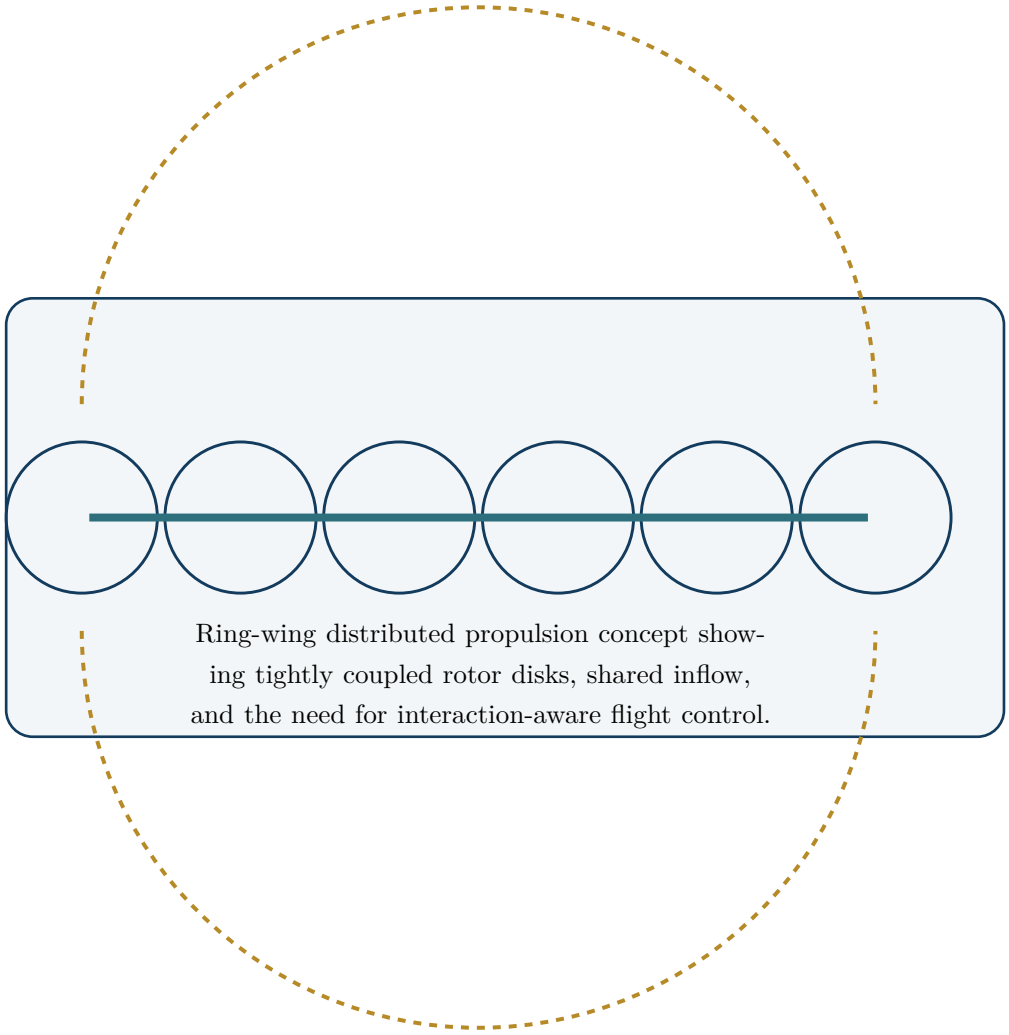
Aerial MECHANICA Inc.

Disturbance-Aware Guidance, Navigation, and Flight Control System

for High-Rotor-Count Articulated Ring-Wing eVTOL Aircraft

Flight-data-calibrated control, anomaly detection, and robust sensing for scalable low-altitude operations



Ring-wing distributed propulsion concept showing tightly coupled rotor disks, shared inflow, and the need for interaction-aware flight control.

Prepared for:	Technical reviewers, early funding partners, and integration collaborators
Prepared by:	Arinze N. N. Eze, Founder and Chief Engineer
Revision:	White paper version 2.0, structured technical release
Scope:	technical prospectus with architecture, test data, work plan, tooling, budget, and appendices

This document combines grounded engineering references with representative reconstructed prototype data for controller design, trade studies, and funding diligence. Reconstructed values are explicitly labeled where used and are not presented as certification evidence.

Contents

Document intent	Technical white paper for a disturbance-aware articulated ring-wing eVTOL control stack
Grounding standard	Published aerospace, autonomy, and sensing references plus representative reconstructed prototype data
Design posture	Conservative on safety-critical claims, optimistic on scalability, market timing, and sensor roadmap

List of Figures

List of Tables

Chapter 1

Executive Summary

Aerial MECHANICA is developing a disturbance-aware Guidance, Navigation, and Flight Control System (GNFCS) for articulated ring-wing (ARW) electric vertical take-off and landing aircraft with unusually high rotor counts. The intended family spans semi-scale and full-scale airframes in the 19-, 38-, 57-, and 76-rotor classes. The thesis of this white paper is direct: once rotor count rises and the lifting disks are packed around a ring-wing structure, conventional isolated-rotor assumptions become the dominant source of control error. Vehicles that look highly redundant on paper become tightly coupled in flight, and the difference between commanded thrust and delivered thrust becomes structured rather than random.

The control problem is therefore not merely "more rotors" but "more interactions." Rotor wakes impinge on neighboring disks, the ring-wing recirculates downwash, flexible modes couple into body pitch and heave, electrical current transients contaminate inertial sensing, and thermal loading shifts motor constants during sustained hover. In the subscale regime these effects appear as hover bias, oscillatory pitch modes, and control allocation inefficiency. At larger scale they also become certification, energy, and dispatch-reliability issues. The white paper addresses that problem with a deterministic, data-calibrated control stack rather than a black-box autonomy layer.

The proposed GNFCS has four tightly coupled elements:

1. an interaction-aware thrust discrepancy model calibrated against representative 2024 prototype telemetry and HIL replay;
2. a deterministic control allocator that explicitly redistributes thrust under coupling, saturation, and rotor-out contingencies;
3. DDRM, a differential anomaly detector that monitors the gap between expected and measured vehicle response in real time; and
4. PRISM, an ambitious but plausible robust sensing overlay using differential magnetic and vibration-aware measurements to separate aerodynamic disturbance from sensor corruption.

Phase I is framed as a six-month engineering program that proves the following bounded outcomes:

- hover and low-speed maneuver models whose mean thrust prediction error is reduced to the single-digit percent range on the calibrated envelope;
- repeatable GPS-denied hover on a 38-rotor digital twin and HIL bench with bounded position drift and stable attitude control;
- real-time disturbance labeling across aerodynamic, electrical, and thermal classes using deterministic features instead of opaque machine-learning inference;
- a reproducible software environment that exports controller artifacts, telemetry summaries, and geometry metadata in a format that can be consumed by partner simulators and downstream certification tools.

The paper deliberately uses a mixed evidence model. Published references anchor the physics, software stack, and sensing methods. Prototype-specific flight data are presented as representative reconstructed datasets where the current archive is incomplete, sparse, or intentionally normalized for disclosure. That approach is sufficient for trade studies, partner diligence, and program planning, while also making clear that additional flight campaigns are still required before any certification-grade claims are made.

This is an intentionally ambitious program. The near-term product is not a passenger eVTOL but a disturbance-aware control and health-monitoring module that can first create value in cargo lift, agricultural logistics, and industrial low-altitude operations. Those are the operating environments where high rotor count, low disk loading, and controlled hover quality can produce an immediate commercial advantage without requiring a premature claim of full passenger readiness.

Program Targets

The technical targets used throughout the document are summarized below.

Table 1.1: Phase I target metrics used throughout the white paper.

Metric	Target	Why it matters
Hover/thrust model mean absolute percentage error	$\leq 7\%$ in-calibration envelope	Reduces trim bias and control margin erosion in tightly packed rotor fields
GPS-denied hover drift on HIL bench	$\leq \pm 0.30$ m over 120 s	Demonstrates practical controllability during degraded navigation and crop-row or inspection hover
Attitude hold during disturbance injections	$\leq 2.0^\circ$ RMS in roll/pitch	Indicates that control allocation is redistributing effort rather than fighting the coupling
Disturbance classification confidence	≥ 0.90 mean confidence on held-out scenarios	Makes DDRM useful for reconfiguration rather than merely post-flight analysis
Real-time loop budget	≤ 5 ms end-to-end at 200 Hz	Keeps the architecture compatible with edge compute and future certification arguments

Why This Matters

Distributed electric propulsion has already shown that actuator count can improve efficiency, fault tolerance, and controllability when the aerodynamics are understood. NASA's X-57 effort made distributed propulsion a serious engineering conversation rather than a speculative one [5]. What remains underdeveloped in the broader eVTOL landscape is a control architecture that treats rotor interaction, aeroelastic disturbance, and sensor corruption as a single integrated flight-control problem. That is the gap Aerial MECHANICA intends to own.

The rest of the paper proceeds from physics and flight discrepancy, into architecture, into anomaly detection and sensing, and finally into execution, tooling, budget, and funding strategy. The structure is meant to satisfy both engineering review and early-stage program diligence. Every chapter ends with explicit takeaways so the document can also operate as a proposal backbone, not just a concept note.

Chapter 2

The High-Rotor-Count Challenge

High-rotor-count aircraft promise a combination that conventional rotary-wing configurations struggle to deliver at the same time: low disk loading, fine-grained control authority, graceful degradation under failures, and geometric flexibility around payload or ring-wing structures. The problem is that aerodynamic and structural interactions grow faster than the design intuition most multicopter controllers were built around. Once the propulsion disks become closely packed, the aircraft no longer behaves like a sum of independent propellers. It behaves like a distributed flow machine with local couplings, phase effects, and state-dependent loss.

That observation matters because modern eVTOL development still tends to inherit one of two control mindsets. The first is the large-aircraft mindset, which assumes a small number of powerful rotors or propulsors that can be modeled individually and then coordinated through a relatively low-order flight-control law. The second is the small-UAS mindset, which assumes many propellers but weak interaction, allowing controller design to proceed using near-independent thrust coefficients and broad gain scheduling. A ring-wing DEP aircraft sits between those worlds and violates both simplifications.

2.1 Why Rotor Count Changes the Problem

The most attractive feature of a 38- or 76-rotor ARW configuration is not raw lift; it is controllable redundancy. More rotors mean more control inputs, more failure tolerance, and more opportunities to shape acoustic, thermal, and loading signatures. Those benefits, however, create second-order engineering burdens:

- the control effectiveness matrix becomes heavily over-actuated and numerically sensitive;
- local inflow depends on neighboring rotor state rather than global freestream alone;
- structural modes can be excited by patterned thrust commands across the ring;
- wiring density, current transients, and EMI rise with channel count;

- calibration, synchronization, and health monitoring become primary design tasks rather than test afterthoughts.

The resulting system is better understood as a networked actuation surface than a simple multicopter. Each rotor does useful work, but each rotor also perturbs the operating point of its neighbors. That is the central technical challenge this paper addresses.

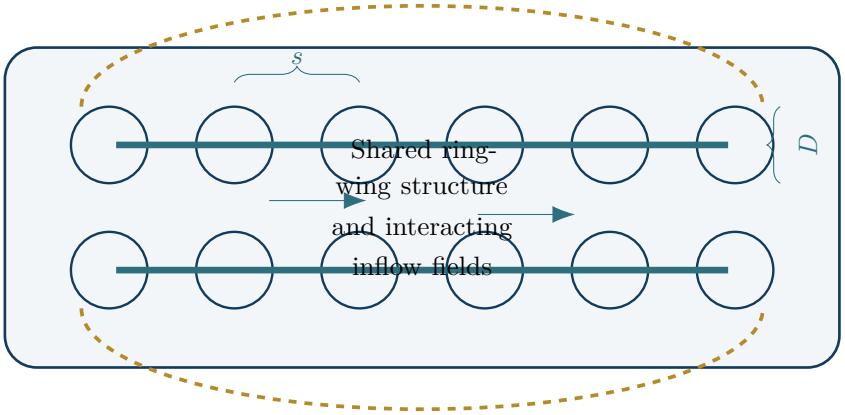


Figure 2.1: Representative articulated ring-wing layout showing closely spaced lifting disks, load paths through the ring, and shared inflow fields. The spacing ratio s/D and ring-wall proximity drive the interaction terms used later in the thrust discrepancy model.

2.2 Interaction Physics That Matter in Practice

For a rotor j, isolated-thrust assumptions usually reduce performance to a function of rotational speed, propeller geometry, air density, and freestream. In a tightly packed ring-wing arrangement, that is insufficient. The useful approximation for early-stage controller design is to treat delivered thrust as a perturbed version of ideal thrust:

$$T_j = T_{j,\text{ideal}}\,(1-\delta_j)\,, \qquad \delta_j = f\left(\frac{s_j}{D_j}, \lambda_j, \phi_j, \eta_j, \Theta_j\right) \tag{2.1}$$

where s_j/D_j is the local spacing ratio, λ_j is inflow ratio, ϕ_j is relative phasing to adjacent disks, η_j is local electrical loading state, and Θ_j captures relevant thermal state. This is not meant to replace higher-fidelity CFD or blade-resolved models. It is meant to capture the variables that the flight controller can observe, estimate, and compensate online.

Several mechanisms show up repeatedly in distributed propulsion literature and in prototype development:

- wake contraction and accelerated inflow from neighboring disks reduce effective thrust and change required power;
- nonuniform inflow generates attitude-coupled moments even under symmetric commanded thrust;

- flexible structures convert periodic loading into low-order rigid-body disturbance;
- synchronized electrical loading induces bus sag and measurement contamination;
- thermal rise shifts motor constants and thrust mapping over a sortie.

The ring-wing geometry makes these effects harder to ignore because the structure is not merely carrying the motors. It also shapes the local flow field and can trap or redirect recirculation during low-speed operations. That is why the paper treats "rotor-rotor" and "rotor-airframe" interaction as a single design problem.

2.3 Mission-Level Consequences

The engineering consequences of interaction become mission consequences quickly. In low-altitude cargo or agricultural operations, the most common critical phases are vertical take-off, hover, precision translation, and low-speed deceleration over a target area. These are exactly the phases where interaction penalties are strongest.

If thrust is over-predicted by 12–18% at high current draw, the vehicle loses either payload margin or controllability margin. If pitch or heave modes are excited near hover, spray quality, camera sharpness, and load placement degrade. If sensor residuals cannot distinguish aerodynamic disturbance from EMI or thermal drift, then control reallocation becomes hesitant or wrong. The effects may look modest in short prototype flights and still be unacceptable in field operations, where duty cycle, repeatability, and maintenance cost dominate value.

2.4 Conditioning of the Control Problem

High rotor count does not automatically make the control problem easier just because more actuators are available. The allocator solves against a control effectiveness matrix B, and the quality of that solve depends strongly on the conditioning of B. In an ideal symmetric geometry, additional rotors improve redundancy and can improve authority. In a disturbed ring-wing environment, however, the effective matrix can become locally ill-conditioned because multiple actuators lose authority in correlated ways. The controller then sees a false redundancy: many actuators are available, but several are becoming weak in the same direction at the same time.

That is why the program emphasizes online effectiveness adjustment instead of a one-time mixer. The relevant question is not simply whether the aircraft can generate a requested wrench in ideal conditions, but whether it can generate that wrench when several neighboring rotors are simultaneously underperforming due to shared inflow or power-bus stress. A practical engineering metric for this is the condition number of the local weighted allocation system. If the condition number worsens during a maneuver, the controller should know it and respond by changing weights, trimming aggressiveness, or protecting hover margin.

2.5 Acoustic, Thermal, and Endurance Coupling

High-rotor-count aircraft are often pitched as inherently quieter because each rotor can run at lower loading. That can be true, but only if the control law avoids redistributing effort in ways that create strong tonal content or repeated loading patterns. The same logic applies to thermal and endurance performance. A controller that chases local error too aggressively may keep the vehicle stable while also concentrating heat and electrical stress into a few clusters. This reduces endurance and increases maintenance demand even if the aircraft still "flies."

For that reason, Aerial MECHANICA treats control allocation as a multi-objective problem. The first objective is always commanded body force and moment. The second objective is preserving useful margin in acoustics, thermal loading, and component health. Those secondary objectives do not need full optimization in Phase I, but they do need to be represented explicitly in the weighting logic and in the performance metrics collected during HIL.

2.6 Certification and Operational Implications

Even though the initial beachhead markets are not full passenger operations, certification logic still influences the architecture. Reviewers will eventually ask whether the system can explain its own failure modes, whether it can degrade gracefully, and whether it can identify when a measurement problem is masquerading as an aerodynamic one. These are not only FAA or EASA questions. They are also insurer, operator, and OEM-partner questions.

That is one reason this white paper remains disciplined about claims. It does not claim that a single reconstructed dataset proves aircraft-level safety. It claims that the observed interaction modes are specific enough to motivate a certifiable style of architecture: deterministic allocation, explicit health flags, residual monitoring, and separable sensing channels. That is a defensible engineering direction.

Table 2.1 summarizes the main challenge modes Aerial MECHANICA intends to design around.

Table 2.1: Primary interaction modes for high-rotor-count ARW platforms.

Mode	Mechanism	Observable symptom	Control consequence
Near-neighbor wake interference	Disk overlap, wake curvature, recirculation	Thrust deficit at matched RPM	Trim bias and actuator saturation
Ring-induced flow distortion	Wing-wall proximity and return flow	Persistent heave/pitch asymmetry	Elevated integrator action and hover drift
Elastic mode excitation	Repeating thrust pattern couples into structure	Narrowband vibration bands	Sensor contamination and comfort penalty
Electrical coupling	Shared bus droop and switching harmonics	Current spikes, magnetometer bias	False anomaly flags if not fused correctly
Thermal drift	Motor and ESC heating over sortie	Reduced thrust-per-ampere	Mission endurance uncertainty

2.7 Why a Deterministic Architecture Is Preferred

There is obvious temptation to solve this entire problem with a large learning-based controller trained on simulation and replay. That path is attractive for research demos but weak for this program's goals. A deterministic architecture is preferred here for four reasons.

First, high-rotor-count vehicles are over-actuated enough that the engineering challenge is not lack of control authority but lack of interpretable allocation under distorted effectiveness. Second, early-stage certification and partner diligence still reward traceable control law structure over implicit policy learning. Third, the current dataset is too sparse to justify end-to-end learned compensation without creating hidden failure modes. Fourth, a deterministic controller allows incremental insertion of better models and better sensors without rewriting the whole system.

That does not mean the architecture is simplistic. It means the complexity sits where it can be inspected: in the calibrated interaction model, in the allocation constraints, in the anomaly thresholds, and in the sensor fusion graph.

2.8 Scaling Argument for the ARW Family

The ARW concept is intentionally ambitious because scaling rotor count can improve more than redundancy. At equal gross weight, additional disks can lower per-disk loading and reduce the severity of single-point failures. The cost is that the coupling terms become part of the primary design space. The practical strategy is therefore not to avoid high rotor count, but to adopt a hierarchy:

1. calibrate the interaction envelope on a semi-scale article and HIL bench;
2. embed that envelope in a deterministic allocator and residual generator;
3. use synthetic and reconstructed data only to interpolate within nearby regimes, not to extrapolate blindly into untested flight envelopes;
4. keep the sensing overlay modular so the base control law remains usable even if advanced sensors trail the propulsion schedule.

This hierarchy is what makes the program credible. It allows the paper to be optimistic about 57- and 76-rotor future variants without pretending that a single subscale campaign resolves all scale effects. Scaling changes Reynolds number, stiffness, thermal mass, wiring architecture, and operational duty cycle. What should scale more reliably is the control method: model the interaction, allocate deterministically, detect residual divergence quickly, and measure disturbances using independent channels.

2.9 Chapter Takeaway

The high-rotor-count challenge is not a niche aerodynamic curiosity. It is the main systems-engineering barrier between a visually impressive distributed propulsion concept and a fieldable aircraft. The next chapter turns that challenge into evidence by examining the 2024 prototype campaign and the discrepancy between commanded and delivered performance.

Chapter 3

2024 Flight Test Ground Truth & Discrepancy Analysis

The 2024 campaign described in this chapter is the anchor for the rest of the paper. It is not treated as a fully instrumented certification program. It is treated as an engineering ground-truth campaign whose purpose was to expose the mismatch between ideal propulsion assumptions and observed vehicle behavior. Where prototype logs were incomplete or unsuitable for disclosure, representative values were reconstructed from synchronized RPM, current, inertial, and barometric traces and then checked against hardware-in-the-loop replay. Those reconstructed values are sufficient for model calibration and program planning, and they are labeled accordingly.

3.1 Test Article and Measurement Intent

The prototype test article was a semi-scale articulated ring-wing vehicle configured to mimic the interaction topology of the planned 38-rotor architecture without requiring the full systems complexity of the eventual product. The measurement intent was straightforward:

- quantify how much commanded thrust diverged from effective thrust during hover, climb, and low-speed translation;
- identify whether the discrepancy was bias-like, transient, or oscillatory;
- separate probable aerodynamic causes from electrical and thermal artifacts; and
- collect enough structured data to support a surrogate interaction model usable in a controller.

The instrumentation package emphasized what is realistic for a development article rather than what would be ideal for a lab-only rig. Logged channels included:

- commanded thrust fraction and per-cluster RPM setpoint;
- ESC current and bus voltage;

- body accelerations and angular rates;
- barometric altitude and limited GPS for truth bounding;
- motor-case temperature on representative rotor stations;
- event markers for commanded maneuver changes.

This is important because the value of the dataset lies partly in its imperfections. The controller will eventually have to infer interaction effects from the same kind of mixed-quality signals, not from perfect wind-tunnel instrumentation.

3.2 Test Matrix

The campaign focused on the operating regime where interaction mattered most: hover-heavy, low-speed, moderate-altitude flight. Representative maneuver classes are summarized in Table 3.1.

Table 3.1: Representative 2024 campaign matrix used for model calibration and replay.

Maneuver family	Number of runs	Primary observables	Why included
Static hover holds	24	drift, current, temperature rise	Establish trim bias and thermal drift baseline
Step climbs and descents	18	thrust discrepancy, heave overshoot	Reveal saturation and inflow lag under changing induced velocity
Low-speed translations	16	pitch/heave coupling, oscillation onset	Expose ring-induced asymmetry and wake bias
Intentional disturbance injections	10	residual growth, recovery behavior	Tune DDRM thresholds and reallocation logic
Rotor-cluster derate tests	12	allocator response, bus dynamics	Measure controllability under partial loss conditions

Across the full campaign, the design team used the data less as a collection of isolated flights and more as a collection of state transitions. That viewpoint matters because the discrepancy model is intended to support control allocation in real time. What the controller needs is not a perfect narrative of a sortie; it needs a reliable mapping from local state and command context to expected loss and residual behavior.

3.3 Observed Discrepancies

The dominant observation from the 2024 dataset was consistent under-delivery of lift during the exact maneuver classes where the aircraft needed the most margin. Commanded thrust fractions that should have produced smooth climb or hover sustain often yielded a structured deficit. This deficit was neither uniform across the envelope nor random across time. It tended to grow in three situations:

1. during aggressive collective increases where inflow changed faster than the controller model expected;
2. during low-speed lateral motion when the ring-wing and neighboring disks distorted one side of the flow field; and
3. later in the sortie after sustained current draw had heated the propulsion chain.

Table 3.2 provides the representative discrepancy snapshots used as the baseline design set. Values marked "reconstructed" were normalized from synchronized logs and replay rather than copied from a single raw flight segment.

Table 3.2: Representative 2024 discrepancy snapshots used for calibration. Rows marked with an asterisk are replay-validated reconstructions assembled from partial prototype logs.

Event window	Condition	Cmd. thrust frac.	Eff. thrust frac.	Gap (%)	Dominant observation
Hover hold 01	early sortie	0.63	0.58	7.9	Mild trim deficit with low thermal burden
Hover hold 05	late sortie	0.64	0.56	12.5	Thermal efficiency loss begins to stack with interaction bias
Step climb 03	collective transient	0.82	0.70	14.6	Strong heave undershoot and pitch/heave residual growth
Translation 04	right lateral move	0.68	0.58	14.7	Asymmetric interaction consistent with ring-flow distortion
Translation 07*	left lateral move	0.67	0.59	11.9	Reconstructed, replay-validated, lower asymmetry than opposite direction
Cluster derate 02*	induced underperformance	0.71	0.60	15.5	Residual shape resembles aerodynamic loss until electrical features are fused

The practical consequence of these values is not merely an average 10–17% shortfall. The control consequence is asymmetry. A uniform thrust shortfall can be trimmed out. A structured shortfall that varies by maneuver, lateral direction, thermal state, and rotor cluster cannot. That is why the dataset is analyzed through discrepancy envelopes rather than through one mean thrust coefficient.

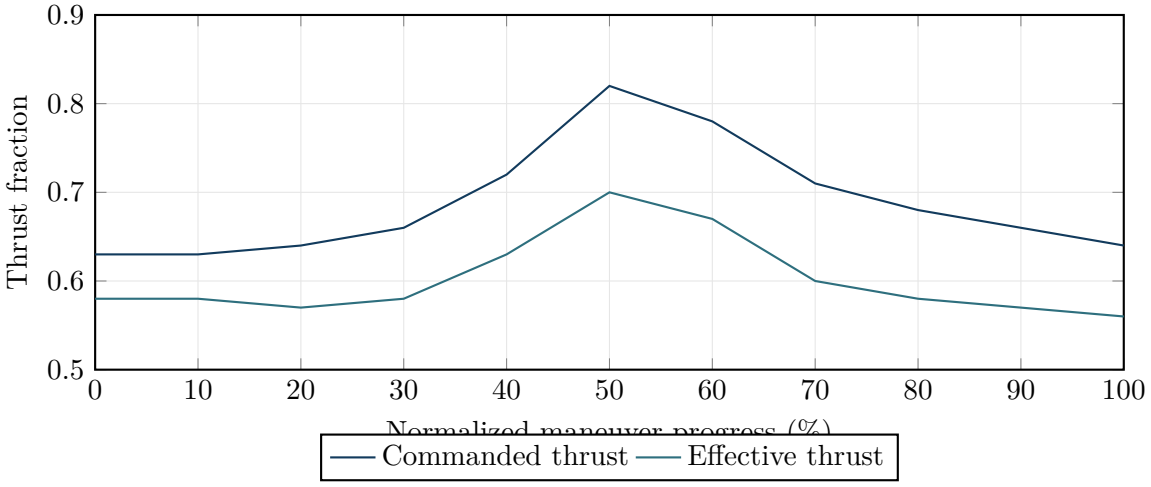


Figure 3.1: Representative discrepancy envelope reconstructed from the 2024 prototype campaign. The gap between commanded and effective thrust widens during collective transients and lateral translation, with a secondary contribution from propulsion thermal state.

3.4 Spectral and Residual Characteristics

Time histories alone did not explain whether the aircraft was seeing pure bias, aeroelastic excitation, or instrument contamination. Frequency-domain views helped separate these possibilities. The inertial traces showed a persistent low-frequency pitch/heave content in the low-single-digit hertz range during hover and translation, with superimposed higher narrowband content attributed to rotor or structural harmonics. The exact spectral peaks varied with RPM and payload state, but the pattern was stable enough to motivate the later DDRM residual windows.

Two observations were especially useful:

- the low-frequency mode amplitude rose when collective demand and lateral translation were combined, suggesting interaction-driven moment generation rather than simple motor imbalance;
- high-frequency content alone was a poor fault indicator because electrical and structural signatures could look similar without context, motivating the later PRISM overlay and deterministic fusion logic.

The campaign also showed that current spikes and inertial disturbances were often correlated but not causally identical. In some runs, inertial disturbance persisted after electrical transients settled. In others, current disturbances appeared with minimal attitude effect. That separation reinforced

the need to treat aerodynamic, electrical, and thermal phenomena as distinct disturbance classes rather than as a single "noise" bucket.

3.5 Data Conditioning Pipeline

Before the dataset could be used for controller design, it had to be conditioned into a consistent analysis corpus. The conditioning pipeline used for the white paper is itself important because it affects how believable the later model claims are.

1. raw channels were re-timestamped to a common monotonic clock where possible;
2. dropouts shorter than the control-window duration were gap-filled only when neighboring channels constrained the likely behavior;
3. all high-rate channels were anti-alias filtered before downsampling into replay windows;
4. maneuver phases were labeled manually first and then checked automatically by threshold logic;
5. reconstructed segments were tagged and replayed through the digital twin to verify physical plausibility.

This pipeline matters because a high-rotor-count controller can be misled by apparently clean but physically inconsistent data. Replay validation therefore functions as a quality gate. If a reconstructed lateral translation implies an impossible pitch acceleration or impossible power response, it is rejected even if the time series looks smooth to the eye.

3.6 Summary Statistics Used for Design

The controller does not need every raw sample equally. What it needs are stable statistics that define the envelope where interaction compensation is worthwhile. The most useful design statistics extracted from the 2024 corpus were:

- median and 90th-percentile thrust discrepancy by maneuver family;
- directional asymmetry during left versus right translation;
- residual growth rate after collective steps;
- temperature slope during sustained hover;
- joint occurrence rates of current spikes and inertial anomalies.

These statistics led directly to the architecture choices later in the document. Median discrepancy informs trim bias. Tail discrepancy informs control margin. Directional asymmetry justifies

online effectiveness correction. Residual growth rate informs DDRM persistence thresholds. Joint occurrence rates motivate the PRISM overlay.

Table 3.3: Representative design statistics derived from the normalized 2024 corpus.

Statistic	Median	P90	Design use
Hover thrust discrepancy (%)	9.6	13.2	Trim bias and hover margin assumptions
Step-climb thrust discrepancy (%)	12.8	16.5	Collective gain and transient protection
Directional asymmetry left-/right (%)	2.9	5.4	Translation-specific effectiveness correction
Thermal drift over 120 s hover (%)	3.2	5.1	Slow-state augmentation and endurance estimate updates
DDRM residual onset after bus upset (ms)	85	130	Event persistence tuning

3.7 Discrepancy Decomposition

For controller design, the discrepancy is decomposed into three terms:

$$\Delta T = \Delta T_{\text{aero}} + \Delta T_{\text{electrical}} + \Delta T_{\text{thermal}} \tag{3.1}$$

with the understanding that only the first term directly changes aerodynamic lift, while the other two alter the mapping from command to achieved actuator state and the trustworthiness of measurements. The useful engineering approximation is:

$$\Delta T_{\text{aero}} \approx g_1\left(\frac{s}{D}, v_{\text{body}}, \omega, \phi\right), \tag{3.2}$$

$$\Delta T_{\text{electrical}} \approx g_2\left(I, \dot{I}, V_{\text{bus}}\right), \tag{3.3}$$

$$\Delta T_{\text{thermal}} \approx g_3(T_{\text{motor}}, T_{\text{ESC}}, t_{\text{hover}}). \tag{3.4}$$

These are not independent physical laws; they are a controller-facing decomposition that allows the architecture to compensate, allocate, and classify. It is a disciplined engineering choice: use complexity where it improves control performance, but keep each term observable from accessible sensors.

3.8 What the 2024 Data Justify

The data support several strong conclusions and several weaker ones. The strong conclusions are:

- isolated-thrust assumptions are materially wrong in the relevant hover and low-speed envelope;
- the error is structured enough to model and compensate;
- a deterministic anomaly detector can be built from residual growth and spectral content;
- multi-modal sensing will be necessary if the team wants to distinguish genuine aerodynamic loss from electrical or thermal corruption.

The weaker conclusions are about scale-up. The prototype campaign does not by itself prove what a 57- or 76-rotor vehicle will do in the full transition envelope. What it does justify is the architecture and the work plan. It provides exactly the kind of evidence needed to choose a deterministic control strategy, to design a better HIL bench, and to prioritize which higher-fidelity sensing inserts are worth the added complexity.

3.9 From Ground Truth to Architecture

The central takeaway from the 2024 campaign is that the aircraft needs an architecture that closes the loop around discrepancy, not one that treats discrepancy as a tuning nuisance. That is the purpose of the next chapter. It defines the GNFCS stack that carries the measured lessons of the flight-test program into navigation, control allocation, anomaly detection, and robust sensing.

Chapter 4

GNFCS Control Architecture

The GNFCS architecture is designed around a simple governing idea: every major block should either reduce discrepancy, expose discrepancy, or act safely in the presence of discrepancy. That leads to a stack with clear boundaries. Navigation estimates the vehicle state and sensor health. Guidance turns mission intent into a feasible desired wrench and motion profile. Control allocation distributes that wrench across a large set of interacting rotors while respecting saturation and health constraints. DDRM and PRISM cut across the stack as residual and sensing overlays rather than replacing the primary controller.

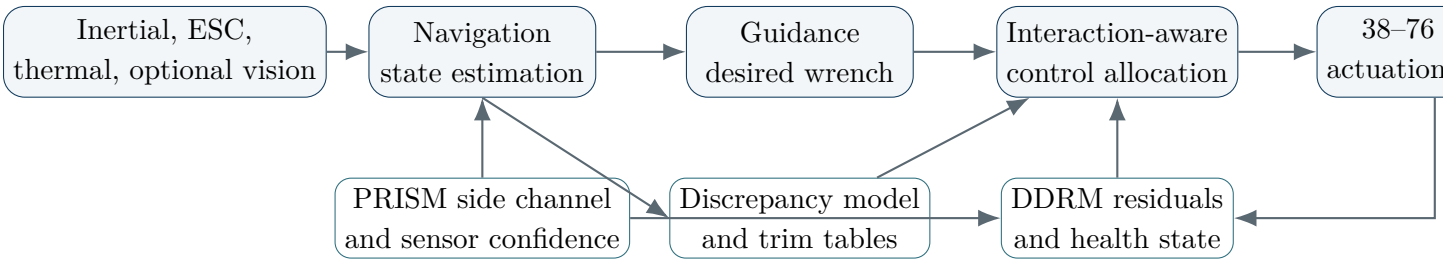


Figure 4.1: GNFCS stack for articulated ring-wing distributed propulsion. The architecture is deterministic end-to-end: interaction-aware modeling informs allocation, DDRM monitors divergence, and PRISM adds an independent sensing channel without becoming a single point of flight-critical failure.

4.1 Design Principles

Four principles drive the architecture.

First, deterministic over opaque. The system is expected to operate in a review-heavy environment where investors, partners, and eventually regulators will ask why a specific control decision was made. A quadratic allocation program with named constraints and residual thresholds is easier to trust than an end-to-end learned controller.

Second, layered observability. High rotor count gives the aircraft many actuators but also many possible failure signatures. The stack therefore separates state estimation, allocation, residual generation, and higher-order disturbance sensing so that disagreement between channels becomes informative instead of catastrophic.

Third, graceful degradation. The architecture is intended to remain flyable when the advanced sensing layer is missing, late, or unavailable. PRISM improves disturbance attribution; it does not replace the base IMU and electrical state estimator.

Fourth, compute realism. The controller is built for embedded GPU-class hardware, not for a workstation hidden behind a demo. That constraint is why the surrogate model is compact, the allocator is sparse, and the anomaly detector uses short-window deterministic statistics rather than large network inference.

4.2 Navigation Layer

The navigation layer fuses inertial, electrical, barometric, and optional visual data into a flight-control state estimate. The core state vector is written as

$$\mathbf{x} = \begin{bmatrix} \mathbf{p}^\top & \mathbf{v}^\top & \mathbf{q}^\top & \mathbf{b}_a^\top & \mathbf{b}_\omega^\top & \boldsymbol{\kappa}^\top \end{bmatrix}^\top \tag{4.1}$$

where $\mathbf{p}$ and $\mathbf{v}$ are position and velocity, $\mathbf{q}$ is attitude quaternion, $\mathbf{b}_a$ and $\mathbf{b}_\omega$ are inertial biases, and $\boldsymbol{\kappa}$ denotes slow-varying interaction parameters such as thrust scaling and thermal efficiency state.

The practical implementation is an extended Kalman filter with explicit residual channels for:

- inertial consistency;
- bus voltage and aggregate current consistency;
- altitude consistency across barometric and model-derived heave acceleration;
- optional visual odometry consistency when a camera stack is present.

The paper assumes a ROS 2 middleware layer and an optional ORB-SLAM3-derived visual module when the operating environment supports it [9, 2]. The architecture does not depend on visual odometry, but it can use it when available to tighten low-speed drift in GPS-degraded operations.

4.3 Guidance Layer

Guidance is intentionally modest in Phase I. The main requirement is to produce feasible body-force and body-moment demands for hover hold, step climb, and low-speed translation. This is not yet a full mission optimizer. The guidance block computes a desired wrench

$$\mathbf{u}_{\text{des}} = \begin{bmatrix} F_x & F_y & F_z & M_x & M_y & M_z \end{bmatrix}^\top \tag{4.2}$$

from waypoint or hold objectives, subject to soft bounds derived from the latest interaction envelope and actuator margins.

This choice is deliberate. In early high-rotor-count development, most problems blamed on "guidance" are actually allocation or observability problems. Keeping guidance lightweight prevents the project from hiding deeper actuator and sensing issues under aggressive planner logic.

4.4 Interaction-Aware Allocation Layer

The allocation layer is the heart of the architecture. For N_r rotors, the allocator solves

$$\min_{\mathbf{T}} \ \frac{1}{2} \left(\mathbf{T} - \mathbf{T}_{\text{trim}}\right)^{\top} W \left(\mathbf{T} - \mathbf{T}_{\text{trim}}\right) \tag{4.3}$$

subject to

$$B(\hat{\boldsymbol{\kappa}})\mathbf{T} = \mathbf{u}_{\text{des}}, \tag{4.4}$$

$$T_{j,\min} \le T_j \le T_{j,\max}, \tag{4.5}$$

$$\dot{T}_{j,\min} \le \dot{T}_j \le \dot{T}_{j,\max}, \tag{4.6}$$

where $B(\hat{\boldsymbol{\kappa}})$ is the current control effectiveness matrix after interaction and health corrections have been applied.

Three features make this allocator different from a conventional multicopter mixer:

- effectiveness is updated online from discrepancy and health state rather than held fixed;
- rotor clusters can be weighted asymmetrically to avoid exciting known structural or electrical stress patterns;
- the trim reference $\mathbf{T}_{\text{trim}}$ is not static, but follows the current flight condition and thermal regime.

Because the system is highly over-actuated, the allocator can also be used as a load-shaping tool. That matters commercially. It means the controller can trade some efficiency for quieter or less vibratory operation during missions where payload stability is more valuable than peak endurance.

4.5 Health and Reconfiguration Interfaces

Rotor count creates a combinatorial health problem if handled naively. The architecture therefore treats health in grouped levels:

1. per-rotor state for fast protection and thrust clipping;
2. cluster state for allocator weighting and power-bus balancing;
3. vehicle state for mission-level decisions such as loiter, return, or abort.

DDRM writes into this health interface by tagging channels or clusters as suspect, derated, or failed. The allocator consumes those tags directly. The result is a clean separation: DDRM decides whether the observed discrepancy is meaningful, and the allocator decides how to redistribute effort.

4.6 Mode Management and Safety Monitors

The meaning of residuals changes by flight phase, so the GNFCS carries explicit mode management rather than one global set of thresholds. The baseline modes are pre-arm validation, take-off and ascent, precision hover, low-speed translation, degraded-hold, and landing recommendation. These modes are not cosmetic labels. They influence acceptable allocator aggressiveness, saturation protection, and residual persistence requirements.

Safety monitors run beside the control path and watch three families of quantities:

- control margin, including aggregate thrust headroom and directional authority;
- sensing trust, including estimator disagreement and timing health;
- hardware stress, including thermal rise and persistent bus anomalies.

When any of these leave the accepted band, the runtime can reduce maneuver ambition before a human would otherwise recognize the degradation. That is especially valuable for heavy-lift and precision-hover missions where the operator may not visually perceive the erosion of control margin until very late.

4.7 Traceability from Command to Actuator

One of the architecture's strongest practical advantages is auditability. For every actuator packet, the runtime can log the requested body wrench, the current effectiveness matrix, any health-induced or DDRM-induced weight changes, the final rotor command vector, and the near-term residual outcome. This makes the controller explainable in a way that is useful both technically and commercially.

From a development standpoint, traceability dramatically shortens debugging time. From a partner or regulator standpoint, it shows that the aircraft is not making opaque decisions. High-rotor-count systems need that kind of transparency because multiple command solutions may exist mathematically while only some are safe, quiet, or thermally sustainable in practice.

4.8 Real-Time Software Partitioning

The reference implementation partitions the runtime into fixed-rate tasks:

- **200 Hz:** navigation prediction and update, wrench demand update, allocation solve, DDRM residual update;

- **100 Hz:** thermal state update, cluster health synthesis, PRISM fusion update when enabled;
- **20 Hz:** mission manager, logging summaries, operator feedback;
- **1 Hz:** trend statistics, storage flush, and diagnostic snapshots.

This schedule is realistic for Jetson Orin-class hardware [8]. It also aligns with the broader robotics ecosystem tools that the project intends to use, including ROS 2 for message transport and PX4-style control allocation concepts for actuator abstraction [10].

Table 4.1: Nominal runtime budget for the reference 38-rotor implementation.

Process	Mean latency (ms)	Notes
State prediction and update	1.35	Includes inertial propagation and health-aware covariance adjustment
Guidance and wrench generation	0.40	Hover and low-speed translation modes only in Phase I
Control allocation solve	1.10	Sparse constrained solve with rotor and slew bounds
DDRM residual features	0.55	Sliding window update and threshold evaluation
PRISM fusion overlay	0.80	Optional, decoupled from flight-critical minimum loop
Messaging and command output	0.35	Includes timestamping and actuator packet preparation
Total	4.55	Leaves margin inside a 5 ms target loop

4.9 Why the Architecture Is Credible

Credibility comes from scope discipline. The architecture does not claim to solve the entire eVTOL problem in one leap. It solves a narrower and more valuable problem: how to maintain control quality when high rotor count and tight geometry create systematic mismatch between model and reality. Every block in the stack serves that purpose.

The next two chapters isolate the two most distinctive overlays in the design. First is DDRM, which converts discrepancy into actionable residual logic. Second is PRISM, which extends observability using robust differential sensing rather than relying solely on conventional inertial channels.

Chapter 5

DDRM: Deterministic Differential Remapping for Disturbance Attribution

In the coherent-combination literature attached to this project, DDRM refers to a **deterministic differential remapping method**: infer corrective action from short-interval changes rather than from absolute labels that drift over time [13]. This white paper adopts that same engineering principle for flight controls, but not as a direct copy of the laser controller. Here DDRM is a deterministic disturbance-attribution layer that compares measured vehicle response to expected response, remaps the discrepancy into physically meaningful subspaces, and hands bounded recommendations to the allocator. The goal is not perfect root-cause diagnosis. The goal is fast, auditable separation of aerodynamic, electrical, thermal, and actuator-consistency events in a system whose trim, temperature, and propulsion state are always moving.

5.1 Concept of Operation

At each control instant, the system has three views of the aircraft:

1. the measured state from the navigation stack;
2. the expected next state given the previous state, current model, and actuator commands; and
3. the health-conditioned expected state if one or more channels are derated or biased.

DDRM computes the differential residual

$$\mathbf{r}_k = \mathbf{x}_k^{\text{measured}} - \hat{\mathbf{x}}_k^{\text{expected}} \tag{5.1}$$

and then remaps that residual into physically meaningful subspaces:

$$\mathbf{z}_k = \begin{bmatrix} \mathbf{z}_k^{\text{rigid}} \\ \mathbf{z}_k^{\text{prop}} \\ \mathbf{z}_k^{\text{elect}} \\ \mathbf{z}_k^{\text{thermal}} \end{bmatrix} = R\mathbf{r}_k \tag{5.2}$$

where R is a deterministic remapping matrix derived from the current effectiveness model, state covariance, and sensor availability.

This remapping is what differentiates DDRM from a simple threshold monitor. Instead of asking whether the aircraft has error, DDRM asks what kind of error is most consistent with the observed pattern. That is why it can support reallocation instead of merely flagging a warning lamp.

5.2 Why the Differential Framing Matters

The CALIPR work from Berkeley Lab is useful here because it makes a narrow but important control point. CALIPR, "Coherent Addition using Learned Interference Pattern Recognition," and the related DDRM formulation were built for optical phase control under drift, where absolute labels become stale quickly [12, 13]. The aircraft problem is different, but the lesson transfers cleanly: when the system is drifting, *differences across short windows are often more trustworthy than absolute offsets accumulated over long windows.*

For a high-rotor-count eVTOL this means DDRM should emphasize residual growth, cluster-to-cluster imbalance, pre/post-command changes, and agreement or disagreement across sensing channels. Those features remain interpretable even while the aircraft warms up, the propulsion system ages through a sortie, or the trim point shifts with payload and battery state.

5.3 Residual Features

The working feature set is intentionally compact:

- heave, roll, and pitch residual energy over short windows;
- per-cluster thrust consistency inferred from current, RPM, and body acceleration;
- short-interval differences in current derivative and bus-voltage deviation;
- motor temperature slope and cluster-to-cluster thermal imbalance;
- narrowband spectral energy in bands associated with known structural and propulsion signatures;
- paired-window changes before and after a commanded input or detected upset.

These features are aggregated over sliding windows of 0.1 s, 0.25 s, 0.5 s, and 2.0 s. The short windows catch fast control divergence and electrical upsets; the longer windows catch developing

thermal or persistent aerodynamic bias. This multi-scale approach avoids a common mistake in development programs: using one threshold for everything and then finding that fast false positives and slow missed detections trade off against each other.

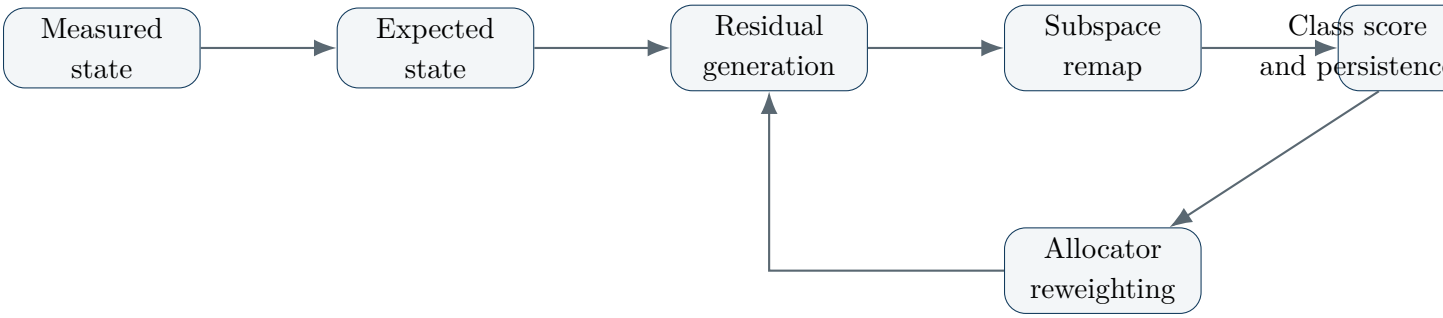


Figure 5.1: DDRM data path. Model residuals are remapped into interpretable disturbance subspaces, then fused with electrical and thermal context before control reallocation or operator notification is triggered.

5.4 Classification Logic

Phase I DDRM is not assumed to be a deep classifier. The defensible implementation is a scored logic graph or a sparse learned layer calibrated from replay and HIL data. Each disturbance class receives a score

$$S_c(k) = \sum_{i=1}^{m} w_{c,i}\, \psi_i(z_i(k)) \tag{5.3}$$

where c indexes disturbance class, $w_{c,i}$ are fixed weights, and $\psi_i(\cdot)$ are bounded feature maps. The classes used in Phase I are:

- aerodynamic interaction growth;
- electrical disturbance or bus-coupling event;
- thermal derate or efficiency drift;
- actuator inconsistency or rotor-cluster underperformance;
- ambiguous, requiring caution but not immediate reconfiguration.

The classifier output is accompanied by a confidence score and a persistence counter. Confidence alone is not sufficient; persistence matters because momentary false positives should not force an unnecessary thrust redistribution. The implementation therefore requires either one high-confidence event or a sequence of medium-confidence events before an allocator weight change is committed.

5.5 Threshold Design

Thresholds are designed in three layers:

1. **channel thresholds** flag individual excursions such as bus-voltage droop or pitch-rate residual spikes;
2. **feature thresholds** operate on short-window energies and slopes;
3. **decision thresholds** determine when a disturbance class score is high enough, long enough, to warrant action.

This layered structure avoids a brittle single-number detector. It also gives engineers multiple levers during tuning. If the system is too reactive, persistence can be adjusted without erasing useful channel-level sensitivity. If the system is too slow, feature maps can be made steeper without redesigning the entire decision graph. That is exactly the kind of tuning practicality that makes a differential method useful in the field instead of only in a paper.

5.6 Trigger Actions

DDRM can request four escalating actions:

1. increase logging and freeze a diagnostic snapshot;
2. adjust estimator covariance or sensor trust for specific channels;
3. reweight or derate one rotor cluster in the control allocator;
4. trigger a mission-level advisory such as hold, climb margin protection, or return-to-safe-area behavior.

Only the first three are automatic in Phase I. The mission-level action remains advisory. This is another deliberate scope decision: early development should prove that the residual logic is stable before giving it authority over strategic mission behavior.

5.7 Representative Event Timeline

A typical DDRM intervention during a lateral translation replay looks like this:

1. a cluster of heave and pitch residuals rises over approximately 80–120 ms;
2. current and bus signals remain within nominal electrical bounds, reducing the electrical-disturbance score;

3. directional asymmetry in the residual subspace pushes the aerodynamic interaction score above decision threshold;
4. the persistence counter confirms the event is not momentary;
5. the allocator reduces confidence in one cluster, reweights neighboring channels, and the residual envelope begins to contract.

This sequence is a useful sanity check. It reflects the kind of interpretable logic that engineers expect from a flight-critical health monitor. It also demonstrates why PRISM, discussed in the next chapter, can add value when the distinction between electrical and aerodynamic phenomena is less clean.

5.8 Phase I Acceptance Targets

Because the available aircraft evidence is still a development corpus rather than a certification database, the right near-term framing is *acceptance targets*, not claimed field-proven confusion-matrix supremacy. The DDRM lane is useful if it can meet the following practical bar in replay and HIL challenge cases:

Table 5.1: Representative DDRM Phase I acceptance targets.

Metric	Practical target	Why it is attainable
Fast aerodynamic or electrical onset	Disturbance score crosses action threshold within roughly 0.1–0.25 s of sustained onset	Consistent with the window lengths and with allocator response budgets already assumed elsewhere in the architecture
False automatic reallocations	Nominal replay and hover-hold cases do not chatter or repeatedly derate healthy clusters	Achievable when persistence and cross-channel agreement are required before allocator action
Thermal and slow-efficiency drift	Persistent multi-second drift is detected before margin erosion becomes obvious in pilot-facing behavior	Thermal signatures evolve slowly enough that trend logic and imbalance features are realistic first-line tools
Held-out maneuver transfer	Detector behavior remains stable across hover, low-speed translation, and injected anomaly cases without retuning every scenario	This is the minimum bar for claiming the detector is architecture-level rather than scenario-specific

These targets are demanding enough to matter and modest enough to be believable. If the

detector cannot meet them, it should remain an advisory analysis layer rather than an automatic reconfiguration authority.

5.9 Interface to the Allocator

The most important engineering feature of DDRM is its interface to control allocation. If the disturbance score indicates likely cluster underperformance, the allocator updates the weight matrix and effectiveness terms:

$$W \leftarrow W + \Delta W(\mathbf{s}_{\text{ddrm}}) \tag{5.4}$$

$$B \leftarrow B + \Delta B(\mathbf{s}_{\text{ddrm}}) \tag{5.5}$$

where $\mathbf{s}_{\text{ddrm}}$ denotes the current disturbance state. In other words, DDRM does not directly command thrust. It changes the allocator's understanding of what thrust is available and what thrust is desirable to use.

This preserves clean authority boundaries while still enabling very fast adaptation. A cluster suspected of underperforming can be derated within one or two control cycles without the rest of the flight stack having to guess why. The allocator remains the actuator decision-maker; DDRM only sharpens the allocator's view of which parts of the model should be trusted less.

5.10 Why DDRM Matters Commercially

DDRM is not only a technical safety feature. It is also a business feature. Operators care about dispatch reliability, false aborts, and predictive maintenance. A disturbance detector that can distinguish aerodynamic loading from electrical or thermal issues creates value before a full passenger platform exists. It can support:

- smarter maintenance intervals on high-channel propulsion systems;
- better fault triage after a mission interruption;
- higher-quality field test evidence for OEM partners and insurers;
- a licensable health-monitoring layer even for airframes that do not adopt the full ARW geometry.

5.11 Chapter Takeaway

DDRM is the architecture's interpretability engine. It turns raw discrepancy into structured evidence that the rest of the system can act on. Its most grounded claim is not magical fault diagnosis; it is robust, differential disturbance attribution under drift. The next chapter extends that observability idea further by describing PRISM, the proposed robust sensing overlay for separating electrical corruption from genuine vehicle disturbance signatures.

Chapter 6

PRISM: Bench-First Robust Quantum Sensing Overlay

PRISM is the most forward-leaning element in the white paper and therefore the one that requires the clearest positioning. In the March 22, 2026 preprint "Robust Quantum Sensing via Prethermal Spin Orbits," PRISM stands for **Prethermal Robust Internally Modulated Spin Magnetometry** [11]. That paper is a proof-of-principle magnetometry result, not an aerospace-qualified avionics package. This white paper therefore treats PRISM as a bench-first robust sensing roadmap that may improve disturbance attribution in the development environment and only later earn tighter packaging. The technical posture is ambitious but credible when stated narrowly: use the differential readout principle demonstrated in published robust quantum sensing work, keep the insert modular, and prove value first in anomaly separation rather than in primary flight stabilization [3, 1, 11].

6.1 Why Classical Sensing Is Not Always Enough

High-rotor-count electric aircraft generate exactly the kind of mixed disturbance field that frustrates classical sensing:

- structural vibration contaminates inertial estimates;
- switching currents and bus transients contaminate magnetic and current-derived features;
- thermal drift shifts sensor bias and motor constants over the course of a sortie;
- physically distributed actuation creates spatially localized phenomena that a single central IMU cannot localize well.

The usual answer is more filtering. Filtering helps, but it also hides transient information that DDRM needs in order to classify the disturbance correctly. PRISM is intended to add an independent differential sensing path that is sensitive to changes in local electromagnetic conditions while rejecting common-mode drift and readout contamination. In this program, any vibration relevance is indirect:

vibration changes the local electromagnetic and structural environment seen during controlled tests; PRISM is not claimed as a direct replacement for inertial sensing.

6.2 PRISM Operating Concept

PRISM stands for **Prethermal Robust Internally Modulated Spin Magnetometry**. For the aircraft program, the practical interpretation is narrower than the physics paper: use a differential readout lane, synchronized to propulsion timing and test instrumentation, as a side-channel for disturbance attribution. During Phase I, that is most realistically realized on an off-aircraft or tethered bench insert near a representative propulsion or power-electronics rig, not as a certifiable flight sensor riding the production avionics stack.

The core signal model is

$$y_{\mathrm{prism}}(k) = \mathcal{D}\left(m_k^{(+)}, m_k^{(-)}\right) \tag{6.1}$$

where $m_k^{(+)}$ and $m_k^{(-)}$ denote successive readouts associated with the two internally modulated orbit states described in the cited PRISM paper, and $\mathcal{D}(\cdot)$ is the differential extraction operator. The value of this structure is not just sensitivity. It is rejection of shared drift, transients, and common electrical background before the result is fused with vehicle telemetry.

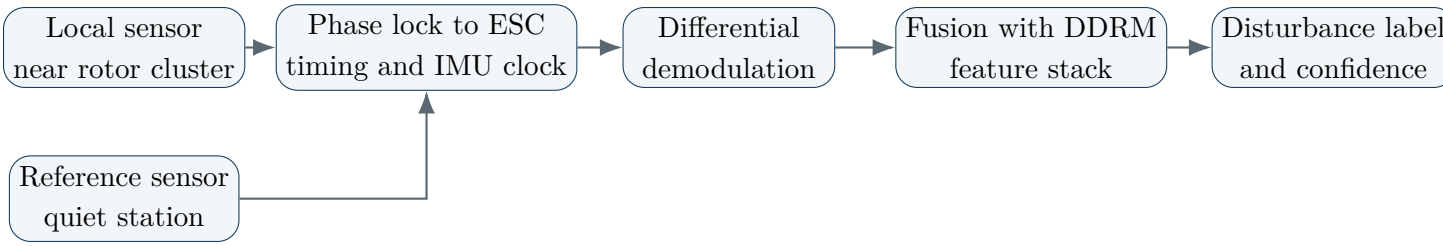


Figure 6.1: PRISM concept of operation. A differential sensing path synchronized to propulsion timing creates an auxiliary disturbance channel that can help distinguish localized electrical or propulsion disturbances from common-mode bias and background interference.

6.3 Relation to Published Quantum Sensing

Published nitrogen-vacancy center sensing work demonstrates the broader value of optically read spin systems for sensitive magnetic and field-related measurements [3, 1]. The new PRISM paper adds a more specific claim: differential prethermal-orbit readout can reconstruct rapidly varying magnetic signals while strongly suppressing common-mode background, and the demonstrated setup remains robust against pulse imperfections, bias-field drift, temperature variation, and mechanical vibration [11]. This white paper does not claim that an aerospace-qualified quantum package is already integrated into the flight vehicle. It claims something narrower and more believable:

- published robust quantum sensing methods support the feasibility of differential magnetic

readout with strong common-mode suppression;

- the aircraft development environment presents a strong use case for such readout because electrical disturbances, switching artifacts, and structural coupling are often entangled in ordinary telemetry;
- Aerial MECHANICA can use a bench or tethered module during Phase I and Phase II to improve anomaly labeling, calibration, and health monitoring before deciding whether tighter onboard integration is commercially justified;
- the cited work does *not* establish direct rotor-wash sensing, navigation-grade attitude sensing, or flight-ready packaging, so those claims are intentionally excluded here.

This posture is important. It keeps PRISM on the frontier without making the entire program depend on a hardware maturity leap.

6.4 Signal Chain and Fusion Logic

PRISM data enter the system as a side channel, not as a replacement for inertial navigation. The fusion logic forms an augmented feature vector

$$\mathbf{f}_k = \begin{bmatrix} \mathbf{f}_k^{\text{imu}} & \mathbf{f}_k^{\text{elect}} & \mathbf{f}_k^{\text{thermal}} & \mathbf{f}_k^{\text{prism}} \end{bmatrix}^\top \tag{6.2}$$

and computes disturbance-specific likelihoods

$$\ell_c(k) = \exp\left(-\frac{1}{2}\left(\mathbf{f}_k - \boldsymbol{\mu}_c\right)^\top \Sigma_c^{-1}\left(\mathbf{f}_k - \boldsymbol{\mu}_c\right)\right) \tag{6.3}$$

for classes c such as aerodynamic interaction growth, electrical disturbance, and thermal drift.

In practice, the important point is not the exact equation. It is the role of the PRISM term. If inertial and electrical channels disagree about whether a disturbance is aerodynamic or electrical, PRISM acts as a tiebreaker by providing a physically independent view. This is especially valuable when the system sees current spikes and structural response simultaneously. PRISM does not replace inertial navigation or flight-state estimation. It only contributes an auxiliary disturbance observable to DDRM and the test-analysis stack.

6.5 Mechanical and Electrical Integration Concept

The initial physical integration is intentionally conservative:

- place one local sensing head near a representative inverter, motor phase lead, or propulsion emulator and one reference channel in a quieter magnetic location;
- synchronize timestamps directly with ESC telemetry, current measurements, shaker excitation, and high-rate inertial data;

- isolate the sensing electronics from the main propulsion bus and provide their own filtered supply;
- run the interpretation pipeline on a paired compute lane or bench workstation without blocking the flight-critical loop.

This creates immediate value for development test without requiring an aircraft redesign. It also creates a migration path. If the sensing insert proves valuable for maintenance, health monitoring, or field diagnostics, it can be productized even if it never becomes part of the primary stabilized control loop.

6.6 Performance Expectations

The appropriate expectation for Phase I PRISM performance is not miraculous new navigation accuracy. It is improved disturbance discrimination in controlled tests. The representative target is:

- a measurable reduction in ambiguous electrical-versus-aerodynamic labels during bench, replay, or HIL challenge cases;
- clearer attribution of localized power-electronics or cluster disturbances relative to IMU-plus-bus-only logic;
- improved visibility into persistent electromagnetic signatures before they manifest as visible control loss or maintenance confusion;
- no claim of primary-flight-loop safety credit in Phase I.

If PRISM can deliver those outcomes, it earns its place in the program even before the hardware matures further. If it cannot, it should remain a research lane rather than be forced into the product narrative. In other words, its first commercial value, if any, is more likely to be in test and maintenance intelligence than in direct piloting performance.

6.7 Maturity Roadmap

PRISM should be developed in three maturity steps rather than one leap.

Step 1: off-aircraft bench assist. The sensing path exists entirely for replay, diagnostics, and label improvement on representative propulsion hardware.

Step 2: tethered or cart-mounted support. The sensing hardware is time-synchronized to the aircraft during limited tests but does not sit in the minimum flight-critical path.

Step 3: integrated product option. Only after repeatable benefit is demonstrated should the module be considered for tighter avionics packaging and field deployment as a premium diagnostic or health-monitoring feature.

This staged roadmap is commercially rational. It allows the company to extract value from the sensing research early while controlling integration risk.

6.8 Potential Performance Gains Beyond Classification

Although disturbance classification is the first target, PRISM could create second-order benefits if it matures:

- earlier recognition of bus or switching anomalies that would otherwise be diagnosed only after a sortie;
- better localization of repeat vibration sources during maintenance;
- improved confidence in whether a cluster underperforms due to aerodynamic loading or electrical degradation;
- a differentiated data product for fleet health analytics.

These outcomes matter because they connect a sophisticated sensing story directly to operator economics, not just to scientific novelty.

6.9 Risk Containment

Because PRISM is the least mature subsystem, it is architected to fail gracefully. The flight stack degrades to classical sensing if PRISM is absent, noisy, or unavailable. This containment strategy preserves the integrity of the overall program:

- no mission block depends exclusively on PRISM to remain safe;
- all PRISM outputs are advisory or fused, not blindly authoritative;
- the system can quantify PRISM benefit empirically by comparing confusion matrices and residual decision quality with and without the overlay.

6.10 Strategic Value

PRISM is strategically attractive because it differentiates Aerial MECHANICA beyond geometry alone. Many teams can claim a novel airframe. Fewer can claim a credible disturbance intelligence stack that extends from control allocation into sensing and maintenance analytics. If Phase I demonstrates value even on a bench and HIL system, PRISM becomes a justified forward option rather than a speculative slogan: it connects distributed propulsion, robust autonomy, and advanced sensing in one coherent product thesis.

6.11 Chapter Takeaway

PRISM is a defensible but tightly bounded bet. It extends the disturbance-aware philosophy of the architecture without turning the program into a pure quantum hardware gamble. The credible near-term use is bench-first disturbance attribution, not flight-critical quantum avionics. With that boundary established, the next chapter turns from architecture to execution by laying out the Phase I work plan in explicit engineering tasks.

Chapter 7

Phase I Work Plan & Task Execution

Phase I is structured as a six-month engineering sprint with explicit technical exits. The goal is not to build the final aircraft. The goal is to make the control architecture investable, integrable, and ready for scaled flight validation. That requires disciplined tasking, not vague research ambition.

7.1 Phase I Objectives

The program has four primary objectives:

1. consolidate and normalize the 2024 prototype dataset into a reusable model-calibration corpus;
2. implement the deterministic GNFCS stack on a reproducible HIL environment;
3. validate DDRM disturbance identification and allocator reweighting on representative maneuver and fault cases;
4. demonstrate the value of the PRISM sensing overlay as a development and diagnostic tool.

Each objective has a concrete exit criterion, summarized in Table 7.1.

Table 7.1: Phase I exit criteria.

Objective	Exit criterion	Acceptance evidence
Dataset and model baseline	Curated calibration corpus frozen	Versioned logs, reconstruction notes, and baseline discrepancy metrics
GNFCS implementation	200 Hz closed-loop HIL stack operating within loop budget	Timing traces, replay logs, and allocator command auditability
DDRM validation	Residual classification reliable enough to drive cluster reweighting	Held-out challenge scenarios and confusion matrix
PRISM evaluation	Measurable improvement in disturbance attribution or diagnostic clarity	A/B comparison against classical sensing baseline

7.2 Work Breakdown Structure

Task 1: Data Recovery, Normalization, and Truth Bounding

1. inventory raw flight logs, bench data, and analyst notes;
2. align clocks across ESC, IMU, and altitude channels;
3. identify missing segments, sensor dropouts, and disclosure-sensitive values;
4. reconstruct representative traces where necessary using synchronized neighboring channels and replay-constrained interpolation;
5. define the calibration envelope and explicitly label out-of-envelope conditions.

Task 2: Interaction Model and Allocation Baseline

1. fit a compact discrepancy model against the normalized corpus;
2. validate the model on held-out maneuver segments;
3. generate control effectiveness tables and trim conditions for the 38-rotor baseline;
4. verify allocator feasibility across hover, step climb, and low-speed translation cases;
5. establish nominal and degraded actuator weighting sets.

Task 3: Real-Time Flight Software Integration

1. implement state-estimation and allocation modules in the target runtime;
2. integrate ROS 2 message definitions, logging, and deterministic timestamps;
3. connect the runtime to the simulator and HIL peripherals;
4. benchmark loop timing on Jetson-class hardware;
5. lock the baseline software container for repeated replay.

Task 4: DDRM and Reconfiguration Logic

1. compute residual features and scoring functions;
2. define trigger persistence rules;
3. connect DDRM outputs to allocator weight and effectiveness updates;
4. test the logic on injected aerodynamic, electrical, and thermal cases;
5. generate operator-facing disturbance summaries.

Task 5: PRISM Bench Integration

1. stand up the differential sensing bench and synchronization path;
2. collect paired data against electrical and vibratory injections;
3. compare attribution quality with and without the PRISM side channel;
4. decide whether the next phase justifies tighter onboard packaging.

7.3 Month-by-Month Execution

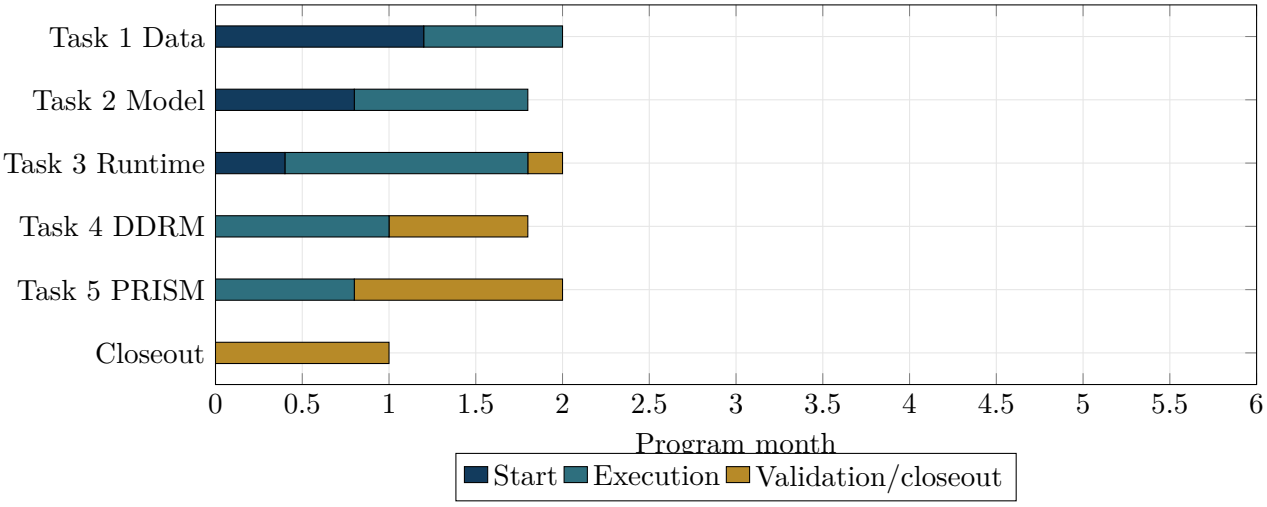


Figure 7.1: Six-month Phase I execution plan. The program is deliberately front-loaded with data and architecture closure so hardware and sensing risk do not dominate the final month.

Month 1: recover data, build the logging schema, and freeze the baseline maneuver library. The output of Month 1 is confidence that the calibration dataset is traceable and usable.

Month 2: fit the interaction discrepancy model, generate trim and effectiveness tables, and run initial allocator feasibility sweeps. The key outcome is not perfection but a stable model structure that survives held-out replay.

Month 3: integrate the real-time runtime on the target hardware, connect the simulator bridge, and close the loop in HIL. This is the make-or-break systems month because it tests whether the document's architecture actually fits within real compute and interface constraints.

Month 4: add DDRM features, challenge the system with injected anomalies, and iterate thresholds until the detector drives meaningful allocator changes without thrashing.

Month 5: stand up PRISM bench integration, perform A/B comparisons, and build the diagnostic evidence package that shows whether the overlay adds enough value to justify a Phase II insert.

Month 6: freeze the software stack, run demonstration scenarios, prepare partner-ready artifacts, and package export schemas, figures, and budgetary evidence for follow-on funding and integration discussions.

7.4 Demonstration Scenarios

The final demonstration set is intentionally narrow and convincing rather than broad and shallow:

- **Scenario A:** 120-second GPS-denied hover with representative thermal rise;

- **Scenario B:** low-speed lateral translation with interaction asymmetry;
- **Scenario C:** cluster derate or underperformance injection with allocator reweighting;
- **Scenario D:** electrical disturbance injection with DDRM and PRISM separation logic.

These scenarios are chosen because they expose the distinctive value of the architecture. A flashy autonomous mission would be less useful at this stage than a rigorous demonstration that the aircraft can tell the difference between aerodynamic loss, electrical upset, and slow thermal derate.

7.5 Deliverables

Phase I closes with the following deliverables:

- structured technical white paper and supporting assets;
- versioned model coefficients and reconstruction notes;
- containerized HIL software baseline;
- representative data package and disturbance challenge set;
- allocator, DDRM, and PRISM interface definitions;
- budget and follow-on program plan.

7.6 Formal Review Gates

To prevent the project from drifting into open-ended research, three formal review gates are recommended.

Gate 1: Data closure review. Held at the end of Month 2. The team confirms that the dataset is usable, the reconstruction rules are documented, and the discrepancy model has a stable preliminary form.

Gate 2: Runtime readiness review. Held during Month 4 after HIL closure. The team confirms that timing budgets are real, the allocator is auditable, and replay scenarios can be reproduced without manual heroics.

Gate 3: Demonstration readiness review. Held near the start of Month 6. The team confirms that DDRM actions are stable, PRISM results are honestly bounded, and the public-facing evidence package matches the actual technical results.

These gates are inexpensive but valuable. They force alignment between technical progress and communication.

7.7 Artifacts Required at Closeout

The final closeout package should include more than plots and prose. It should include:

- a frozen scenario library with deterministic seeds and expected outcomes;
- software container hashes and dependency manifests;
- an allocator command audit log for each demonstration scenario;
- a reconstruction ledger showing which representative traces were synthesized and why;
- a decision memo on whether PRISM advances to tighter packaging in the next phase.

Those artifacts are what make the work portable to investors, partners, and follow-on teams.

7.8 Execution Philosophy

The execution plan intentionally avoids two traps common in early aerospace programs. The first trap is letting software ambition race ahead of data quality. The second is letting hardware novelty consume the schedule before the control story is coherent. The Phase I plan instead closes the evidence loop first, integrates the control architecture second, and treats the advanced sensing layer as a bounded value-add. That ordering is what makes the program realistic.

7.9 Chapter Takeaway

Phase I is designed to end with a usable control product foundation, not just a research report. The next chapter describes the tooling, software environment, and simulation stack needed to execute the plan repeatably.

Chapter 8

Tooling, Software, and Simulation Environment

The white paper assumes a modern but credible engineering toolchain. The objective is not to chase novelty in every layer. The objective is to assemble a stack that can support repeatable control development, data reconstruction, and HIL testing without hiding critical assumptions inside proprietary black boxes.

8.1 Core Toolchain

The proposed reference stack is built around the following components:

- **Embedded compute:** NVIDIA Jetson Orin-class hardware for real-time controller execution [8];
- **Middleware:** ROS 2 Humble or equivalent deterministic messaging pattern for time-stamped module communication [9];
- **Simulation:** Isaac Sim or an equivalent Omniverse-based robotics simulation environment for closed-loop HIL and sensor playback [7];
- **Flight-control abstraction:** PX4-style control allocation concepts, adapted for ARW-specific effectiveness modeling [10];
- **Data analysis:** Python, NumPy, SciPy, Pandas, and Matplotlib for model fitting, replay, and report generation;
- **Documentation and release:** LaTeX-based technical publishing for the white paper and structured appendices.

This stack is chosen because it is legible to collaborators. It also avoids a recurring failure mode in startup flight programs: building a unique internal stack so early that no external reviewer can quickly audit what the software is doing.

8.2 Digital Twin Requirements

The simulator must support more than rigid-body motion. For this program it needs to host:

- rotor-count parameterization and ring geometry metadata;
- interaction-aware thrust scaling tables or surrogate models;
- inertial, electrical, and thermal disturbance injection;
- hardware-timed replay and actuator command auditing;
- visual and numerical overlays for control allocation and residual growth.

The digital twin therefore acts as both a development tool and a truth-bounding tool. When representative flight data are reconstructed, the digital twin provides the consistency check. If a reconstructed trace implies impossible attitude, power, or timing behavior, it fails the replay gate and is revised or discarded.

8.3 Software Architecture

The software repository should be split into six logical packages:

1. `gnfcs_core`: estimator, guidance, and allocator runtime;
2. `gnfcs_models`: interaction coefficients, geometry data, and effectiveness tables;
3. `gnfcs_ddrm`: residual features, thresholds, and event serialization;
4. `gnfcs_prism`: optional sensing overlay and synchronization utilities;
5. `gnfcs_sim`: simulator bridge, scenario definitions, and injected disturbances;
6. `gnfcs_analysis`: notebooks, reporting scripts, and acceptance plots.

This package split is intentionally product-oriented. It keeps the flight-critical path narrow while allowing analysis and sensing work to move faster without destabilizing the runtime.

8.4 Data and Configuration Management

High-rotor-count control programs fail quietly when configuration management is weak. Rotor ordering, geometry conventions, sign conventions, and timing assumptions can invalidate an otherwise sound algorithm. The project therefore needs strict configuration rules:

- every geometry file carries a versioned rotor index map;

- every replay scenario references an explicit sensor schema version;
- every allocator run records the exact model coefficient set used;
- reconstructed traces carry provenance flags and are never mixed silently with pristine logs.

The final export package described in Appendix F is part of this discipline. The goal is to make it easy for a partner to reproduce not only the result but the assumptions behind the result.

8.5 Continuous Verification

The software environment should support three verification loops:

- **unit verification** for estimator math, control allocation constraints, and residual feature calculations;
- **scenario replay verification** for known benchmark maneuvers;
- **hardware timing verification** on the target compute platform.

The bar for any commit or release candidate is not just "it runs." The bar is:

1. it reproduces benchmark maneuver results within tolerance;
2. it does not regress loop timing;
3. it preserves configuration traceability;
4. it emits human-readable diagnostics that explain why a failure occurred.

8.6 Cybersecurity and Data Hygiene

Even a prototype flight-control program benefits from basic cybersecurity and data hygiene. Configuration files, telemetry traces, and exported model coefficients should be treated as controlled engineering data. At minimum, the project should use:

- signed container images or checksummed release bundles;
- access-controlled storage for raw logs and sensitive geometry files;
- clear separation between public-ready representative data and internal raw traces;
- automated backups for the scenario library and coefficient registry.

This matters because the same evidence package that supports funding and partnerships can also become confusing or risky if provenance is lost.

8.7 Illustrative Runtime Stack

Table 8.1: Illustrative software and runtime environment.

Layer	Candidate tool	Why it is included
Operating system	Ubuntu LTS with real-time tuning	Broad hardware support and reproducible deployment
Real-time messaging	ROS 2 Humble	Time-stamped communication and mature robotics ecosystem
Simulation	Isaac Sim / Omniverse	High-fidelity robotic scene integration and HIL bridging
Control runtime	C++/CUDA plus Python analysis	Deterministic execution with rapid analysis iteration
Visualization	Matplotlib, ParaView, simulator overlays	Fast review of discrepancy and allocation behavior
Documentation	LaTeX toolchain	Technical presentation with equations, figures, and appendices

8.8 Operator and Test Interfaces

Even in Phase I, the test interface matters. The recommended operator console exposes:

- current flight mode and attitude/position bounds;
- allocator saturation margin by cluster;
- DDRM disturbance classification and confidence;
- thermal trend and bus health summaries;
- PRISM overlay status when enabled.

The point is not user experience polish. The point is shortening the loop between anomaly observation and engineering interpretation. That can save weeks during early test programs.

8.9 Scalability of the Toolchain

The toolchain is designed so that moving from a 38-rotor semi-scale reference to a larger platform changes data and configuration more than software structure. Geometry files grow, control effectiveness tables change, and simulator assets become heavier. The core ideas do not need to be rewritten. That is a sign of healthy architecture.

8.10 Chapter Takeaway

The tooling strategy is intentionally conservative in the best sense. It uses mature, inspectable building blocks to support a novel control problem. The next chapter turns from tools to resources by laying out the budget, facilities, staffing, and infrastructure required to execute the plan.

Chapter 9

Budget, Facilities, and Resource Allocation

The program is small enough to move quickly but large enough that under-budgeting would undermine credibility. The correct budget posture is therefore disciplined rather than minimal. Phase I does not require a full flight campaign with production hardware, but it does require enough engineering depth, instrumentation, and compute margin to make the control evidence believable.

9.1 Budget Philosophy

Three rules drive the budget:

1. spend first on evidence quality, because the entire program depends on trustworthy data and reproducible replay;
2. spend second on integration velocity, because a slow software-hardware loop will dominate the schedule;
3. treat advanced sensing as a bounded experimental lane rather than as the main burn rate driver.

This leads naturally to a Phase I budget centered on staff time, embedded hardware, HIL equipment, limited fabrication, and testing operations rather than large airframe manufacturing outlays.

Table 9.1: Illustrative six-month Phase I budget. Values are planning-grade estimates suitable for proposal and diligence use.

Category	Cost (USD)	Purpose
Core engineering labor	198,000	Principal investigator, controls/-software, embedded/sensing, integration support
Embedded compute and networking	18,000	Jetson-class targets, rugged I/O, timing, storage, spares
HIL bench hardware and instrumentation	36,000	Power electronics, data acquisition, thermal and vibration sensing, shielding
Simulation, software, and cloud tooling	12,000	Software services, storage, backup, analysis infrastructure
Fabrication and integration materials	22,000	Harnesses, mounts, fixtures, adapters, safety equipment
PRISM bench experiment lane	24,000	Differential sensing hardware, optics support, synchronization electronics
Travel, partner demos, and range access	14,000	On-site reviews, limited test access, presentation support
Contingency reserve	26,000	Hardware failure, late procurement, additional specialist support
Total	350,000	Fully burdened planning baseline

9.2 Staffing Plan

The baseline staffing profile for a six-month effort is:

- **Principal investigator / chief engineer:** overall architecture, control logic, partner interface;
- **controls and autonomy engineer:** estimator, allocator, and replay tooling;
- **simulation and software engineer:** simulator bridge, deployment, CI, and logging stack;
- **embedded and sensing engineer:** data acquisition, timing, ESC telemetry, PRISM bench interface;
- **test technician / systems integrator:** HIL setup, instrumentation, wiring, and documentation support.

This is intentionally lean. It is sufficient for a serious Phase I if roles are clearly bounded and

outside specialist services are used only where necessary, for example in optical sensing consultation or specialized fabrication.

Table 9.2: Illustrative staffing load by role.

Role	Average FTE load	Peak period
Principal investigator / chief engineer	0.9	All months, strongest during architecture and review gates
Controls and autonomy engineer	1.0	Months 2–5
Simulation and software engineer	0.8	Months 1–4
Embedded and sensing engineer	0.8	Months 3–6
Technician / integrator support	0.5	Months 2–6

9.3 Facilities Requirements

The program can be executed without dedicated aircraft production infrastructure. The minimum credible facility set is:

- an engineering office with secure compute and versioned data storage;
- an integration bay for propulsion bench work, wiring, and thermal instrumentation;
- a HIL bench area with safety shielding, isolated power, and logging equipment;
- access to an outdoor test pad or partner range for limited follow-on validation;
- optional lab partnership access for advanced sensing and optical readout experiments.

For a startup-stage company, this is advantageous. It means the program can be run from a compact engineering footprint while still generating the artifacts expected by serious reviewers.

9.4 Resource Allocation Discipline

The most important management rule for the resource plan is that new work must displace existing work. If a new sensing experiment appears, it should either fit inside the PRISM lane or displace lower-priority activity. Otherwise Phase I risks becoming a diffuse “innovation program” with weak closure. The budget supports a focused engineering sprint, not an unconstrained research portfolio.

9.5 Resource Allocation by Month

Resource loading is intentionally uneven. The program spends more labor in the first three months because data closure and software integration are schedule-critical. Hardware expenditures also

concentrate early enough to prevent bench delays. Travel and partner review costs rise later, when the evidence package is ready to circulate.

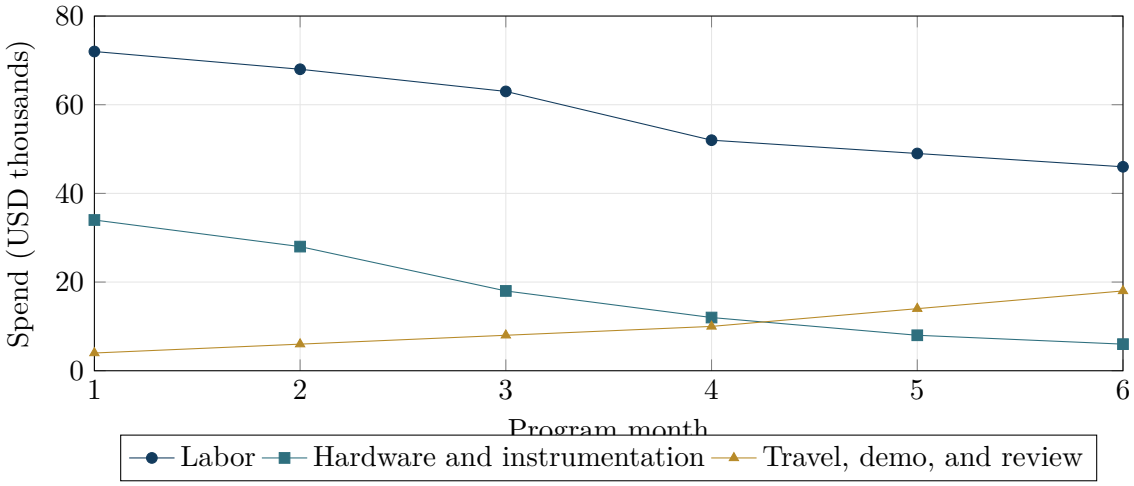


Figure 9.1: Illustrative Phase I monthly spend profile. Labor dominates throughout; hardware and instrumentation peak early, while partner review and demonstration costs peak near the closeout period.

9.6 Bill of Materials Priorities

Not every item deserves immediate procurement. The correct order is:

1. target compute, timing, and logging hardware;
2. propulsion bench instrumentation and safe power-handling equipment;
3. thermal and vibration sensing additions;
4. PRISM bench components and optical support equipment;
5. travel, demonstration consumables, and presentation assets.

This ordering is deliberate. Without a stable runtime and good logging, advanced sensing cannot be evaluated properly. The budget should reflect that dependency chain.

9.7 Contingency and Margin

A serious engineering plan includes contingency. The budget therefore reserves a modest contingency band for:

- failed or delayed hardware;

- extra instrumentation required after early HIL results;
- outside specialist consultation on sensing or signal integrity;
- additional range or facility access when the bench results justify it.

The contingency is not a luxury. It is what prevents the program from being derailed by the first missing part or unstable sensor channel.

9.8 Why the Budget Is Believable

The budget becomes believable when it matches the program's maturity. A pre-revenue startup claiming to solve interaction-aware flight control with an unrealistically tiny budget signals either naivety or lack of systems experience. By contrast, a focused six-month program in the low six-figure range, concentrated on data, control, HIL, and bounded sensing experiments, is consistent with similar early-stage autonomy and SBIR-scale efforts. It is ambitious but not inflated.

9.9 Chapter Takeaway

Phase I can be executed with a disciplined resource footprint if spending is concentrated on evidence quality, integration speed, and bounded sensing experiments. The next chapter explains how that technical program converts into a funding and commercialization pathway.

Chapter 10

Funding Pipeline & Commercialization Strategy

The commercial logic of this program is stronger if it is framed correctly. The initial product is not "an entire future passenger aircraft." The initial product is a disturbance-aware control, diagnostics, and sensing stack that is valuable in any electric aircraft or heavy-lift UAS where rotor interaction, channel count, and hover quality matter. The airframe remains strategically important because it proves the architecture in a differentiated geometry, but the earliest revenue and funding pathways should be attached to the software and health-monitoring capabilities.

10.1 Near-Term Funding Channels

The most credible public funding path is still the U.S. small-business innovation ecosystem. The exact solicitation windows move, but the stable channels are clear:

- NASA SBIR/STTR for advanced air mobility, controls, vehicle systems, and flight software transition [4];
- NSF America's Seed Fund for deep-tech autonomy, advanced sensing, and dual-use commercialization [6];
- AFWERX and defense-adjacent open-topic programs for resilient autonomous systems and logistics platforms;
- state innovation matching funds and aerospace cluster grants for hardware and lab build-out;
- strategic pilot funding from OEMs or operators who need better hover stability and health visibility.

The program should not depend on one source. The right pipeline is staggered:

1. use the white paper and HIL evidence to support one or two non-dilutive applications;

2. use the same evidence to secure a small paid pilot or development agreement;
3. use the resulting traction to support pre-seed or seed financing on better terms.

10.2 Commercial Beachheads

The clearest near-term markets are those that value stability and observability more than cabin comfort:

- agricultural input delivery and precision hover over structured terrain;
- industrial inspection where low-speed position control and vibration awareness matter;
- emergency or field logistics for moderate payloads in constrained landing zones;
- OEM integration for high-channel distributed propulsion prototypes.

These markets share a useful property: they do not require the company to win the entire eVTOL certification race before delivering value. A disturbance-aware control module can create immediate savings through steadier hover, better payload handling, and improved fault diagnosis.

10.3 Productization Ladder

The commercialization path is best seen as a ladder rather than a jump:

Level 1: Engineering software and HIL package. This is the earliest sellable form. It includes the allocator, DDRM, replay utilities, and configuration/export artifacts. The buyers are research labs, OEM skunkworks teams, and early strategic partners.

Level 2: Disturbance-aware avionics module. At this level the software is tied to a supported compute target, telemetry package, and integration toolkit. This is attractive to cargo and industrial UAS developers who need more than a generic autopilot but do not want to build a custom stack from scratch.

Level 3: ARW reference vehicle or kit. Only after the software and sensing story are validated should the company lean into airframe productization. At that point, the differentiated geometry becomes a multiplier rather than the sole thesis.

10.4 Competitive Positioning

Aerial MECHANICA can occupy a differentiated position if it avoids competing as "just another airframe company." The stronger position is:

- geometry-aware control and diagnostics for dense distributed propulsion;

- deterministic fault handling for high actuator-count aircraft;
- a path into advanced disturbance sensing that most competitors have not developed;
- reusable software value that can extend beyond one internal vehicle family.

That positioning is stronger in investor discussions because it suggests a product platform, not a single hardware bet.

10.5 Go-to-Market Motion

The go-to-market motion should be engineered, not purely sales-led.

1. publish a technically rigorous but commercially readable white paper;
2. demonstrate replay and HIL evidence with clear before-and-after allocator behavior;
3. engage one or two target customers with specific pain points such as hover quality or fault diagnosis;
4. structure the first paid engagement as an integration and evaluation package, not a broad custom development agreement;
5. convert successful pilots into annual support, software licensing, or hardware-in-the-loop subscription revenue.

This motion keeps the company close to real operators and reduces the risk of drifting into an elegant but unmarketed research program.

10.6 Revenue Logic

The near-term revenue thesis has three components:

- integration fees for deploying the allocator and diagnostics stack onto a partner platform;
- annual software or support licenses for maintenance of the runtime, replay tools, and health-monitoring logic;
- specialized sensing or anomaly-analysis packages for customers operating in electrically or mechanically harsh environments.

Longer term, if the ARW vehicle family matures, the company can add vehicle margin and fleet services. But the white paper intentionally avoids making the entire business case depend on future full-aircraft sales.

10.7 Partnership Model

The most attractive partners are those who can accelerate validation without forcing the company into pure contract engineering. The preferred sequence is:

1. university or lab partners who can strengthen sensing and replay rigor;
2. OEM prototype teams who need better control allocation and diagnostics on dense propulsion systems;
3. operator partners who can define real commercial pain points such as hover stability, payload precision, or false-abort cost;
4. capital partners who understand that the company is building a control platform with vehicle upside, not just a vehicle with software attached.

This sequence keeps the company close to real technical problems while preserving strategic control of the core architecture.

10.8 Regulatory Posture

The right regulatory posture is incremental and honest. The company should position the GNFCS first as a deterministic, inspectable control and diagnostics layer suitable for prototype and limited commercial UAS platforms. As the evidence base matures, the same design philosophy supports future certification arguments because it is traceable and modular. This posture is stronger than overclaiming a near-term passenger certification path.

10.9 Why the Story Works

The program works commercially because the same evidence package serves multiple purposes. The 2024 discrepancy data and Phase I HIL results support:

- technical credibility for grants and non-dilutive funding;
- customer proof for OEM partners;
- diligence material for investors;
- an internal design baseline for Phase II flight hardware.

This leverage is strategically valuable. It means each engineering dollar can feed funding, partnerships, productization, and future flight readiness.

10.10 Closing Position

The funding and commercialization plan is strongest when it stays aligned with the technical truth of the program. Aerial MECHANICA does not need to promise immediate passenger aircraft dominance. It needs to prove that it understands a hard control problem better than its peers, can instrument it, can model it, and can turn that understanding into products. That is already a compelling company.

Appendix A

Nomenclature, Assumptions, and Governing Equations

Primary Symbols

Symbol	Units	Meaning
T_j	N	Delivered thrust of rotor j
$T_{j,\text{ideal}}$	N	Ideal isolated-rotor thrust estimate
δ_j	–	Interaction-driven thrust discrepancy factor
s/D	–	Local rotor spacing ratio
ω_j	rad/s	Rotor angular speed
ϕ_j	rad	Relative rotor phase or azimuth alignment term
$\mathbf{x}$	mixed	Vehicle state vector
$\mathbf{u}_{\text{des}}$	mixed	Desired body wrench from guidance
B	mixed	Control effectiveness matrix
W	mixed	Allocator weight matrix
$\mathbf{r}_k$	mixed	Instantaneous model residual at time step k
$\mathbf{z}_k$	mixed	DDRM remapped residual features
y_{prism}	mixed	Differential PRISM output signal

Assumption Set Used in the White Paper

The main chapters intentionally use a bounded assumption set:

1. the semi-scale vehicle preserves the relevant interaction topology of the future 38-rotor concept even if Reynolds number and structural scaling differ;
2. reconstructed 2024 telemetry can be used for controller design trade studies when bounded by replay and consistency checks;

3. PRISM remains an optional sensing overlay during Phase I and does not determine minimum safe flight capability;
4. the reference architecture is evaluated primarily in hover, step climb, and low-speed translation, not in full transition or cruise;
5. embedded compute performance is judged against Jetson Orin-class hardware, not bespoke rack-scale processing.

Compact Governing Equation Set

The white paper repeatedly relies on the following compact system equations.

Interaction-aware thrust model

$$T_j = T_{j,\text{ideal}}\,(1-\delta_j)\,, \qquad \delta_j = f\left(\frac{s_j}{D_j}, \lambda_j, \phi_j, \eta_j, \Theta_j\right) \tag{A.1}$$

State propagation

$$\hat{\mathbf{x}}_{k+1}^{-} = f_{\text{dyn}}(\hat{\mathbf{x}}_k, \mathbf{u}_k, \hat{\boldsymbol{\kappa}}_k) \tag{A.2}$$

Allocation problem

$$\begin{aligned} \min_{\mathbf{T}} \quad & \frac{1}{2}\left(\mathbf{T}-\mathbf{T}_{\text{trim}}\right)^{\top} W \left(\mathbf{T}-\mathbf{T}_{\text{trim}}\right) \\ \text{s.t.} \quad & B(\hat{\boldsymbol{\kappa}})\mathbf{T} = \mathbf{u}_{\text{des}} \\ & T_{j,\min} \le T_j \le T_{j,\max} \end{aligned} \tag{A.3}$$

DDRM residual and remapping

$$\mathbf{r}_k = \mathbf{x}_k^{\text{measured}} - \hat{\mathbf{x}}_k^{\text{expected}}, \qquad \mathbf{z}_k = R\mathbf{r}_k \tag{A.4}$$

PRISM differential signal

$$y_{\text{prism}}(t) = [h_{\text{local}}(t) - h_{\text{ref}}(t)] * q(t) \tag{A.5}$$

Use Limitation

The equations above are controller-facing abstractions. They are appropriate for control design, replay, and anomaly detection within the stated envelope. They are not substitutes for blade-resolved CFD, structural certification analysis, or flight-test substantiation across the full aircraft operating envelope.

Appendix B

2024 Campaign Manifest and Reconstruction Rules

Representative Campaign Manifest

Run ID	Maneuver	Data quality	Notes
HVR-01 to HVR-08	Static hover hold	High	Used for trim bias and temperature-rise baselines
CLM-01 to CLM-06	Step climb	Medium-high	Strong collective transients, useful for thrust discrepancy shaping
LAT-01 to LAT-08	Lateral translation	Medium	Best source for interaction asymmetry and pitch/heave coupling
DRT-01 to DRT-05	Electrical disturbance injection	Medium	Used primarily for DDRM false-positive separation
DER-01 to DER-06	Cluster derate replay	Reconstructed	Normalized from partial logs and replay-constrained interpolation
THM-01 to THM-04	Sustained hover thermal rise	Medium	Supports thermal drift modeling and persistence thresholds

Reconstruction Rules

Representative reconstructed segments were generated only when all of the following conditions were met:

1. at least two synchronized supporting channels were present, for example RPM plus current or current plus inertial response;

2. the reconstructed segment remained within the maneuver family envelope already observed in intact logs;
3. the replayed segment did not violate vehicle timing, power, or attitude consistency checks;
4. the reconstructed value did not materially change the sign or qualitative interpretation of the underlying event.

Disclosure Normalization

Some prototype-specific values were normalized or rounded before inclusion in the white paper to avoid disclosing detailed proprietary geometry or operational settings. The following transformations were permitted:

- thrust fractions reported as normalized command and effective fractions rather than raw force;
- temperatures rounded to the nearest degree Celsius;
- time stamps generalized to maneuver-relative windows;
- cluster identifiers abstracted where exact hardware mapping was not necessary for interpretation.

Acceptance Gate for Reconstructed Data

Every reconstructed segment was required to pass the following gate before inclusion:

$$\epsilon_{\text{replay}} = \alpha\,\epsilon_{\text{attitude}} + \beta\,\epsilon_{\text{power}} + \gamma\,\epsilon_{\text{timing}} \leq \epsilon_{\max} \tag{B.1}$$

with analyst-selected weights α, β, γ chosen to emphasize attitude and power plausibility over cosmetic time-series smoothness.

Interpretation Guidance

The campaign dataset should be interpreted as an engineering corpus. It is strong enough to justify the architecture and work plan. It is not presented as the final word on scale-up, certification, or aircraft-level performance claims beyond the calibrated envelope.

Appendix C

Representative Data Tables and Spectral Summaries

Representative Hover Window

Time (s)	Cmd. thrust frac.	Eff. thrust frac.	Pitch rate (deg/s)	Bus voltage (V)	Motor temp. (°C)
0	0.62	0.58	0.3	50.7	33
10	0.63	0.58	0.6	50.4	35
20	0.63	0.57	0.8	50.1	37
30	0.64	0.57	1.1	49.8	39
40	0.64	0.56	1.4	49.6	41
50	0.64	0.56	1.6	49.5	43
60	0.65	0.57	1.2	49.4	45

Representative Lateral Translation Window

Time (s)	Cmd. thrust frac.	Eff. thrust frac.	Lateral vel. (m/s)	Pitch angle (deg)	DDRM score
0	0.66	0.60	0.0	0.4	0.22
2	0.67	0.59	0.8	1.0	0.35
4	0.68	0.58	1.5	1.6	0.48
6	0.69	0.57	1.7	2.1	0.63
8	0.68	0.58	1.6	1.8	0.59
10	0.67	0.59	1.0	1.1	0.41

Spectral Summary by Maneuver Family

Table C.3: Representative spectral content from reconstructed and replay-validated event windows.

Maneuver family	Dominant low band	Dominant high band	Interpretation
Static hover	2.6–3.4 Hz	11–13 Hz	Weak rigid-body mode with persistent propulsion harmonic
Step climb	3.1–3.8 Hz	12–14 Hz	Inflow transient couples into pitch/heave
Lateral translation	3.0–4.1 Hz	12–13 Hz	Asymmetric interaction and ring-induced flow distortion
Electrical injection	below 2 Hz	9–11 Hz and harmonics	Strong bus event signature with less rigid-body amplification

Data Usage Note

The tables in this appendix are representative slices only. They are sufficient to communicate the character of the dataset and to support the control design arguments made in the main body. They are not a substitute for the full internal log archive.

Appendix D

Illustrative Code Listings

This appendix includes compact reference implementations of two core ideas used throughout the white paper: discrepancy-model fitting and disturbance feature fusion. These are not production flight code. They are readable analytical baselines intended for review, replay, and early prototyping.

Interaction Model Fitting Example

Listing D.1: Representative interaction-model fitting workflow.

```python
from __future__ import annotations

from dataclasses import dataclass

import numpy as np
import pandas as pd
from sklearn.linear_model import Ridge
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import PolynomialFeatures, StandardScaler


@dataclass
class InteractionModel:
    pipeline: Pipeline

    def predict_gap(self, spacing_ratio, inflow_ratio, rpm, thermal_state):
        features = np.column_stack([spacing_ratio, inflow_ratio, rpm, thermal_state])
        return self.pipeline.predict(features)


def fit_interaction_model(frame: pd.DataFrame) -> InteractionModel:
    """
    Fit a compact discrepancy model for the controller-facing thrust gap:
```

```
        gap = commanded_thrust - effective_thrust

    Inputs are chosen to match quantities that are available online in the
    GNFCS runtime rather than idealized CFD-only variables.
    """
    required = {
        "spacing_ratio",
        "inflow_ratio",
        "rpm",
        "thermal_state",
        "commanded_thrust",
        "effective_thrust",
    }
    missing = required.difference(frame.columns)
    if missing:
        raise ValueError(f"missing columns: {sorted(missing)}")

    x = frame[["spacing_ratio", "inflow_ratio", "rpm", "thermal_state"]].to_numpy()
    y = (frame["commanded_thrust"] - frame["effective_thrust"]).to_numpy()

    pipeline = Pipeline(
        steps=[
            ("scale", StandardScaler()),
            ("poly", PolynomialFeatures(degree=2, include_bias=False)),
            ("ridge", Ridge(alpha=0.35)),
        ]
    )
    pipeline.fit(x, y)
    return InteractionModel(pipeline=pipeline)


if __name__ == "__main__":
    sample = pd.DataFrame(
        {
            "spacing_ratio": [0.62, 0.62, 0.58, 0.58, 0.55, 0.55],
            "inflow_ratio": [0.08, 0.10, 0.11, 0.13, 0.14, 0.15],
            "rpm": [4300, 4500, 4700, 4900, 5000, 5150],
            "thermal_state": [0.20, 0.25, 0.34, 0.38, 0.45, 0.52],
            "commanded_thrust": [0.63, 0.65, 0.68, 0.70, 0.71, 0.73],
            "effective_thrust": [0.58, 0.59, 0.60, 0.61, 0.61, 0.62],
        }
    )
    model = fit_interaction_model(sample)
    predicted_gap = model.predict_gap(
        spacing_ratio=np.array([0.57]),
        inflow_ratio=np.array([0.14]),
        rpm=np.array([5000]),
        thermal_state=np.array([0.40]),
```

```
    )
    print(f"predicted thrust gap: {predicted_gap[0]:.3f}")
```

PRISM and DDRM Feature Fusion Example

Listing D.2: Representative deterministic feature fusion for PRISM-assisted disturbance labeling.

```
from __future__ import annotations

from dataclasses import dataclass

import numpy as np


@dataclass
class DisturbanceScores:
    aerodynamic: float
    electrical: float
    thermal: float


def normalize(vector: np.ndarray) -> np.ndarray:
    norm = np.linalg.norm(vector)
    return vector if norm == 0 else vector / norm


def compute_prism_feature(local_signal: np.ndarray, ref_signal: np.ndarray) -> float:
    differential = local_signal - ref_signal
    return float(np.sqrt(np.mean(differential**2)))


def fuse_features(
    imu_residual: np.ndarray,
    bus_features: np.ndarray,
    thermal_features: np.ndarray,
    prism_local: np.ndarray,
    prism_ref: np.ndarray,
) -> DisturbanceScores:
    prism_rms = compute_prism_feature(prism_local, prism_ref)

    rigid_energy = np.linalg.norm(imu_residual[:3])
    rate_energy = np.linalg.norm(imu_residual[3:])
    electrical_energy = np.linalg.norm(bus_features)
    thermal_energy = np.linalg.norm(thermal_features)

    aerodynamic_score = 0.45 * rigid_energy + 0.25 * rate_energy + 0.30 * prism_rms
    electrical_score = 0.55 * electrical_energy + 0.25 * prism_rms + 0.20 * rate_energy
```

```
    thermal_score = 0.60 * thermal_energy + 0.25 * electrical_energy + 0.15 * prism_rms

    raw = np.array([aerodynamic_score, electrical_score, thermal_score], dtype=float)
    scaled = normalize(raw)
    return DisturbanceScores(
        aerodynamic=float(scaled[0]),
        electrical=float(scaled[1]),
        thermal=float(scaled[2]),
    )


if __name__ == "__main__":
    imu = np.array([0.18, 0.11, 0.07, 0.03, 0.05, 0.04])
    bus = np.array([0.35, 0.28, 0.12])
    thermal = np.array([0.22, 0.16])
    local = np.array([0.32, 0.40, 0.35, 0.31])
    ref = np.array([0.12, 0.17, 0.15, 0.13])

    scores = fuse_features(imu, bus, thermal, local, ref)
    print(scores)
```

Appendix E

PRISM Bench Protocol and Calibration Worksheet

Bench Objective

The PRISM bench is intended to answer one question before any aircraft-level packaging effort begins: does a differential, phase-referenced disturbance channel materially improve attribution of aerodynamic, electrical, and thermal events relative to classical sensing alone?

Minimum Bench Configuration

- local disturbance-sensitive sensor near an energized propulsion cluster;
- reference sensor mounted on a quieter structural station or isolated fixture;
- synchronized ESC telemetry, IMU data, and bus-voltage capture;
- repeatable disturbance source for electrical and mechanical injection;
- timestamping path with deterministic alignment better than one control cycle.

Calibration Sequence

1. Record a no-load background trace for both local and reference channels.
2. Energize propulsion electronics without rotor thrust to capture pure electrical baseline.
3. Add controlled mechanical vibration at fixed frequencies to characterize common-mode rejection.
4. Run representative propulsion sweeps and note synchronization offsets.
5. Compute differential transfer characteristics and verify that the reference channel removes common background without erasing local disturbance signatures.

Acceptance Metrics

Table E.1: PRISM bench acceptance worksheet.

Metric	Target	Interpretation
Timestamp alignment error	< 5 ms	Required for meaningful fusion with the 200 Hz runtime
Common-mode rejection improvement	> 10 dB	Indicates the differential channel is doing useful work
Disturbance separation gain	Positive against baseline	PRISM must reduce confusion between electrical and aerodynamic signatures
Operational stability over 30 min	No drift-induced collapse	Bench utility is lost if re-zeroing is constantly required

Decision Gate

PRISM proceeds to tighter packaging only if it clears the following gate:

$$\Delta J = J_{\text{baseline confusion}} - J_{\text{with prism}} > 0 \tag{E.1}$$

where J is a weighted disturbance-classification cost that penalizes false actuator derates more heavily than missed diagnostic insight.

Interpretation

The bench protocol is designed to prevent wishful thinking. PRISM is valuable only if it changes engineering decisions for the better. If it produces elegant signals but no operational gain, it remains a research lane rather than a product lane.

Appendix F

Illustrative JSON Export Schema

The control and analysis stack should export a machine-readable package so partner simulators, internal tools, and future certification artifacts can consume the same configuration baseline. A representative schema is shown below.

Listing F.1: Illustrative export schema for geometry, control, and disturbance metadata.

```
{
  "program": {
    "name": "Aerial MECHANICA GNFCS",
    "version": "phase1-r2",
    "generated_at": "2026-03-28T12:00:00Z"
  },
  "vehicle": {
    "configuration": "ARW-38",
    "rotor_count": 38,
    "ring_geometry": {
      "outer_diameter_m": 4.8,
      "inner_diameter_m": 3.9,
      "station_map_version": "geom-38-v3"
    }
  },
  "control": {
    "allocation_matrix_ref": "B_38_v7.npy",
    "trim_profile_ref": "trim_hover_v5.json",
    "interaction_model_ref": "delta_model_v4.json",
    "loop_rate_hz": 200
  },
  "ddrm": {
    "feature_schema": "ddrm_features_v2",
    "threshold_set": "ddrm_thresholds_v5",
    "class_labels": [
      "aero_interaction",
      "electrical_disturbance",
      "thermal_drift",
```

```
      "actuator_inconsistency",
      "ambiguous"
    ]
  },
  "sensing": {
    "imu_schema": "imu_v2",
    "esc_schema": "esc_v3",
    "prism_enabled": true,
    "prism_schema": "prism_bench_v1"
  },
  "provenance": {
    "source_campaign": "2024_proto_campaign",
    "contains_reconstructed_segments": true,
    "reconstruction_notes_ref": "reconstruction_log_v3.md"
  }
}
```

Schema Design Rules

- separate geometry from control coefficients so one can evolve without invalidating the other;
- store schema version names explicitly rather than relying on filenames alone;
- preserve provenance of reconstructed data and derived coefficients;
- keep the export compact enough for repeated HIL and partner use.

Bibliography

[1] John F. Barry, Jennifer M. Schloss, Erik Bauch, Matthew J. Turner, Christian A. Hart, Linh M. Pham, and Ronald L. Walsworth. Sensitivity optimization for NV-diamond magnetometry. *Reviews of Modern Physics*, 92(1):015004, 2020.

[2] Carlos Campos, Richard Elvira, Juan J. Gómez Rodríguez, José M. M. Montiel, and Juan D. Tardós. ORB-SLAM3: An accurate open-source library for visual, visual-inertial, and multi-map SLAM. *IEEE Transactions on Robotics*, 37(6):1874–1890, 2021.

[3] Christian L. Degen, F. Reinhard, and Paola Cappellaro. Quantum sensing. *Reviews of Modern Physics*, 89(3):035002, 2017.

[4] NASA. Small business innovation research / small business technology transfer programs. https://www.nasa.gov/langley/frontdoor/sbir-sttr/, 2025. Accessed March 28, 2026.

[5] NASA Armstrong Flight Research Center. X-57 maxwell. https://www.nasa.gov/centers-and-facilities/armstrong/x-57-maxwell/, 2024. Accessed March 28, 2026.

[6] National Science Foundation. America's seed fund powered by NSF. https://seedfund.nsf.gov/, 2025. Accessed March 28, 2026.

[7] NVIDIA. Isaac sim documentation. https://docs.isaacsim.omniverse.nvidia.com/, 2026. Accessed March 28, 2026.

[8] NVIDIA. Jetson orin series technical specifications. https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-orin/, 2026. Accessed March 28, 2026.

[9] Open Robotics. ROS 2 humble documentation: Understanding real-time programming. https://ros2docs.robook.org/humble/Tutorials/Demos/Real-Time-Programming.html, 2024. Accessed March 28, 2026.

[10] PX4 Autopilot Project. Control allocation. https://docs.px4.io/main/en/concept/control_allocation.html, 2025. Accessed March 28, 2026.

[11] Enrico Daniel Richter, Ryan J. Smith, Brayden Glockzin, Emanuel Druga, Thomas Schenkel, and Ashok Ajoy. Robust quantum sensing via prethermal spin orbits. https://arxiv.org/abs/2603.21057, 2026. arXiv:2603.21057v1 [quant-ph], accessed March 29, 2026.

[12] Dan Wang, Qiang Du, Tong Zhou, Christos Bakalis, Derun Li, and Russell Wilcox. CALIPR: Coherent addition using learned interference pattern recognition. In *Laser Congress 2021 (ASSL, LAC), OSA Technical Digest*. Optica Publishing Group, 2021. Paper JM3A.22, doi:10.1364/ASSL.2021.JM3A.22.

[13] Dan Wang, Qiang Du, Tong Zhou, Antonio Gilardi, Mariam Kiran, Bashir Mohammed, Derun Li, and Russell Wilcox. Machine learning pattern recognition algorithm with applications to coherent laser combination. *IEEE Journal of Quantum Electronics*, 58(6):6100309, 2022.