International Research Journal on Advanced Engineering and Management
https://goldncloudpublications.com
https://doi.org/10.47392/IRJAEM.2026.0178
e ISSN: 2584-2854
Volume: 04 Issue: 05
May 2026
Page No: 1185 - 1190

Implementation and Validation of a Thrust-Mass-Acceleration Consistency Invariant for Quadcopter UAV Safety

Sivakumar Kaliyappan[1], Meenakshi Saravanan[2], Mirunalini Sureshkumar[3], Mahalakshmi Lakashmanan[4]
1Assistant professor, Dept. of ECE, Panimalar Engineering College, Chennai, Tamil Nadu, India
[2,3,4]UG Scholar, Dept. of ECE, Panimalar Engineering College, Chennai, Tamil Nadu, India
Emails: *ksivakumar.ece@panimalar.ac.in[1], meenakshigsw30@gmail.com[2], mirunalinisuresh81@gmail.com3, mahalakshmilakshmanan2005@gmail.com[4]*

Abstract
This paper introduces a thrust-mass-acceleration consistency invariant, a physics-based supervisory validation technique that utilizes Newton's Second Law to detect discrepancies in vertical thrust, mass, and acceleration during autonomous aerial vehicle operation. As most flight stacks currently utilize sensors and software to manage their operations, they generally assume that all sensor data received by the vehicle is accurate and trustworthy. This invariant was developed to provide a lightweight supervisory monitoring system that does not require modification of the flight stack but can detect when there is a discrepancy in one or more of the physical variables required for safe vertical flight. It uses the PX4 SITL with ROS2 integration to implement its supervisory validation capability within the constraints of a strictly defined North-East-Down frame of reference. In contrast to many existing approaches, this invariant leaves the control logic unchanged. Therefore, this invariant provides a modular framework for developing supervisory safety architectures based upon invariant relationships. Simulation studies have demonstrated that this invariant can effectively detect faults in the vertical thrust generation system and inconsistent vertical motion, while restricting its scope strictly to vertical dynamic consistency.

Keywords: *Fault Detection; Physics-Based Invariant; Quadcopter UAV; Supervisory Control; ROS2.*

1. Introduction
Increasingly autonomous quadcopters are operating in contexts requiring high reliability and safety [5], [16], [18]. Current flight controllers extensively use sensor fusion, state estimation, and on-board control software to provide stable flight [9], [11], [12]. These are well-designed but assume that the sensor readings and actuator outputs are correlated to physical behaviors as assumed by a given model. But in the real-world conditions, motor fatigue, mass variation, actuator latency or sensor offset might cause small deviations that are difficult to detect without traditional fault flags.The current fault detection schemes can be complex, relying on the use of observers, redundancy or data driven models [3]. An observer or redundancy-based scheme will naturally add to the computational load, whereas controller modification will affect the control scheme itself. Therefore, there is a need for a less computationally demanding supervisory layer which provides a validation of the vehicle response against a set of physical rules. This work investigates such a mechanism, building up the invariant. This invariant is an observer that does not redesign the controller but checks if the vertical motion of the vehicle behaves according to the fundamental relationships between force and acceleration. Using the invariant check as a validation layer within a PX4 SITL and ROS2, this work investigates how invariant based supervision can enhance fault detection robustness while preserving modular controller architecture.

1.1. Thrust-Mass-Acceleration Invariant Modelling and Architecture
The thrust-mass-acceleration invariant is written as a physics-based supervisory condition to check the real-time vertical dynamic consistency. The supervisory condition acts as an observer on top of the flight controller, where it checks if there is a consistent relationship among the commanded thrust, vehicle mass and the vertical acceleration measured by sensors. As opposed to modifying the control law,

International Research Journal on Advanced Engineering and Management
https://goldncloudpublications.com
https://doi.org/10.47392/IRJAEM.2026.0178
e ISSN: 2584-2854
Volume: 04 Issue: 05
May 2026
Page No: 1185 - 1190

the supervisory condition monitors the vehicle's vertical response and flags sustained deviations from physically acceptable behaviour. This formulation establishes a straightforward relationship of cause and effect between the commanded actuator and the resultant motion that is used to develop the supervisory logic for this system in the next section.

- **Mathematical Formulation:**

Starting from Newton's Second Law in the inertial frame [2], [13]:

$$\sum F = ma$$

Under the PX4 North-East-Down convention, where the positive Z-axis is downward and g=9.81" " m/s^2, All accelerations and forces are expressed in the PX4 North-East-Down frame, the vertical dynamics of the quadcopter can be written as:

$$ma_z = T_{total}cos\phi cos\theta - mg$$

Rearranging,

$$m(g + a_z) = T_{total}cos\phi cos\theta$$

To account for actuator lag, a conservative delay Δt=150" " ms is incorporated. The residual is therefore defined as:

$$r(t) = m\left(g + a_z(t)\right) - T_{total}(t - \Delta t)cos\phi(t)cos\theta(t)$$

Under nominal flight conditions, r(t)≈0. Persistent deviation indicates a violation of the invariant.

- **Thrust Modeling:** Since thrust is proportional to the square of motor speed [4], [14], [17], and motor speed relates approximately linearly to input command, thrust is modeled quadratically as:

$$T(u) = k_2u^2 + k_1u + k_0$$

The coefficients are obtained through SITL-based quadratic regression to prevent linear approximation errors.

- **Supervisory Logic and Threshold Design**

An error threshold of, ε is selected based on combined uncertainties from IMU noise, thrust modeling error, mass variation, and drag effects. The value corresponds to a conservative bound exceeding the empirically observed 3σ residual during nominal hover conditions. To avoid false triggers from transient disturbances, violations must persist for a time window:

$$\tau = 200\,ms$$

At 50 Hz update rate, this corresponds to 10 consecutive violations.

- **Architectural Integration**

The Invariant is built into PX4 SITL as an independent ROS2 Supervisor Node that will monitor for violations of the physical consistency constraint in real time. This supervisor does not modify inner-loop control signals. Upon persistent violation, it issues a high-level mode transition command.
Therefore, this design enables real time monitoring of the physical constraints that define consistency, while also ensuring that the flight control system remains stable.

2. Implementation Framework

The invariant is implemented within a Software-In-The-Loop environment using the PX4 flight stack [9] in Gazebo simulation, integrated with a ROS2 Humble supervisory node. The implementation is intentionally designed to remain external to the primary flight controller, ensuring that the invariant operates as a monitoring layer rather than a control modification.

2.1. System Integration

The supervisory node subscribes to important PX4 topics needed in invariant evaluation: Thrust command input to estimate total thrust. All signals are interpreted according to the PX4 North-East-Down (NED) coordinate convention, in which the positive Z-axis is directed downward and gravity acts along the positive Z-axis. The supervisory node calculates the residual at every control update. To achieve this delay a circular buffer is used to store the thrust command. The residual makes the comparison between the present acceleration and delayed thrust to avoid false infractions in quick throttle switches. At each update cycle, the residual is evaluated in real time.

$$r(t) = m(g + az(t)) - Ttotal(t - \Delta t)cos\phi(t)cos\theta(t)$$

2.2. Supervisory Decision Logic

A violation flag will be issued if the residual

International Research Journal on Advanced Engineering and Management
https://goldncloudpublications.com
https://doi.org/10.47392/IRJAEM.2026.0178
e ISSN: 2584-2854
Volume: 04 Issue: 05
May 2026
Page No: 1185 - 1190

magnitude exceeds the pre-defined threshold ε for a time greater than τ. The reason for this decision is to prevent possible integrator wind-up or conflict in controllers by instead switching the vehicle into a controlled flight mode using ROS2.

2.3. Safety Gating Conditions

In order to reduce false positives, this invariant will only operate when roll and pitch angles remain within ± 45° and vertical velocity remains within acceptable ranges. Additionally, the system will check estimator health flags before performing an invariant evaluation. If the estimator has reduced confidence, the safety threshold will temporarily widen to minimize transient false positive alarms. The design choice described above will ensure that the invariant maintains a level of modularity, consistency of frame, and compatibility with current PX4 control architecture and will require no invasive changes to the existing architecture.

3. Experimental Validation Using Gazebo

The effectiveness of the invariant will be evaluated through controlled simulation experiments by means of the PX4 SITL environment linked to Gazebo, as a way to check if the residual-based formulation is able to identify physically anomalous behaviors, without triggering normal flight dynamics.

3.1. Simulation Configuration

The quadcopter is run within Gazebo with the whole rigid body dynamics and gravitational effects activated. PX4 is responsible for the flight control logic [9], [15]; on the other hand, the supervisory computations are run remotely via a ROS2 node [8]. All the signals needed for evaluation purposes - namely vertical acceleration, angle of rotation, and thrust commands are obtained directly from the state of the simulated vehicle. The evaluation frequency is set, and an actuator delay compensation has been realized via a circular buffer so that the commanded thrust can always be consistent with the measure of the acceleration.

3.2. Nominal Flight Characterization

Prior to fault injection, a set of steady state hover and low climb tests are run on Gazebo, and the residual statistics are collected to understand its steady state mean and variance. From this we develop an expected envelope and check that our threshold isn't triggered by normal operation.

3.3. Fault Injection Strategy

To test detection capability, controlled disturbances are introduced in the Gazebo simulation:

- Motor wear simulation through gradual decrease of thrust [1].
- Sudden drop in thrust that occurs in a propeller damage scenario represents the false bias that is added vertically [4&7].
- Fast thrust transients for delay compensation measurement.
- Every injection condition is held for a sufficiently long time for it to pass the given time window and assess the temporal filtering of the invariant.

3.4. Performance Metrics

Performance of the System will be evaluated by the following performance metrics:

- The amount of time it takes to determine whether an error has occurred.
- Absence of false alarms during normal operation.
- Stability of detection during aggressive maneuvers.
- Smooth transition back to normal detection once normal operating conditions are restored.

Detection is considered successful if there exists a period where the residuals exceed the thresholds [5], causing a consistent violation flag without resulting in oscillating triggers, this method of validation provides assurance that the invariant is functioning as a reliable supervisor to the monitoring system, in the Gazebo simulation environment, while maintaining appropriate levels of sensitivity to true faults and resistance to transients.

4. Simulation Methodology

4.1. Environment Setup

The invariant supervisor has been implemented as part of a PX4 SITL environment, which includes a Gazebo-Classic simulation environment [10]. In this environment, PX4 is used for flight control, and a ROS 2 node is used to perform supervisory monitoring [10]. A Micro-XRCE-DDS bridge provides communication at a rate of 50 Hz and enables consistent telemetry updates.

International Research Journal on Advanced Engineering and Management
https://goldncloudpublications.com
https://doi.org/10.47392/IRJAEM.2026.0178
e ISSN: 2584-2854
Volume: 04 Issue: 05
May 2026
Page No: 1185 - 1190

4.2. Node Implementation

To enable data consistency with the PX4 estimator, all the data was processed in the North-East-Down coordinate system. Calibrations can be enabled on the vehicle once it reaches a height of 2.0 meters to avoid any disturbances caused by the ground effect, the node follows a four-state sequence: take off, calibrating, landing, and processing. These states provide repeatable and controlled data acquisition processes.

4.3. Baseline Calibration

The vehicle was able to execute an automated throttle sweep from 0.1 to 1.1 at a constant height of 3.0 meters. The vertical acceleration measurements were very close to the expected gravity measurement in NED and exhibited only minor deviations that were below tolerance limits shown in Figure 1.

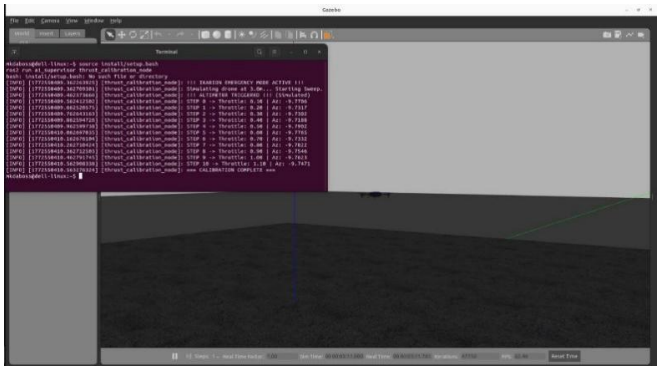

Figure 1 Real-time SITL Execution of the Thrust Calibration Protocol

4.4. Thrust Model Characterization

A quadratic thrust model, was developed based upon the acquired data [5]. The model had an R^2 value of 0.989, which represents the degree of correlation between measured and predicted thrust values. The actuator exhibited smooth and consistent force response across the range of throttle inputs.

$$T(u) = k_2u^2 + k_1u + k_0$$

4.5. Residual and Trigger Validation

The residual signal remained within a range of ±0.5 m/s² during steady state hovering conditions and was validated. A persistence filter was used to prevent false triggers due to transients in the residual signal [5]. Persistent residuals resulted in a correct triggering of severity responses shown in Table 1.

Step	Throttle	$A_z(m/s)^2$	Status
0	0.10	-9.7706	Hover Baseline
1	0.20	-9.7317	Steady
2	0.30	-9.7302	Steady
3	0.40	-9.7188	Steady
4	0.50	-9.7902	Noise Peak
5	0.60	-9.7765	Steady
6	0.70	-9.7332	Steady
7	0.80	-9.7822	Steady
8	0.90	-9.7546	Steady
9	1.00	-9.7623	Full Throttle
10	1.10	-9.7471	Limit Check

Table 1 Empirical Mapping of Motor Throttle to Vertical Acceleration (A_z) in PX4 SITL

5. Results and Discussion

5.1. Result

The experiments were planned to test how well the invariant can reliably track the vertical flight behavior in controlled simulation conditions. Structured hover tests, gradual throttle variations, and simulated fault scenarios were run in the PX4-Gazebo environment. It was found that the results were stable in normal flight and stable in detecting sustained inconsistencies in case of introducing abnormal thrust conditions. The supervisory layer was functioning well without interfering with the primary flight controller.

5.2. Discussion

The observation of the behavior of the supervisory framework validates the utility of such an approach for ensuring the safety of UAV flights. The ability of the invariant to provide such an outcome without requiring any complicated observer or controller redesign suggests that such an approach can encourage safety through relatively simple checks of the system's behavior. Additionally, because the

International Research Journal on Advanced Engineering and Management
https://goldncloudpublications.com
https://doi.org/10.47392/IRJAEM.2026.0178
e ISSN: 2584-2854
Volume: 04 Issue: 05
May 2026
Page No: 1185 - 1190

framework's performance was entirely independent of the controllers themselves, the finding also suggests that the incorporation of the fundamental physical models can enhance safety without risking any undesired interactions with those control architectures. Thus, this outcome reinforces the notion that the safety of autonomous systems can be improved through safety checks that are based upon the fundamental principles of the system's dynamics. [12], [16], [18]

Conclusion

The thrust-mass-acceleration invariant is written as a physics-based supervisory condition to check the real-time vertical dynamic consistency. The supervisory condition acts as an observer on top of the flight controller, where it checks if there is a consistent relationship among the commanded thrust, vehicle mass, and the vertical acceleration measured by sensors. As opposed to modifying the control law, the supervisory condition monitors the vehicle's vertical response and flags sustained deviations from physically acceptable behavior. Therefore, this formulation establishes a direct, understandable relation between the commanded actuation and the observed motion.

References

[1]. Zhou, L., Jin, H., Chen, P., Zhang, X., & Liu, Y. (2025). Actuator fault detection method of quadcopter UAV based on dual-channel inertial sensors. Discover Applied Sciences, 7, 736. doi: 10.1007/s42452-025-07403-5.

[2]. Wu, Y., Ling, G., & Shi, Y. (2025). Robust trajectory tracking fault-tolerant control for quadcopter UAVs based on adaptive sliding mode and fault estimation. Computation, 13(7), 162. doi: 10.3390/computation13070162.

[3]. Zhang, D., Meng, L., & Hao, Y. (2025). Adaptive sliding mode fault-tolerant control of UAV systems based on radial basis function neural networks. Scientific Reports, 15, 27504. doi: 10.1038/s41598-025-13659-z.

[4]. Simha, A., & Ambroziak, L. (2025). Nonlinear control of quadcopter UAV under rotor failure for robust trajectory tracking. Scientific Reports, 15, 42243. doi: 10.1038/s41598-025-26264-x.

[5]. Mazare, M., Taghizadeh, M., Ghaf-Ghanbari, P., & Davoodi, E. (2024). Robust fault detection and adaptive fixed-time fault-tolerant control for quadcopter UAVs. Robotics and Autonomous Systems, 179, 104747. doi: 10.1016/j.robot.2024.104747.

[6]. Wu, Y. (2025). Adaptive finite-time fault-tolerant control scheme of UAV against combined faults. Scientific Reports. doi: 10.1038/s41598-025-31619-5.

[7]. Zaludin, Z. A. (2023). Fault tolerance conceptual strategy for a quadcopter drone with rotor failure. Asian Review of Mechanical Engineering, 12(2), 1-8. doi: 10.51983/arme-2023.12.2.3849.

[8]. Alqaisi, W., Soliman, M., Badawi, A., & Youssef, M. (2025). Detection and tracking quadcopter using SURF and feedback linearization sliding mode control. Robomech Journal, 12, 36. doi: 10.1186/s40648-025-00312-7.

[9]. Jing, Y., Wang, X., Heredia-Juesas, J., Fortner, C., Giacomo, C., Sipahi, R., & Martinez-Lorenzo, J. (2022). PX4 simulation results of a quadcopter with a disturbance-observer-based and PSO-optimized sliding mode surface controller. Drones, 6(9), 261. doi: 10.3390/drones6090261.

[10]. Macenski, S., Foote, T., Gerkey, B., Lalancette, C., & Woodall, W. (2022). Robot Operating System 2: Design, architecture, and uses in the wild. Science Robotics, 7(66), eabm6074. doi: 10.1126/scirobotics. abm6074.

[11]. Sabatini, A. M., & Genovese, V. (2013). A sensor fusion method for tracking vertical motion in quadrotor UAVs using inertial measurements. IEEE Sensors Journal, 13(9), 3512–3521.

[12]. Kendoul, F. (2012). Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems. Journal of Field Robotics, 29(2), 315–378.

[13]. Bouabdallah, S., Murrieri, P., & Siegwart,

International Research Journal on Advanced Engineering and Management
https://goldncloudpublications.com
https://doi.org/10.47392/IRJAEM.2026.0178
e ISSN: 2584-2854
Volume: 04 Issue: 05
May 2026
Page No: 1185 - 1190

R. (2004). Design and control of an indoor micro quadrotor. Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 4393–4398.

[14]. Hoffmann, G. M., Huang, H., Waslander, S. L., & Tomlin, C. J. (2007). Quadrotor helicopter flight dynamics and control. AIAA Guidance, Navigation and Control Conference.

[15]. Castillo, P., Lozano, R., & Dzul, A. (2005). Modelling and control of mini-flying machines. Springer Tracts in Advanced Robotics.

[16]. Mahony, R., Kumar, V., & Corke, P. (2012). Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor. IEEE Robotics & Automation Magazine, 19(3), 20–32. doi:10.1109/MRA.2012.2206474.

[17]. Pounds, P., Mahony, R., & Corke, P. (2010). Modelling and control of a quad-rotor robot. Control Engineering Practice, 18(7), 691–699. doi: 10.1016/j.conengprac.2010.01.001.

[18]. Saied, M., Lussier, B., Fantoni, I., Francis, C., & Shraim, H. (2016). Fault diagnosis and fault-tolerant control of UAVs: A review. Annual Reviews in Control, 42, 226–241. doi: 10.1016/j.arcontrol.2016.09.008.

[19]. Bouabdallah, S., & Siegwart, R. (2007). Full control of a quadrotor. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 153–158.

[20]. Ryll, M., Bulthoff, H. H., & Giordano, P. R. (2015). Modeling and control of a quadrotor UAV with tilting propellers. IEEE International Conference on Robotics and Automation (ICRA), 4606–4613.