Journal of Intelligent & Robotic Systems (2026) 112:28
https://doi.org/10.1007/s10846-026-02369-x

REGULAR PAPER

Propeller Fault Detection and Isolation for Multirotor Drones with Adaptation to Battery Voltage Drop

Alessandro Baldini[1] · **Riccardo Felicetti**[1] · **Francesco Ferracuti**[1] · **Alessandro Freddi**[1] · **Andrea Monteriù**[1]

Received: 5 November 2024 / Accepted: 5 February 2026

Abstract

This paper addresses the problem of propeller fault detection and isolation in multirotor aerial vehicles using inertial data, explicitly accounting for the impact of battery voltage drop to ensure reliable residual generation. A complete mathematical model is presented, including the vehicle's kinematics, dynamics, and powertrain. From this model, an experimentally fitted static powertrain model is developed, which encompasses PWM commands, supply voltage, and blade faults. This model enables effective estimation of the lift force by incorporating battery voltage measurements, which is then used by a bank of observers designed for actuator fault detection and isolation. The resulting residuals are fed to a lightweight neural network classifier, achieving 95.04% fault isolation accuracy despite considering small faults (starting from a 5% reduction in one propeller blade length), varying operating conditions, sensor noise, and model mismatches. The proposed method is validated through Monte Carlo simulations, and its real-time feasibility is demonstrated using processor in the loop experiments on a standard flight controller.

Keywords Fault detection and Isolation · Unmanned aerial vehicles

1 Introduction

Multirotors represent a versatile class of Unmanned Aerial Vehicles (UAVs) which are commonly employed in many application scenarios, such as aerial exploration [1], agriculture [2], load [3] and passenger transportation [4]. Compared to helicopters, which possess similar properties such as the capability to hover and to perform vertical take off and landing, multirotors are characterized by a simpler construction and a lower cost [5]. However, multirotor UAVs have several limitations, such as low energy autonomy, especially if compared to fixed wing UAVs [6]. Also, the presence of several actuators makes them prone to actuator faults, presenting significant challenges to achieve reliability in autonomous flights. Nevertheless, redundant configurations to make multirotor UAVs fault tolerant are investigated in the literature [7].

Each component of the powertrain is a potential source of faults. Unlike other UAVs, multirotors (including large scale ones) typically feature an electric propulsion system. Their powertrain consists of a battery, Electronic Speed Controllers (ESCs), brushless DC motors, and propellers. Faults affecting the ESC or the DC motor usually result in an outage of the affected rotor. Hence, Fault Detection and Isolation (FDI) is well established in these cases [8], and faults are tackled in flight using either reconfiguration strategies [9], when redundancy can be exploited, or dedicated strategies for emergency landing [8], also including drone parachutes.

Lithium Polymery (LiPo) batteries represent the standard in commercial UAVs, and their limited energy density constrains the UAVs' energy autonomy. The output voltage of a LiPo battery is far from being constant over time; it can drop significantly due to factors like its State Of Charge (SoC), output current, and health status [10, 11]. The UAV's

✉ Riccardo Felicetti
r.felicetti@univpm.it

Alessandro Baldini
a.baldini@univpm.it

Francesco Ferracuti
f.ferracuti@univpm.it

Alessandro Freddi
a.freddi@univpm.it

Andrea Monteriù
a.monteriu@univpm.it

[1] Department of Information Engineering, Università Politecnica delle Marche, Via Brecce Bianche, Ancona 60131, Italy

responsiveness is affected as a consequence, as battery output voltage directly influences the rotational speed achievable by the propeller at a given control command (e.g., Pulse Width Modulation (PWM) duty cycle).

In addition, multirotor UAVs are acknowledged to suffer a high loss rate [12], mostly related to equipment issues [13], and one of the most common faults consists in a propeller damage [14]. The propellers can experience impacts with obstacles, such as birds, other vehicles, or the ground during the landing phase in adverse conditions, and the outcome is a blade damage of variable severity.

As a result of such uncertainties, non-adaptive control laws fail to function properly, unless they are specifically tuned for different operating conditions [15]. Though, multirotors are often controlled by using PID regulators, and one of their greatest benefits is their tolerance to small variations in the DC gain [16] of the actuators, mainly thanks to the integral term. Such robustness allows for normal flight even when the battery voltage is low and/or when one or more blades experience a loss of effectiveness due to small damages. This is the main reason why the time-varying nature of the battery output voltage is usually neglected for the purpose of control design for UAVs.

Even though the UAV can continue to fly with minor damages, detecting and isolating actuator faults is crucial for safety-critical multirotor applications, such as passenger drones. This approach can enhance safety and integrity, which are at risk in the event of defective blades. Such properties are central to civil aviation, where strict airworthiness certifications are required [17], primarily to eliminate harm to those onboard the aircraft. Even for UAVs, airworthiness regulations are oriented toward protecting people and property being overflown, as well as other airspace users within the operational environment.

Detecting and isolating a blade damage during a flight in real conditions is challenging due to the many sources of uncertainty, including time-varying voltage supply, individual differences in the motors, unknown parameters, etc. Moreover, in the case of redundant actuators, the control effort is distributed among many actuators, potentially masking minor blade defects. While redundancy increases robustness, it also makes FDI harder.

Unlike the flight electronics, which are powered through regulated voltages, the motors are directly connected to the LiPo battery via the ESCs. As a result, their supply voltage is not stabilized and decreases during flight, leading to reduced motor speed and rotor thrust. This behavior is typically accepted, as voltage regulation for motors would add inefficiency or limit performance. However, this introduces significant challenges for FDI, as the motor behavior becomes tightly coupled to the varying battery voltage. To investigate these main sources of uncertainty, in our previous work [18] we proposed a full characterization of both lift force and rotor speed vs. control command (i.e., the PWM duty cycle) at different voltage levels, due to battery discharge in a drone during flight, also taking into consideration faulty propellers. Such work aimed to fill a gap in the literature regarding the experimental evaluation of the impact of battery discharge and blade faults on rotor lift. In the pre-existing literature, the authors of [19] and [20] characterized multirotor UAVs propulsion systems through static maps, but they did not address battery discharge effects. The authors of [15] described the impact of reduced battery voltage on rotational speed and introduced an adaptive proportional derivative controller, but actual force and torque estimations were not provided. In [21], a black-box Neural Network (NN) model for battery discharge compensation was trained and, similarly, [22] employed fuzzy logic for battery voltage compensation, but they provided limited details on lift and rotor speed characterization. In [23, 24] thrust variation during battery discharge was analyzed, but under constant control commands only thus limiting its applicability. Finally, [25] characterized rotor speed versus battery voltage for the Qball-X4 UAV, but only one profile is reported, restricting replicability.

Therefore, the objective of this paper is to develop a model-based strategy to detect and isolate propeller faults in multirotor drones, requiring it to be robust by design to the battery voltage variations. The proposed method does not require any additional hardware, thus minimizing costs and payload, as it simply uses navigation data and battery voltage, which are always available as they enable the control of the UAV. To this aim, the experimental results from our work [18], related to the influence of battery voltage on motor dynamics, are exploited in order to enhance actuator FDI in multirotor drones. In fact, both mechanical faults and battery voltage dynamics affect the lift; taking them into account simultaneously allows for minimizing potential false alarms due to normal battery dynamics.

Some alternatives are available in the literature to detect and/or isolate actuator faults in multirotor drones. Two primary approaches are utilized in the literature for performing Fault Detection (FD) in UAVs: model-based and signal-based methods. Techniques such as Thau observers [26], linear and Linear Parameter Varying (LPV) proportional-integral observers [27], and banks of nonlinear observers [28] have been proposed in the scientific literature. These strategies extend beyond FD, allowing for FDI (and even fault estimation, in some favorable conditions) across various fault types. Nevertheless, model-based approaches demand an increased modeling effort and often face challenges due to the lack of availability of numerous parameters in practical applications. Signal-based strategies, instead, do not require complex system models or assumptions on the fault. Provided that a suitable amount of data is available, including all of the possible faults of practical interest, an Artificial Intelligence (AI) based FD system can be defined, which consists of signal-based methods for feature generation and a classification tool [29]. Many AI-based FD strategies for UAVs have been investigated in the last years: [30] proposed a random forest to detect sensor faults, [31] employed several variants of Kernel Principal Component Analysis to detect actuator and sensor faults, while [32] suggested to use deep NNs to detect and isolate an actuator failure. AI-based solutions show some drawbacks as well. First, they need a suitable amount of data from the specific UAV under consideration, including faulty data, that is not often available. Second, they are usually limited to FD, i.e., they are designed to provide alerts due to abnormal behaviour, but they generally lack the capability to isolate the faulty actuator. Third, many AI-based methods require a significant amount of computational power to run in real time, that is not natively available in commercial devices. Some notable exceptions have been proposed in the literature. In [14], data from additional sensors (microphones and additional IMUs) were exploited to train the reported NNs; FDI was performed in real-time, however, it came at the price of equipping the drone with extra sensors and microcontrollers, increasing its cost and its weight. In [33], a computationally cheap strategy for FD, based on vibration data, was proven to be feasible online on a conventional flight controller, however, it did not include fault isolation.

The main contributions of this work are:

1. Comprehensive powertrain modeling: We propose a detailed yet tractable grey-box model of the UAV powertrain, including the battery, ESCs, motors, and propellers. The model captures the impact of battery voltage variations (typically unregulated for motors) on thrust generation and is supported by experimental validation.
2. Effects of voltage drop and propeller faults: We analyze how both battery voltage drop and propeller blade faults affect motor speed, actuator lift, and drag, highlighting their possible implications for closed-loop control performance.
3. Real-time fault detection and isolation under uncertainty: We provide a method to detect and isolate faults on individual propellers despite modeling uncertainties and the time-varying nature of the motor supply voltage. The method is validated via Monte Carlo simulations and Processor-In-the-Loop (PIL) experiments on a conventional flight controller.

Compared to data-driven methods, this work leverages reliable residuals to enable lightweight fault detection and isolation, while avoiding methods that may be infeasible in real time on embedded hardware. Thanks to residuals that are largely independent of operating conditions, it can isolate faulty actuators using only inertial and voltage measurements. In contrast to existing model-based strategies, the key novelty lies in the ability to remain effective across a wide range of practical operating conditions, including battery discharge, which is often overlooked in the literature.

The paper is structured as follows. In Section 2, we detail the mathematical model of the multirotor and the powertrain. Then, in Section 3, we validate the powertrain model using experimental data. Section 4 formalizes the residual generator for FDI, which is implemented using a bank of observers. In Section 5, the residuals are analyzed, and residual evaluation policies are proposed. Section 6 presents the simulation results for a DJI F550 hexarotor during tracking, together with ablation studies, comparisons, and the Processor-In-the-Loop validation. Finally, Section 7 concludes the paper.

2 Mathematical Model

In this Section, we describe the mathematical model of a multirotor UAV and its powertrain, which consists of the battery, the ESCs, the DC motors, and the propellers. First of all, we briefly report the conventional kinematics and dynamics model, as well as the conventional lift and drag model from the literature. Then, we introduce the novel modeling part, which consists of a simplified representation of the battery dynamics, the ESCs, and the brushless DC motors. The final objective is to model the dependency of the motor speed and lift on the battery voltage, as well as on propeller faults.

2.1 Kinematics and Dynamics

The kinematics and dynamics of a typical multirotor can be modeled as those of a single rigid body with six degrees of freedom [7]. Consider a North-East-Down (NED) earth-fixed reference frame $R_E = (O_E, x_E, y_E, z_E)$, treated as inertial, and a body-fixed reference frame $R_B = (O_B, x_B, y_B, z_B)$, located at the UAV's center of mass. To simplify the analysis, we assume that the center of mass aligns with the geometric center of the multirotor. Let us define e_1, e_2, e_3 as the standard basis vectors of $\mathbb{R}^3$. Let also $p_F = \mathrm{col}(x_F, y_F, z_F)$ be the position of the center of mass with respect to R_E, and let $\omega = \mathrm{col}(p, q, r)$ be the UAV's angular velocity in R_B. Assuming that secondary effects (e.g., rotor dynamics, blade flapping, gyroscopic and inertial effects due to the rotors) can be neglected, the quadrotor kinematics and dynamics can be expressed as [7]

$$\begin{aligned} m\ddot{p}_F &= -k_t\dot{p}_F + mge_3 + RF_m^B \\ J\dot{\omega} &= -k_r\omega - \omega \times J\omega + \tau_m^B \\ \dot{\eta} &= T(\eta)\omega. \end{aligned} \tag{1}$$

In Eq. 1, m represents the total mass of the system, while $J = \mathrm{diag}(J_x, J_y, J_z)$ denotes the inertia tensor along the

x_B, y_B, and z_B axes. The parameters k_t and k_r are the linear and angular friction coefficients, respectively. The vector $\eta = \mathrm{col}(\varphi, \theta, \psi)$ contains the attitude angles (roll, pitch, and yaw) that define the rotation from R_B to R_E. Defining $c_{(\cdot)} = \cos(\cdot)$, $s_{(\cdot)} = \sin(\cdot)$, and $t_{(\cdot)} = \tan(\cdot)$ for brevity, the rotation matrix R is given by

$$R = \begin{bmatrix} c_\psi\, c_\theta & c_\psi\, s_\varphi\, s_\theta - c_\varphi\, s_\psi & s_\varphi\, s_\psi + c_\varphi\, c_\psi\, s_\theta \\ c_\theta\, s_\psi & c_\varphi\, c_\psi + s_\varphi\, s_\psi\, s_\theta & c_\varphi\, s_\psi\, s_\theta - c_\psi\, s_\varphi \\ -s_\theta & c_\theta\, s_\varphi & c_\varphi\, c_\theta \end{bmatrix}. \tag{2}$$

The terms F_m^B and τ_m^B represent the total force and torque generated by the motors (expressed in R_B). The matrix $T(\eta)$ corresponds to the kinematic coordinate transformation associated with the roll-pitch-yaw rotation, given by [34]

$$T(\eta) = \begin{bmatrix} 1 & s_\varphi t_\theta & c_\varphi t_\theta \\ 0 & c_\varphi & -s_\varphi \\ 0 & s_\varphi / c_\theta & c_\varphi / c_\theta \end{bmatrix}. \tag{3}$$

2.2 Propeller and Control Effectiveness

For the ith motor, let $f_i \in \mathbb{R}$ represent the magnitude of lift force, $d_i \in \mathbb{R}$ the corresponding drag magnitude, and $\Omega_i \in \mathbb{R}$ the angular velocity of the propeller. The lift force and drag generated by each rotor can be expressed as functions of the propeller's rotational speed [35]

$$f_i = c_{L_i} \Omega_i^2 \qquad\qquad d_i = c_{D_i} \Omega_i^2, \tag{4}$$

where c_{L_i} and c_{D_i} are the lift and drag coefficients, determined by the blade geometry. In-flight collisions can lead to propeller damage, such as blade edge wear or fractures [36]. Therefore, a damaged propeller has coefficients c_{L_i} and c_{D_i} that differ from their nominal values.

The total force and torque are given by

$$\begin{aligned} F_m^B &= -\sum_{i=1}^{n_a} f_i e_3 \\ \tau_m^B &= -\sum_{i=1}^{n_a} (l_i \times f_i e_3) - \sum_{i=1}^{n_a} (-1)^i \, d_i e_3, \end{aligned} \tag{5}$$

where $l_i \in \mathbb{R}^3$ is the position of the motor in the body frame, $\times$ denotes the cross product, and $(-1)^i$ accounts for the alternate motor rotation directions. Denoting with $f = col(f_1, \ldots, f_{n_a})$ the vector composed by the lift forces, it is possible to express the linear relation

$$\begin{bmatrix} F_m^B \\ \tau_m^B \end{bmatrix} = F f, \tag{6}$$

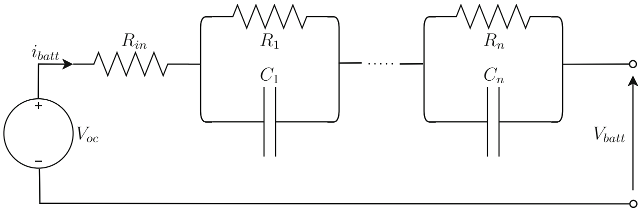

Fig. 1 Battery model with model order n

where $F \in \mathbb{R}^{6 \times n_a}$ denotes the control effectiveness matrix, directly obtained from Eq. 5.

2.3 Battery

The output voltage of a LiPo battery at its terminals $V_{batt}(t)$ mainly depends on the battery SoC and the output current $i_{batt}(t)$, and a well known static map [10]

$$V_{oc}(t) = \phi_{ocv}^{soc}(SoC(t)) \tag{7}$$

relates the SoC and the Open Circuit Voltage (OCV), denoted hereafter by $V_{oc}(t)$. A simple equivalent circuit model from the literature [37] is shown in Fig. 1, where R_{in} models the series internal resistance and R_i and C_i are additional internal resistor-capacitor pairs for $i = 1, \ldots, n$, where n is the model order. The current i_{batt} is assumed to be positive during discharge. For simplicity, we model the battery dynamics as

$$V_{batt}(t) = V_{oc}(t) - R_{in} i_{batt}(t) \tag{8}$$

$$SoC(t) = SoC(t_0) - \frac{1}{3600 Q_{nom}} \int_{t_0}^{t} i_{batt}(\tau) d\tau, \tag{9}$$

where Eq. 8 is a crude (0 order) equivalent battery model [10], Eq. 9 performs Coulomb counting to track the battery SoC, Q_{nom} is the nominal battery capacity in Ah, 3600 is a unit-conversion factor from Ah to C, and the resulting $V_{oc}(t)$ is a static function of $SoC(t)$ (see [10]).

Please note that $V_{oc}(t)$ cannot be measured during a flight. In fact, multirotors continuously draw a significant amount of current to counteract the gravity force, while measuring the $V_{oc}(t)$ requires to disconnect the loads for a certain time. On the contrary, $V_{batt}(t)$ is available and measured.

2.4 Electronic Speed Controller

The ESCs convert the battery's DC voltage into a three-phase voltage to drive the brushless DC motors through high-frequency electronic switching. In the literature, the ESC model is usually static, mainly focusing on power efficiency due to conduction losses, switching losses, and standby power draw [38]. The control input to the ESC is typically a PWM signal, whose duty cycle u_i determines the average voltage delivered to the ith motor and thus its lift force. In the practice, the actual control input commanded by the control law is the PWM duty cycle $u = \mathrm{col}(u_1, \ldots, u_{n_a})$, where each u_i is constrained to reside in [0, 1] due to saturation limits. Since the PWM frequency operates in the tens of kHz range, the associated delay is on the order of milliseconds and is therefore commonly neglected.

In the literature, models for ESC efficiency are typically based on empirical fitting. In [38], the authors adopt a grey box rational approximation, involving the battery voltage, the internal ESC resistance, the duty cycle, and the switching frequency. The authors of [39], instead, use an affine model of just the duty cycle and the input current. Neglecting the switching time and the idle power draw for simplicity, the ESC model can be derived from [38] as

$$V_{esc,i} = u_i (V_{batt} - R_{esc,i} I_i), \tag{10}$$

where $V_{esc,i}$ represents the output voltage provided by the ith ESC, V_{batt} is the input voltage from the battery, u_i is the duty cycle, $R_{esc,i}$ models the internal resistance of the MOSFETs, and I_i is the current.

2.5 DC Motor

The model of DC motors is well established in the literature [40]. The electrical dynamics are modeled by:

$$V_{esc,i} = R_{mot,i} I_i + K_{e,i} \Omega_i + L_{mot,i} \dot{I}_i, \tag{11}$$

where $V_{esc,i}$ is the input voltage to the ith motor, I_i the current, $R_{mot,i}$ the resistance, $L_{mot,i}$ the inductance, Ω_i the angular speed, and $K_{e,i}$ the back-EMF constant.

The mechanical dynamics follow:

$$J_i \dot{\Omega}_i = K_{t,i} (I_i - I_{0,i}) - B_{mot,i} \Omega_i - d_i, \tag{12}$$

where J_i is the inertia, $B_{mot,i}$ the viscous friction coefficient, $K_{t,i} = K_{e,i}$ the torque constant, $I_{0,i}$ the idle current, and d_i the load torque. The load torque in multirotor applications is modeled as $d_i = c_{D_i} \Omega_i^2$ [35] and the friction is usually neglected, as $B_{mot,i} \Omega_i$ is negligible with respect to d_i.

In low-frequency applications such as UAV flight, where the electrical time constant $\tau_{e,i} = L_{mot,i}/R_{mot,i}$ is on the order of microseconds, the term $L_{mot,i} \dot{I}_i$ is usually neglected [40]. This corresponds to assuming that the current I_i in Eq. 11 reaches steady state instantaneously compared to the slower mechanical dynamics in Eq. 12. Therefore, from Eq. 11 we obtain

$$I_i = \frac{V_{esc,i} - K_{e,i} \Omega_i}{R_{mot,i}}, \tag{13}$$

which can be substituted in Eq. 12, obtaining the full model:

$$\dot{\Omega}_i = \frac{1}{J_i} \left[K_{e,i} \left(\frac{V_{esc,i} - K_{e,i} \Omega_i}{R_{mot,i}} - I_{0,i} \right) - c_{D_i} \Omega_i^2 \right]. \tag{14}$$

2.6 Static Speed and Lift Models

From Eq. 14, it follows that the steady-state relation between Ω_i and V_{batt} is

$$c_{D_i} \Omega_i^2 + \left(\frac{K_{e,i}^2}{R_{mot,i}} \right) \Omega_i + \left(K_{e,i} I_{0,i} - \frac{K_{e,i} V_{esc,i}}{R_{mot,i}} \right) = 0. \tag{15}$$

Please note that $I_{0,i} < V_{esc,i}/R_{mot,i}$ is guaranteed under normal operating conditions due to the dead zone commonly present in these motors, so Eq. 15 yields one positive and one negative solution for Ω_i, of which only the positive one is physically meaningful, i.e.,

$$\Omega_i = \frac{1}{2c_{D_i}} \left[-\frac{K_{e,i}^2}{R_{mot,i}} + \sqrt{\left(\frac{K_{e,i}^2}{R_{mot,i}} \right)^2 + 4c_{D_i} \left(\frac{K_{e,i} u_i (V_{batt} - R_{esc,i} I_i)}{R_{mot,i}} - K_{e,i} I_{0,i} \right)} \right]. \tag{16}$$

The quantities related to the motor ($K_{e,i}$, $R_{mot,i}$, and $I_{0,i}$) are usually reported in the datasheet and can be assumed constant, as well as $R_{esc,i}$. In fact, possible faults affecting the motor or the ESC are typically catastrophic, resulting in a complete loss of thrust. As such, they are easy to detect and are of limited relevance to the present work on fault detection and isolation. Nonetheless, deviations between individual motors and their nominal specifications may be significant and should not be neglected. Also, I_i is linearly dependent on u_i [38], and can therefore be eliminated by substitution.

Conversely, the remaining variables are potentially time varying: u_i varies during the flight as it represents a control input, while c_{D_i} and V_{batt} are related to propeller faults and battery discharge, respectively. Hence, Eq. 16 represents a static relation $g_{speed,i}(\cdot)$ between each duty cycle u_i, the battery voltage V_{batt}, the motor speed in steady state Ω_i, and the drag coefficient c_{D_i}

$$\Omega_i = g_{speed,i}(u_i, c_{D_i}, V_{batt}). \tag{17}$$

The motor speed Ω_i is positively correlated with V_{batt}, and a square-root dependence is evident in Eq. 16. Also, from $f_i = c_{L_i} \Omega_i^2$, we can infer that the lift force is positively correlated with V_{batt} as well, and the linear dependency on V_{batt} dominates:

$$f_i = g_{lift,i}(u_i, c_{D_i}, V_{batt}). \tag{18}$$

The models in Eqs. 16–18 are parameterized with respect to the battery voltage and refer to a single motor. As a result, the model remains valid for multirotors of different sizes and shape, as it explicitly accounts for battery variability (e.g., number of LiPo cells in series), while being independent of the number of motors.

Considering (16)–(18) for compensating battery voltage is impractical, because it requires precise knowledge of many parameters, including c_{L_i} and c_{D_i}, which can be affected by unknown faults. Therefore, in [18], we have adopted a grey box approximated model. In [18], we have shown that there exist five parameters a_{L0}, a_{L1}, a_{L2}, a_{L3}, and v_{L0} such that the fractional order polynomial

$$f_i = (V_{batt} - v_{L0}) \cdot \left(a_{L0}\sqrt{u_i} + a_{L1}u_i + a_{L2}u_i^2 + a_{L3}u_i^3\right) \tag{19}$$

models accurately the lift force. Note that the linear dependency of f_i on V_{batt} is consistent with the previous considerations. The identification of these parameters can be performed using data from a single flight in which the battery discharges over its full usable range. Since the equation is used for battery voltage compensation, only data from healthy propellers is required. These parameters are expected to remain constant, provided that no faults occur. In the following Section, we address the validation of the model, taking into account battery discharge and propeller faults.

3 Model Validation

In this Section, we validate the mathematical model of the UAV, focusing on the battery model and the dependency of the motor speed and lift on the battery model, as well as on propeller faults.

3.1 Experimental Setup

To acquire voltage, speed, and lift measurements, a T-Motor Air Gear 350 outrunner motor, together with the T-Motor T9545 self-tightening propellers, was installed on a support, leaving 40cm from the worktable to limit the flow interaction with the surface (see Fig. 2). The motor was fed by a Tekko Holybro D-Shot 125 ESC that also provides motor speed measurements. Such devices are commonly paired with lightweight commercial UAVs. A strain gauge load cell [41] was employed for measuring the force exerted by the motor. The measured quantity from the load cell was acquired using a Seneca Z-Sg converter [42], together with a low cost Data AcQuisition (DAQ) USB Device [43] from National Instruments. We employed a 4 cells 6750 mAh LiPo battery with a maximum C-rate of 25 C, restricting our analysis to a SoC within a safe range from 40% to 95% (i.e., OCV from 15.2 V to 16.6V). Please note that we assume the battery state of health is constant for simplicity, meaning the battery aging can be neglected. In fact, battery aging typically requires hundreds of battery cycles to manifest [44].

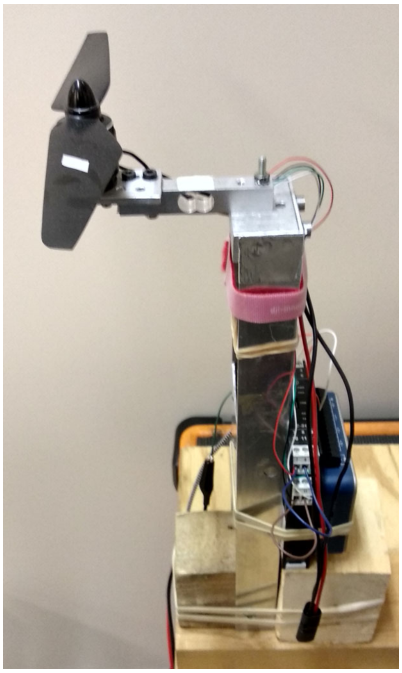

Fig. 2 Experimental setup

3.2 Battery Model

Experimental data has been acquired to validate the proposed battery model in Eq. 8. Figure 3 shows a notable decrease in voltage V_{batt} when the duty cycle (using a single motor) is greater than zero. The model in Eq. 8 is sufficient to capture the main battery dynamics illustrated in Fig. 3. The resulting internal resistance in the experiment is $R_{in} \approx 80\,\text{m}\Omega$, corresponding to a severely worn-out 4 cells LiPo battery.

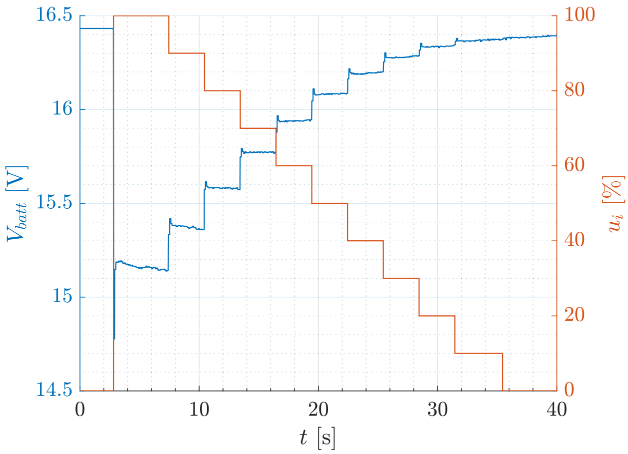

Fig. 3 Battery voltage drop due to internal resistance

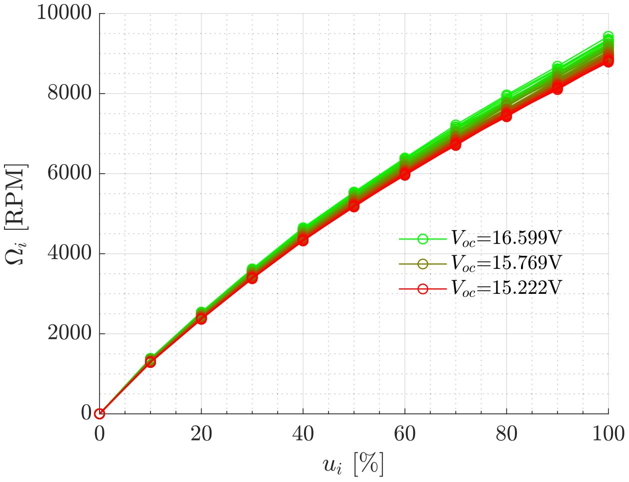

Fig. 4 Speed vs. duty cycle - regular blade (**a**)

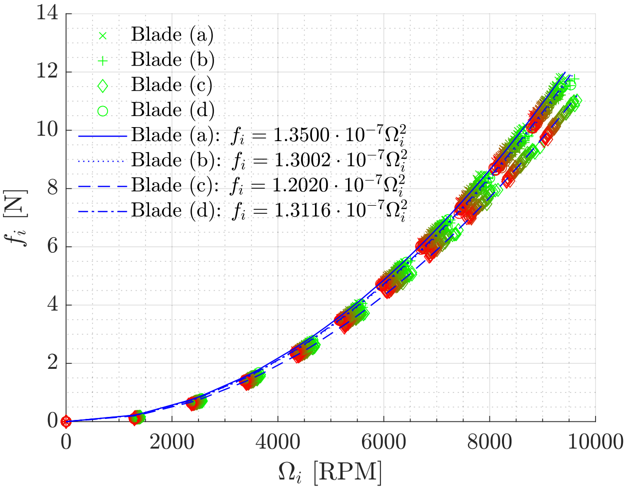

Fig. 5 Lift vs. speed - regular and faulty blades

3.3 Motor Speed

The experimental data reported in Fig. 4 describes relation (17), i.e., the dependency of the motor speed on the battery voltage, where the duty cycle u_i is the independent variable and V_{oc} is a parameter. The motor speed increases with the V_{oc} as expected. Also note that the motor speed shows a sub-linear increase with the duty cycle u_i, due to the increasing internal resistance voltage drop.

3.4 Lift Force, Battery Voltage and Propeller Faults

In this Section, we validate the model (19) and we show that the lift and drag coefficients depend on the propeller faults. We employ three damaged blades for the experimental tests. Given a nominal blade length of 11 cm and a 2 cm hub diameter, the propeller with two regular blades (Fig. 6a) has a diameter of 24 cm. We compare it with three damaged propellers: one with a regular blade and an artificially chipped blade measuring 10.45 cm in remaining length (Fig. 6b), another with a regular blade and an artificially chipped blade measuring 9.9 cm in remaining length (Fig. 6c), and finally, one with a regular blade and a scratched blade from a crash measuring 10.1 cm in remaining length (Fig. 6d). The reason for choosing the fault severity levels of 5% and 10% blade damage is that these faults are minor, with negligible impact on the UAV's manoeuvrability and unlikely to be noticed by a human pilot during normal flight. Therefore, investigating smaller faults is of limited practical relevance, while more severe faults are easier to detect with the proposed algorithm.

First of all, we note that the lift and drag coefficients in Eq. 4 are agnostic to the battery voltage. Figure 5 reports the lift force against the motor speed for the healthy and faulty propellers in Fig. 6. Each dot represents the median lift over a 1 s steady state acquisition (constant duty cycle). The dot color indicates the OCV: the independence of f_i from V_{batt} is evident.

The experimental data in Fig. 7 show that, for any fixed duty cycle u_i, the lift force u_i increases with the battery voltage as expected. This is due to the increase of Ω_i, as the ESC drives the motor in an open-loop configuration.

By accounting for battery voltage variation, the fractional-order polynomial in Eq. 19 achieves a fit with $R^2 = 0.99999$. The corresponding parameters are provided in [18] and reported in Table 1. For the acquired data, the resulting Root Mean Square Error (RMSE) is 10.756 mN. For comparison, the average lift force in the hexarotor under investigation is 2.534 N; consequently, the RMSE is less than 0.5% of the hovering lift.

Figure 8 shows the combined effects of blade fault and battery discharge on the lift force. Blade (a) provides the largest lift, followed by blade (b), blade (d), and blade (c) (see Fig. 6). Figure 8 clearly depicts red dots consistently below green dots, indicating that battery SoC has a greater impact than a 1.1 cm blade chipping. This underscores the importance of accounting for the SoC of the battery in the assessment of the health status of the propellers.

4 Residual Generator

The objective of this Section is to design a residual generator for a multirotor with $n_a \geq 4$ non-coaxial propellers. The produced residuals are required to be robust with respect to the battery voltage internal dynamics, avoiding then false positive fault isolation due to battery discharge. More precisely, we set the following problem.

Fig. 6 Healthy and faulty propellers

(a) Regular blade (b) 5% fault (artificial chipping)

(c) 10% fault (artificial chipping) (d) Scratched blade due to a crash

Problem 1 Find a set of observer based residual generators

$$\dot{z}_i = \gamma_i(z_i, p_F, \dot{p}_F, \omega, \eta, u, V_{batt}) \qquad i = 1, \ldots, n_a \qquad (20)$$

$$r_i = \chi_i(z_i, p_F, \dot{p}_F, \omega, \eta, u, V_{batt}) \qquad i = 1, \ldots, n_a, \qquad (21)$$

where z_i and r_i are the observer internal state and the output residual, respectively, and the functions $\gamma_i(\cdot)$ and $\chi_i(\cdot)$ are to be designed, such that

1. if, for each $k \neq i$, the kth propeller is faultless, then r_i exhibits an asymptotically stable dynamics, and thus it converges to zero;
2. if, for at least one $k \neq i$, the kth propeller is faulty, then r_i does not converge to zero.

Actually, Problem 1 requires the design of a bank of observers, and the overall residual set is a generalized structured residual (see [45]). Note that the convergence property of each residual r_i is required to be dependent only on the fault status, and therefore the overall set of residuals can be qualified as robust with respect to battery discharge.

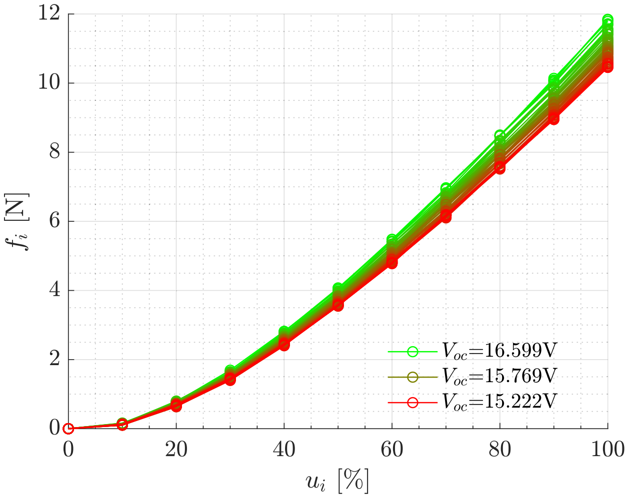

Fig. 7 Lift vs. duty cycle - regular blade (**a**)

4.1 Faults Symptoms

The lift force produced by the ith propeller, according to Section 2, depends on the battery voltage V_{batt}, the PWM u_i, and the fault condition. Let $\bar{f}_i$ be the lift force in the faultless case, calculated as in Eq. 18 and using the nominal parameters (blade (a) in Table 1). Assuming that V_{batt} is measured (and u_i is clearly available), $\bar{f}_i$ can be computed online. Therefore, $f_i = \bar{f}_i$ holds if the ith propeller is faultless. However, in case of blade damage, the propellers experience reduced lift coefficients, leading to $f_i \neq \bar{f}_i$. Defining then $\Delta f_i = f_i - \bar{f}_i$, the lift force can be trivially rewritten as

$$f_i = \bar{f}_i + \Delta f_i, \qquad (22)$$

where $\Delta f_i = 0$ if and only if there is no fault affecting the ith propeller (regardless the value of V_{batt}). Therefore, the terms Δf_i can be taken as blade fault symptoms, which are modeled additively, according to the residual generator problem [45, 46].

4.2 Residual Generator Design

In the following, we design the residual generator for FDI for a generic multirotor with non-coaxial propellers. This assumption entails that the columns F_j of the control effectiveness matrix F are pairwise linearly independent. Conversely, the method does not apply in the case of coaxial multirotors.

Let F_i be the ith column of the control effectiveness matrix F. It is useful to compactly denote $x_1 = \mathrm{col}(p_F, \eta)$ and $x_2 = \mathrm{col}(\dot{p}_F, \omega)$, leading to the compact multirotor model

$$\dot{x}_1 = g_1(x_1, x_2) \qquad (23)$$

$$\dot{x}_2 = g_2(x_1, x_2) + \sum_{k=1}^{n_a} G(x_1) F_i (\bar{f}_i + \Delta f_i), \qquad (24)$$

Table 1 Nonlinear multivariable fit: lift (19)

	a_{L0}	a_{L1}	a_{L2}	a_{L3}	v_{L0}	R^2	RMSE
Blade (a)	$-1.4310{\cdot}10^{-2}$	$4.6339{\cdot}10^{-3}$	$1.0350{\cdot}10^{-4}$	$-1.7318{\cdot}10^{-7}$	5.2859	0.99999	$8.6427{\cdot}10^{-3}$
Blade (b)	$-9.3949{\cdot}10^{-3}$	$3.2136{\cdot}10^{-3}$	$1.1373{\cdot}10^{-4}$	$-2.5089{\cdot}10^{-7}$	5.0339	0.99999	$1.0756{\cdot}10^{-2}$
Blade (c)	$-9.2213{\cdot}10^{-3}$	$3.0675{\cdot}10^{-3}$	$1.1057{\cdot}10^{-4}$	$-2.6715{\cdot}10^{-7}$	5.0107	0.99999	$9.3121{\cdot}10^{-3}$
Blade (d)	$-1.4103{\cdot}10^{-2}$	$4.5117{\cdot}10^{-3}$	$9.8717{\cdot}10^{-5}$	$-1.7527{\cdot}10^{-7}$	5.1339	0.99999	$8.4475{\cdot}10^{-3}$

where $g_1(\cdot)$ and $g_2(\cdot)$ are known functions directly obtained from the multirotor model (1), and

$$G(x_1) = \begin{bmatrix} \frac{1}{m}R & 0_{3\times 3} \\ 0_{3\times 3} & J^{-1} \end{bmatrix}. \tag{25}$$

Note that, for each vector x_1, the matrix $G(x_1)$ is non singular. Therefore, a couple of remarks could be pointed out in order to solve Problem 1.

Remark 1 For any vector $v \in \mathbb{R}^6$, there exists a smooth real valued function $\lambda_v : \mathbb{R}^6 \times \mathbb{R}^6 \to \mathbb{R}$ such that

$$\frac{\partial \lambda_v(x_1, x_2)}{\partial x_2} G(x_1) = v^T. \tag{26}$$

Moreover, the function $\lambda_v(\cdot)$ is affine in v and it can be directly defined as

$$\lambda_v(x_1, x_2) = v^T \mathrm{col}\left(m R^T \dot p_F, J\omega\right) + \bar\lambda(x_1), \tag{27}$$

where $\bar\lambda(x_1)$ is any smooth function depending on x_1, which can be arbitrarily chosen.

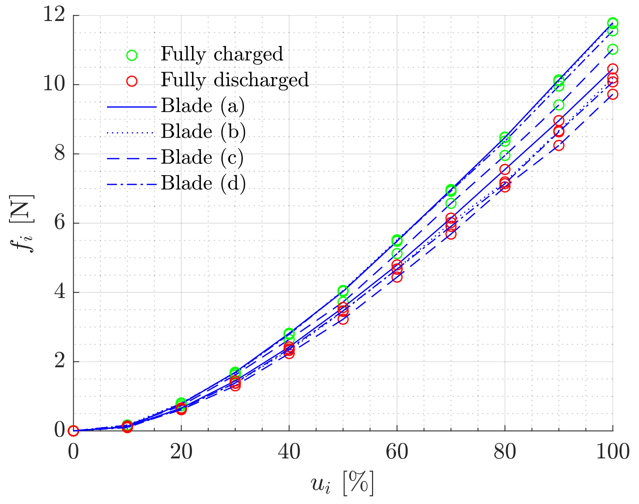

Fig. 8 Lift vs. duty cycle - regular and faulty blades

In the following, the null function $\bar\lambda(x_1) = 0$ will be considered.

Remark 2 There exist n_a smooth maps $\Phi_1, \ldots, \Phi_{n_a} : \mathbb{R}^6 \times \mathbb{R}^6 \to \mathbb{R}^5$ such that

$$\frac{\partial \Phi_i(x_1, x_2)}{\partial x_2} G(x_1) F_j = p_{ij}, \tag{28}$$

where $p_{ij} = 0$ if and only if $i = j$. Each map $\Phi_i(x_1, x_2)$ can be algorithmically constructed, since it is sufficient to consider any basis $v_{i_1}, \ldots, v_{i_5}$ of $\ker(F_i^T)$, and then to define

$$\Phi_i(x_1, x_2) = \mathrm{col}\left(\lambda_{v_{i_1}}(x_1, x_2), \ldots, \lambda_{v_{i_5}}(x_1, x_2)\right). \tag{29}$$

Equivalently, defining the matrix $V_i = \left[v_{i_1} \ \ldots \ v_{i_5}\right]$, it is possible to define

$$\Phi_i(x_1, x_2) = V_i^T \mathrm{col}\left(m R^T \dot p_F, J\omega\right). \tag{30}$$

The transformations $\Phi_i(x_1, x_2)$, for $i = 1, \ldots, n_a$, possess the key features for the desired decoupling conditions, and thus for the residual generator design. To prove this, consider the set of the observer based residual generators

$$\dot z_i = \tilde g_i(x_1, x_2) + \sum_{\substack{k=1 \\ k \neq i}}^{n_a} p_{ik} \bar f_k + H_i r_i \tag{31}$$

$$r_i = \Phi_i(x_1, x_2) - z_i, \tag{32}$$

for each $i = 1, \ldots, n_a$ and

$$\tilde g_i(x_1, x_2) = V_i^T \begin{bmatrix} -m\hat\omega R^T \dot p_F - k_t R^T \dot p_F + mgR^T e_3 \\ -k_r\omega - \omega \times J\omega \end{bmatrix},$$

where $\hat\omega$ denotes the skew matrix associated with the angular velocity vector ω.

Proposition 1 *The set of residual generators (31)–(32), with $-H_i$ Hurwitz for $i = 1, \ldots, n_a$, solves Problem 1.*

Proof It is a matter of calculation to check that

$$\frac{\partial \Phi_i(x_1,x_2)}{\partial x_1} g_1(x_1,x_2) = V_i^T \begin{bmatrix} -m\hat{\omega}R^T \dot{p}_F \\ 0 \end{bmatrix} \tag{33}$$

and thus

$$\sum_{k=1}^{2} \frac{\partial \Phi_i(x_1,x_2)}{\partial x_k} g_k(x_1,x_2) = \tilde{g}_i(x_1,x_2). \tag{34}$$

Therefore, for $i = 1,\dots,n_a$, we can directly differentiate (32) to get

$$\dot{r}_i = -H_i r_i + \sum_{\substack{k=1\\k\neq i}}^{n_a} p_{ik}\Delta f_k. \tag{35}$$

Therefore, the residual r_i is statically decoupled from Δf_i, and so it is decoupled from the fault on the ith propeller. In contrast, since $p_{ik} \neq 0$ for each $k \neq i$, there is a coupling effect between r_i and any other fault. Moreover, r_i can converge to the origin if and only if $f_k = 0$ for each $k \neq i$. □

Remark 3 The residual error is then linear and time invariant, and it can be made asymptotically stable by any Hurwitz matrix $-H_i$. Moreover, all the eigenvalues of the error dynamics can be arbitrarily chosen, according to the fact that we are considering an accessible state system.

4.3 Choice of the Basis for Residual Interpretation

The maps $\Phi_i(x_1,x_2)$ from Eq. 29 depend on the choice of a basis $v_{i_1},\dots,v_{i_5}$ of $\ker(F_i^T)$. Among the possible choices, we present the one used in the remainder of the paper, as it provides better insight into the meaning of each scalar residual. Denoting with l the length of the ith arm, the position of ith propeller in the body fixed frame can be identified with $l_i = l\cos(\theta_i)e_1 + l\sin(\theta_i)e_2$. Therefore, using Eq. 5, it is possible to express the ith column F_i of F as

$$F_i = \begin{bmatrix} 0 & 0 & -1 & -ls_{\theta_i} & lc_{\theta_i} & -(-1)^i \frac{c_L}{c_D} \end{bmatrix}^T. \tag{36}$$

The basis $v_{i_1},\dots,v_{i_5}$ of $\ker(F_i^T)$ can be chosen according to a physical interpretation rather than being just a kernel. For instance, a possible choice is $v_{i_1} = e_1$, $v_{i_2} = e_2$, and

$$v_{i_3} = \begin{bmatrix} 0 & 0 & 0 & c_{\theta_i} & s_{\theta_i} & 0 \end{bmatrix}^T \tag{37}$$

$$v_{i_4} = \begin{bmatrix} 0 & 0 & l & -s_{\theta_i} & c_{\theta_i} & 0 \end{bmatrix}^T \tag{38}$$

$$v_{i_5} = \begin{bmatrix} 0 & 0 & (-1)^i \frac{c_{D_i}}{c_{L_i}} & 0 & 0 & -1 \end{bmatrix}^T. \tag{39}$$

Denoting $\mathrm{col}(v_x, v_y, v_z) = R^T p_F$, where v_x, v_y, and v_z are the components of the linear velocity the body frame, the related transformation is

$$\Phi_i(x_1,x_2) = \begin{bmatrix} mv_x \\ mv_y \\ \cos(\theta_i)I_x p - \sin(\theta_i)I_y q \\ lv_z - \sin(\theta_i)I_x p + \cos(\theta_i)I_y q \\ (-1)^i \frac{c_{D_i}}{c_{L_i}} mv_z - I_z r \end{bmatrix}. \tag{40}$$

With this choice, we can make some additional observations on each component of $\Phi_i(x_1,x_2)$. The third component of $\Phi_i(x_1,x_2)$, which is the one associated with v_{i_3}, is expected to be dominant for the fault isolation task. This component represents a combination of the torques around the pitch and roll axes, which are the most relevant control actions. We expect the fifth component of $\Phi_i(x_1,x_2)$ to be less sensitive to faults, since the yaw angle is less responsive to actuator commands compared to pitch and roll. The first and second components of $\Phi_i(x_1,x_2)$ are expected to be null, as actuator faults do not directly affect the linear acceleration along the x_B and y_B axes (i.e., they are non-matching). Moreover, these vectors are identical for all motors and therefore do not contribute to fault isolation. As a result, the first and second components are not useful for FDI and are discarded to avoid unnecessary computation. Finally, note that all the variables in $\Phi_i(x_1,x_2)$ can be reliably measured, with the exception of ratio c_{D_i}/c_{L_i}, which can experience variations due to faults.

5 Residual Evaluation

In this Section, we detail how to perform FDI from the model-based residuals proposed in the previous Section.

5.1 Statistical Residual Model

The residual generator (20) depends on the measured state space variables, which can be in general corrupted by noise. As $r_i \in \mathbb{R}^3$, for $i = 1,\dots,n_a$, Eq. 20 returns $3n_a$ scalar residuals. In case of propeller faults, each residual r_i is designed to be sensitive to the faults on the jth propeller, with $j \neq i$. Also, the nominal lift estimation $\bar{f}_i$ relies on Eq. 18 and the estimated parameters from blade (a) in Table 1, which show an RMSE up to 10.756 mN [18]. Moreover, model mismatch may affect the residual, such as unmodeled rotor dynamics. For these practical reasons, the residuals are not exactly zero even when no propeller fault has occurred, so thresholding is not straightforward.

We do not consider each residual as a stochastic process in this analysis. Instead, we assume each realization represents a different observation of the same underlying random variable. We assume that the residuals are characterized by Gaussian distributions, due to the presence of sensor noise. Thus, we expect the distributions to change in the presence of a propeller fault, while remaining constant if no modifications are made to the powertrain. In particular, we focus on the mean value for FDI.

5.2 Neural Network Based Classification

To improve the robustness and accuracy of FDI, we employ a shallow NN that jointly interprets the means of the residuals rather than treating them individually. The dataset for training and testing consists of $8n_a$ simulations, each lasting 90 s, comprising:

- four trajectories, including two smooth helicoid trajectories, one hovering and one waypoint navigation,
- two initial SoC levels (40% and 100%),
- n_a faulty motors.

A fault is injected at 50% of the flight, with blades (b), (c), and (d) from Table 1 applied sequentially over equal time intervals. For each simulation, we randomize the noise seed (additive white gaussian noise according to IMU datasheet) and each motor's lift bias to take into account the powertrain identification error. The FDI procedure is performed on non-overlapping windows at a constant rate. Further details on the simulations are provided in Section 6.

The NN features an input layer ($3n_a$ neurons), a hidden layer ($3n_a$ neurons with ReLU activation function), and a output layer ($n_a + 1$ neurons with softmax activation). Each input corresponds to the mean value, computed over 25 consecutive samples, of one of the $3n_a$ residuals proposed in our method. Each output neuron corresponds to the probability of either a specific actuator fault among the n_a possible faults or no fault (represented by the extra neuron). Training employs Stochastic Gradient Descent with Momentum using the following hyperparameters: learning rate 10^{-6}, mini-batch size 32, L2 regularization 10^{-4}.

To assess generalizability with respect to the trajectory, we perform a leave-one-trajectory-out cross-validation: the dataset is split into four groups of $2n_a$ flights, each group corresponding to a different trajectory. Hence, one trajectory is kept as a completely separate test set, while the remaining three are used for training and validation. The dataset undergoes stratified partitioning with 70% training and 30% validation splits to address class imbalance. Feature normalization is applied using z-score standardization computed exclusively from the training set. The corresponding target labels are computed as the median value of the fault labels within each temporal segment. Please note that no mean removal is applied to the residuals to compensate for individual motor variability.

5.3 Statistical Test and Decision Making

For comparison, we also design a conventional statistical test on the mean of each residual to perform FDI. To assess deviations due to faults, we employ a two-sided Student's T-test, comparing the sample mean of each scalar residual against its nominal mean under the null hypothesis. The test can be efficiently executed online using a recursive implementation that only requires the number of samples, the current data point, and the nominal mean. Both the sample mean and standard deviation are updated using simple recursive formulas, distributing the computational load across samples. Once the p-value is computed, it is compared to the critical value of the desired significance level: the test returns `true` (i.e., fault) if the null hypothesis is rejected, and `false` otherwise.

To account for inter-motor variability, the nominal mean of each residual is computed during the first 10 seconds of flight, excluding the initial transient (e.g., take-off) for simplicity. The FDI algorithm is enabled only after these statistics are established. To this end, motors and propellers are assumed to be fault-free at take-off, as verified by standard pre-flight checks such as visual inspection.

Let $r_i \in \mathbb{R}^3$, for $i = 1, \ldots, n_a$ be the ith residual. The algorithm first performs FD; if a fault is detected, FDI follows. FD is based on T-tests applied to the first element of r_i, for $i = 1, \ldots, n_a$, which offer the best signal-to-noise ratio, using a high significance level α_{FD}. Since opposite motors exhibit similar responses in the first element of r_i due to symmetry, we expect $n_a - 2$ out of n_a residuals to trigger in the presence of a fault. Thus, if at least $n_a - 2$ tests fail, a fault is detected.

Once a fault is detected, FDI is performed by evaluating the T-test component-wise to each component of r_i, possibly using a different significance level α_{FDI}. Notably, changing the significance level does not require recalculating the

Table 2 Hexarotor [47] and battery parameters

Parameter	Value	Meas. Unit
Total system mass (m)	1.55	kg
Inertia along x_B (J_x)	0.0266	kg m^2
Inertia along y_B (J_y)	0.0266	kg m^2
Inertia along z_B (J_z)	0.0498	kg m^2
Gravitational acceleration (g)	9.81	m/s^2
Arm length (l)	0.275	m
Number of actuators (n_a)	6	–
Friction coefficient (k_t)	0	s^{-1}
Friction coefficient (k_r)	0	s^{-1}
Internal resistance (R_{in})	10	Ω
Nominal capacity (Q_{nom})	6.75	Ah
Fully charged voltage (V_{oc})	16.8	V

p-value, but only an additional comparison. We define the triggering pattern $\rho \in \mathbb{R}^{n_a}$, where $\rho_i = 1$ if any of the three components of the residual r_i fail the T-test, and $\rho_i = 0$ otherwise. Finally, we compare ρ with the expected fault signatures. If ρ matches the i-th column of the fault signature matrix, we identify the ith motor as faulty; otherwise, we report an unknown fault.

6 Results

In this Section, we report the simulation results for a DJI F550 hexarotor UAV (i.e., $n_a = 6$). We consider a hexarotor platform due to its higher intrinsic robustness to propeller faults, in order to prevent tracking failures that would unnecessarily complicate the analysis. The UAV parameters can be found in [47], and they are reported in Table 2 for completeness. The results presented in this Section refer to faults of varying severity, as already detailed in Section 5.2, specifically simulating the presence of blades (b), (c), and (d) in Fig. 6. These faults were experimentally identified in previous work [18], and their parameters are reported in Table 1. The corresponding fault signature matrix is reported in Table 3.

Table 3 Fault signature matrix for $n_a = 6$ (0 if decoupled, 1 otherwise)

	Δf_1	Δf_2	Δf_3	Δf_4	Δf_5	Δf_6
r_1	0	1	1	1	1	1
r_2	1	0	1	1	1	1
r_3	1	1	0	1	1	1
r_4	1	1	1	0	1	1
r_5	1	1	1	1	0	1
r_6	1	1	1	1	1	0

Table 4 Additive output white gaussian noise

Affected variable	Standard deviation
$\ddot{p}_F = \mathrm{col}(\ddot{x}_F, \ddot{y}_F, \ddot{z}_F)$	0.0785
$\dot{p}_F = \mathrm{col}(\dot{x}_F, \dot{y}_F, \dot{z}_F)$	0.0078
$p_F = \mathrm{col}(x_F, y_F, z_F)$	$7.8480e - 04$
$\omega = \mathrm{col}(p, q, r)$	0.0055
$\eta = \mathrm{col}(\varphi, \theta, \psi)$	$5.5192e - 04$

Additive sensor noise is simulated according to the datasheet of the MPU-9250 IMU [48], which is commonly adopted by the commercial Cube autopilot (also known as Pixhawk 2 autopilot). Accelerometer and gyroscope noise is directly injected, while noise on the remaining state variables is assumed to be one order of magnitude lower due to Kalman filtering, as reported in Table 4.

The controller and FDI parameters are shown in Table 5. The control law parameters refer to a classical inner-outer loop control structure, where three PID controllers are implemented in each loop. A control frequency of 400 Hz is used, consistent with the default main loop rate in ArduPilot firmware, reflecting the execution frequency on Pixhawk-class hardware.

Table 5 Control and FDI parameters. The PID parameters refer to the classical K_P, K_I, and K_D coefficients, respectively

Parameters	Values
Sampling rate	400 Hz
PID: x_F	{4.98, 2.08, 3.90}
PID: y_F	{4.98, 2.08, 3.90}
PID: z_F	{4.98, 2.08, 3.90}
PID: φ	{242, 720, 27}
PID: θ	{242, 720, 27}
PID: ψ	{4.98, 2.08, 3.90}
Residual generators: $H_1, \ldots, H_6$	$5I_5$
NN: hidden layer	18 neurons (ReLU)
NN: learning rate	10^{-3}
NN: mini-batch size	32
NN: L2 regularization	10^{-4}
NN: maximum epochs	5
T-test: α_{FD}	10^{-10}
T-test: α_{FDI}	10^{-8}
T-test: buffer size	25
T-test: sampling interval	0.0625 s

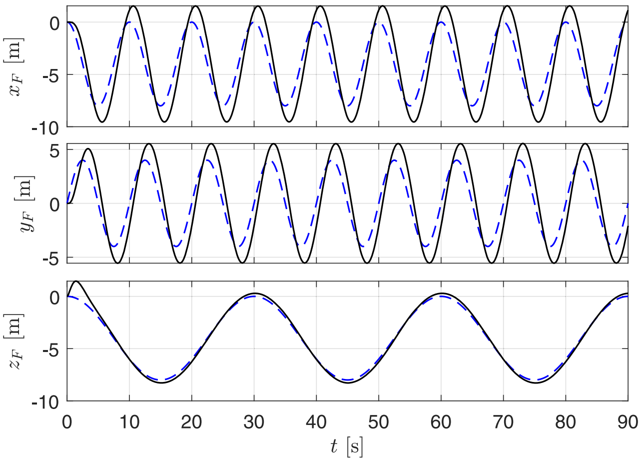

Fig. 9 Tracking performances about the linear position: reference (blue dashed line) and state (black solid line)

As for PWM signals, each control input is constrained to $u_i \in [0, 1]$, where the value $u_i = 1$ corresponds to the maximum lift thrust f_i obtainable from the ith motor, i.e., up to 12.0059 N under fault-free and fully charged battery conditions. The motor dynamics are finally included as first order systems with a time constant equal to 0.05 s, while neglecting them in the FDI code. As a result, the residuals differ from white Gaussian noise in the no-fault scenario, thereby making FDI more challenging and realistic.

To test the robustness of the approach regarding the individual parameter variations of each motor, we also inject an unknown bias on each motor lift f_i. The magnitude of the bias is a uniform random variable in $[-10.756, 10.756]$ mN, where 10.756 mN is the lift RMSE due to motor identification [18].

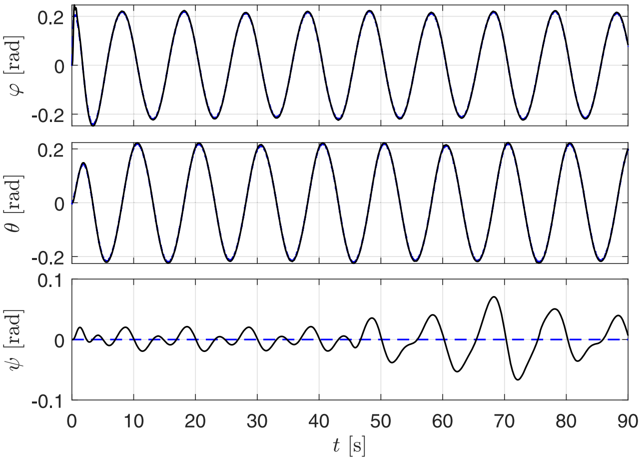

Fig. 10 Attitude tracking performances: reference (blue dashed line) and state (black solid line)

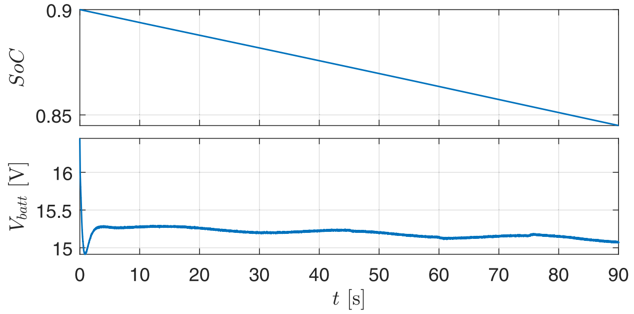

Fig. 11 SoC and battery voltage during the flight

As first, we report the results during a single tracking task in Sections 6.1–6.3. Then, we propose a Monte Carlo validation in Section 6.4. Section 6.5 proposes several comparisons and an ablation study to validate both the residuals and the classifier. Finally, we report the results of the PIL tests in Section 6.6.

6.1 Trajectory Tracking

The simulated flight consists of a tracking task whose total duration is 90 s and the initial SoC is 90%. The hexarotor is required to follow an helicoid, alternating ascending and descending reference. The initial 45 s of flight are faultless. Then, a fault on propeller 3 is injected. Firstly, we replace its nominal parameters with the ones of the faulty blade (b) from Table 1. Then, from $t = 60$ s, we employ the parameters of the faulty blade (c), and finally the ones of the faulty blade (d) from $t = 75$ s.

Figures 9 and 10 show the tracking performance. The yaw trajectory ψ_r is externally given as a reference, while the roll φ_r and pitch θ_r trajectories are calculated in a classical inner-outer loop control structure. The PID controllers prove to be robust to battery discharge and minor blade damages. In fact, the UAV is still capable to perform the tracking task with negligible performance degradation (see, for example, the yaw angle in Fig. 10). Please note that the battery volt-

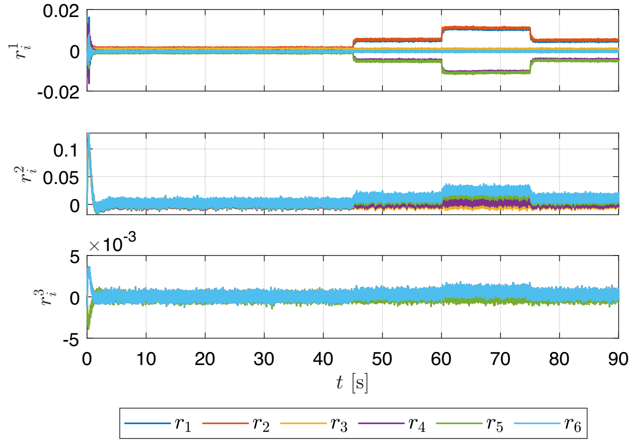

Fig. 12 Residuals (with battery compensation included in the residual generator)

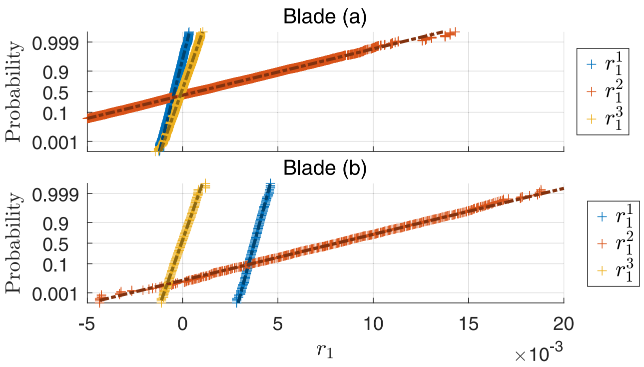

Fig. 13 Normal plot for r_1 (including battery compensation): healthy blade (**a**) vs. faulty blade (**b**)

age (Fig. 11) changes in time, due to battery discharge and internal resistance voltage drop.

6.2 Residual Analysis

Figure 12 shows the residuals r_i over time, with the first, second, and third components of each residual plotted separately. In presence of faults, a residual offset can be noticed, exception made for r_3, as it is insensitive by design to faults on motor 3. The offset is proportional to the severity of the fault (also see Fig. 6). However, the effects of sensor noise are evident.

Figure 13 shows the normal probability plots of the scalar residuals r_1 collected during the flight, focusing on the first residual r_1 for brevity. Each line corresponds to one element of the residual vector, plotted against the theoretical quantiles of a normal distribution.

In the fault-free case (healthy blade (a)), the residuals closely follow the reference lines, confirming that the Gaussian distribution assumption made in Section 5.1 is well satisfied. This agreement holds despite the battery discharge dynamics, indicating that variations in battery voltage do not significantly affect the statistical properties of the residuals under normal conditions, thanks to the proposed method for compensating battery discharge.

When a faulty blade (b) is mounted on motor 3, the residual distribution changes noticeably. Specifically, the first component of r_1 exhibits a clear mean shift with a high signal-to-noise ratio, evidencing a deviation from the expected behavior and highlighting sensitivity to the blade fault, consistent with the observations made in Section 4.3. The remaining components show a less pronounced change in statistical distribution, with the third component exhibiting the smallest change. Overall, this figure supports the use of residuals r_i as reliable indicators for detecting and isolating faults related to blade damage, even in the presence of battery voltage fluctuations.

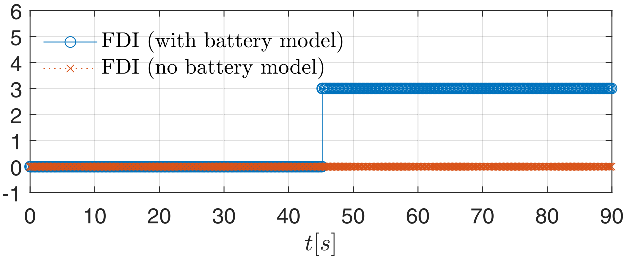

Fig. 14 Fault isolation using the NN classifier

6.3 Fault Isolation

Figure 14 reports the fault isolation outcomes using the proposed NN and residuals. Each input to the NN is the mean of one of the 18 proposed residuals, computed iteratively over a window of 25 consecutive samples. Hence, the NN classifier is triggered every 0.0625 s. The training requires less than 5 minutes, and the results stabilize within 5 epochs. During the fault-free portion of the flight, the proposed method (blue line) returns 0 (i.e., no fault). In the second part of the flight, where faults of varying severity are introduced in propeller 3, the proposed method accurately returns 3 as soon as the fault is injected.

6.4 Monte Carlo Validation

A Monte Carlo validation is performed using the 48 randomized scenarios from Section 5.2, with cross-validation results reported in Table 6 using the NN described therein. The results indicate high classification accuracy and low variance on both training and validation sets, suggesting effective learning without overfitting. The test accuracy, performed on the held out set of trajectories, remains high, although the increased standard deviation suggests sensitivity to more aggressive trajectories.

Table 7 presents the FDI confusion matrix obtained from the separate test set only. The accuracy of the proposed method is 95.04%, with a recall of 96.60% for the no-fault

Table 6 Cross-validation using a NN on the mean value of the residuals

Dataset	Mean accuracy	Standard deviation
Training	98.20%	±0.97%
Validation	98.18%	±0.96%
Test	95.04%	±6.28%

Table 7 Monte Carlo validation: FDI confusion matrix (test set only) using the NN and the battery-compensated residuals. FM$_i$: Faulty Motor i; NF: No Fault; UF: Unknown Fault. Values are row-wise normalized percentages

	Predicted class							
True class	NF	FM1	FM2	FM3	FM4	FM5	FM6	UF
NF	96.60%	0.35%	0.62%	0.58%	0.45%	0.58%	0.82%	0.00%
FM1	4.18%	93.92%	0.66%	0.33%	0.00%	0.10%	0.80%	0.00%
FM2	4.38%	0.42%	94.36%	0.75%	0.10%	0.00%	0.00%	0.00%
FM3	5.83%	0.23%	0.52%	93.00%	0.35%	0.00%	0.07%	0.00%
FM4	5.12%	0.00%	0.16%	0.43%	93.80%	0.30%	0.19%	0.00%
FM5	6.15%	0.09%	0.05%	0.09%	0.71%	92.64%	0.28%	0.00%
FM6	5.66%	0.49%	0.10%	0.10%	0.07%	0.42%	93.16%	0.00%

class and approximately 93% recall for each remaining fault class.

6.5 Ablation Study and Comparisons

6.5.1 Ablation Study: Removing Battery Voltage Compensation

To highlight the importance of compensating battery voltage in residual generation, Fig. 15 reports the residuals without taking into account the battery model, i.e., employing a conventional linear regression $f_i = k_{pwm}\, u_i$, where $k_{pwm} = 12.0059\,\mathrm{N}$ is the lift force at full speed. Differently from Fig. 12, the residuals in Fig. 15 are unreliable: during faults, the residuals show only marginal deviations, so defining proper decision criteria is not straightforward. Also, the Gaussian approximation (Fig. 16) is less accurate if compared to Fig. 13, as data points deviate significantly from the reference line. This deviation is mainly due to model mismatch. Also, their mean values deviate significantly from zero even in the no-fault case, indicating a systematic bias primarily due to battery discharge and individual variations in motor characteristics. In contrast, mean shifts due to faults are not prominent enough to be discernible in the plots. This suggests that such an uncompensated large bias due to battery mismatch actually hinders fault isolation.

Figure 14 also compares the fault isolation outcomes with those of the same method, but without using the lift force estimation from Eq. 19. Table 8 reports the results of the Monte Carlo validation. When residuals without battery compensation are used instead of the proposed ones, the NN classifier returns no-fault in most cases, despite retraining the NN. The method fails to perform FDI because the residuals, without compensation for battery voltage, do not provide sufficient information for such a small and shallow NN as one suitable for implementation on a conventional flight controller. In contrast, the same NN achieves accurate fault isolation (see Table 6) when using battery-compensated residuals.

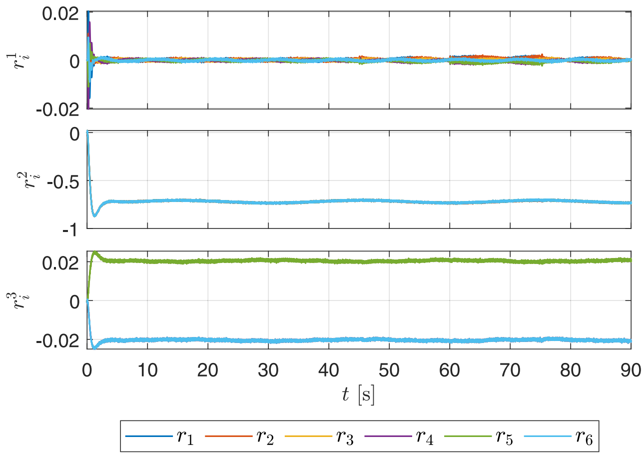

Fig. 15 Residuals without battery compensation in the residual generator

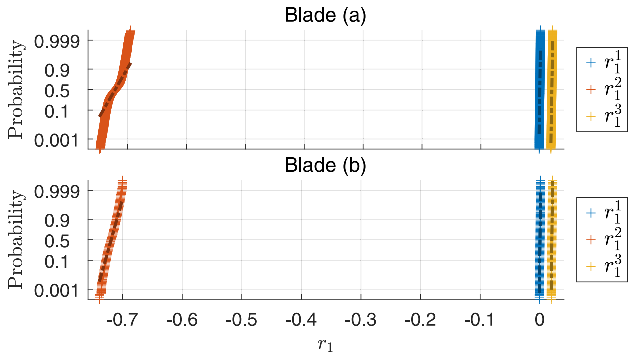

Fig. 16 Normal plot for r_1 without battery compensation: healthy blade (**a**) vs. faulty blade (**b**)

Table 8 Cross-validation using a NN on the mean value of the non-compensated residuals

Dataset	Mean accuracy	Standard deviation
Training	50.74%	±0.65%
Validation	50.56%	±0.58%
Test	48.41%	±1.88%

6.5.2 Ablation Study: T-test vs. Neural Network

For comparison, the T-test, with significance levels $\alpha_{FD} = 10^{-10}$ and $\alpha_{FDI} = 10^{-8}$, is applied to the residuals r_i every 0.0625 s, on a buffer including only the last 25 samples of each residual. Figure 17 compares the T-test results on the investigated flight, obtained using residuals with battery compensation versus those without compensation. During the fault-free portion of the flight, the T-test method (blue line) returns no fault, with the exception of isolated instances where an unknown fault (FDI result -1) is indicated. These infrequent false positives are due to sensor noise and actuator model mismatch and could be filtered out by applying a moving mode. In the second part of the flight, the T-test method correctly returns 3, with several false negatives where the FDI returns -1 (unknown fault) because the residuals does not match any column (i.e., fault signature) in Table 3. Applying the same statistical tests and decision-making policy without the battery model yields incorrect results (see Fig. 17), as in the case of the NN classifier, confirming that uncompensated residuals are unsuitable for FDI.

Results obtained on the Monte Carlo validation using the T-test-based method are summarized in Table 9. The accuracy of the T-test-based method is 80.35%, with a comparable recall for the no-fault class (94.25%) with respect to the NN method (see Table 7), but a significantly lower recall for each fault class (approximately 69%). With the T-test, the most frequent misclassification involves assigning an actual fault to an unknown fault. Therefore, a fault is detected but not successfully isolated. The remaining misclassifications involve assigning a fault to the wrong faulty motor, with most errors occurring between opposite motors (i.e., 1 and 4, 2 and 5, 3 and 6). This behaviour is related to the symmetry discussed in Section 5.3.

6.5.3 Comparison with Purely Data-Driven FDI

We compare the proposed strategy with the method presented in [32], which employs batches of roll-pitch error measurements to perform FDI. The training dataset is the same as in Section 5.2, where two-dimensional roll-pitch error measurements are included. Each time series is systematically segmented into non-overlapping windows of 120 consecutive samples, which are then flattened and fed to a NN featuring a input layer (240 neurons), two hidden layer 1 with (30 neurons with a sigmoid activation function), and an output layer (7 neurons with softmax activation), as detailed in [32]. Training employs Stochastic Gradient Descent with Momentum using the following hyperparameters [32]: learning rate 10^{-6}, maximum epochs 2000, mini-batch size 32, L2 regularization 10^{-4}.

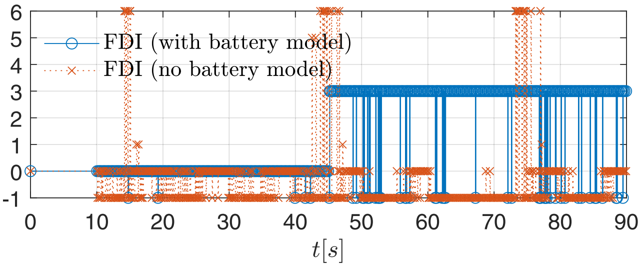

Fig. 17 Fault isolation using the T-test

Table 10 reports the results. The method provides approximately 50% accuracy, as the no-fault classification is always returned, being the most represented class in the dataset, which is clearly unsatisfactory. The method fails to perform FDI because it is based only on the tracking error, which is a poor indicator in trajectory tracking: differently from [32], the tracking error is primarily driven by the reference trajectory and affected by sensor noise, whereas the impact of faults is effectively compensated by the PID controllers, as evident in Section 6.2. By contrast, the simpler NN detailed in Section 5.2 can provide accurate FDI (see Table 6) because the proposed residuals contain fault information in a distilled form.

6.6 Processor-In-the-Loop Validation

To ensure the real-time executability on conventional control platforms, a PIL validation is performed. We consider the PixHawk 6c as the target platform, a widely used commercial controller equipped with a 32-bit STM32H743 Arm® Cortex®-M7 MicroController Unit (MCU) running at 480 MHz, with 2 MB of flash memory and 1 MB of SRAM.

The deployed code for the PIL test includes the control law, the residual generator, and the FDI algorithm for a hexarotor. The target hardware is a Arm® Cortex®-M7 (MCU) running at 480 MHz. All computations are implemented in single-precision floating-point format to facilitate deployment on MCUs. The deployed code yields the PIL results shown in Table 11.

Using the NN solution, the maximum CPU time (0.1098 ms) is larger than the T-test (0.0813 ms), because only the resid-

Table 9 Monte Carlo validation: FDI confusion matrix using the T-test and the battery-compensated residuals. FM_i: Faulty Motor i; NF: No Fault; UF: Unknown Fault. Values are row-wise normalized percentages

True class	Predicted class							
	NF	FM1	FM2	FM3	FM4	FM5	FM6	UF
NF	94.25%	0.08%	0.07%	0.04%	0.04%	0.01%	0.07%	5.42%
FM1	0.55%	69.17%	0.03%	0.05%	3.80%	0.00%	0.05%	26.33%
FM2	0.52%	0.17%	70.04%	0.05%	0.03%	3.55%	0.00%	25.62%
FM3	0.64%	0.02%	0.00%	70.20%	0.07%	0.00%	3.62%	25.45%
FM4	0.62%	3.99%	0.00%	0.03%	69.26%	0.09%	0.00%	26.01%
FM5	0.55%	0.03%	3.64%	0.00%	0.12%	69.37%	0.00%	26.28%
FM6	0.64%	0.05%	0.00%	3.31%	0.03%	0.00%	69.23%	26.73%

ual mean can be updated iteratively, while the NN activation is performed once every 25 samples. Compared to the 2.5 ms control cycle period (i.e., 400 Hz), the worst-case execution time corresponds to approximately 4.390% of the cycle, accounting for control, residual generation, and FDI. This guarantees that the NN classifier can run online, as well as the T-test alternative solution. Conversely, the average CPU time is larger using the T-test, because mean and standard deviation are updated iteratively, while the test performed once every 25 samples is lightweight. As anticipated, the results also show that the behavior of the C implementation on the MCU closely matches that of the reference simulation.

7 Conclusions

In this paper, we have presented a model-based FDI approach for multirotor UAVs. To make the residuals less sensitive to battery voltage drop, we have incorporated an experimental characterization of the powertrain. A hexarotor is presented as a study case, showing the necessity of the battery model to achieve fault isolation in realistic conditions. The analysis of the residuals shows that certain residuals are more relevant for FDI; in particular, the torque mismatch exhibits the best sensitivity in presence of sensor noise.

The proposed NN classifier outperforms the T-test-based method, achieving higher overall accuracy (95.04% vs. 80.35%) and notably better fault class recall (around 93% vs. 69%), thereby demonstrating the capability to identify even small faults, such as a single blade loss of approximately 5%. The main advantage of the NN approach lies in its ability to jointly interpret residual information in a multivariate manner, rather than relying on separate threshold-based checks for each residual as in traditional methods like the T-test. This integrated analysis enables more accurate fault detection and isolation, reducing misclassifications and improving robustness. The performance evaluation is supported by a solid validation strategy employing leave-one-trajectory-out cross-validation to ensure generalization to unseen flight conditions. Additional realism is incorporated by including motor biases, unmodeled motor dynamics affecting the FDI, and realistic sensor noise. Finally, the effectiveness of the proposed method is further confirmed by validation on the PIL setup, demonstrating its applicability in real-time embedded environments.

The proposed method applies to any multirotor with non-coaxial rotors. Since the computational complexity scales linearly with the number of actuators n_a, the reduced load makes the method feasible even for configurations with many motors. However, as the number of motors increases, a higher probability of fault misclassification is expected. To extend the method to coaxial multirotors, additional considerations regarding the predefined motor rotation direction are necessary; moreover, active fault isolation strategies that deliberately alter individual motor speeds may also prove beneficial.

Our future work will focus on two primary directions for improvement. First, to increase the robustness and adaptability of the fault isolation process, we are exploring the integration of observer-based techniques with frequency-domain methods. Second, we plan to extend the approach

Table 10 Cross-validation using the DNN from [32] on angular tracking errors

Dataset	Mean accuracy	Standard deviation
Training	49.39%	±0.05%
Validation	49.26%	±0.00%
Test	49.33%	±0.00%

Table 11 PIL test results using the proposed FDI solutions

	CPU time [ms]		CPU utilization [%]	
	Maximum	Average	Maximum	Average
NN	0.1098	0.0534	4.390	2.137
T-test	0.0813	0.0741	3.252	2.963

by expanding the spectrum of faults considered within the system.

Acknowledgements Not applicable.

Author Contributions Alessandro Baldini: formal analysis, investigation, methodology, software, validation, visualization, writing - original draft. Riccardo Felicetti: formal analysis, investigation, methodology, software, validation, visualization, writing - original draft. Francesco Ferracuti: conceptualization, supervision, validation, writing - original draft. Alessandro Freddi: conceptualization, methodology, project administration, supervision, writing - review & editing. Andrea Monteriù: conceptualization, project administration, resources, supervision, writing - review & editing.

Funding Open access funding provided by Università Politecnica delle Marche within the CRUI-CARE Agreement. The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

Data Availability We do not analyse or generate any datasets, because our work proceeds within a theoretical and mathematical approach.

Declarations

Ethics approval Not applicable.

Consent to participate Not applicable.

Consent for publication Not applicable.

Competing interests The authors have no relevant financial or non-financial interests to disclose.

References

1. Goel, K., Corah, M., Boirum, C., Michael, N.: Fast exploration using multirotors: Analysis, planning, and experimentation. In: Field and Service Robotics: Results of the 12th International Conference, pp. 291–305. Springer (2021)
2. Basiri, A., Mariani, V., Silano, G., Aatif, M., Iannelli, L., Glielmo, L.: A survey on the application of path-planning algorithms for multi-rotor UAVs in precision agriculture. J. Navig. **75**(2), 364–383 (2022)
3. Villa, D.K., Brandao, A.S., Sarcinelli-Filho, M.: A survey on load transportation using multirotor UAVs. J. Intell. Robot. Syst. **98**, 267–296 (2020)
4. Kellermann, R., Biehle, T., Fischer, L.: Drones for parcel and passenger transportation: A literature review. Transp. Res. Interdiscip. Perspect. **4**, 100088 (2020)
5. Hamandi, M., Usai, F., Sablé, Q., Staub, N., Tognon, M., Franchi, A.: Design of multirotor aerial vehicles: A taxonomy based on input allocation. Int. J. Robot. Res. **40**(8–9), 1015–1044 (2021)
6. Yuksek, B., Vuruskan, A., Ozdemir, U., Yukselen, M., Inalhan, G.: Transition flight modeling of a fixed-wing VTOL UAV. J. Intell. Robot. Syst. **84**, 83–105 (2016)
7. Michieletto, G., Ryll, M., Franchi, A.: Fundamental actuation properties of multirotors: Force-moment decoupling and fail-safe robustness. IEEE Trans. Rob. **34**(3), 702–715 (2018). https://doi.org/10.1109/TRO.2018.2821155
8. Freddi, A., Lanzon, A., Longhi, S.: A feedback linearization approach to fault tolerance in quadrotor vehicles. IFAC Proc. Vol. **44**(1), 5413–5418 (2011)
9. Baldini, A., Felicetti, R., Freddi, A., Longhi, S., Monteriù, A.: Actuator fault-tolerant control architecture for multirotor vehicles in presence of disturbances. J. Intell. Robot. Syst. **99**, 859–874 (2020)
10. Plett, G.L.: Battery management systems, volume I: Battery modeling. Artech House (2015)
11. Rahimi-Eichi, H., Baronti, F., Chow, M.-Y.: Online adaptive parameter identification and state-of-charge coestimation for lithium-polymer battery cells. IEEE Trans. Industr. Electron. **61**(4), 2053–2061 (2013)
12. Shraim, H., Awada, A., Youness, R.: A survey on quadrotors: Configurations, modeling and identification, control, collision avoidance, fault diagnosis and tolerant control. IEEE Aerosp. Electron. Syst. Mag. **33**(7), 14–33 (2018)
13. Wild, G., Murray, J., Baxter, G.: Exploring civil drone accidents and incidents to help prevent potential air disasters. Aerospace **3**(3), 22 (2016)
14. Puchalski, R., Ha, Q., Giernacki, W., Nguyen, H.A.D., Nguyen, L.V.: Padre-a repository for research on fault detection and isolation of unmanned aerial vehicle propellers. J. Intell. Robot. Syst. **110**(2), 74 (2024)
15. Mohammadi, M., Shahri, A.M.: Adaptive nonlinear stabilization control for a quadrotor UAV: Theory, simulation and experimentation. J. Intell. Robot. Syst. **72**, 105–122 (2013)
16. Ho, M.-T., Lin, C.-Y.: PID controller design for robust performance. IEEE Trans. Autom. Control **48**(8), 1404–1409 (2003)
17. Clothier, R.A., Palmer, J.L., Walker, R.A., Fulton, N.L.: Definition of an airworthiness certification framework for civil unmanned aircraft systems. Saf. Sci. **49**(6), 871–885 (2011)
18. Baldini, A., Felicetti, R., Ferracuti, F., Freddi, A., Monteriù, A., Scalella, S., Zhang, Y.: Multirotor lift estimation under battery discharge and blade faults. In 2024 International Conference on Unmanned Aircraft Systems (ICUAS), pp. 8–14. IEEE (2024)
19. Piljek, P., Kotarski, D., Krznar, M.: Method for characterization of a multirotor UAV electric propulsion system. Appl. Sci. **10**(22), 8229 (2020)
20. Kotarski, D., Krznar, M., Piljek, P., Simunic, N.: Experimental identification and characterization of multirotor UAV propulsion. In Journal of Physics: Conference Series, vol. 870, p. 012003. IOP Publishing (2017)
21. Efe, M.Ö.: Battery power loss compensated fractional order sliding mode control of a quadrotor UAV. Asian J. Control. **14**(2), 413–425 (2012)
22. Lin, X., Yu, Y., Sun, C.-Y.: A decoupling control for quadrotor UAV using dynamic surface control and sliding mode disturbance observer. Nonlinear Dyn. **97**(1), 781–795 (2019)
23. Podhradskỳ, M., Bone, J., Coopmans, C., Jensen, A.: Battery model-based thrust controller for a small, low cost multirotor unmanned aerial vehicles. In 2013 International Conference on Unmanned Aircraft Systems (ICUAS), pp. 105–113. IEEE (2013)

24. Podhradskỳ, M., Coopmans, C., Jensen, A.: Battery state-of-charge based altitude controller for small, low cost multirotor unmanned aerial vehicles. J. Intell. Robot. Syst. **74**, 193–207 (2014)
25. Wang, C., Nahon, M., Trentini, M.: Controller development and validation for a small quadrotor with compensation for model variation. In: 2014 International Conference on Unmanned Aircraft Systems (ICUAS), pp. 902–909. IEEE (2014)
26. Freddi, A., Longhi, S., Monteriù, A.: A diagnostic Thau observer for a class of unmanned vehicles. J. Intell. Robot. Syst. **67**(1), 61–73 (2012)
27. Ortiz-Torres, G., Castillo, P., Sorcia-Vázquez, F.D., Rumbo-Morales, J.Y., Brizuela-Mendoza, J.A., De La Cruz-Soto, J., Martínez-García, M.: Fault estimation and fault tolerant control strategies applied to VTOL aerial vehicles with soft and aggressive actuator faults. IEEE Access **8**, 10649–10661 (2020)
28. Baldini, A., Felicetti, R., Freddi, A., Longhi, S., Monteriù, A.: Hexarotor fault tolerant control using a bank of disturbance observers. In: 2022 International Conference on Unmanned Aircraft Systems (ICUAS), pp. 608–616. IEEE (2022)
29. Gangsar, P., Tiwari, R.: Signal based condition monitoring techniques for fault detection and diagnosis of induction motors: A state-of-the-art review. Mech. Syst. Signal Process. **144**, 106908 (2020)
30. Ai, S., Song, J., Cai, G., Zhao, K.: Active fault-tolerant control for quadrotor UAV against sensor fault diagnosed by the auto sequential random forest. Aerospace **9**(9), 518 (2022)
31. Liang, S., Zhang, S., Huang, Y., Zheng, X., Cheng, J., Wu, S.: Data-driven fault diagnosis of FW-UAVs with consideration of multiple operation conditions. ISA Trans. **126**, 472–485 (2022)
32. Park, J., Jung, Y., Kim, J.-H.: Multiclass classification fault diagnosis of multirotor UAVs utilizing a deep neural network. Int. J. Control Autom. Syst. **20**(4), 1316–1326 (2022)
33. Baldini, A., Felicetti, R., Ferracuti, F., Freddi, A., Iarlori, S., Monteriù, A.: Real-time propeller fault detection for multirotor drones based on vibration data analysis. Eng. Appl. Artif. Intell. **123**, 106343 (2023)
34. Fossen, T.I.: Guidance and control of ocean vehicles. Wiley (1994)
35. Bouabdallah, S., Noth, A., Siegwart, R.: PID vs LQ control techniques applied to an indoor micro quadrotor. In: 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No. 04CH37566), vol. 3, pp. 2451–2456. IEEE (2004)
36. Bondyra, A., Gasior, P., Gardecki, S., Kasiński, A.: Fault diagnosis and condition monitoring of UAV rotor using signal processing. In: 2017 Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA), pp. 233–238. IEEE (2017)
37. Wang, Y., Tian, J., Sun, Z., Wang, L., Xu, R., Li, M., Chen, Z.: A comprehensive review of battery modeling and state estimation approaches for advanced battery management systems. Renew. Sustain. Energy Rev. **131**, 110015 (2020)
38. Gong, A., Verstraete, D.: Experimental testing of electronic speed controllers for UAVs. In: 53rd AIAA/SAE/ASEE Joint Propulsion Conference, pp. 49–55 (2017)
39. Gong, A., MacNeill, R., Verstraete, D.: Performance testing and modeling of a brushless DC motor, electronic speed controller and propeller for a small UAV application. In: 2018 Joint Propulsion Conference, pp. 45–84 (2018)
40. Bishop, R.C., H, D.R.: Modern control systems (2011)
41. Adafruit: (2024). https://www.adafruit.com/product/4541
42. Seneca S.r.l.: (2024). https://www.seneca.it/media/3938/z-sgz-sg2_2002_en.pdf
43. National Instruments: (2024). https://www.ni.com/docs/en-US/bundle/usb-6001-specs/resource/374369a.pdf
44. Galeotti, M., Cinà, L., Giammanco, C., Cordiner, S., Di Carlo, A.: Performance analysis and SOH (state of health) evaluation of lithium polymer batteries through electrochemical impedance spectroscopy. Energy **89**, 678–686 (2015)
45. Chen, J., Patton, R.J.: Robust model-based fault diagnosis for dynamic systems, vol. 3. Springer (2012)
46. De Persis, C., Isidori, A.: A geometric approach to nonlinear fault detection and isolation. IEEE Trans. Autom. Control **46**(6), 853–865 (2001)
47. Niemiec, R., Ivler, C., Gandhi, F., Sanders, F.: Multirotor electric aerial vehicle model identification with flight data with corrections to physics-based models. CEAS Aeronaut. J. **13**(3), 575–596 (2022)
48. TDK InvenSense: Mpu-9250, nine-axis (gyro+accelerometer+compass) mems motiontracking™device (2016). https://invensense.tdk.com/download-pdf/mpu-9250-datasheet/

Alessandro Baldini was born in Ancona, Italy, in 1991. He graduated in Computer and Automation Engineering at Università Politecnica delle Marche (Ancona, Italy) in 2016, and obtained Ph.D. in 2021. He is currently a Researcher at the Department of Information Engineering, Università Politecnica delle Marche. His research interest is mainly focused in geometric control, nonlinear control and nonlinear observers, with a special attention to disturbance observers. His main application field is that of Fault Tolerant Control applied to unmanned vehicles.

Riccardo Felicetti received his Master's Degree cum laude in Computer and Automation Engineering and his Ph.D. degree cum laude in Information Engineering from Università Politecnica delle Marche, in 2016 and 2021, respectively. He is currently a researcher with Università Politecnica delle Marche. His main research interests are fault detection and diagnosis, fault-tolerant control, and optimization with applications to unmanned vehicles and energy management systems.

Francesco Ferracuti received his M.Sc. degree in Automation Engineering and his Ph.D. degree in Automation, Information and Management Engineering from Università Politecnica delle Marche, Italy, in 2010 and 2014, respectively. He is currently an Associate Professor at the Department of Information Engineering, Università Politecnica delle Marche. His research interests include data-driven fault diagnosis, signal processing, statistical pattern recognition, and system identification, with applications to industrial, robotic and energy systems.

Alessandro Freddi is an Associate Professor at Università Politecnica delle Marche (Ancona, Italy), where he teaches “Preventive Maintenance for Robotics and Smart Automation”, “Fault Diagnosis and Predictive Maintenance”, and “Control of Mechanical Systems”. His main research activities include fault diagnosis, fault prognosis, fault-tolerant control, and human-robot interaction, with applications to unmanned vehicles, autonomous and robotic systems. He is the co-author of more than 130 publications in these fields and the co-editor of six books. He currently serves as Associate Editor for several major conferences and journals in the fields of Automation and Control Systems. Since 2012, he has participated in 12 research projects in the areas of automation, robotics, and assistive technologies. He also collaborates with several industrial partners with a particular focus on diagnosis and predictive maintenance.

Andrea Monteriù received his M.Sc. in Electronic Engineering (2003) and Ph.D. in Artificial Intelligent Systems (2006) from Università Politecnica delle Marche, Italy. He is Associate Professor of Systems and Control Engineering and Director of the Laboratory of Artificially Intelligent Robotics (LAIR). His research interests primarily include fault diagnosis, fault tolerant control, nonlinear dynamics and control, periodic and stochastic system control, applied in different fields including aerospace, marine, robotic and unmanned artificial intelligent systems. He has authored over 250 peer-reviewed publications and currently serves as Chair of the IFAC Technical Committee 7.2 on Marine Systems.