from merely compensating (Fig. 12.5). Both results were then looked for on the aircraft in flight 17, and the simulated twin of that flight, run under both stacks at the severity the aircraft actually had, is reported next to it in Part II.

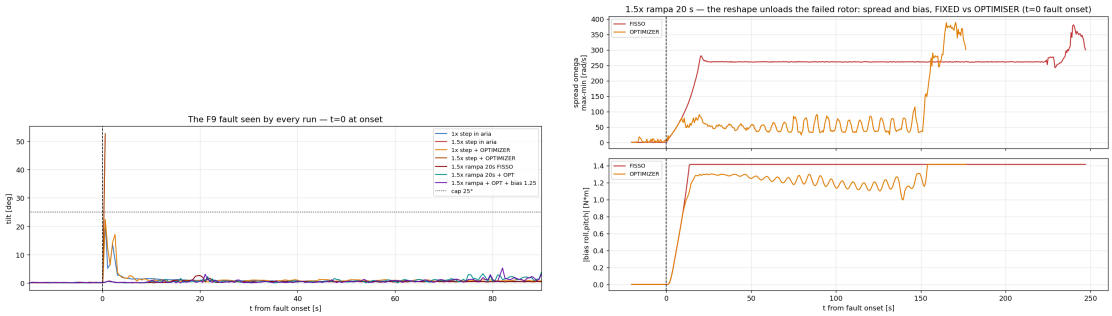


Figure 12.5: Left: tilt across all ten runs of the 31 July campaign, aligned on fault onset; the traces separate by *engagement*, not by geometry. Right: motor-speed spread and learned trim magnitude for the 1.5× 20 s ramp, fixed X geometry against the optimiser. Final plant, single runs.

12.1.6 Supervision and failover of the predictive stack

The predictive stack never flies alone: its launch also starts the sliding-mode stack, muted, and the supervisor of Sec. 9.7, which hands the actuators over on a single latched event. The three standing failover cases were flown again in the final campaign. On a frozen command stream the sliding-mode stack takes over after 0.206 s, with a hand-over excursion of 5.6 cm and a hover tilt afterwards of 0.18° at the 99th percentile; on a solver that keeps failing, after 0.512 s and 7.1 cm; and in the null case, where nothing is wrong, the guard stays silent and the mission completes. The latencies are the same to a few milliseconds as those measured in August (0.202 and 0.515 s). Only the frozen-stream case races the offboard-loss window, and at 0.206 s it sits well inside the 0.5 s of `COM_OF_LOSS_T` (Fig. 9.3); the

solver case takes longer than that window but never opens it, since the offboard stream stays alive throughout. In neither case does the vehicle notice the exchange.

Silence in the null case is a weaker statement than it sounds, and the campaign is more useful for what it rejected than for what it passed. Of the predictive runs attempted, twenty-two were accepted and thirteen were not. Two were rotor-fault cells flown as an internal robustness check and not reported in this thesis, in which the vehicle never took off. Six were flown with the actuator lag of the model set to 0.08 s against a simulated plant at 0.15 s: in every one the quadratic program collapsed and the standby took over, which measures the mismatch between the two constants and not the value of either (Sec. 12.1.7). Three were the first gusty figure-of-eights in O, T and H, in which the oscillation detector added after flight 19 tripped on the roll transient that follows a reconfiguration in wind — about 15° peak to peak in O and T, and 25° in H during the morph itself, the first two decaying within seconds and H over some twenty — because its hard branch fired on two consecutive readings above 14° without asking whether the amplitude was growing. Two were lost to a stall of the predictive process of 200 ms to 340 ms, about a second and a half after the bag recorder started, on a twenty-core workstation that was otherwise idle; the cause is not known, the runs were re-flown clean, and the fact is recorded here because a companion computer has far less headroom than that workstation.

The detector was revised on 10 September to trip only on an oscillation that is still growing after several cycles — present for at least 2.5 s and grown by a factor 1.8 in a gradual way, with a hard backstop moved to 30° — and the three gusty cells were re-flown and completed under the predictive stack alone. Replayed offline on the flight-19 log, the revised criterion fires at 104.0 s, at 11° peak to peak and 1.8 s before the body-rate trip that actually fired in flight, and it does not fire on the decaying transients of the campaign. That is a replay, not a flight: the revised detector, like the one it replaces, has never fired on the aircraft.

12.1.7 Discussion

In simulation the custom stack flies all four geometries, and it keeps flying them through the transitions between them; the stock autopilot flies three and loses the fourth, for a structural reason rather than a tuning one. Part II qualifies the first half of that sentence without touching the second: on the aircraft O is not a hover shape, because holding it loads one arm servo to 44 % of stall and trips its protection within seconds (Sec. 6.7), and H cannot take off because its pitch stiffness demand exceeds the take-off moment cap, though it flies once airborne. The claim the two parts support together is the more careful one: a geometry-aware allocator keeps X, T and H flyable through morphing, and in O the limit is servo torque, not the control law.

The energy-aware optimiser is a real but not a free win. On the payload it takes 63 % to 81 % out of the rotor-speed spread and about 7 % out of the propeller power without touching the drift; under the replayed rotor fault it is the difference between an authority-dead airframe and a completed mission; and on the aircraft, with a physically damaged propeller (flight 17, three trials), it roughly halves the roll trim in every trial and cuts the spread in two of the three. What it never does is improve tracking, and in the June run that was lost it had pushed the geometry further than the sliding-mode model can follow. That is the role Chapter 8 claims for it — authority and thermal margin rather than accuracy — and the reason it matters most under a fault. The excursion bound it still lacks is the open item recorded in Sec. 12.1.4.

Sliding mode versus predictive control: the trade-offs. No global winner survives the evidence. In simulation, on one plant and one matrix, the sliding-mode stack is a little tighter almost everywhere and no more than that; on hardware, in the nominal geometry, the predictive stack is the tighter controller by 29 % on the one figure-of-eight it flew, and it lost the aircraft on its first real reconfiguration. Table 9.3 condenses the structural contrast, and Table 12.2 and Part II carry the measured figures. The pattern is the one the quadrotor literature reports on a smaller scale. The most direct comparative study available, on real agile quadrotors, finds that NMPC tracks better, especially near actuator limits where prediction and constraint handling pay, but that it is heavier and more prone to convergence trouble under large unmodelled disturbances, and it makes the point that a fast, robust inner loop does more for disturbance rejection than model fidelity inside the predictor [49]. This project shows the same trend in stark form: in the T geometry the real plant returned about three times the model gain at 3 Hz (Sec. 6.8), and the predictive controller, whose margin is bought with model fidelity, lost the aircraft, while the sliding-mode controller, which buys its margin with loop bandwidth, flew the same geometry on three separate flights. The comparison is therefore about model error more than about controller quality: prediction paid off in the geometry whose model was good and lost the aircraft in the one whose model was not, and a morphing platform changes its own model every time it reconfigures. Constraint handling remains the provable advantage of receding-horizon control [50], made real-time feasible by code-generated RTI solvers [16] such as the acados implementation used here [17]; the sliding-mode side of the ledger is equally classical, invariance to matched uncertainty at the price of chattering managed through a boundary layer [32, 33]. The practical conclusion is complementarity, which is no longer only a design direction: the deployed predictive launch carries the sliding-mode stack as a hot standby, and in flight 19 the standby did take over — correctly, and about a second too late to matter (Sec. 12.1.6).

The actuator constant. One lesson of the predictive campaigns is worth stating on its own, because it is easy to draw the wrong one. The lag of the actuator model has taken two values in this project: 0.08 s from small-signal measurements, and 0.15 s, the large-step bench value, adopted when the quadratic program proved ill-conditioned at the smaller one (Sec. 6.6.1). With the model matched to its plant the controller worked; with the two apart by a factor of two it failed in the least forgiving cases. At 0.08 s against a simulated plant at 0.15 s the solver collapsed in six runs out of six. At 0.15 s against an aircraft that, with its two-blade propellers, answers in about 0.08 s, the figure-of-eight in X flew — it is the best tracking of the whole project — and the T geometry oscillated until the aircraft was lost. The numbers reported in this part were therefore obtained with the model matched to the simulated plant, and the flight configuration carries the constant matched to the aircraft. What the evidence supports is not that one value is right and the other wrong, but that the predictive controller is sensitive to the agreement between its actuator model and the actuator it drives, to the point of losing the vehicle when they disagree by a factor of two, and that the constant has to be re-identified whenever the propulsion changes.

Limitations. Four limitations bound Part I, and Chapter 13 carries them forward. First, all of it is PX4 SITL with Gazebo, which validates the control logic and the closed-loop dynamics (Sec. 10.1.5) but not sensing, actuation or aerodynamics; and because the autopilot and the simulator advance in lockstep while the controllers run on the host clock, their effective rate in vehicle time equals the nominal one only when the real-time factor is close to one, which was not logged. Second, almost every condition is a single run, so a large and consistent difference is a realisation and not a mean; apart from the three-trial twin of flight 17, the repeated trials of Part II are the only place this thesis has a standard deviation. Third, the payload and the stock-autopilot results were obtained on the design-stage twin and are not figures for the final platform. Fourth, the predictive stack was not taken through the fault matrix at the full injected severity of Sec. 12.1.5, and it was never flown under a fault on the aircraft: the comparison of the two stacks under a rotor fault rests on the twin of flight 17 (Part II), at the severity the aircraft actually had.

Part I is simulation evidence for the control architecture, checked in Part II against real flight on the geometries and disturbances the platform could sustain. Of the two central claims taken to the field, one was tested and partly survived: the custom stack does keep X, T and H flyable through morphing, whereas O proved limited by servo torque and became a transit-only configuration. The other, decimetre-level payload rejection, was never tested on hardware, because no payload flight was made. And the campaign found a limit the simulation had not anticipated: the arm-and-servo compliance mode that ended flight 19.

12.2 Part II — Real-world flight tests

This part reports the quantitative results of the outdoor campaign of Chapter 11 on the flights in which the vehicle completed the commanded experiment: the hover of flight 7, the figure-of-eight tours of flights 10 and 15, the helix tours of flights 14 and 16, the fault-tolerant morphing trials of flight 17 and the predictive-control flights 18 and 19 (for flight 19, the X-configuration phase; the loss of control that followed its reconfiguration to T is analysed in Sec. 6.8 and is not a result of this part). Two streamed references were flown: the 3×1 m figure-of-eight with a 20 s period and the ascending helix of 1.5 m radius, 12 s period and 5 m climb, both from a 2 m hover. Wherever a quantity was measured more than once it is reported as mean ± sample standard deviation with the number of repetitions n — flights, runs, laps or hover phases, as stated; single measurements are marked $n = 1$. The simulation figures quoted alongside come from the final campaign of Part I, flown on the same references (Table 12.2).

12.2.1 Hover and in-flight reconfiguration

Table 12.4 collects the steady hover phases at 2 m of the sliding-mode tour flights (phases of at least ten seconds, flights 14 and 15). The hover altitude is held within 0.06 m in every phase and every shape; tilt at the 95th percentile lies between 5 and 6° in X and T and rises in H, whose roll lever is the shortest of the three presets. The motor-command spread measures the allocation imbalance of each geometry: in T two rotors carry the longitudinal lever alone and the spread is seven times the X value, while H stays within a factor of 1.3 of X. The first long hover of the platform (flight 7) held 101 s at 2 m with a maximum drift of 0.62 m and a median tilt of 1.17°; under the predictive stack (flight 18) the hover position standard deviation was $[0.089, 0.026, 0.018]$ m.

Table 12.4: Steady hover at 2 m on the real aircraft under the sliding-mode stack: mean ± SD over the steady hover phases (≥10 s) of flights 14 and 15, n phases. The altitude error is the 95th percentile of the deviation from the 2 m set-point; the motor spread is the mean difference between the highest and lowest normalised motor command.

Shape	n	Tilt p_{95} [°]	Altitude error p_{95} [m]	Motor spread
X	4	5.46 ± 1.11	0.032 ± 0.022	0.048 ± 0.006
T	2	5.15 ± 0.63	0.010 ± 0.008	0.349 ± 0.001
H	2	6.30 ± 1.16	0.020 ± 0.014	0.063 ± 0.012

Reconfiguration in flight was exercised on every tour: the X→T→H→X sequence of flights 10, 14 and 15 and the four X↔T and X↔H transitions of flight 16 were all completed. At the profile velocity of $9.6\,^\circ\,\mathrm{s}^{-1}$ used on flights 14 and 15 a 45°

transition takes 4.92 ± 0.10 s ($n = 6$) and the 90° T→H transition 9.25 ± 0.07 s ($n = 2$); at the deployed profile velocity of $19.24\,^{\circ}\,\mathrm{s}^{-1}$, flown on flight 16, the 45° transition takes 2.68 ± 0.05 s ($n = 4$), against 2.34 s of pure kinematics. During the transitions the altitude stays within 0.03 m of the set-point and the tilt at the 95th percentile is $6.8 \pm 1.2\,^{\circ}$ ($n = 6$), one to two degrees above the steady-hover value: the online update of centre of gravity, inertia and allocation keeps the transient of a reconfiguration at the level of an ordinary hover disturbance.

12.2.2 Trajectory tracking

Table 12.5 reports the tracking error on the two references, per flight and aggregated. For the figure-of-eight the planar RMSE over the laps of each flight is quoted and, where the same shape was flown twice, the mean ± SD across flights; the lap-to-lap SD within flights 15 and 19 measures the repeatability inside one flight. For the helix, flown once per shape on flight 14 and twice per shape on flight 16, the three-dimensional RMSE is the mean ± SD over the three runs, and Table 12.6 decomposes it by channel. Figure 12.6 shows the within-flight spread that the per-lap statistics summarise.

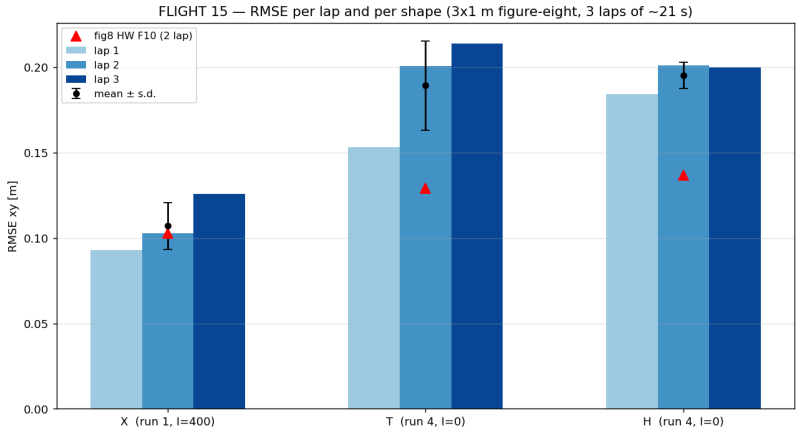


Figure 12.6: Planar RMSE of each lap of the 3×1 m figure-of-eight on Flight 15, by configuration. Treating every lap as an independent realisation is what turns a single flight into a repeatability measurement: the lap-to-lap spread within one flight is small against the difference between configurations, so the $\mathrm{X} < \mathrm{T} \approx \mathrm{H}$ ordering is not an artefact of which lap happened to be measured. It also bounds how much of the cross-shape difference in Table 12.5 can be attributed to geometry rather than to the day.

Three observations follow. First, X is the most repeatable condition of the campaign: the two sliding-mode figure-of-eight flights agree to 4 % (0.103 and 0.108 m, four days apart on different batteries) with a lap-to-lap SD of 0.017 m, and the predictive stack closes the same figure at 0.072 ± 0.013 m per lap (0.079, 0.058 and 0.080 m; Figure 12.7). Second, on the morphed shapes the planar error

Table 12.5: Tracking error in real flight: planar RMSE of the figure-of-eight and 3D RMSE of the helix, in metres. Per-flight values are the RMSE over the laps of that flight (two laps on flight 10, three elsewhere); "across flights" is mean ± SD over flights; "per lap" is mean ± SD over the three laps of flight 15 (SMC) or flight 19 (NMPC); the helix statistics are over the three runs per shape (flight 14 once, flight 16 twice). SITL rows: the final simulation campaign on the same references (Table 12.2), clean air / gusts, one run per cell.

Reference	Stack	Statistic	X	T	H
fig-8	SMC	F10 ($n = 1$)	0.103	0.129	0.137
fig-8	SMC	F15 ($n = 1$)	0.108	0.191	0.195
fig-8	SMC	across flights ($n = 2$)	0.106 ± 0.004	0.160 ± 0.044	0.166 ± 0.041
fig-8	SMC	per lap, F15 ($n = 3$)	0.107 ± 0.017	0.189 ± 0.032	0.195 ± 0.009
fig-8	NMPC	F19, 3 laps ($n = 1$)	0.073	—	—
fig-8	NMPC	per lap, F19 ($n = 3$)	0.072 ± 0.013	—	—
helix	SMC	F14 ($n = 1$)	0.118	0.221	0.230
helix	SMC	F16, two runs	0.128, 0.170	0.128, 0.141	0.162, 0.187
helix	SMC	across runs ($n = 3$)	0.139 ± 0.028	0.163 ± 0.050	0.193 ± 0.035
fig-8	SMC	SITL clean / gusts	0.031 / 0.052	0.025 / 0.050	0.024 / 0.053
fig-8	NMPC	SITL clean / gusts	0.033 / 0.067	0.063 / 0.077	0.034 / 0.058
helix	SMC	SITL clean / gusts	0.051 / 0.059	0.055 / 0.058	0.048 / 0.062

Table 12.6: Helix tracking on hardware decomposed by channel, mean ± SD over the three runs per shape ($n = 3$: flight 14 and the two runs of flight 16). Tilt is the 95th percentile of the total tilt over the helix window; no motor command exceeded 0.95 in any run.

Quantity	X	T	H
planar RMSE [m]	0.135 ± 0.029	0.158 ± 0.049	0.188 ± 0.035
vertical RMSE [m]	0.031 ± 0.005	0.042 ± 0.013	0.044 ± 0.002
planar error p_{95} [m]	0.25 ± 0.10	0.29 ± 0.08	0.34 ± 0.08
max vertical error [m]	0.065 ± 0.010	0.070 ± 0.008	0.086 ± 0.019
tilt p_{95} [°]	6.7 ± 0.8	6.3 ± 0.8	7.4 ± 1.0
roll RMS [°]	2.4 ± 0.5	2.2 ± 0.1	2.9 ± 0.2
pitch RMS [°]	2.1 ± 0.2	2.0 ± 0.5	2.1 ± 0.3
yaw SD [°]	1.7 ± 0.4	1.2 ± 0.2	1.0 ± 0.2
motor spread	0.047 ± 0.006	0.343 ± 0.013	0.071 ± 0.007

grows by 50 % to 60 % relative to X and its flight-to-flight spread widens, with an SD of 0.04 m against 0.004 m in X: the morphed geometries are trackable, but less repeatable across sessions, the T helix in particular ranging from 0.128 to 0.221 m over three runs. Third, the vertical channel is the quiet success of the campaign: on the helix the altitude RMSE is 0.03 m to 0.045 m in every shape, with a maximum excursion below 0.08 m on flight 14 and below 0.11 m over all three runs of every shape, on a 5 m climb (Figure 12.8), and on the figure-of-eight the altitude span never exceeds 0.09 m. Attitude activity is essentially independent of the geometry — roll and pitch RMS between 2 and 3°, tilt at the 95th percentile between 6 and 7.5° — so the tracking penalty of T and H is paid in position, not in attitude, consistent with the shorter control levers of those shapes.

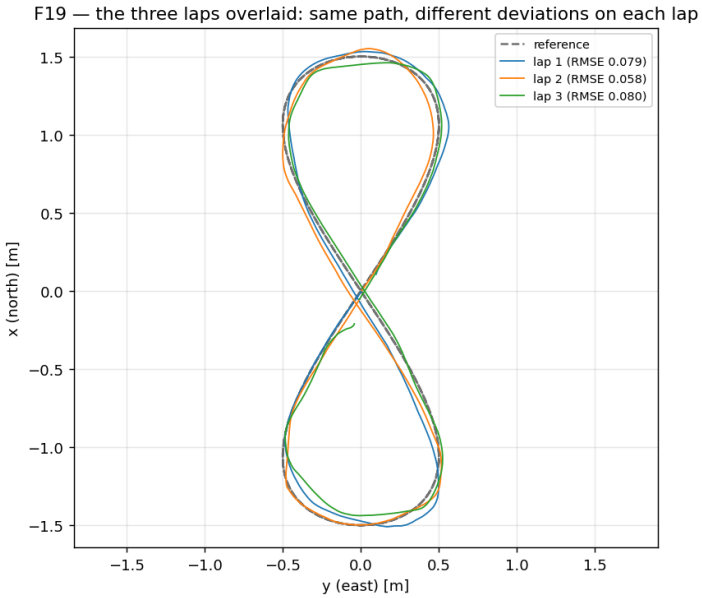


Figure 12.7: The three figure-of-eight laps of flight 19 (real flight, NMPC, shape X) overlaid on the reference: per-lap RMSE 0.079, 0.058 and 0.080 m, mean 0.072 ± 0.013 m. The best tracking the platform has produced; $n = 1$ flight.

12.2.3 Sliding mode versus NMPC in the nominal geometry

The hardware comparison between the two stacks is confined to the X configuration, the only geometry in which both flew the same trajectory (Table 12.7); the predictive stack's hardware evidence consists of two flights on one morning, so every NMPC figure is $n = 1$ at the flight level. On the figure-of-eight the predictive controller closes at 0.073 m against 0.103 m for the best sliding-mode flight and 0.106 ± 0.004 m for the two of them — a 29 % reduction of the planar error on the identical reference (31 % against the mean), obtained with the altitude held to 0.021 m RMS, roll and

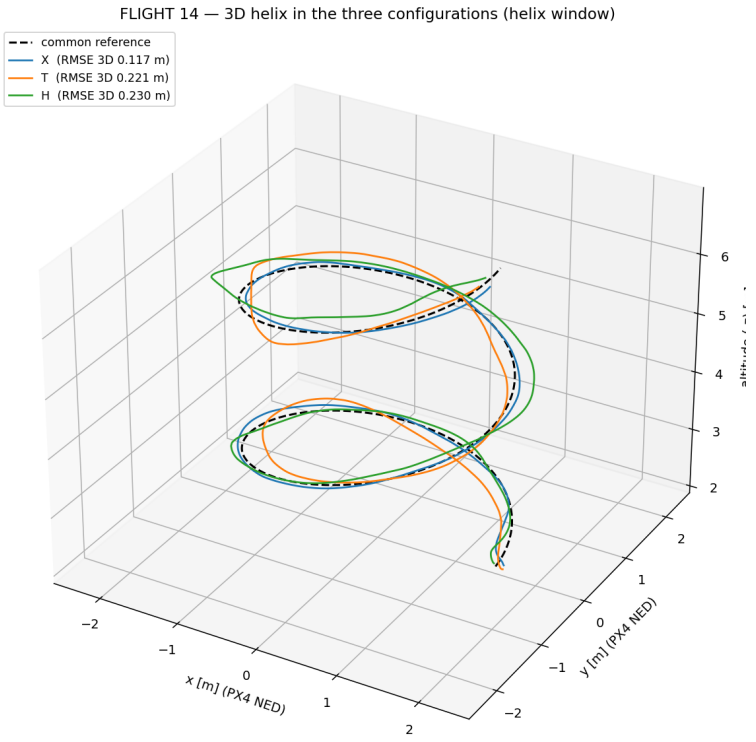


Figure 12.8: The ascending helix of flight 14 (real flight, SMC) flown in the X, T and H configurations against the common reference: 3D RMSE 0.118, 0.221 and 0.230 m — the first three-dimensional trajectory streaming on the hardware. One flight, one battery, $n = 1$ per shape; the repeated runs of flight 16 are in Table 12.5.

pitch RMS of 1.35° and a solver that never approached its 10 ms budget: median 0.90 ms, maximum 4.2 ms, no failed QP on flight 18. The one channel on which the predictive stack is worse is heading: its yaw wandered by 44° peak to peak over flight 18 against a 95th-percentile yaw error of 2.6° for the sliding-mode stack in hover, a consequence of the small yaw weight of the deployed cost (Sec. 9.3.1); the corrected weighting of Chapter 9 has not been flown. Morphing tours and the fault-tolerance experiment were flown under the sliding-mode stack only, so the comparison outside X rests on the simulation results of Sec. 12.1.3, and the comparison under a rotor fault on the simulated twin of flight 17 (Sec. 12.2.4).

12.2.4 Fault-tolerant morphing

Flight 17 carries the thesis claim on hardware. The propeller damaged in flight 9 was refitted on rotor 2 and three trials were flown, each within a single battery and each containing its own baseline: hover with the fault and the geometry fixed in X, then the geometry optimiser enabled in flight for 63 s to 72 s, then an

Table 12.7: The two stacks on the real aircraft in the X configuration. Sliding-mode figure-of-eight: flights 10 and 15; predictive: flight 19 (tracking) and flight 18 (hover and solver).

Quantity	SMC	NMPC
fig-8 planar RMSE, all laps	0.103 / 0.108 m (F10, F15)	0.073 m (F19), −29% against the best SMC flight
fig-8 planar RMSE per lap	0.107±0.017 m (F15, $n = 3$)	0.072±0.013 m (F19, $n = 3$)
altitude in the fig-8	span 0.078 m (F15)	0.021 m RMS, span 0.092 m
attitude in the fig-8	roll/pitch RMS 1.74/1.95°, tilt p_{95} 5.8° (F15)	roll/pitch RMS 1.35°, tilt p_{95} 4.1°
hover at 2 m	max drift 0.62 m over 101 s (F7)	position std [0.089, 0.026, 0.018] m (F18)
yaw	p_{95} error 2.6° (F10, hover)	44° peak to peak (F18)
solver, in flight	closed-form law	p_{50} 0.90, p_{95} 1.02, max 4.2 ms; 0 QP failures (F18)

autonomous landing with the optimiser still active. The within-flight design makes each comparison immune to session and battery drift. The measured thrust deficit of the damaged rotor was 25 ± 3 % ($n = 3$; 28, 26 and 22 %), about 0.65 of the severity injected in the simulation campaigns. Table 12.8 gives the fixed-geometry phase against the optimiser tail — the last 20 s before the landing command — for each trial and as mean ± SD over the three.

Table 12.8: Flight 17, per-trial metrics: fixed-geometry phase → optimiser tail (last 20 s before landing), and mean ± SD over the three trials. Damaged propeller on rotor 2; $n = 3$ within-flight trials of one fault realisation.

	Trial 1	Trial 2	Trial 3	Mean ± SD
thrust deficit of rotor 2	−28%	−26%	−22%	-25 ± 3%
ω_2 [rad s^{-1}]	544.9 → 519.4	554.5 → 522.5	556.1 → 548.4	$552 \pm 6 \rightarrow 530 \pm 16$
rotor-speed spread [rad s^{-1}]	92.1 → 58.1	87.0 → 55.2	74.5 → 79.6	$84.5 \pm 9.1 \rightarrow 64.3 \pm 13.3$
roll trim bias [N m]	0.615 → 0.297	0.617 → 0.266	0.457 → 0.282	$0.56 \pm 0.09 \rightarrow 0.28 \pm 0.02$
pitch trim bias [N m]	−0.544 → −0.086	−0.472 → +0.007	−0.473 → −0.111	$-0.50 \pm 0.04 \rightarrow -0.06 \pm 0.06$
tilt p_{95} [°]	2.88 → 3.93	3.10 → 4.13	4.29 → 3.45	$3.4 \pm 0.8 \rightarrow 3.8 \pm 0.3$
lateral position SD [m]	0.091 → 0.137	0.153 → 0.055	0.128 → 0.078	$0.12 \pm 0.03 \rightarrow 0.09 \pm 0.04$
max lateral excursion [m]	0.407 → 0.494	0.792 → 0.279	0.577 → 0.366	$0.59 \pm 0.19 \rightarrow 0.38 \pm 0.11$

The central observation is the thesis of this work stated as a measurement: *the trim migrates from actuation to geometry.* The roll moment the controller must hold falls from 0.56 ± 0.09 to 0.28 ± 0.02 N m, a reduction of 49 ± 10 % present in all three trials, and the pitch bias is driven from 0.50 N m to within 0.11 N m of zero (-87 ± 13 %), while the optimiser is never told that a propeller is broken: it sees only the commanded moments and minimises thrust spread, and in doing so unloads the damaged rotor. The reconfiguration is repeatable: the final arm angles

agree across the three trials to within 4° on every arm ([55.5, 246.9, 286.9, 139.5]°, SD ≤4.1°) and the final geometric centre of gravity to within 2 mm (-5.0 ± 2.0 and -8.8 ± 0.3 mm). The spread reduction, by contrast, is not the robust effect: it is 37 % in trials 1 and 2 and absent in trial 3, where the optimiser moved load onto the diagonal partner instead, so the mean over trials is -22 ± 25 %. Tracking is not improved by the reshape — the lateral position SD moves from 0.12 ± 0.03 to 0.09 ± 0.04 m and the tilt at the 95th percentile from 3.4 ± 0.8 to 3.8 ± 0.3 °, both within their spread — and no protection tripped in any trial. The predictive-stack counterpart of this experiment was not flown on the aircraft.

The simulated twin of the experiment was flown in the final campaign under both stacks, three trials each, with the fault injected at the severity the aircraft actually had (0.65 of the modelled deficit) and the same fixed-then-optimiser protocol. The reshape is the same one. Under the sliding-mode stack each arm repeats its final angle across the three trials to an SD of 0.6°, under the predictive stack to 1.3°; the two stacks agree with each other to one or two degrees on every arm; and both end within 1° to 6° of the mean geometry the aircraft found (Fig. 12.9). The damaged rotor is unloaded to the same speed under both (546 to 505 rad s^{-1} and 545 to 504 rad s^{-1}), the spread falls from 74 to 20 rad s^{-1} under the sliding-mode stack and from 80 to 19 rad s^{-1} under the predictive one, and at this severity the predictive solver never failed. What differs is how far the relief goes: the sliding-mode twin takes 73 ± 1 % out of the spread where the aircraft took 22 ± 25 %, and 88 ± 0.3 % out of the roll trim (0.41 to 0.05 N m) where the aircraft took 49 ± 10 %. The simulated hover is also much quieter than the real one. Tilt at the 95th percentile stays below 1° in the sliding-mode twin and in the optimiser tail of the predictive twin, whose fixed-geometry phase reaches 1.4° to 3.7°, and lateral excursion is 3 cm to 7 cm in the twins against 28 cm to 49 cm on the aircraft. The simulator has no wind, no vibration and no motor-to-motor scatter. The twin therefore predicts the mechanism and the geometry, and overstates how much the reshape relieves the controller. One caveat concerns the predictive side: the observer estimate is not the same quantity as the sliding-mode bias, and its roll component does not decay with the reshape (-0.90 to -0.87 N m) while its pitch component does (0.85 to 0.49 N m), so the comparison of trim relief is made on the sliding-mode stack only.

12.2.5 The cost of each geometry

The tour flights measured what no simulation had: the operating cost of each geometry in battery power and allocation imbalance (Table 12.9). Hover power is the battery-side mean over the steady hover phases of flights 14 and 15 and over the whole shape session of flight 16, one value per flight and shape; because the three packs did not start at equal charge, the per-flight ratio to X is the comparable quantity and the absolute values are indicative.

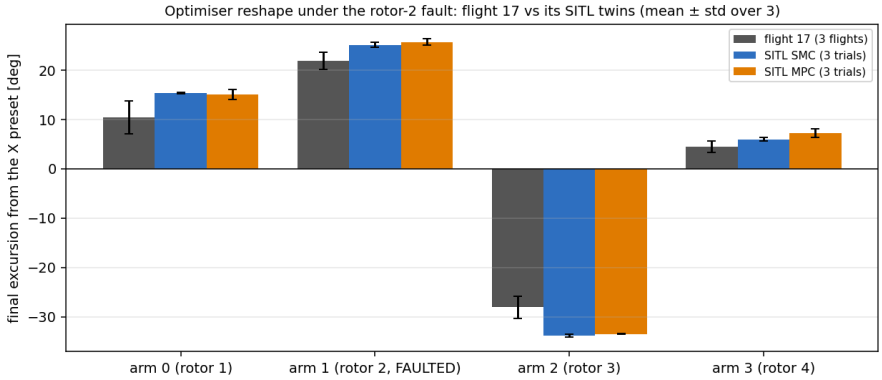


Figure 12.9: Final excursion of each arm from the X preset under the rotor-2 fault, mean ± SD over three: flight 17 against the sliding-mode and predictive twins. The same arm moves the same way in all three; the simulated reshape is a few degrees deeper.

Table 12.9: The price of each geometry on the real aircraft. Hover power per flight and mean ± SD over the three tour flights; the power premium is the mean ± SD of the per-flight ratios to X; the motor spread is the helix statistic of Table 12.6. Servo loads per Sec. 6.7.

Shape	Hover power [W] F14	F15	F16	Mean ± SD [W]	Δ vs X [%]	Motor spread ($n = 3$)
X	750	745	712	736 ± 21	ref.	0.047 ± 0.006
T	920	875	800	865 ± 60	$+17.5 \pm 5.1$	0.343 ± 0.013
H	775	780	723	759 ± 31	$+3.1 \pm 1.6$	0.071 ± 0.007
O	+25.1% vs X, one flight (Sec. 6.9)					—

The two expensive shapes are expensive for different reasons. T pays for allocation: the two rotors that carry the longitudinal lever run hot while the other two run light, and the aerodynamics barely change. About a fifth of the O premium is aerodynamic — 25.1 % of measured power against a modelled thrust demand of 19.6 %, the remaining five and a half points coming from disc overlap and wash — but what bounds O is servo torque rather than energy, which is why it is a transit shape (Sec. 6.7). H is the cheap shape: its 3.1 % premium sits inside the flight-to-flight scatter of the measurement, and what it cannot do is take off, its pitch stiffness demand exceeding the take-off moment cap. Choosing a geometry on this platform is therefore a trade between energy and actuation limits, with X the cheapest shape for a healthy vehicle — which is why the optimiser leaves a healthy vehicle near X and earns its place only under a fault (Chapter 8).

12.2.6 Simulation versus flight

Table 12.10 places the final simulation campaign next to the hardware statistics of this part on every quantity that was measured in both, and Fig. 12.10 plots the tracking rows. Where the hardware side was repeated the mean ± SD is quoted, and the ratio is taken against the gusty-world simulation, the closer of the two simulated conditions to an outdoor flight.

Table 12.10: Simulator against aircraft on matched quantities. SITL values are from the final simulation campaign (September 2026) and, for the fault rows, from the sliding-mode twin of flight 17 at the flown severity; hardware values are the statistics of this part. Hover tilt: hover phases of flights 14 and 15 against the hover phase of the simulated X figure-of-eight runs; the hover-collective residual is the ESC-map effect of Sec. 6.4, and the collective row is the model prediction through the flight ESC map, not the simulator's own map, on which the plant hovers at 0.56 (Appendix B); the reconfiguration reference is the kinematic time at the deployed profile velocity.

Quantity	SITL clean	SITL gusts	Hardware	Ratio
fig-8 RMSE, SMC, X [m]	0.031	0.052	0.106 ± 0.004 ($n = 2$)	2.0×
fig-8 RMSE, SMC, T [m]	0.025	0.050	0.160 ± 0.044 ($n = 2$)	3.2×
fig-8 RMSE, SMC, H [m]	0.024	0.053	0.166 ± 0.041 ($n = 2$)	3.1×
fig-8 RMSE, NMPC, X [m]	0.033	0.067	0.072 ± 0.013 ($n = 3$ laps)	1.1×
helix RMSE, SMC, X [m]	0.051	0.059	0.139 ± 0.028 ($n = 3$)	2.4×
helix RMSE, SMC, T [m]	0.055	0.058	0.163 ± 0.050 ($n = 3$)	2.8×
helix RMSE, SMC, H [m]	0.048	0.062	0.193 ± 0.035 ($n = 3$)	3.1×
hover tilt p_{95}, SMC, X [°]	1.5	4.3	5.46 ± 1.11 ($n = 4$)	1.3×
hover collective (model)	0.415–0.416		0.381–0.417	≈ 1
FTM: rotor-speed spread change	−73%		$-22 \pm 25\%$ ($n = 3$)	0.3×
FTM: roll-trim change	−88%		$-49 \pm 10\%$ ($n = 3$)	0.6×
45° reconfiguration [s]	2.34 (kinematic)		2.68 ± 0.05 ($n = 4$)	1.15×

On the gross physics the twin is calibrated: hover collective, thrust ratios and altitude dynamics match the aircraft within a few percent, which is what the identification of Chapter 6 paid for, and a reconfiguration takes 15 % longer than its kinematic time. On position error the twin is optimistic by a factor of two to three for the sliding-mode stack even against its gusty world, with the largest gap on the morphed shapes and on the helix. For the predictive stack in X the gusty prediction matches the flight within its lap-to-lap spread, the one regime in which the simulator was quantitatively predictive — with the qualification that the simulated stack ran with its actuator model matched to the simulated plant rather than to the aircraft (Sec. 12.1.7). On attitude the twin is optimistic as well: the hover tilt of the real vehicle is about 1.3 times the gusty simulation and three and a half times the clean one, because propeller vibration, servo jitter and real wind are not in the model. On fault tolerance the twin reproduces the reshape but not its effect: at the flown severity it takes three times more out of the rotor-speed spread

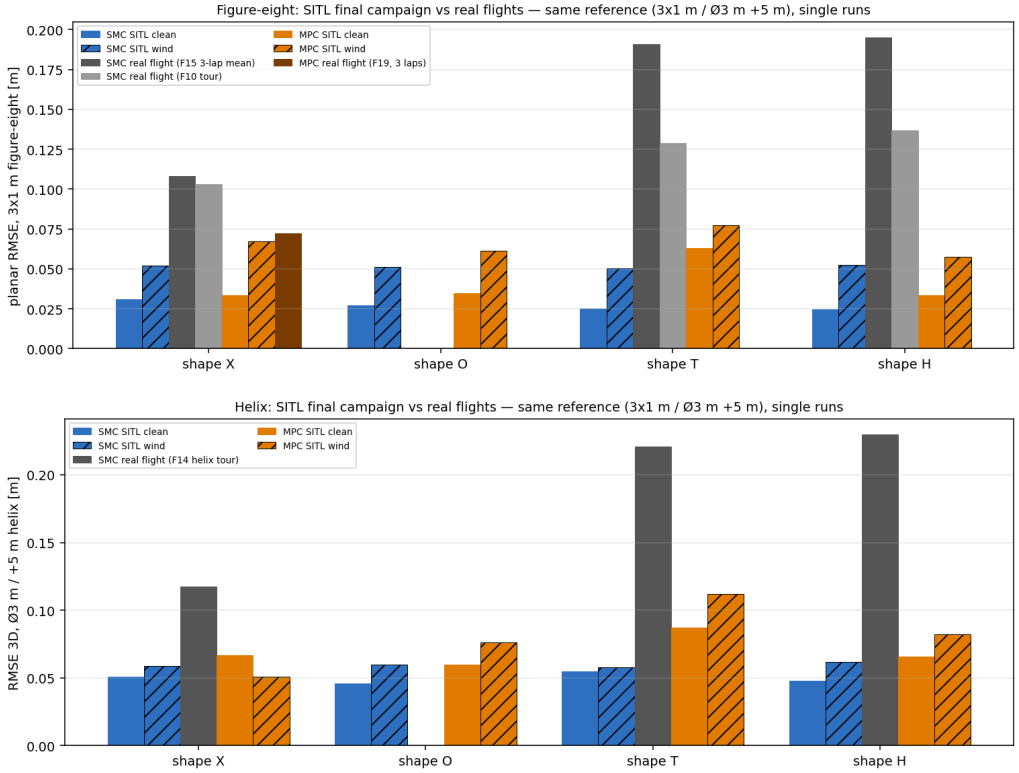


Figure 12.10: The final simulation campaign against the real flights on the same references, per shape: figure-of-eight (top) and helix (bottom), sliding-mode and predictive stacks in clean air and gusts, against flights 10 and 15 (figure-of-eight, SMC), flight 14 (helix, SMC) and flight 19 (figure-of-eight, NMPC). No flight exists in O.

and nearly twice as much out of the roll trim than the aircraft did (Sec. 12.2.4). The practical reading for the next campaign is that the simulation figures of this platform are lower bounds on error, by a factor of about three on position, except for the predictive controller inside its modelling envelope, where they can be taken at face value.

12.2.7 Summary of the flight results

Stated without inflation, the flights established the following on hardware. The platform hovers with a tilt of 5.5 ± 1.1 ° at the 95th percentile and holds altitude within 0.06 m; it tracks streamed planar and three-dimensional references in three configurations, with a planar RMSE of 0.11 m to 0.14 m in X and 0.16 m to 0.19 m in T and H and a vertical RMSE below 0.045 m everywhere; it reconfigures in flight in 2.7 ± 0.1 s per 45° with an altitude transient below 0.03 m; the geometry optimiser, blind to the fault, halves the roll trim of a rotor with a 25 % thrust deficit in three trials out of three, with a reshape repeatable to 4°; the predictive

controller runs at a median solve time below 1 ms and tracks 29 % tighter than the sliding-mode stack in the nominal geometry; and the cost of each geometry is measured, $+17.5 \pm 5.1$ % of hover power for T and $+3.1 \pm 1.6$ % for H. Every statistic rests on two to six repetitions, and the predictive-stack figures on single flights; the repetition campaign that would turn them into publishable statistics, and the predictive-stack morphing and fault-tolerance flights, are the declared next steps (Sec. 11.7).

Chapter 13

Conclusions and future work

13.1 Conclusions

This thesis set out to design, build and control a morphing quadrotor equipped with independently rotating arms, ensuring accurate control *across* its configuration space. The resulting contributions include an original airframe that was designed, built, and flown; an online variable-geometry model (Chapter 5) whose constants were identified through a dedicated bench-and-flight campaign (Chapter 6); a sliding-mode control stack featuring a moment-bias estimator that rejects unmodelled disturbances without requiring prior knowledge of their source (Chapter 7); an energy-aware geometry optimiser (Chapter 8); and a nonlinear model-predictive controller incorporating a supervisory sliding-mode fallback (Chapter 9). Validation was conducted on two parallel tracks: a multi-stage simulation pipeline (Chapter 10) and an outdoor campaign comprising nineteen instrumented flights, executed between 19 July and 6 August 2026 over twelve field days (Chapter 11).

The flight campaign is the critical element that transforms the design into a verified result. It documents the vehicle's progression from its first sustained hover (Flight 7), which achieved 101.1 s of hovering at 2.02 m altitude with a maximum drift of 0.62 m, to protected in-air reconfiguration (Flight 13). Subsequently, the hardware demonstrated the first three-dimensional trajectory streaming, executing an ascending helix in X, T, and H geometries with 3D RMSE values of 0.118, 0.221, and 0.230 m, respectively. During this manoeuvre, the altitude error never exceeded 0.08 m over a 5 m climb (Flight 14). The campaign further showcased fault-tolerant morphing with a physically destroyed propeller (Flight 17) and concluded with the first two hardware flights utilising the predictive control stack (Flights 18 and 19). All headline metrics regarding the real vehicle are reported with their corresponding sample sizes. Most results rely on one or two flights; these are point measurements rather than statistical distributions, a limitation reiterated in Sec. 13.2 where

relevant.

13.1.1 Objectives revisited

Table 13.1 returns the verdict on the eight objectives stated in Sec. 1.2, grouped into six themes, each with the measurement that supports it. All were met, two of them with conditions that the campaign — not the simulations — imposed: the predictive controller is safe only in the geometry its model describes, and the validation objective was met by flying the campaign, which then falsified part of what the simulations had claimed.

The four questions, answered. Through those objectives the four questions of Sec. 1.2 receive four answers, none of them uniform. *Can one control stack span the configuration space?* The sliding-mode stack succeeded without per-shape retuning beyond the online recomputation of inertia and allocation matrices, whereas the predictive stack did not. It emerged that actuation limits and structural constraints, rather than the control law itself, bounded the flyable set: the O configuration was rejected due to servo torque limitations, while the T configuration encountered a mechanical mode instability. *What is the morphing degree of freedom worth?* Its value lies in trim margin and residual actuator authority, not in tracking accuracy; moreover, every geometry deviating from X incurs a penalty in hover power consumption. *Can geometry be used as a fault-tolerance mechanism?* Yes, on hardware and repeatably, by an optimiser that is never told the fault exists. *Does a predictive controller pay for itself?* In the geometry its model describes, by 29 % on the only trajectory both stacks flew; outside it, the same flight that proved the first half proved the second. In the X configuration the predictive controller flew the figure-of-eight at 0.0732 m of planar RMSE over three laps against 0.103 m for the best sliding-mode flight of the identical trajectory, with the solver closing every step inside its 10 ms budget; in T, where the sliding-mode stack had flown three clean laps, it went unstable in hover and the aircraft was lost. The sliding-mode stack — whose gains are scheduled on the inertia of the current shape, and which never claimed a plant model beyond it — remained the safe stack across the configuration space, and the one the supervisor adopted when the predictive controller failed.

A methodological result: flight is the identification. Every parameter identified on the single-rotor bench and subsequently re-identified in flight was found to be incorrect. Specifically, the thrust coefficient differed by a factor of 1.78, the torque coefficient deviated by more than its own confidence band width, the transport delay varied by an order of magnitude, and both servo tuning decisions promoted at the bench were reversed by in-flight A/B tests. These bench measurements are not errors; they are accurate within their specific experimental

Table 13.1: The objectives of Sec. 1.2 against what was measured. Fn = flight n; sample sizes as stated in Chapter 12.

Objective	**Verdict and evidence**	**Reference**
Build and characterise the platform	*Met, with its own design failure on the record*: the airframe flew nineteen times; propulsion and actuation were characterised on the bench and re-identified in flight	Ch. 3, 4
Variable-geometry model	*Met*: mass, centre of gravity, inertia and allocation are analytic in the measured arm angles and are recomputed at every control solve; the declared limit of the model is the rigid arm	Ch. 5; Sec. 6.8
Stable across configurations; rejection of unsensed disturbances	*Met over the flyable configuration space*: X, T and H fly and morph, three tours out of three on one gain set; H is not a take-off geometry and O is transit-only for actuator reasons, not control ones. What the estimator demonstrably rejected in flight is the trim of a damaged rotor, not a payload	F13–F17; Sec. 12.2.2
Morphing as a degree of freedom, and as fault tolerance	*Met, and demonstrated on hardware under a real fault*: roll trim roughly halved in all three trials, from 0.46–0.62 to 0.27–0.30 N m, with the cost of each geometry measured (T +12 to +23%, H +2 to +5%, O +25.1% of hover power)	F17; Secs. 12.2.4 and 12.2.5
Predictive controller with a fallback	*Met with conditions*: the NMPC runs on the companion computer at 0.90–0.97 ms of median solve time, every step inside its 10 ms budget, and tracks 29 % tighter than the sliding-mode stack in X — but the fallback is *supervisory*, not certified, and its one real activation came late	F18, F19; Secs. 12.2.3 and 6.8
Validate in simulation, and fly the campaign	*Met, and partly self-falsifying*: the nineteen flights were flown as stated, and of the two Part I claims taken to the field one survived with conditions and one was never tested, and the campaign found a limit the simulation had not anticipated	Sec. 12.2.7

context, with one value agreeing with an independent vendor bench to 3 %. What the bench cannot reproduce is the *coupling* — it measures one rotor, rigidly restrained, at constant speed, on an unloaded arm, while the vehicle flies a plant in which rotor, airframe, servo-actuated arms and estimator are one system. For a morphing vehicle, flight is not the validation of the identification: it *is* the identification. The protection architecture built on the same principle earned its place: each of its three layers was written in response to a real failure and validated on the log of that failure, and under it every load trip taken in the air ended in an automatic landing with the aircraft intact.

Flight 19 is a result, not an accident. The flight that produced the best tracking of the campaign ended, twenty seconds after its first real reconfiguration, in a growing 3.03 Hz roll oscillation in pure hover. The root cause was identified from the crash log itself: an arm/servo compliance mode that makes the real plant carry about three times the modelled gain near 3 Hz in the T geometry — a resonant amplification, not a delay. The predictive model treats the arms as rigid, an assumption accurate below 2 Hz and false inside the closed-loop bandwidth, so the stability the design believed it had was computed for a plant whose loop gain at the resonance was in fact $|L| = 1.33$. This is the sharpest conclusion of the thesis, and it is a negative one: *a compliance mode is not containable by a rigid-arm model, and no choice of weights repairs a model that does not contain the mode.*

13.2 Limitations

The evidence in this thesis is no longer simulation-bound, but it is bounded in three other ways: by a crash whose fix has never been flown, by the shapes and controllers the campaign could not close, and by the number of repetitions behind every figure. The complete open-item register is Sec. 11.7; what follows is the perimeter of the claims made above.

- **The fix for the Flight 19 instability has never been flown.** Restoring the actuator lag to its measured small-signal value of 0.08 s reduces the loop gain in T from 1.33 to 0.78 on the identified plant. This yields a margin of 2 dB to 3 dB, falling short of the 6 dB assumed by the rigid-arm design, yet this correction has so far been checked only against the measured frequency response, not in flight. Flight cards 20–22 were prepared and committed but never executed; consequently, hardware evidence under the predictive controller is limited to one shape and one flight: T, H, the helices and any reconfiguration under prediction remain undemonstrated. The campaign concluded with an interim policy: predictive trajectories in X configuration with locked arms, while all multi-shape tours relied on the sliding-mode stack. This operational constraint represents a limitation of the current hardware validation rather than a demonstrated capability.

- **The compliance mode lies outside both the prediction model and the simulator.** The simulated arms are rigid, so the mode that destroyed the aircraft cannot be reproduced or anticipated on the current plant at any fidelity; the crash was reproduced afterwards only by substituting the flight-measured arm response for the nominal actuator model.

- **The O configuration is not a hover shape.** At +100° the most loaded servo sits at 40 % to 44 % of stall with 6.3° of droop and trips the load watchdog after 3.6 s: with the present actuators O is a transit shape, and the binding constraint is servo torque, not the control law.

- **The supervisory fallback has never been exercised as a test on hardware.** It is validated in closed-loop simulation — failover in 0.20 s on a stream freeze and 0.52 s on sustained solver failure — and by unit tests. Its one real activation fired correctly on the hard body-rate threshold but roughly twenty seconds after the onset of the instability, because no threshold then existing watched an oscillation that *grows*.

- **The statistics rest on one to six repetitions per condition, mostly two or three.** Nothing in Part II of Chapter 12 reaches the three matched repetitions a publishable statistic would need; every mean ± SD there is taken over laps, hover phases or within-flight trials, and cross-shape comparisons remain confounded by protocol variations, battery state and session drift.

- **The thrust coefficient is a composite.** Since rotor speed is not measured in flight, the identified parameter represents k_T composed with the ESC map at a specific operating point. These two elements are inseparable within the current data set, and the torque ratio likewise remains an interval.

- **Two comparisons are one-sided.** Only the sliding-mode half of the fault-tolerance experiment was flown on hardware — the predictive twin is the unflown Flight 22, although both stacks were run against the simulated fault at the flown severity — and the comparison against the native autopilot never reached hardware, the geometry that produced its sharpest contrast being now off-limits for the custom stack as well.

- **Heading under the predictive controller wanders**: peak-to-peak deviations reached 44.2° on Flight 18 and 71° on Flight 19, compared to a 95th-percentile deviation of 2.55° for the sliding-mode stack on the same trajectory. The cause was identified post-campaign, and while the correction has been implemented, it remains unvalidated in flight.

- **Payload rejection is evidenced only in simulation.** The design case involving a 3 kg movable payload, which sized the moment-bias estimator, relies on simulations against a plant configuration that no longer exists; no actual flight carried a payload. In practice, the estimator demonstrated its ability to reject a different and arguably more challenging unmodelled disturbance: the trim shift induced by a physically damaged rotor.

- **Two estimator anomalies are mitigated but not fully understood**: a bias of approximately 2 m between EKF altitude and lidar readings, which is absorbed by the take-off datum, and a yaw reset ranging from 12° to 21° occurring within seconds of spin-up.

13.2.1 Negative results

Four measured refutations are reported as conclusions in their own right, because each cost a flight or a field day. First, **a servo integral gain that removes stick–slip on an unloaded bench does not remove it in flight**: it produced two static load trips and roughly three times the in-band roll vibration in a within-flight A/B, because under rotor-torque load the integrator nulls the error by pushing *through* the stiction. Second, **doubling the commanded servo velocity does not move the ∼6 Hz oscillation**: the stiction indicators improved by about a factor of three and the spectral peak did not move at all, which promoted the oscillation from motion artefact to structural mode — the root cause of the Flight 19 crash. The cure is a stiffer actuator, not a faster one. Third, **scaling the attitude weights does not detune the predictive controller's attitude stiffness**: the roll and pitch loop gain is inherited from the position and velocity weights through the model, so the per-shape derating machinery was measured to be a placebo and was neutralised, and the instability was cured by fixing the model rather than by softening the controller. Fourth, **the geometry optimiser does not improve tracking**: on the payload sequence the worst drift moved from 0.061 to 0.074 m with the optimiser enabled, and on hardware under a real rotor fault the lateral position SD and the tilt at the 95th percentile both stayed inside their spread. The optimiser redistributes thrust and halves the trim; its benefit lies in energy efficiency and residual control authority rather than disturbance rejection. At high fault severity, this distinction determines whether a mission is completed or the vehicle loses all control authority.

13.3 Future work

The campaign of Chapter 11 is closed, not pending; what follows are the directions it opened, in the order in which they should be taken.

1. **Fly the actuator-lag fix.** The small-signal constant $\tau_{\text{act}} = 0.08\,\text{s}$ represents the identified cure for the Flight 19 instability, so far checked only against the measured frequency response. Until the deliberate T- and H-settle test specified in the Flight 20 card is executed, morphing under the predictive controller cannot be considered proven safe on hardware.

2. **Remove the compliance mode, or model it.** The lag fix provides a margin of 2 dB to 3 dB. Structural remedies include replacing the 1.6 N m position-controlled servos with current-controlled units offering roughly twice the stall torque, or explicitly modelling the ∼6 Hz mode as a second-order attitude channel in the prediction model, at the cost of a stiffer optimal control problem. This same actuator upgrade would also enable the O configuration to function as a hover shape rather than merely a transit one.

3. **Re-measure the actuator lag with the current propellers across step amplitudes.** The constant is amplitude-dependent, rising from 0.079 s in small-signal conditions to 0.153 s for large steps (measured on the previous propeller set). The take-off regime with 15×5.5 propellers has never been measured; consequently, both the simulation plant and the controller constant must be updated simultaneously.

4. **Close the fault-tolerance comparison.** The sliding-mode side of this comparison is established (Flight 17), whereas the predictive side remains incomplete. Testing the same physical propeller on the same rotor under the identical three-trial protocol using NMPC constitutes the single missing data point for the thesis question, and the corresponding flight card has already been prepared.

5. **Move the sliding-mode loop into the autopilot.** The custom stacks run as ROS 2 processes outside the flight controller [46, 26], which is what made a geometry-aware allocator and an NMPC possible on this timescale, but it caps the feedback path at the bridge rate and adds about 30 ms of round trip (Sec. 6.3.1), so the offboard loop closes at 250 Hz against a native inner loop at 400 Hz to 800 Hz [51]. For a sliding-mode law that is not cosmetic: the width of the quasi-sliding band scales with the sampling period [52, 53]. The structural fix is to port controller, allocator and estimator into the firmware as a work-queue module on the angular-velocity callback [54].

6. **Bound the arm excursion in the optimiser cost.** The box constraint limits each joint separately and says nothing about the depth of a reshape; in the June payload runs that depth (55° to 70°), not the payload direction, was the controlling variable of the sliding-mode loss of control (Sec. 12.1.4), and an explicit excursion penalty is the missing term.

7. **Raise the statistics to $n \geq 3$.** Three repetitions of each shape and trajectory under a matched protocol — one battery, one wind condition, one session — separate the measurements of this thesis from publishable results, and are the only way to settle whether the cross-shape differences of Part II are physics or protocol. The same protocol should carry the one baseline the campaign

never flew: the native autopilot, on this airframe, in the shapes it is expected to lose. Pushing the fault severity past what the physical propeller provides — about 0.65 of the simulated ladder — belongs to the same programme.

8. **Co-optimise the arm angles inside the optimal control problem.** The predictive controller is already morphing-*aware*, but the geometry is an exogenous input produced by a separate 1 Hz optimiser gated on hover; making the arm angles decision variables of the same problem is the natural continuation, and it presupposes the compliance mode being in the model.

9. **Close the two estimator gaps, and instrument the rotors.** The EKF-versus-lidar altitude bias and the yaw reset at spin-up are measured and unexplained; and a single hover with working rotor-speed telemetry would make k_T and the ESC map independently observable, which no post-processing of the present data can do.

The central claim of this work — that geometry can be spent as a control resource under a real, physical rotor fault — was demonstrated in the air, three times out of three, by a supervisory layer that was never told a propeller was broken. The boundary of that claim was measured in the same fortnight and by the same aircraft: reconfiguration is a control resource only up to the frequency at which the mechanism that provides it stops being rigid. Neither half of that sentence could have been written from simulation, and the vehicle paid for the second one.

Appendix A

Jacobians of the predictive model

This appendix gives in closed form the Jacobians of the prediction model of Chapter 9: the partial derivatives of the continuous vector field $\boldsymbol{f}(\boldsymbol{z}, \boldsymbol{\omega}_c)$ with respect to the augmented state $\boldsymbol{z}$ and the rotor-speed command $\boldsymbol{\omega}_c$. They are the quantities evaluated at every shooting node, the real-time-iteration SQP relinearising the dynamics about the current trajectory and building a Gauss–Newton QP from them. The `acados`-generated code computes them by automatic differentiation; the blocks below document what that code evaluates and expose the two couplings that dominate the morphing regime — the thrust–tilt coupling and the gyroscopic term.

A.1 Linearisation point and block structure

The Jacobians are evaluated at a node $(\bar{\boldsymbol{z}}_k, \bar{\boldsymbol{\omega}}_{c,k})$ of the reference trajectory, with the geometry triple $(\mathbf{I}, \boldsymbol{x}_m, \boldsymbol{y}_m)$ recomputed from the measured arm angles and the additive disturbances $(\boldsymbol{d}_v, \boldsymbol{d}_M)$ held constant over the horizon (Sec. 9.2.3); all terms below are understood to be evaluated there, and the bar is dropped. Because the disturbances enter additively and are constant over the prediction, their partial derivatives vanish. The same holds structurally for the geometry: the triple is a per-solve *parameter*, not a state, so no block $\partial \boldsymbol{f}/\partial \boldsymbol{\theta}$ appears and the morphing transition itself is never predicted. The approximation is exact in a held shape and bounded elsewhere — at the deployed profile velocity of $19.24\,^\circ\mathrm{s}^{-1}$ the arms sweep about 5.8° within the 0.30 s horizon, a change the model sees only at the next solve, and the flight protocol commands shape transitions only in hover.

Ordering the augmented state of Sec. 9.2.1 as $\boldsymbol{z} = (\boldsymbol{p},\, \boldsymbol{v},\, \boldsymbol{\Theta},\, \boldsymbol{\omega}_b,\, \boldsymbol{\omega}_a)^\top \in \mathbb{R}^{16}$,

with block widths (3,3,3,3,4), the state Jacobian inherits the sparse form

$$\boldsymbol{A} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{z}} = \begin{bmatrix} \mathbf{0} & \boldsymbol{I}_3 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{A}_{v\Theta} & \mathbf{0} & \boldsymbol{A}_{v\omega} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{A}_{\Theta\Theta} & \boldsymbol{W} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{G} & \boldsymbol{A}_{\omega\omega} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & -\dfrac{1}{\tau_{\text{act}}}\boldsymbol{I}_4 \end{bmatrix}, \tag{A.1}$$

while the command enters exclusively through the actuator-lag state defined in Eq. (9.2), resulting in $\boldsymbol{B} = \partial \boldsymbol{f}/\partial \boldsymbol{\omega}_c = [\,\mathbf{0}\ \mathbf{0}\ \mathbf{0}\ \mathbf{0}\ \tau_{\text{act}}^{-1}\boldsymbol{I}_4\,]^\top$. Consequently, physical authority reaches the body indirectly: the command drives $\boldsymbol{\omega}_a$ through the lag dynamics, which in turn influences the body via the blocks $\boldsymbol{A}_{v\omega}$ and $\boldsymbol{A}_{\omega\omega}$ defined below.

A.2 Translational and kinematic blocks

From $\dot{\boldsymbol{v}} = g\,\boldsymbol{e}_3 - (T/M)\,\boldsymbol{b}_z(\boldsymbol{\Theta}) + \boldsymbol{d}_v$, where $T = k_T \sum_i \omega_{a,i}^2$ and $\boldsymbol{b}_z$ denotes the body up-axis direction, the velocity equation is independent of $\boldsymbol{p}$, $\boldsymbol{v}$, and $\boldsymbol{\omega}_b$. The absence of a velocity dependence reflects the fact that the rigid-body model excludes aerodynamic drag. The attitude dependence manifests as the *thrust–tilt coupling*

$$\boldsymbol{A}_{v\Theta} = -\frac{T}{M}\frac{\partial \boldsymbol{b}_z}{\partial \boldsymbol{\Theta}}, \qquad \frac{\partial \boldsymbol{b}_z}{\partial \boldsymbol{\Theta}} = \begin{bmatrix} s_\psi c_\phi - c_\psi s_\theta s_\phi & c_\psi c_\theta c_\phi & c_\psi s_\phi - s_\psi s_\theta c_\phi \\ -c_\psi c_\phi - s_\psi s_\theta s_\phi & s_\psi c_\theta c_\phi & c_\psi s_\theta c_\phi + s_\psi s_\phi \\ -c_\theta s_\phi & -s_\theta c_\phi & 0 \end{bmatrix}, \tag{A.2}$$

the block that lets the optimiser trade attitude for horizontal acceleration, and the dependence on the applied rotor speeds is the rank-one block

$$\boldsymbol{A}_{v\omega} = -\frac{2k_T}{M}\,\boldsymbol{b}_z(\boldsymbol{\Theta})\,\boldsymbol{\omega}_a^\top \in \mathbb{R}^{3\times 4}, \tag{A.3}$$

each rotor changing the thrust magnitude by $2k_T\omega_{a,i}$ along the common direction $-\boldsymbol{b}_z/M$.

The Euler-rate map $\dot{\boldsymbol{\Theta}} = \boldsymbol{W}(\boldsymbol{\Theta})\,\boldsymbol{\omega}_b$ of Eq. (9.5) gives two further blocks: the kinematic matrix itself,

$$\boldsymbol{W} = \begin{bmatrix} 1 & s_\phi t_\theta & c_\phi t_\theta \\ 0 & c_\phi & -s_\phi \\ 0 & s_\phi/c_\theta & c_\phi/c_\theta \end{bmatrix}, \qquad t_\theta = \tan\theta, \tag{A.4}$$

well conditioned because the tilt constraint of Eq. (9.8) keeps θ away from the $\pm\pi/2$ singularity, and $\boldsymbol{A}_{\Theta\Theta} = \partial\dot{\boldsymbol{\Theta}}/\partial\boldsymbol{\Theta}$, obtained by differentiating $\boldsymbol{W}\boldsymbol{\omega}_b$ with respect to ϕ and θ only, since $\boldsymbol{W}$ does not depend on ψ.

A.3 Rotational block: gyroscopic term and control effectiveness

With the diagonal configuration-dependent inertia $\mathbf{I} = \mathrm{diag}(I_{xx}, I_{yy}, I_{zz})$, differentiating the body-rate dynamics of Eq. (9.6) with respect to the body rates yields the *gyroscopic* block, zero on the diagonal and carried entirely by the inertia differences,

$$\boldsymbol{G} = \begin{bmatrix} 0 & -\dfrac{(I_{zz} - I_{yy})\, w_z}{I_{xx}} & -\dfrac{(I_{zz} - I_{yy})\, w_y}{I_{xx}} \\ -\dfrac{(I_{xx} - I_{zz})\, w_z}{I_{yy}} & 0 & -\dfrac{(I_{xx} - I_{zz})\, w_x}{I_{yy}} \\ -\dfrac{(I_{yy} - I_{xx})\, w_y}{I_{zz}} & -\dfrac{(I_{yy} - I_{xx})\, w_x}{I_{zz}} & 0 \end{bmatrix}. \tag{A.5}$$

Negligible near hover, it grows with the rates and with the inertia asymmetry that morphing induces — one of the reasons the nonlinear solve is preferred over a hover-only linearisation. The control effectiveness follows from the wrench of Eq. (9.3) through the per-rotor sensitivities $\partial M_\phi/\partial\omega_{a,i} = -2k_T y_i \omega_{a,i}$, $\partial M_\theta/\partial\omega_{a,i} = 2k_T x_i \omega_{a,i}$ and $\partial M_\psi/\partial\omega_{a,i} = 2k_Q \sigma_i \omega_{a,i}$, with $\boldsymbol{\sigma} = (+1, +1, -1, -1)$ the spin directions:

$$\boldsymbol{A}_{\omega\omega} = 2 \begin{bmatrix} -k_T\, y_0\, \omega_{a,0}/I_{xx} & \cdots & -k_T\, y_3\, \omega_{a,3}/I_{xx} \\ k_T\, x_0\, \omega_{a,0}/I_{yy} & \cdots & k_T\, x_3\, \omega_{a,3}/I_{yy} \\ k_Q\, \sigma_0\, \omega_{a,0}/I_{zz} & \cdots & k_Q\, \sigma_3\, \omega_{a,3}/I_{zz} \end{bmatrix} \in \mathbb{R}^{3\times 4}. \tag{A.6}$$

The motor positions (x_i, y_i) are the centre-of-mass-relative lever arms of Eq. (5.18), so this block carries the morphing dependence directly: as the arms rotate, the moment arms — and hence the roll/pitch control effectiveness — change with them.

What these Jacobians do not contain. The blocks above carry the *kinematic* morphing dependence but assume the arms to be rigid links, an assumption falsified in flight: on the loss-of-control record the true loop gain in the T configuration was about three times (10 dB) what the prediction model assumed at the limit-cycle frequency — not a transport delay, which would show as phase without magnitude, but the resonant amplification of an arm/servo compliance mode, and the entire design margin (Sec. 6.8). Capturing the mode would require augmenting the state with a second-order arm-compliance model, so that $\partial\dot{\boldsymbol{\omega}}_b/\partial\boldsymbol{\theta}$ and the corresponding stiffness and damping blocks appear in $\boldsymbol{A}$; neither the prediction model nor the simulation plant carries such a mode today.

A.4 Actuator lag and the discrete Jacobians

The first-order lag defined in Eq. (9.2) is linear and decoupled, contributing $-\boldsymbol{I}_4/\tau_{\text{act}}$ to $\boldsymbol{A}$ and constituting the entirety of $\boldsymbol{B}$. As the control input affects the state solely through this block, τ_{act} determines the scale of the entire input Jacobian. Although measured experimentally, it is not a fixed plant constant: the first-order fit increases monotonically with step amplitude, nearly doubling from the small-signal regime near hover to take-off-scale steps (Sec. 6.6), whereas the model accommodates only a single value. The associated risk is asymmetric. Underestimating τ_{act} leads the optimiser to assume control authority that the hardware lacks, while overestimation results in merely conservative behaviour. Consequently, the deployed value is the small-signal constant of 0.08 s, independently recovered from in-flight frequency-response fits (Sec. 6.6.1).

The solver works with the discrete map obtained by integrating $\boldsymbol{f}$ over one node spacing Δt. For an explicit single step the discrete sensitivities are $\boldsymbol{A}_d = \boldsymbol{I}_{16} + \Delta t\, \boldsymbol{A}$ and $\boldsymbol{B}_d = \Delta t\, \boldsymbol{B}$, already exposing the structure: $\boldsymbol{A}_d$ is a perturbation of the identity inheriting the sparsity of Eq. (A.1). For the deployed explicit Runge–Kutta integrator they are obtained by propagating the variational equations through the stages — precisely the variational integration `acados` generates — so the continuous blocks remain the building blocks regardless of integration order. One deployed modification acts downstream: since 2026-08-06 the Gauss–Newton Hessian assembled from them is regularised by a Levenberg–Marquardt term of 0.01, which changes not the Jacobians but the curvature of the QP built from them.

Appendix B

Physical parameters and symbol-to-code map

This appendix is the consolidated reference for the physical parameters of the morphing quadrotor and the map from the symbols used in this thesis to the identifiers in the implementation. Every constant lives in a single module of the flight code — `drone_params.py` for physics, `arm_geometry.py` for geometry, `dynamixel_driver.py` and `mission_manager.py` for the servo and failsafe thresholds — and the values below were verified by executing those modules with `MORPHING_REAL_HW=1`, the switch that selects the hardware ESC table; quantities computed from other constants at import time are marked *derived*. The identification evidence for each value is Chapter 6; the divergences between what the files declare and what actually flew are collected in Sec. B.4. One usage note applies to everything here: any offline re-analysis of real flight data must export `MORPHING_REAL_HW=1`, otherwise the linear SITL map is loaded and every recomputed rotor speed and thrust is meaningless — the error that once produced a spurious "k_T high by 14.3 %" estimate, later retracted (Sec. 6.4).

B.1 Geometry, mass and inertia

B.2 Propulsion, actuation and failsafe thresholds

The hardware throttle-to-speed table below is the thirteen-point map measured at the bench on a fresh pack under load (2026-07-15, 6S); it is selected only when `MORPHING_REAL_HW=1` and is guarded at import time (monotonicity, range, coverage). In simulation the linear $100\,\mathrm{rad\,s^{-1}}$ to $780\,\mathrm{rad\,s^{-1}}$ map is used instead, on both sides of the loop — which is why the thrust coefficient cancels in SITL

Table B.1: Airframe geometry and mass model of the vehicle (CAD 2026-07-31). Angles are absolute rotor-arm headings in the body FRD frame.

Quantity	Value
Arm length (hinge → rotor)	0.235 m
Body half-corner offset D	0.113072 m
X-neutral headings	$[45°, 225°, 315°, 135°]$
Rotor z-stagger	60 mm, by diagonals: arms {0,1} at 140 mm, {2,3} at 200 mm
Joint clamp	±135°
Propeller radius (15 × 5.5)	0.1905 m (measured)
Body mass (incl. battery)	1.790 kg
Connector mass (each of four)	0.180 kg
Arm link masses	$[0.085, 0.085, 0.092, 0.092]$ kg
Rotor assembly mass (each of four)	0.275 kg
Total rigid mass	3.964 kg (derived)
Hover thrust	38.887 N (derived)
CoG height above bottom plate	101.6 mm
CoG planar offset	$(-0.30, +0.83)$ mm

Table B.2: Shape presets as absolute FRD headings with the corresponding joint angles, and the configuration quantities derived at runtime by `inertia_diag` and `mean_lever_arm_frd`. All presets sit inside the ±135° joint clamp with at least 35° of margin.

Shape	Headings / joints [°]	Roll lever m	Pitch lever m	$[J_{xx}, J_{yy}, J_{zz}]$ kg m^2
X	$[45, 225, 315, 135]$ / [0,0,0,0]	0.2792	0.2792	$[0.1217, 0.1198, 0.2030]$
T	$[90, 180, 270, 180]$ / $[-45, +45, +45, -45]$	0.2306	0.2306	$[0.1067, 0.0990, 0.1720]$
H	$[0, 180, 0, 180]$ / $[+45, +45, -45, -45]$	0.1131	0.3481	$[0.0354, 0.1752, 0.1773]$
O	$[145, 325, 55, 235]$ / −100 on all four	0.1636	0.1636	$[0.0611, 0.0611, 0.0999]$

(Sec. 11.2.1).

B.3 Symbol-to-code map

Symbols map onto the implementation as follows, all files living in the `morphing_drone` package: the total mass is `TOTAL_RIGID_MASS`; the configuration-dependent inertia and mean levers are `inertia_diag(theta)` and `mean_lever_arm_frd(theta)` in `arm_geometry`; the absolute headings are `SHAPE_PRESETS` and the joint angle `wrap180(THETA_BASE - theta_abs)`; the rotor coefficients are `kT` and `kT * KQ_OVER_KT`; the rotor-speed box is `N_MIN`/`N_MAX` and the normalised command `px4_control_from_omega()`; the NMPC actuator lag is `TAU_ACT` (file 0.15, 0.08 supplied by the launch line since 2026-08-06 but not yet flown); hover

Table B.3: Propulsion and actuation constants; the flight-identified provenance of k_T, k_Q and τ_{act} is Chapter 6.

Quantity	Value
Thrust coefficient k_T	$4.1547 \times 10^{-5}\,\text{N}\,\text{s}^2$
Torque ratio k_Q/k_T ; k_Q	0.024 m ; $9.97128 \times 10^{-7}\,\text{N}\,\text{m}\,\text{s}^2$ (derived)
Rotor speed floor / ceiling	120.0 / 779.8 $\text{rad}\,\text{s}^{-1}$
Motor lag (SMC invariants)	0.15 s
NMPC actuator lag τ_{act}	0.15 s in the file; 0.08 s supplied at launch, not yet flown
Transport round trip	0.030 s
Hover rotor speed ; collective	483.73 $\text{rad}\,\text{s}^{-1}$; 0.416 hardware map, 0.56 SITL map (derived)
Max thrust per rotor ; total ; T/W	25.264 N ; 101.06 N ; 2.599 (derived)
Servo model ; supply	Dynamixel XC430-T150BB-T (no. 1070) ; dedicated 12 V rail (11.9 V measured)
Stall torque ; resolution	1.6 N m at 12 V ; 0.088° per tick
Current sensor	none: the load register is estimated PWM duty
Operating mode ; position PID	extended position (mode 4) ; 640/0/6000
Profile velocity ; status poll	14 units = 19.24 $^\circ\,\text{s}^{-1}$; 2 Hz
Neutral ticks	$\{2023, 2036, 2037, 2037\}$
Runaway trip	35 $^\circ\,\text{s}^{-1}$, 18°, 0.10 s
Droop / static-load / transit-load trips	6.0° for 2.0 s ; 40 % for 3.0 s ; 85 % for 2.0 s
Temperature ; stale-read trips	65 °C ; 20 consecutive failed reads
Ground-gate freshness ; landing reshape guard	2.0 s ; 16.0 s with 8.0° tolerance

Table B.4: The thirteen-point hardware ESC map, $\omega \to$ normalised command.

ω [$\text{rad}\,\text{s}^{-1}$]	148.3	215.2	275.3	353.7	425.6	492.5	553.3
norm	0.060	0.123	0.182	0.264	0.345	0.427	0.509
ω [$\text{rad}\,\text{s}^{-1}$]	602.4	652.7	698.9	739.5	760.0	779.8	
norm	0.591	0.673	0.754	0.836	0.918	1.000	

thrust and speed are `T_HOVER` and `OMEGA_HOVER`; the sliding-mode gains are `OMEGA_N`, `ZETA`, `LAM` and `DELTA`, with the equivalent attitude stiffness printed as `ks` in the controller `CTRL` log line; the NMPC weights are `Q_POS`, `Q_VEL`, `Q_ATT`, `Q_RATE` and `RD_LEAD`, with horizon `N_HORIZON` = 30 and step `DT` = 0.01; and the disturbance-observer states are the first six per-solve parameters of the NMPC.

B.4 Known divergences between the code and what flies

Reading only the repository defaults gives, in a handful of places, values that are *not* the ones that flew or that are known to be stale. Table B.5 declares them; it is the honest complement to the tables above and the correct starting point for anyone attempting to reproduce the flight results.

Table B.5: Values for which the repository default and the operating state differ.

Quantity	In the repository	In flight	Note
τ_{act}	0.15 s	0.08 s	supplied by the launch line since 2026-08-06; flights 18 and 19 flew 0.15 s and the override has not yet been flown (Sec. 6.6.1)
`MPC_THR_HOVER` on the flight controller	last dump (2026-07-24): 0.262	expected 0.416	no later dump in the repository proves the write
`tools/pi` parameter files	flight and SIH parameter dumps	—	declared stale by the flight cards themselves
Simulation actuator lag	0.15/0.12 s (up/down)	model at 0.15 s in the final campaign, 0.08 s flown	in each venue the model matches its plant (Sec. 12.1.7)
Simulation inertia	$+5/+7/+12\,\%$ over CAD ($xx/yy/zz$)	CAD values	documented, conservative direction

Appendix C

Flight-test protocol

This appendix records the operational flight-test protocol referenced in Chapter 11, as it was actually used across the nineteen flights of the campaign. The general protocol was superseded in practice by one self-contained *card per flight* (twenty-one cards in the repository), described in Sec. C.3.

C.1 Entry gates and standard setup

The conditions closed before the first flight included the propulsion bench campaign (concluded on 2026-07-15, with two coefficients later superseded by flight evidence); rotor-speed telemetry assessment, which determined that bidirectional DShot is unreliable for this FC–ESC combination and was permanently disabled; servo bring-up on the complete vehicle; vehicle weighing; PX4 failsafe configuration with readback verification (offboard loss 0.5 s → land, RC loss → land, disarm on land 2.0 s, low battery → land, geofence 50 m horizontal / 30 m vertical); companion latency and jitter measurement; and the presence of a briefed safety pilot familiar with the override semantics described below. The campaign proceeded knowingly without measured rotor speed due to the telemetry limitations. One gate was deliberately waived, which is recorded as a safety fact: the planned "flight 0", intended to validate the native PX4 controller on the bare airframe, was skipped by explicit decision. The accepted consequence of this waiver was that the RC-takeover path had never been tested before it became necessary during actual operations.

The standard operating setup required an open field with a free radius of at least 30 m around the hover point and no personnel under the trajectory. Wind limits were set below $3\,\mathrm{m\,s^{-1}}$ for early test cards and $5\,\mathrm{m\,s^{-1}}$ thereafter. Battery resting voltage was required to be 4.15 V per cell for identification flights and 3.9 V otherwise. A fixed power-up sequence was mandated: pre-flight inspection, companion boot, DDS agent initialisation, topic verification, EKF ready confirmation, recording

initiation *before* arming, followed by stack launch; the node remained unarmed until a mission was explicitly commanded. Real-time pinning was applied per process *after* launch rather than by wrapping, to prevent starving the servo serial driver. The pre-flight checklist verified that propellers were torqued and handed correctly, arm play was within measured limits, servo cables were routed clear of rotor discs, the vehicle was balanced in X configuration, and the GNSS antenna was clear of power wiring. Additionally, the battery was checked for charge and security; the commit hash was noted; and the parameter fingerprint for the specific test card was executed on the ground with a successful result. This step printed file values, including the actuator-lag default, to ensure launch overrides were visible, accompanied by a fresh PX4 parameter readback. The kill switch was tested *that day* on the ground.

Data recorded per flight included the PX4 ulog, a session rosbag plus one bag per trial, the controller node log, a parameter dump, and written notes. Two mandatory rules were adopted following incidents involving data loss: every log copy must be verified after transfer, and the logger must write to disk rather than tmpfs. A specific warning for analysis is that per-trial bags start at the trajectory initiation; therefore, hover and settle windows are available only in the session bag.

C.2 Override and emergency semantics

On **RC takeover**, the node mutes all outputs, including motors, within approximately 200 ms and never attempts to reclaim control. The pilot assumes full authority, with PX4 operating via its native controller, which is reliable only in X configuration. In non-X shapes, takeover performance is *degraded* because the native controller assumes a centred vehicle geometry; however, above 5 m, this mode remains preferred to an immediate kill switch activation. The **kill switch** halts motors immediately under an unconditional latch, and the node does not re-arm autonomously; **re-engagement** is possible only with the vehicle disarmed on the ground and a fresh mission command. The priority order for emergency response is: controllable anomaly → RC takeover and land; violent loss of control or trajectory toward personnel → kill (a fall from 2 m to 5 m is considered repairable); link loss → execution of configured PX4 failsafes with pilot supervision. The final autonomous safety net is the airframe-lost trip at 75° tilt, active only in flight, which mutes and disarms the vehicle; this mechanism serves as a backstop for the pilot rather than a replacement for the kill switch.

C.3 Test cards and abort criteria

From flight 3 onward, the operational unit was the per-flight card: a self-contained document specifying explicit entry gates under the cascade rule (the next card is flown only if the current one completes successfully), the on-ground parameter fingerprint, the declared envelope, a three-level quick stop procedure (commanded land; manual failover to SMC from NMPC stack; RC takeover or kill), abort criteria, and post-flight routines. Commands to mission topics were always published with `--times 5 --rate 2`, never using the `--once` flag. The executed cards and their outcomes are archived with the flight data.

The abort criteria in force are: *position* — XY excursion beyond 1 m sustained for more than 2 s in hover, or altitude error beyond 0.5 m; *dynamics* — any attitude oscillation whose amplitude *grows*, doubling within a few cycles (flight 19 demonstrated both that this is the right criterion and that no automatic threshold was watching it); an explicit *non-criterion* — do not abort for 3°–4° of steady tilt in wind with a firm hold; and, on a *guard failover in flight*, land, download the logs and stop the NMPC flying for the day. Automatic gates implemented in every card from flight 12 onward include: take-off refusal if any latched arm fault is present; an overload-risk trip that disables the optimiser, blocks morphing, and initiates a normal landing with a reshape to X configuration, whereas any other arm fault triggers an immediate emergency landing in the current shape; the on-ground runaway gate; and, following flight 19, a hard attitude gate prior to streaming (requiring roll peak-to-peak deviation below 4° over 2 s) along with a growing-oscillation detector. The measured failover times are part of the safety contract: guard adoption in 0.206 s on a frozen controller and 0.512 s on persistent QP failure, manual lever in 0.251 s (Fig. 9.3), with three closed-loop failovers all ending in clean landings.

C.4 Standard post-flight metrics

The following metrics are computed for every flight by the committed analysis scripts and are the operational definition of the numbers reported in Chapter 12: hover tilt at the 95th and 99th percentiles; tracking RMSE (planar for the figure-eights, three-dimensional and datum-aware for the helices); XY and altitude hold error; motor-command spread in hover; hover collective and the measured-to-modelled thrust ratio against operating point; servo duty p_{50}/p_{95} per window; pack power per configuration; energy and charge consumed; roll spectral peak frequency and band power — computed with a linear detrend *before* the high-pass filter and over windows that never straddle a reconfiguration, otherwise the 45° arm ramp reads as several degrees of nonexistent jitter; and, for the NMPC stack, solve time

$p_{50}/p_{95}/p_{99}/$ max, consecutive QP failures and the solve-success fraction. Three data traps were encountered and are handled in the scripts: the run summary file keeps only the *last* session when an output directory is reused; a ulog whose first sample carries a zero timestamp makes unsigned time subtraction wrap; and every offline re-analysis of real flight data must export `MORPHING_REAL_HW=1` (Sec. B.3).

Bibliography

[1] Davide Falanga, Kevin Kleber, Stefano Mintchev, Dario Floreano, and Davide Scaramuzza. «The Foldable Drone: A Morphing Quadrotor That Can Squeeze and Fly». In: *IEEE Robotics and Automation Letters* 4.2 (2019), pp. 209–216. DOI: 10.1109/LRA.2018.2885575 (cit. on pp. 1, 10–13, 15–19, 51, 52, 84).

[2] Guiyang Cui, Ruihao Xia, Xin Jin, and Yang Tang. *Motion Planning and Control of A Morphing Quadrotor in Restricted Scenarios*. 2023. arXiv: 2312.07075 [cs.RO] (cit. on pp. 1, 10, 12, 13, 15–17, 19, 52, 84).

[3] Stefano Mintchev, Ludovic Daler, Gregoire L'Eplattenier, Laurent Saint-Raymond, and Dario Floreano. «Foldable and Self-Deployable Pocket Sized Quadrotor». In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. Seattle, WA, USA, 2015, pp. 2190–2195. DOI: 10.1109/ICRA.2015.7139489 (cit. on pp. 1, 10, 12).

[4] Zhiping Liu, Guoshao Chen, and Shuping Xu. «Modeling and Control of Reconfigurable Quadrotors Based on Model Reference Adaptive Control». In: *Aerospace* 11.8 (2024), p. 687. DOI: 10.3390/aerospace11080687 (cit. on pp. 1, 11–14, 16, 18, 19).

[5] Turcan Tuna, Salih Ertug Ovur, Etka Gokbel, and Tufan Kumbasar. «Design and development of FOLLY: A self-foldable and self-deployable quadcopter». In: *Aerospace Science and Technology* 100 (2020), p. 105807. DOI: 10.1016/j.ast.2020.105807 (cit. on pp. 10, 12).

[6] Aleligne Yohannes Shiferaw, Balasubramanian Esakki, Tamilarasan Pari, Elangovan Elumalai, Saleh Mobayen, and Andrzej Bartoszewicz. «Design and Implementation of Morphed Multi-Rotor Vehicles with Real-Time Obstacle Detection and Sensing System». In: *Sensors* 21.18 (2021), p. 6192. DOI: 10.3390/s21186192 (cit. on pp. 10, 12, 14).

[7] Nathan Bucki and Mark W. Mueller. «Design and Control of a Passively Morphing Quadcopter». In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 9116–9122. DOI: 10.1109/icra.2019.8794373 (cit. on p. 10).

[8] Valentin Riviere, Augustin Manecy, and Stéphane Viollet. «Agile Robotic Fliers: A Morphing-Based Approach». In: *Soft Robotics* 5.5 (2018), pp. 541–553. DOI: 10.1089/soro.2017.0120 (cit. on p. 10).

[9] Harsh Modi, Hao Su, Xiao Liang, and Minghui Zheng. *MorphoCopter: Design, Modeling, and Control of a New Transformable Quad-Bi Copter*. 2025. arXiv: 2506.07204 [cs.RO] (cit. on pp. 11, 12, 14, 15, 17–19).

[10] Marco Ruggia. *MOMAV: A highly symmetrical fully-actuated multirotor drone using optimizing control allocation*. 2025. arXiv: 2506.08868 [cs.RO] (cit. on pp. 11–13, 15–17).

[11] Ruben Raphael D'Sa. «Design of a Transformable Unmanned Aerial Vehicle». Advisor: N. Papanikolopoulos. PhD thesis. University of Minnesota, Department of Mechanical Engineering, 2020 (cit. on pp. 11, 12).

[12] Stamatina C. Barakou, Costas S. Tzafestas, and Kimon P. Valavanis. «A Review of Real-Time Implementable Cooperative Aerial Manipulation Systems». In: *Drones* 8.5 (2024). Also available as arXiv:2403.14347, p. 196. DOI: 10.3390/drones8050196 (cit. on pp. 11, 14, 15, 17).

[13] Michail G. Michailidis, Kimon P. Valavanis, and Matthew J. Rutherford. *Nonlinear Control of Fixed-Wing UAVs with Time-Varying and Unstructured Uncertainties*. Vol. 1. Springer Tracts in Autonomous Systems. Springer, 2020. DOI: 10.1007/978-3-030-40716-2 (cit. on pp. 11, 14, 15, 18).

[14] George K. Fourlas and George C. Karras. «A Survey on Fault Diagnosis and Fault-Tolerant Control Methods for Unmanned Aerial Vehicles». In: *Machines* 9.9 (2021), p. 197. DOI: 10.3390/machines9090197 (cit. on pp. 13, 16).

[15] Remus C. Avram, Xiaodong Zhang, and Jonathan Muse. «Quadrotor Actuator Fault Diagnosis and Accommodation Using Nonlinear Adaptive Estimators». In: *IEEE Transactions on Control Systems Technology* 25.6 (2017), pp. 2219–2226. DOI: 10.1109/TCST.2016.2640941 (cit. on pp. 13, 16).

[16] Boris Houska, Hans Joachim Ferreau, and Moritz Diehl. «An Auto-Generated Real-Time Iteration Algorithm for Nonlinear MPC in the Microsecond Range». In: *Automatica* 47.10 (2011), pp. 2279–2285. DOI: 10.1016/j.automatica.2011.08.020 (cit. on pp. 15, 152).

[17] Robin Verschueren et al. «acados—A Modular Open-Source Framework for Fast Embedded Optimal Control». In: *Mathematical Programming Computation* 14.1 (2022), pp. 147–183. DOI: 10.1007/s12532-021-00208-8 (cit. on pp. 15, 105, 152).

[18] Davide Bicego, Jacopo Mazzetto, Ruggero Carli, Marcello Farina, and Antonio Franchi. «Nonlinear Model Predictive Control with Enhanced Actuator Model for Multi-Rotor Aerial Vehicles with Generic Designs». In: *Journal of Intelligent & Robotic Systems* 100.3–4 (2020), pp. 1213–1247. DOI: 10.1007/s10846-020-01250-9 (cit. on p. 15).

[19] Tiago Nascimento and Martin Saska. «Embedded Fast Nonlinear Model Predictive Control for Micro Aerial Vehicles». In: *Journal of Intelligent & Robotic Systems* 103.4 (2021), p. 74. DOI: 10.1007/s10846-021-01522-y (cit. on p. 15).

[20] Youmin Zhang and Jin Jiang. «Bibliographical Review on Reconfigurable Fault-Tolerant Control Systems». In: *Annual Reviews in Control* 32.2 (2008), pp. 229–252. DOI: 10.1016/j.arcontrol.2008.03.008 (cit. on p. 16).

[21] Mark W. Mueller and Raffaello D'Andrea. «Stability and Control of a Quadrocopter Despite the Complete Loss of One, Two, or Three Propellers». In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. 2014, pp. 45–52. DOI: 10.1109/ICRA.2014.6906588 (cit. on p. 16).

[22] Fang Nan, Sihao Sun, Philipp Foehn, and Davide Scaramuzza. «Nonlinear MPC for Quadrotor Fault-Tolerant Control». In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 5047–5054. DOI: 10.1109/LRA.2022.3154033 (cit. on p. 17).

[23] Sihao Sun, Giovanni Cioffi, Coen de Visser, and Davide Scaramuzza. «Autonomous Quadrotor Flight Despite Rotor Failure With Onboard Vision Sensors: Frames vs. Events». In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 580–587. DOI: 10.1109/lra.2020.3048875 (cit. on p. 17).

[24] Serhat Sönmez. «Model-Based Navigation and Control of Multirotor UAVs: A Machine Learning Approach». PhD thesis. University of Denver, Daniel Felix Ritchie School of Engineering and Computer Science, 2024 (cit. on p. 18).

[25] PX4 Autopilot Development Team. *DShot ESCs — PX4 Guide*. URL: https://docs.px4.io/main/en/peripherals/dshot.html (visited on 07/12/2026) (cit. on pp. 33, 36).

[26] PX4 Autopilot Development Team. *uXRCE-DDS (PX4-ROS 2/DDS Bridge) — PX4 Guide*. URL: https://docs.px4.io/main/en/middleware/uxrce_dds.html (visited on 07/01/2026) (cit. on pp. 36, 118, 171).

[27] J. Gordon Leishman. *Principles of Helicopter Aerodynamics*. 2nd ed. Cambridge Aerospace Series. Cambridge, UK: Cambridge University Press, 2006 (cit. on pp. 38, 87).

[28] John Brandt and Michael Selig. «Propeller Performance Data at Low Reynolds Numbers». In: *49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*. American Institute of Aeronautics and Astronautics, 2011. DOI: 10.2514/6.2011-1255 (cit. on p. 39).

[29] Robert Mahony, Vijay Kumar, and Peter Corke. «Multirotor Aerial Vehicles: Modeling, Estimation, and Control of Quadrotor». In: *IEEE Robotics & Automation Magazine* 19.3 (2012), pp. 20–32. DOI: 10.1109/mra.2012.2206474 (cit. on p. 48).

[30] Gabriel Hoffmann, Haomiao Huang, Steven Waslander, and Claire Tomlin. «Quadrotor Helicopter Flight Dynamics and Control: Theory and Experiment». In: *AIAA Guidance, Navigation and Control Conference and Exhibit*. American Institute of Aeronautics and Astronautics, 2007. DOI: 10.2514/6.2007-6461 (cit. on p. 48).

[31] PX4 Autopilot Development Team. *Modules Reference: Estimator (mc_hover_thrust_estimator) — PX4 Guide*. URL: https://docs.px4.io/main/en/modules/modules_estimator.html (visited on 07/01/2026) (cit. on p. 63).

[32] Vadim I. Utkin. «Variable Structure Systems with Sliding Modes». In: *IEEE Transactions on Automatic Control* 22.2 (1977), pp. 212–222. DOI: 10.1109/TAC.1977.1101446 (cit. on pp. 71, 152).

[33] Jean-Jacques E. Slotine and Weiping Li. *Applied Nonlinear Control*. Englewood Cliffs, NJ: Prentice Hall, 1991 (cit. on pp. 71, 82, 152).

[34] Rong Xu and Umit Ozguner. «Sliding Mode Control of a Quadrotor Helicopter». In: *Proceedings of the 45th IEEE Conference on Decision and Control*. IEEE, 2006, pp. 4957–4962. DOI: 10.1109/cdc.2006.377588 (cit. on p. 71).

[35] PX4 Autopilot Development Team. *Controller Diagrams — PX4 Guide*. URL: https://docs.px4.io/main/en/flight_stack/controller_diagrams.html (visited on 07/01/2026) (cit. on p. 75).

[36] Gabriele Pannocchia and James B. Rawlings. «Disturbance Models for Offset-Free Model-Predictive Control». In: *AIChE Journal* 49.2 (2003), pp. 426–437. DOI: 10.1002/aic.690490213 (cit. on p. 102).

[37] Urban Maeder and Manfred Morari. «Offset-free reference tracking with model predictive control». In: *Automatica* 46.9 (2010), pp. 1469–1476. DOI: 10.1016/j.automatica.2010.05.023 (cit. on p. 102).

[38] Wen-Hua Chen, D.J. Ballance, P.J. Gawthrop, and J. O'Reilly. «A nonlinear disturbance observer for robotic manipulators». In: *IEEE Transactions on Industrial Electronics* 47.4 (2000), pp. 932–938. DOI: 10.1109/41.857974 (cit. on p. 106).

[39] Teodor Tomic, Christian Ott, and Sami Haddadin. «External Wrench Estimation, Collision Detection, and Reflex Reaction for Flying Robots». In: *IEEE Transactions on Robotics* 33.6 (2017), pp. 1467–1482. DOI: 10.1109/tro.2017.2750703 (cit. on p. 106).

[40] Moritz Diehl, Hans Georg Bock, and Johannes P. Schlöder. «A Real-Time Iteration Scheme for Nonlinear Optimization in Optimal Feedback Control». In: *SIAM Journal on Control and Optimization* 43.5 (2005), pp. 1714–1736. DOI: 10.1137/s0363012902400713 (cit. on p. 108).

[41] G. Grimm, M.J. Messina, S.E. Tuna, and A.R. Teel. «Nominally Robust Model Predictive Control With State Constraints». In: *IEEE Transactions on Automatic Control* 52.10 (2007), pp. 1856–1870. DOI: 10.1109/tac.2007.906187 (cit. on p. 108).

[42] Mina Kamel, Thomas Stastny, Kostas Alexis, and Roland Siegwart. «Model Predictive Control for Trajectory Tracking of Unmanned Aerial Vehicles Using Robot Operating System». In: *Studies in Computational Intelligence*. Springer International Publishing, 2017, pp. 3–39. DOI: 10.1007/978-3-319-54927-9_1 (cit. on p. 115).

[43] Davide Falanga, Philipp Foehn, Peng Lu, and Davide Scaramuzza. «PAMPC: Perception-Aware Model Predictive Control for Quadrotors». In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1–8. DOI: 10.1109/iros.2018.8593739 (cit. on p. 115).

[44] Guillem Torrente, Elia Kaufmann, Philipp Fohn, and Davide Scaramuzza. «Data-Driven MPC for Quadrotors». In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 3769–3776. DOI: 10.1109/lra.2021.3061307 (cit. on p. 115).

[45] PX4 Autopilot Development Team. *Gazebo Simulation — PX4 Guide*. URL: https://docs.px4.io/main/en/sim_gazebo_gz/ (visited on 07/01/2026) (cit. on p. 118).

[46] PX4 Autopilot Development Team. *Offboard Mode — PX4 Guide*. URL: https://docs.px4.io/main/en/flight_modes/offboard.html (visited on 07/01/2026) (cit. on pp. 125, 171).

[47] Daniel Mellinger and Vijay Kumar. «Minimum snap trajectory generation and control for quadrotors». In: *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 2520–2525. DOI: 10.1109/icra.2011.5980409 (cit. on p. 130).

[48] Boyu Zhou, Fei Gao, Luqi Wang, Chuhao Liu, and Shaojie Shen. «Robust and Efficient Quadrotor Trajectory Generation for Fast Autonomous Flight». In: *IEEE Robotics and Automation Letters* 4.4 (2019), pp. 3529–3536. DOI: 10.1109/lra.2019.2927938 (cit. on p. 130).

[49] Sihao Sun, Angel Romero, Philipp Foehn, Elia Kaufmann, and Davide Scaramuzza. «A Comparative Study of Nonlinear MPC and Differential-Flatness-Based Control for Quadrotor Agile Flight». In: *IEEE Transactions on Robotics* 38.6 (2022), pp. 3357–3373. DOI: 10.1109/TRO.2022.3177279 (cit. on p. 152).

[50] David Q. Mayne, James B. Rawlings, Christopher V. Rao, and Pierre O. M. Scokaert. «Constrained Model Predictive Control: Stability and Optimality». In: *Automatica* 36.6 (2000), pp. 789–814. DOI: 10.1016/S0005-1098(99)00214-9 (cit. on p. 152).

[51] PX4 Autopilot Development Team. *Parameter Reference (IMU_GYRO_RATEMAX) — PX4 Guide*. URL: https://docs.px4.io/main/en/advanced_config/parameter_reference.html (visited on 07/01/2026) (cit. on p. 171).

[52] Weibing Gao, Yufu Wang, and Abbas Homaifa. «Discrete-Time Variable Structure Control Systems». In: *IEEE Transactions on Industrial Electronics* 42.2 (1995), pp. 117–122. DOI: 10.1109/41.370376 (cit. on p. 171).

[53] Arie Levant. «Sliding Order and Sliding Accuracy in Sliding Mode Control». In: *International Journal of Control* 58.6 (1993), pp. 1247–1263. DOI: 10.1080/00207179308923053 (cit. on p. 171).

[54] PX4 Autopilot Development Team. *PX4 Architectural Overview — PX4 Guide*. URL: https://docs.px4.io/main/en/concept/architecture.html (visited on 07/01/2026) (cit. on p. 171).