MARS: Defending Unmanned Aerial Vehicles From Attacks on Inertial Sensors with Model-based Anomaly Detection and Recovery

Haocheng Meng
Duke University

Shaocheng Luo
Duke University

Zhenyuan Liang
Duke University

Qing Huang
Duke University

Amir Khazraei
Duke University

Miroslav Pajic
Duke University

Abstract

Unmanned Aerial Vehicles (UAVs) rely on measurements from Inertial Measurement Units (IMUs) to maintain stable flight. However, IMUs are susceptible to physical attacks, including acoustic resonant and electromagnetic interference attacks, resulting in immediate UAV crashes. Consequently, we introduce a *Model-based Anomaly detection and Recovery System* (MARS) that enables UAVs to quickly detect adversarial attacks on inertial sensors and achieve dynamic flight recovery. MARS features an attack-resilient state estimator based on the Extended Kalman Filter, which incorporates position, velocity, heading, and rotor speed measurements to reconstruct accurate attitude and angular velocity information for UAV control. Moreover, a statistical anomaly detection system monitors IMU sensor data, raising a system-level alert if an attack is detected. Upon receiving the alert, a multi-stage dynamic flight recovery strategy suspends the ongoing mission, stabilizes the drone in a hovering condition, and then resumes tasks under the resilient control. Experimental results in PX4 software-in-the-loop environments as well as real-world MARS-PX4 autopilot-equipped drones demonstrate the superiority of our approach over existing IMU-defense frameworks, showcasing the ability of the UAVs to survive attacks and complete the missions.

1 Introduction

Inertial Measurement Units (IMUs) are used in any Unmanned Aerial Vehicle (UAV) to maintain the flight stability. IMU incorporates the orientation, angular velocity, and acceleration from the accelerometer, gyroscope, and magnetometer, providing vital information on body-frame acceleration and attitude to keep the drone flying smoothly (e.g., [1]). However, UAV inertial sensors have been shown vulnerable to various adversarial attacks. Onboard gyroscopes and accelerometers are typically Micro-Electro-Mechanical Systems (MEMS) devices, which makes them sensitive to acoustic noise at resonant frequencies [2]. While the resonant frequencies are typically in the ultrasonic band, commercially available IMU sensors can resonate within the audible band, making attacks feasible using consumer-grade speakers. Specifically, the resonant oscillation caused by acoustic injection can severely disrupt the drone's attitude controllers [3,4]. Moreover, attackers can gain implicit control over the output of compromised sensors by manipulating the injected analog signals [5].

IMUs are also vulnerable to Electromagnetic Interference (EMI) attacks. EMI waves typically require significant power to distort the circuits [6,7]. Recent EMI attacks on commercial flight controller boards have been shown to incapacitate UAVs by corrupting the communication between IMU sensors and the flight control unit [8].

Both types of attacks result in safety-critical incidents, as a UAV is rendered uncontrollable under the inertial sensor attacks, crashing within seconds; thus, igniting recent interest in defenses against IMU attacks [9–14]. Yet, existing prevention and mitigation techniques fail to neutralize the impact of attacks; e.g., [10–13] fail against IMU attacks [3, 5, 8] that quickly crash the UAVs (see [8] for details). Installing redundant IMUs is also ineffective, as attackers would compromise all IMUs that share the same physical characteristics and communication protocols. Shielding methods are constrained by the physical properties of drone sensors and control boards; metal shields can interfere with wireless communication, increase the risk of overheating, degrade performance, and reduce payload capacity due to their weight (e.g., [15, 16]).

Thus, filtering or de-noising compromised IMU readings has been considered (e.g., [17, 18]). Traditional frequency-based filters struggle because attack signals can disperse across the in-band domain, so Deep Auto Encoders (DAE) have been deployed to learn the acoustic resonance signature [14]; however, as we show in this work, they are unable to cope with sensor saturation, unexpected sensor measurements, and communication drops. Moreover, they lack real-time guarantees for online recovery [14].

Instead of attempting to de-noise the compromised IMU measurements, a complementary attitude reconstruction method leverages geometric relationships to restore attitude

control [9]. Yet, this approach requires unrealistically high accuracy and a high update frequency (as fast as the IMU sampling rate) of the position sensors; this is unfeasible in real-world scenarios where GPS is usually sampled 10 times slower than IMUs (e.g., $5 \sim 10\ Hz$ vs. $100\ Hz$).

Consequently, in this work, we introduce a *Model-based Anomaly detection and Recovery System* (MARS) to provide resiliency for UAVs against physical attacks on the inertial sensors. By utilizing tachometers that measure propeller rotational speeds, we identify an alternative solution to UAV body-frame state estimation, independent of IMUs. Tachometers are small, very low-cost sensors consisting of pulse counters and sensing probes, which can be easily installed on UAV arms. These sensors have optical sensing and magnetic sensing mode, and both cannot be compromised with acoustic and electromagnetic interference. As encoders, they can measure rotor speeds with high resolution and frequency, sufficient for real-time estimation and control [19].

By exploiting our knowledge (i.e., the model) of the physical dynamics of a UAV, we find that *rotor speed measurements can be combined with existing non-IMU position and heading data to estimate the net thrust and torque applied at the UAV's center of gravity*. This information is then used to produce body-frame attitude estimates through a *novel non-linear sensor fusion*. This approach offers resiliency to IMU attacks from two perspectives: first, it can be integrated with anomaly detection methods to accurately and swiftly detect IMU attacks and trigger system-level alarms; second, it enables position and attitude control in the absence of trustworthy IMU data. Based on these findings, we design a *flight recovery strategy* that effectively handles adversarial situations in UAV missions by detecting attacks, restoring stability, and continuing missions with minimal performance degradation.

In brief, MARS combines resilient sensor fusion, anomaly detection, and flight recovery, thereby safeguarding UAVs from corrupted inertial sensor measurements and maintaining an adequate level of the UAV flight performance. Our contributions can be summarized as:

• We introduce a general attack-resilient state estimator that does not require IMUs but incorporates position, heading and tachometer sensors to estimate net thrust and torque, and then generate high-fidelity body-frame orientation and angular velocity estimates.

• We design a MARS-based CUSUM sliding window anomaly detector (AD) that significantly outperforms existing ADs (e.g., [20, 21]) by swiftly and accurately detecting IMU attacks and providing system-level alarms for UAV security-aware decision-making.

• We develop a multi-stage flight recovery strategy that prioritizes security without significantly sacrificing performance. Our method is universally applicable to arbitrary UAV missions, regardless of the UAV's state, ensuring freedom in UAV maneuverability when equipped with the MARS detection and recovery framework.

• We evaluate MARS methodology on the PX4 open-source autopilot firmware [22, 23] in large-scale simulations as well as real-world physical experiments; we demonstrate MARS effectiveness, even against attacks for which other security methods (e.g., [9, 14]) fail.

The paper is organized as follows. Sec. 2 introduces preliminaries on nonlinear UAV models, IMU vulnerabilities and other related work. Sec. 3 presents the MARS methodology, based on a novel model-based resilient state estimator, MARS-based anomaly detection, and the multi-stage flight recovery. Sec. 4 presents large-scale evaluation in a simulated environment, compares MARS with existing frameworks in a hovering mission and explores the efficacy of flight recovery in a dynamic tracking mission. Sec. 5 presents real drone case studies with our customized MARS-PX4 autopilot and demonstrating MARS effectiveness in real-world implementations. We provide concluding remarks and discuss MARS limitations and future work (Sec. 6).

2 Preliminaries

2.1 UAV Modeling and Control

Nonlinear UAV/quadrotor model. UAVs are modeled as a nonlinear 6-Degree of Freedom (6-DoF) rigid body system with 3 translational and 3 rotational movements, utilizing 4 rotors to generate aerodynamic force for flight [24]. We consider two reference frames: the *Earth* (inertial frame) and the *body* frame. The Earth frame is defined as the right-handed North, East, Down (NED) frame, denoted by $\mathcal{F}_{\mathcal{E}}$. The right-handed body frame, defined as Forward, Right, Down (FRD) frame, is denoted by $\mathcal{F}_{\mathcal{B}}$. The rotation from the Earth to the body frame is captured by the rotation matrix $\mathbf{R}(\mathbf{q}) \in \mathbb{R}^{3\times3}$, characterized by quaternion $\mathbf{q} = \left[q_w, q_x, q_y, q_z\right]^T$. The translational motion is captured by the Earth frame positions expressed as $\xi = \left[x, y, z\right]^T$. The rotor speeds of the four propellers are given as $\omega = \left[\omega_1, \omega_2, \omega_3, \omega_4\right]^T$. App. 7.1 provides detailed description of the nonlinear model capturing the UAV physics (i.e., its *physical* dynamics).

Onboard sensing. A UAV relies on onboard sensors to achieve stable and precise flight control. The key sensors are IMU, compass, barometer, and GPS, which provide real-time attitude and position information. IMU units typically consist of accelerometers, gyroscopes, and magnetometers, measuring changes in the drone's linear velocity, angular velocity, and magnetic field. Also, sensors such as optical flow, depth cameras, radar, and Lidar can further improve the UAV's situational-awareness. In this paper, we focus on the two major components inside the IMU–accelerometers and gyroscopes, as well as the GPS and compass. These sensors produce body-frame attitude and angular velocity, earth-frame position, and heading measurements.

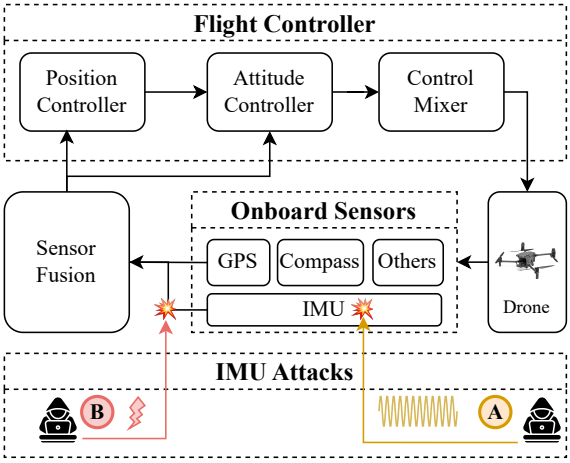

Figure 1: UAV onboard sensing and control architecture highlighting physical vulnerabilities of the IMUs: A) Acoustic resonant; B) Electromagnetic interference attack.

Standard UAV state-estimators. A standard UAV state estimator, used for fusion of the multi-modal sensing, is built upon GPS-IMU sensor fusion, where the IMU is *currently irreplaceable* due to its direct observation of fast-changing translational and rotational motions [25, 26].

Cascade position and attitude flight controllers. With accurate position and attitude estimates from an IMU-based state estimator, an efficient flight controller calculates the desired rotor speed commands for the four rotors. A commonly used control architecture is the cascaded position and attitude PID control: the position controller takes the position set-points provided by high-level modules and the position estimates from a low-level filter, producing attitude and thrust commands. The attitude controller then computes the error between the desired attitude and the current estimate, yielding the desired body-frame torque and thrust, which are translated into rotor speed commands by the control mixer. Fig. 1 illustrates the overall high-level sensing and control architecture.

2.2 Threat Model on IMU Attacks

In general, physical attack vectors on UAVs involve different spoofing properties, operations, victim characteristics and goals. They could affect GPS, LiDAR, Camera, IMU, audio and ultrasonic components available on UAVs (e.g., see [27]). In this work, we focus on the reported inertial sensor vulnerabilities. Recently, several physical attacks have compromised the IMU physical sensing mechanisms or disrupted communication pathways between the sensors and control units. Specifically, for the threat model from Fig. 1, adversaries can employ two ***physical*** attacks—*acoustic resonant attacks* or *electromagnetic interference attacks*, to achieve the ***attack objective***: *destabilize the low-level attitude control, leading to UAV crashes.* The following discusses the *attacker knowledge and capability* of these two types of attacks.

Acoustic resonant attacks. As MEMS devices, accelerometers and gyroscopes can experience resonance in their proof masses. Acoustic injection can induce resonant oscillations within these sensors, leading to significant fluctuations in the measurements. Attackers can exploit this inherent vulnerability by learning the acoustic resonance signature and identifying the susceptible range for successful attacks. The attacker can then propagate resonant signals using signal generators and transmitters to effectively trigger resonance in the UAV's onboard IMUs. A detailed description of acoustic resonant attacks is provided in App. 7.2.1.

Electromagnetic interference attacks. Attackers can also leverage electromagnetic interference to affect the data transmission between the IMU sensors and the flight controller board, resulting in communication failures [8]; these in turn, cause devastating estimation and control errors. The attack requirements, effect, and the comparison with acoustic resonant attacks are summarized in App. 7.2.2.

2.3 Anomaly Detection (AD) in UAVs

Model-based and data-driven AD methods have been explored for UAVs [28]. Model-based methods focus on the physical invariants of the drone dynamics [21], and have the advantage of simplicity and cost-efficiency [29], but require expert knowledge of the system dynamics and accurate estimation of system parameters [30]. Data-driven methods exploit the hidden characteristics in sensor/control data, without the need for complicated modeling of the physical platform. The real-time model-free data analysis may provide fast and accurate detection of sensor faults [31], but requires significant training data to fully utilize the generality of neural networks; otherwise, as we show, may not perform well.

3 MARS Methodology

MARS is a general anomaly detection and recovery framework that is universally applicable for defending UAVs against cyber-physical threats targeting the onboard IMUs. MARS uses the dynamical model (i.e., physics) of the UAV to quickly detect an attack on inertial sensors, and isolate the corrupted state estimation caused by the attacks as well as reconstruct the missing attitude information required by the UAV controllers; this is achieved by utilizing additional sensors that are unaffected by the IMU attack vectors. Specifically, when inertial sensors are under attack, additional low-cost rotor speed sensors (i.e., tachometers) may provide indirect body-frame attitude information using the unique UAV physical dynamics. Fig. 2 shows the high-level architecture of MARS.

MARS employs two sets of state estimators and control modes: *standard* and *resilient*. In the *standard mode*, which provides optimal control performance when the UAV is not

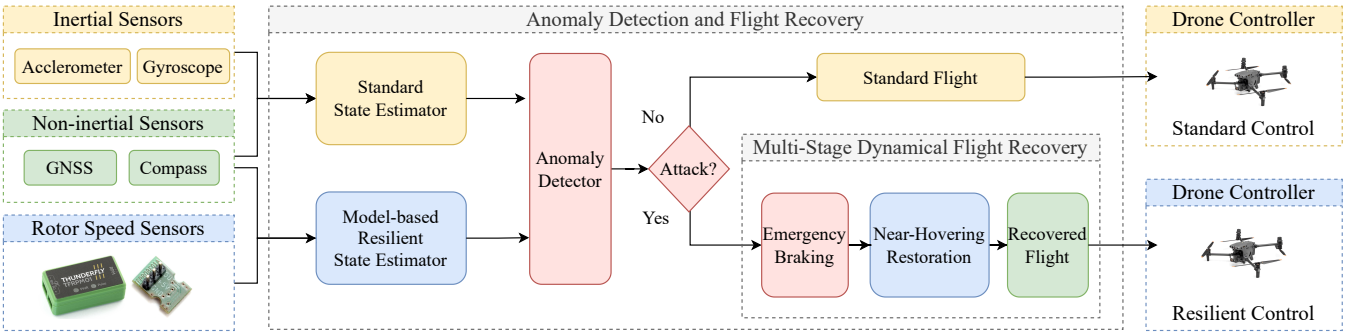

Figure 2: MARS architecture with resilient sensor hubs (i.e., all sensors except the IMU), AD, and multi-stage flight recovery, operating at two different modes: *performance-focused standard control* and *security-focused resilient control* mode.

under attack, conventional UAV sensors are used, including inertial accelerometers, gyroscopes, GPS, and compass. In the *resilient mode*, MARS rejects IMU readings and incorporates additional rotor speed measurements, achieving strong attack resiliency with *acceptable* control performance to continue mission flights. The switching between the modes is determined by a *real-time anomaly detector* (AD) that monitors the estimation residuals between the IMU measurements and the resilient state estimates. If an anomaly is detected, the system switches from the standard mode to the resilient mode under a *multi-stage dynamic flight recovery strategy*, securing the UAV into near-hovering conditions before resuming the flight.

We now describe MARS in more detail. We first introduce the model-based resilient state estimation (RSE), including the approach to estimate the UAV net thrust and torque from rotor speeds and earth-frame velocities; as well as fusing the high-fidelity thrust and torque estimates with other sensor measurements to obtain robust UAV attitude estimates using an Extended Kalman Filter (EKF) (Sec. 3.1 and Fig. 3). We then introduce the AD that combines CUSUM statistics and sliding window detection (Sec. 3.2), followed by the MARS multi-stage dynamic flight recovery strategy (Sec. 3.3).

3.1 Model-based Resilient State Estimation

Determining the body-frame attitude states, including the attitude quaternions and angular velocity, is crucial to perform real-time position and attitude control. In a standard UAV state estimator, the attitude information is acquired from IMU data, which are the measured angular velocities and accelerations that define the motion of the quadrotor. This is the reason why IMU attacks are so effective. On the other hand, we note that the attitude states can also be estimated from the ***net*** thrust (**f**) and torque (**τ**) applied at the quadrotor's center of mass; thus, we define $\mathbf{u} = \left[\mathbf{f}^{\mathcal{B}}, \boldsymbol{\tau}^{\mathcal{B}}\right]^T$. We now show that accurate estimates of the *net* torque and thrust can be obtained from the propeller rotational speed and earth-frame velocities *under near-hovering assumptions*. To achieve this we employ the vehicle's dynamical model.

3.1.1 Near-hovering Thrust and Torque Estimation

We consider the force $\vec{F}_i$ and torque $\vec{M}_i$ generated by the i^{th} propeller, applied on the UAV rotor's center of mass A_i, under near-hovering condition, which can be modeled as [1]:

$$\begin{aligned}\vec{F}_i &= -a\omega_i^2\vec{k}_b - \omega_i\left(\lambda_1\vec{V}_{A_i}^{\perp} - \lambda_2\vec{\Omega}\times\vec{k}_b\right)\\ &\quad + \varepsilon_i\omega_i\left(\lambda_3\vec{V}_{A_i}\times\vec{k}_b - \lambda_4\vec{\Omega}^{\perp}\right)\\ \vec{M}_i &= -b\varepsilon_i\omega_i^2\vec{k}_b - \omega_i\left(\mu_1\vec{V}_{A_i}^{\perp} + \mu_2\vec{\Omega}\times\vec{k}_b\right)\\ &\quad - \varepsilon_i\omega_i\left(\mu_3\vec{V}_{A_i}\times\vec{k}_b + \mu_4\vec{\Omega}^{\perp}\right);\end{aligned} \tag{1}$$

here, ω_i is the propeller rotational angular velocity with direction ε ($\varepsilon > 0$ represents counter clock-wise (CCW) revolution and $\varepsilon < 0$ clock-wise (CW) revolution); $\vec{\Omega}$ and $\vec{\Omega}^{\perp}$ denote the angular velocity of the quadrotor and its perpendicular projection on the rotor plane; unit vector $\vec{k}_b$ points down and is perpendicular to the rotor plane; $\vec{V}_{A_i}$ and $\vec{V}_{A_i}^{\perp}$ represent the earth-frame linear velocity and the its perpendicular projection on the rotor plane; and $a, b, \lambda_1, \lambda_2, \lambda_3, \lambda_4, \mu_1, \mu_2, \mu_3, \mu_4$ are rotor's physical parameters.

While near-hovering, the term associated with body-frame angular velocity can be neglected, and the four rotors have (almost) the same earth-frame velocities as the quadrotor center of mass. Thus, (1) can be simplified as:

$$\begin{aligned}\vec{F}_i &\approx -a\omega_i^2\vec{k}_b - \omega_i\lambda_1\vec{V}_{A_i}^{\perp} + \varepsilon_i\omega_i\lambda_3\vec{V}_{A_i},\\ \vec{M}_i &\approx -b\varepsilon_i\omega_i^2\vec{k}_b - \omega_i\mu_1\vec{V}_{A_i}^{\perp} - \varepsilon_i\omega_i\mu_3\vec{V}_{A_i}.\end{aligned} \tag{2}$$

From (2), under the near-hovering assumptions, the torque and thrust generated by an individual propeller are related only to the squared rotor speed and the earth-frame velocity of the UAV's center of mass. The net thrust and torque applied at the center of mass, denoted by C, can then be approximated as the sum (over 4 propellers/rotors)

$$\mathbf{f}^{\mathcal{B}} = \sum_{i=1}^{4}\vec{F}_i, \quad \boldsymbol{\tau}^{\mathcal{B}} = \sum_{i=1}^{4}\vec{CA}_i\times\vec{F}_i + \vec{M}_i. \tag{3}$$

From (3), the net torque and thrust of the quadrotor, which drive the drone dynamics as captured in Sec.7.1 (i.e.,(8)

and (9)) can be computed using individual propeller torque and thrust, which rely *only on rotor speed and earth-frame velocity measurements*. This approach enables UAV net thrust and torque estimation without onboard inertial sensors, which can be further tuned (Sec. 3.1.2) to facilitate robust estimation of the attitude state (Sec. 3.1.3).

3.1.2 Dynamic Torque Compensation

(3) provides a thrust and torque estimate under hovering conditions, using the rotor speed and earth-frame velocity measurements. Yet, drones are unlikely to remain in near-hovering conditions during various flights, and attacks may occur in both hovering and non-hovering situations. Thus, when an attack is detected, it is critical to temporarily 'stop' the drone, restore near-hovering, and then resume flight. To expedite the braking and restoration process and enhance the UAV's stability during this procedure, a dynamic torque compensation related to the body-frame velocity, which indicates the direction of local drone movement, can be added to (3).

The compensated torque can be expressed as:

$$\boldsymbol{\tau}_{cp}^{\mathcal{B}} = \boldsymbol{\tau}^{\mathcal{B}} + K_{cp}\vec{V}_C^{\mathcal{B}} + \boldsymbol{\tau}_b^{\mathcal{B}}, \tag{4}$$

where $K_{cp} = \left[k_{cp,x}, k_{cp,y}, k_{cp,z}\right]^T$ is a set of tuned coefficients from experiments for the braking mechanism; $\vec{V}_C^{\mathcal{B}}$ is the projection of the earth frame linear velocity $\vec{V}_C$ onto the body frame. Here, we only consider the rotation along the Z-axis characterized by the measured yaw angle ψ from the compass in this projection, as roll and pitch angles are not directly accessible. Also, $\boldsymbol{\tau}_b^{\mathcal{B}}$ is the torque bias term used to tune the net torques for optimal hovering stability, and K_{cp} and $\boldsymbol{\tau}_b^{\mathcal{B}}$ can be experimentally tuned for best performance.

The intuition behind this torque compensation for UAV 'braking' is the coupling effect of the quadrotor's translational and rotational movements—when the UAV is moving at a certain velocity, it has a corresponding roll and pitch angle that enables it to move in that direction. Hence, to 'stop' the drone, an opposite angular momentum is needed, which can be achieved by adding a torque vector in the correct direction. The faster the drone is moving, the larger the opposite angular momentum that needs to be applied.

To fully estimate the net thrust and torque, we need to consider all contributing factors, including not only the rotation speed of the four rotors but also the effects of propeller angular acceleration and the quadrotor's body-frame angular velocity. The challenge lies in the absence of useful information from the IMUs when they are under attack. Hence, the estimate of the thrust and torque solely using propeller speeds and earth-frame velocities will inevitably deteriorate as the drone moves further away from near-hovering conditions and exhibits more acute dynamic behavior. This is an intrinsic limitation when trustworthy accelerometers and gyroscopes are unavailable. Still, we demonstrate that even with this level of thrust and torque estimation, we can design a *nonlinear resilient state estimator that provides accurate body-frame estimates*, enabling drone attitude controllers to stabilize the drone and prevent crashes.

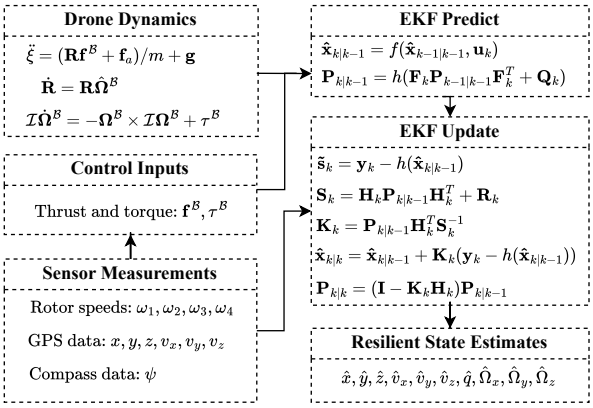

Figure 3: Model-based resilient state estimator architecture.

3.1.3 EKF-based Attitude Estimation

Typically, IMUs directly provide angular velocity and acceleration information $\boldsymbol{\Omega}^{\mathcal{B}} = \left[\Omega_x, \Omega_y, \Omega_z\right]^T$ and $a = \left[a_x, a_y, a_z\right]^T$ for the drone controllers as the information is directly observable from the accelerometer and gyroscope sensors. However, when IMU's sensor readings cannot be trusted, it is critical to estimate the missing attitude information (i.e., the attitude state – both the attitude quaternion and body-frame angular velocity) from another robust source; we achieve this using the combination of dynamic net thrust and torque estimates, and the remaining unaffected sensors. This is a nonlinear state estimation problem, and we adopt the EKF to achieve better accuracy than linear estimation models.

To capture the UAV's motion and enable necessary position and attitude control, we define the state variable $\mathbf{x} = \left[x, y, z, v_x, v_y, v_z, q_w, q_x, q_y, q_z, \Omega_x, \Omega_y, \Omega_z\right]^T$, which includes the position, velocity, attitude and body-frame angular velocity. After IMU attacks, we rely on position and heading sensors, such as GPS and compass to provide *noisy* observation data $\mathbf{y} = \left[x, y, z, v_x, v_y, v_z, \psi\right]^T$. We also use tachometers to provide rotor speed measurements $\omega = \left[\omega_1, \omega_2, \omega_3, \omega_4\right]^T$, then used to estimate the net thrust and torque $\mathbf{u} = \left[\mathbf{f}^{\mathcal{B}}, \boldsymbol{\tau}^{\mathcal{B}}\right]^T$, as discussed in Sec.3.1.1-3.1.2.

The intuition behind this estimation approach lies in replacing the IMU measured acceleration and angular velocity to thrust and torque from individual rotor speeds to perform body-frame attitude and angular velocity estimation, incorporating the UAV dynamics. Fig. 3 illustrates the process of generating resilient state estimates by incorporating sensor measurements with the drone's physical dynamics under the

condition of compromised IMUs. A detailed formulation for this nonlinear EKF is provided in App. 7.3 – note that we used the vector $\mathbf{u} = \left[\mathbf{f}^{\mathcal{B}}, \boldsymbol{\tau}^{\mathcal{B}}\right]^T$ to denote the net thrust and torque estimates, as they provide inputs to the UAV dynamical model used in the EKF formulation.

3.2 Statistical Anomaly Detection

To detect and respond to ongoing IMU attacks, an efficient AD is required. Statistical anomaly detection algorithms, including χ^2, cumulative sum (CUSUM) and sequential probability ratio test (SPRT) have been deployed in robotic systems (e.g., [21, 32, 33]). Machine learning (ML)-based ADs have also been adopted for discovering embedded attack signatures from system logs (e.g., [34]). In App. 7.4.1, we provide a setup of anomaly detection as a hypothesis testing problem.

3.2.1 MARS-based CUSUM detection

To practically solve the hypothesis testing problem, we focus on one outstanding statistics, the *MARS resilient estimate residual*, as the anomaly score to 'evaluate' the level of anomaly. This anomaly score represents how much the observed data deviates from the estimation, and can raise an alarm if that deviation exceeds some manually set threshold. In our design, we adopt CUSUM statistics on the estimate produced by the MARS resilient state estimator.

Specifically, we continuously monitor accelerometer and gyroscope measurements and compute the residual:

$$r_k = Y_{s,k} - h_r(\hat{x}_{r,k}), \tag{5}$$

where r_k is the residual between current resilient estimate $\hat{x}_{r,k}$ and current inertial sensor observation data $Y_{s,k}$, computed through the observation function h_r. Then the CUSUM statistics describing this difference is: $S_k = max(0, S_{k-1} + r_{k-1} - b)$, where S_k and S_{k-1} are the current and previous step CUSUM statistics; r_{k-1} is the previous step estimate residual and b is a constant to control the accumulation of error.

If CUSUM statistics is larger than a pre-defined threshold λ, then a point anomaly α_k is raised, and the current CUSUM statistics is set back to zero ($S_k = 0$); otherwise, this point is non-anomaly: $\alpha_k = \begin{cases} 1, & \text{if } S_k > \lambda, \\ 0, & \text{otherwise.} \end{cases}$

3.2.2 Sliding window in online detection

While CUSUM statistics reflect the accumulation of error over time, simply relying on point anomalies alone increases the possibility of false positives especially when the drone experiences sudden disturbances. Therefore, in online detection scenarios, we adopt a sliding-window technique to further analyze the CUSUM statistics and build a more robust anomaly decision-making process. The status of a system anomaly is determined based on the ratio of CUSUM anomalies within a certain time window. By grouping the point anomalies into a sliding window with length l and defining a system anomaly indicator, the current detection rate DR_k is

$$DR_k = \frac{1}{l} \sum_{i=0}^{l} \alpha_{k-l+i+1}. \tag{6}$$

Based on this detection rate, we can check if the UAV is under attack and if further action needs to be taken:

$$\alpha_{s,k} = \begin{cases} \text{True}, & \text{if } DR_k > p, \\ \text{False}, & \text{otherwise,} \end{cases} \tag{7}$$

where p refers to the false positive rate that the system tolerates without giving an alarm. Since the existing attacks on inertial sensors are generally destructive with significant impact at time of attack, we choose a relatively small $p = 0.005$. Defining the starting time of the AD at $k = 0$, a detailed description of our detection and recovery process is shown in Alg. 1, where $Y_{s,k}$ and $Y_{r,k}$ are sensor measurements in the standard and resilient control modes – i.e., $Y_{s,k}$ contains IMU readings and other sensor measurements, while $Y_{r,k}$ excludes IMU readings but includes rotor speed measurements.

Adding a sliding window on top of CUSUM anomalies inevitably increases the AD response time if an actual attack occurs. However, this window reduces the likelihood of false

Algorithm 1 MARS Anomaly Detection

Input: $Y_{s,k}$ for sensor readings in standard control mode, and $Y_{r,k}$ for sensor reading in resilient control mode.
Output: System anomaly $\alpha_{s,k} \in \{True, False\}$.
Initialization: $k = 0, S_{-1} = 0, r_{-1} = 0$.
while system is running **do**
  Update current sensor readings $Y_{s,k}$ and $Y_{r,k}$
  Standard estimate $\hat{x}_{s,k} = f_s(Y_{s,k})$
  Resilient estimate $\hat{x}_{r,k} = f_r(Y_{r,k})$
  Estimate residual $r_k = Y_{s,k} - h_r(\hat{x}_{r,k})$
  CUSUM statistics $S_k = max(0, S_{k-1} + r_{k-1} - b)$
  if $S_k > \lambda$ **then**
    CUSUM point anomaly $\alpha_k = 1$
    $S_k = 0$
  else
    CUSUM point anomaly $\alpha_k = 0$
  end if
  Detection rate $DR_k = \frac{1}{l} \sum_{i=0}^{l} \alpha_{k-l+i+1}$
  if $DR_k > p$ **then**
    System anomaly $\alpha_{s,k} =$ True. System under attack, $\hat{x}_{r,k}$ is sent to drone controllers.
  else
    System anomaly $\alpha_{s,k} =$ False. No attack detected, $\hat{x}_{s,k}$ is sent to drone controllers.
  end if
end while

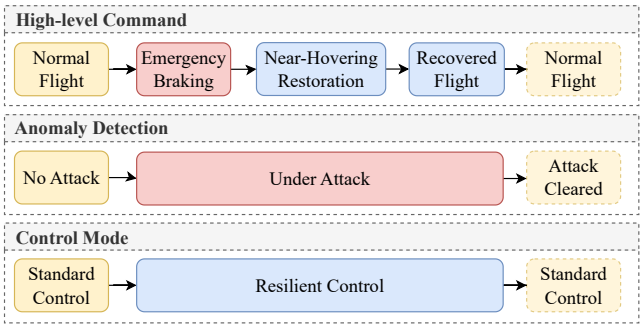

Figure 4: Multi-stage dynamical recovery for arbitrary autonomous missions under the coordination of high-level command, AD status and control mode selection.

switches from the standard control to the resilient mode in the context of online detection and recovery. Later, in our evaluations, we demonstrate that this trade-off does not affect the recovery capability of MARS.

The main novelty of MARS anomaly detection lies in the use of the MARS resilient estimate residual, rather than relying on a specific statistical algorithm. Hence, besides CUSUM, other algorithms that track historical anomalies can be tuned to work with MARS. MARS anomaly detector outperforms existing benchmarks in both detection rates and time-to-detection statistics, discussed in Section 4.3.

3.3 Multi-Stage Dynamical Flight Recovery

While stable hovering ensures the capability of surviving inertial sensor attacks, it is highly likely for UAVs to encounter attacks during more dynamic mission segments. Thus, it is necessary to analyze the potential for recovering during e.g., aggressive maneuvers, beyond just hovering. However, as is discussed in Sec. 3.1, when a UAV is moving at a high speed, it deviates from near-hovering conditions, increasing the error in the thrust and torque estimates. This means that at the time of an attack, if the drone has a large pitch or roll angle, the torque and thrust estimates would suffer, providing less accurate attitude information.

Thus, we need to design a multi-stage dynamical recovery strategy to handle attack regardless of the maneuver the UAV is performing during the attack. First, a 'braking' stage is necessary for the UAV to restore near-hovering conditions (as discussed in Sec. 3.1.2), allowing it to stabilize using accurate resilient state estimates. After recovering from the impact of the attacks and achieving a stable hovering state, the UAV resumes its mission under resilient control until the threats are cleared. This is summarized in Fig. 4; the high-level controller (i.e., command) sends an immediate braking command once the onboard AD detects an anomaly. Simultaneously, the controller switches to the resilient mode to begin the recovery process. The UAV then enters the near-hovering restoration stage, where the drone's control objective is set to hovering, in order to enable obtaining accurate resilient control estimates and optimal resilient control performance. Once the UAV achieves stable hovering, it can resume its mission using the resilient control, entering the next stage: *recovered flight* (using the estimator from Sec. 3.1.3). Finally, if the attack stops (i.e., AD flag clears), the UAV switches back to normal flight, fully using its IMU.

4 Attack Detection and Recovery Analysis

4.1 MARS Implementation

We implemented MARS in the open-source autopilot PX4 [22, 23]; a customized MARS-PX4 autopilot with MARS modules was integrated into the PX4 uORB messaging environment and incorporated Robot Operating System (ROS) and MAVLink communication protocols for data exchange (Fig. 5). For evaluation, we also implemented realistic inertial sensor attacks following the approaches from [3, 5, 8]. For simulation-based evaluations, we used the open-source vehicle simulation platform Prometheus [35], powered by ROS and the Gazebo robot simulator.

The ROS component consists of MARS high-level controllers, where the MARS recovery module and the position controller are instantiated, interacting with MAVLink communication through the MAVROS package. MAVLink messaging interconnects ROS high-level communication with PX4 uORB messaging via MAVLink streaming and receiving mechanisms, which are specially customized to transmit resilient state estimates. The core flight controller modules inside PX4 receive control commands from the high-level controllers, compute the desired rotor speeds, and send them to the Electronic Speed Control (ESC) units. The state estimates required by the controllers can be provided either by the PX4 standard state estimator or our MARS Resilient State Estimator (MARS-RSE), as determined by the MARS Anomaly Detector (MARS-AD).

We first conducted MARS analysis in Software-in-the-Loop (SITL) environments, 'connecting' Gazebo simulated drones controlled by our customized MARS-PX4 autopilot. Tables 7 and 8 in App. 7.5.1 list the virtual sensors, estimators, and controller specifications for PX4 SITL evaluations.

Using this setup, we initially evaluated the accuracy of MARS resilient state estimation by computing the thrust and torque and angular information estimate error (Sec. 4.2). Then, for four different inertial sensor attacks, we evaluated the effectiveness of MARS-AD against benchmark detectors (Sec. 4.3). Moreover, we evaluated the complete MARS framework on drone hovering missions and compare its performance to three representative inertial sensor recovery approaches from [9, 14]. We demonstrated the superiority of MARS over existing methods in survival time and control smoothness (Sec. 4.4). Finally, we evaluated the MARS multi-stage dynamical recovery strat-

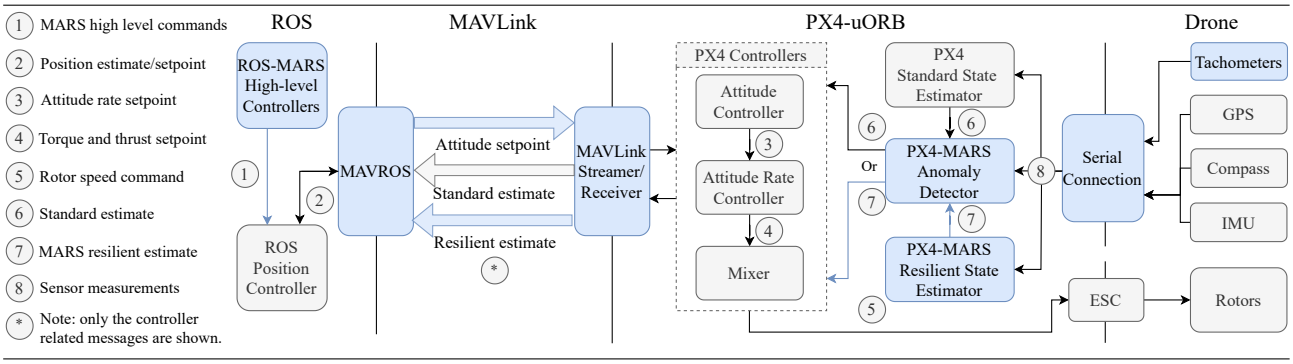

Figure 5: MARS-PX4 autopilot architecture. Gray blocks: PX4 default modules; blue blocks: our customized or specially tailored PX4 modules and communication bridges for MARS compatibility in the PX4 autopilot ecosystem.

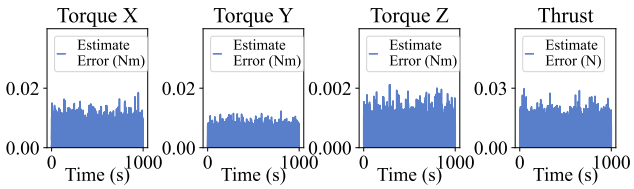

Figure 6: FRD frame thrust and torque estimation errors.

egy in dynamic tracking missions (Sec. 4.5), showing MARS capability to complete flight missions even under attack, with only small performance degradation.

4.2 Resilient State Estimate Effectiveness

The MARS resilient state estimator is designed to provide accurate state estimates for drone controllers, to ensure stable flights even under attack. The sensor group that feeds into the MARS resilient state estimator consists of position, heading, and rotor speed sensors. The key to MARS resilient state estimation is leveraging tachometer rotor speed measurements to calculate quadrotor thrust and torque, which are then used to generate body-frame attitude and angular velocity state estimates using the EKF. These estimates must meet the requirements of the drone attitude controller in terms of *high accuracy* and *proper update frequency*.

To verify the accuracy, we conducted sample runs in the SITL environment with virtual tachometers providing rotor speed measurements. Comparing with the ground truth torques and thrust applied at the center of gravity of the simulated quadcopter, Fig. 6 shows torque and thrust estimate error in a 1000 *s* hovering sample run; the X-Y axis torque error stays below 0.02 *Nm*, the Z axis below 0.002 *Nm*, and the thrust below 0.03 *N*, way within the required levels.

With stable and accurate estimates for the applied thrust and torque, Fig. 7 shows that MARS resilient state estimator produces high-fidelity attitude and body-frame angular velocity data, enabling the drone to swiftly recover attitude

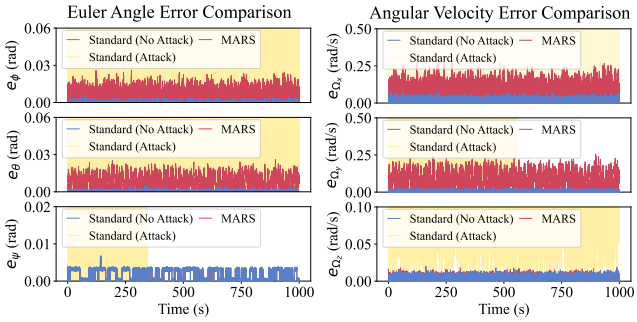

Figure 7: Attitude and angular velocity estimate error comparison between MARS resilient state estimator and standard estimator. Left: Euler angle; right: angular velocity.

control disrupted due to attacks on inertial sensors. Specifically, Fig. 7 compares the estimation error for three different scenarios: standard estimator without attack (blue), standard estimator under attack (yellow line), and MARS-RSE regardless of attack (red). While the standard estimator achieves optimal accuracy without attacks, due to the use of IMUs, it is highly vulnerable to IMU attacks (note the levels of the yellow line). In comparison, MARS-RSE achieves low error levels that allow for flight stabilization while being *robust to IMU attacks*. The Z-axis estimates have much lower error, both in angle and angular velocity, as the yaw angle is directly observable from the compass sensor. In Sec. 4.4, we show that these level of accuracy is sufficient for reconstructing stable attitude control under IMU attacks.

4.3 Detection of Inertial Sensor Attacks

MARS-RSE is only effective when IMU attacks are quickly and accurately detected, prompting a timely switch to resilient control. Thus, we also evaluated the MARS-AD (i.e., accuracy and time-to-detection), focusing on both the reported and potentially more subtle IMU attacks.

4.3.1 Design of Inertial Sensor Attacks

We consider two categories of feasible and destructive inertial sensor attacks: acoustic resonant attacks [3, 5] and EMI attacks [8]. For the former, we examine three different attack variations: Denial of Service (DoS), Side-Swing, and Switch attacks. The DoS attack involves un-modulated resonance injection [5]; whereas Side-Swing and Switch attacks are modulated versions of the DoS attack through attack manipulation. Intuitively, the Side-Swing attack retains only the positive or negative components of the attack signal, setting the unwanted half to zero. The Switch attack flips the unwanted half to the opposite direction, resulting in entirely positive or negative attack values. All three resonant attacks account for the sampling drift effect, which refers to the imprecise sampling intervals of the sensors, causing the attack signal to disperse over the bandwidth. To evaluate MARS under the worst-case scenarios, we inject resonant signals into all three sensing axes of both accelerometers and gyroscopes simultaneously, achieving the maximum possible oscillation.

Unlike acoustic resonant attacks, the effects of EMI attacks are more unpredictable, potentially causing sensor saturation, inaccurate values from unexpected data packets, and persistent data loss [8]. To maximize the attack impact, we focus on the representative case of sensor saturation for EMI attacks, injecting signals at the sensing limits into both accelerometers and gyroscopes. More details about attack design are provided in App. 7.5.2 (e.g., Fig. 19 and Table 11).

4.3.2 Accuracy and Time-to-Detection for MARS-AD

To evaluate the MARS-AD based on the MARS-RSE residual and the CUSUM statistical test, we compare it with two benchmark ADs: standard χ^2 and standard CUSUM. Both benchmarks utilize the Mahalanobis distance produced by an IMU-based standard state estimator as their decision statistic but employ different statistical algorithms. All ADs employ the same sliding window approach to determine the system anomaly status from point anomalies. For thorough comparison, we considered 10 datasets for each of the four attack profiles, where each dataset consists of a no-attack window and an attack window of the same length (20 s), resulting in a total of 40 datasets $\times$ 20 s each $\times$ 250 Hz sampling rate $=$ 20,000 data points. Moreover, the maximum attack power is used as the worst-case scenario aiming at drone crashes.

Fig. 8 illustrates the detection accuracy of the three detectors using Receiver Operating Characteristic (ROC) curves; e.g., when the false alarm rate is set at 0.01, for AR-DoS attacks the true detection rates of χ^2, Standard CUSUM and MARS CUSUM are 0.792, 0.937, and 0.998 respectively. The results show that the MARS-AD outperforms the two standard estimator-based detectors by achieving higher true detection rates and lower false alarm rates.

In addition to accuracy, timely attack detection is critical for the success of subsequent real-time mitigation and recovery

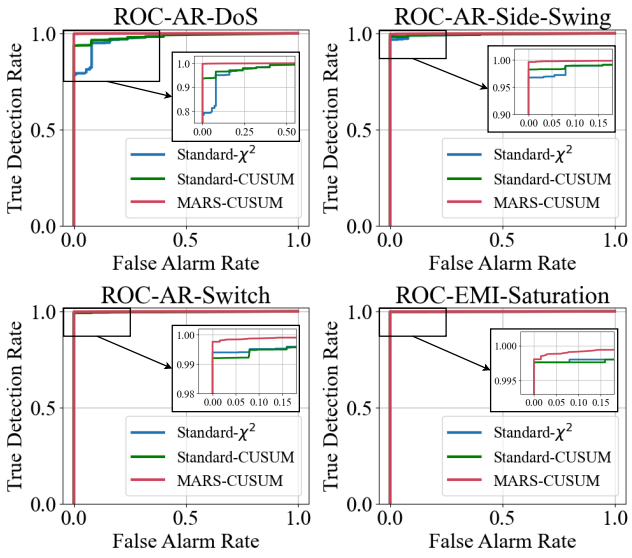

Figure 8: Anomaly detectors ROC curves.

Table 1: Anomaly detector average response time.

Anomaly Detectors	Anomaly Detector Average Response Time (s)			
	AR-DoS	AR-Side-Swing	AR-Switch	EMI-Saturation
Standard χ^2	1.512	0.323	0.060	0.024
Standard CUSUM	0.634	0.176	0.079	0.024
MARS CUSUM	0.024	0.039	0.024	0.020

measures. Table 1 shows the average AD response times for different attacks. Again, our MARS-AD achieves low detection times across all attack profiles, while the standard estimator-based detectors take longer to identify anomalies and struggle with less acute attack vectors.

Further, since the reported IMU attacks have overwhelmingly powerful effects, ADs detect them with reasonably good accuracy and response times (in Fig. 8, all 3 ADs have decent accuracy). However, it is highly likely that more sophisticated attack variants could be developed in cyber attack paths, or future physical IMU attacks might have capabilities to compromise the UAV system in a more stealthy manner. Thus, we also conducted tests with two more modulated attacks for the four attack profiles: *Step Amplitude* and *Ramp Amplitude*. The detailed results shown in App. 7.4.2, further demonstrate that the MARS-AD significantly outperforms the benchmark detectors even with these more stealthy attacks.

4.4 Hovering Stabilization under Attack

Once an anomaly is detected, the system switches to the resilient control mode to avoid immediate crashes. MARS involves reconstructing body-frame attitude information through a robust sensor fusion that incorporates position, heading, and propeller revolutions.

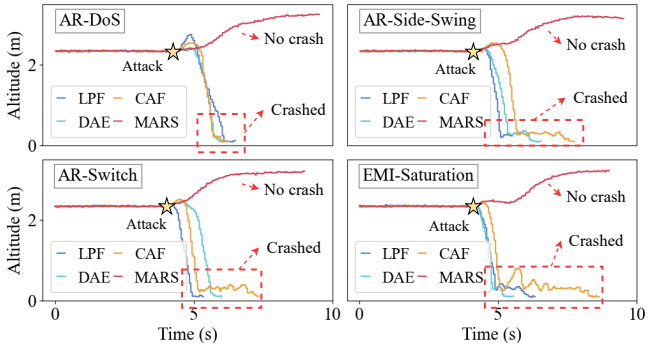

Figure 9: The UAV altitude in hovering missions under four inertial sensor attacks with different recovery frameworks.

We compared MARS with three existing recovery methods:

Low-pass filtering (LPF): LPF is one of the most commonly used heuristic filters in autopilots for its simplicity and effective noise reduction (e.g., [36]). We adopted the PX4 built-in low-pass filter functions with a 30 Hz cutoff frequency.

Deep-Auto-Encoder (DAE) [14] (App. 7.6.1 provides the training details). To ensure that the DAE model meets the 250 Hz update frequency required for attitude controllers, which was highlighted as one of the limitations in [14], we slowed down simulation by setting the real-time factor to 0.2 (i.e., 5x slower than real-time) and monitored the ROS data flow to confirm an actual 4 ms model inference time.

Complementary Attitude Feedback (CAF): CAF is an attitude reconstruction method using position information and geometric relationship [9]. It relies on position and heading data to geometrically calculate the Euler angles and feed them into drone controllers. Therefore, the update frequency of CAF is tied to the position sensor, not with the attitude controller, which is unrealistic in practice. For evaluation, we maintain a 250 Hz output frequency but only update the estimate at the GPS frequency.

4.4.1 Evaluation Results

We first tuned all methods to ensure their estimates can *control the drone stably in attack-free scenarios*. Hence, any emerging drone crash does not result from poor control design but from the unmitigated impact of the IMU attacks. Evaluations we done in SITL hovering missions with the UAV taking off and hovering at a given location when the IMU attacks start. Fig. 9 tracks the altitude of the quadrotor in the the hovering missions under the four recovery frameworks when four attack vectors were injected. The three representative methods failed to maintain the drone's stability, causing it to fall to the ground within 2 s, while MARS made the UAV survive the attacks.

For more comprehensive comparisons, we considered 4 attacks $\times$ 4 recoveries $\times$ 10 hovering missions, resulting in 160 trials. This generated approximately 160 trials $\times$ 20 s per trial

Table 2: Average Survival Time in Hovering Mission.

Recovery Methods	Hovering Mission Average Survival Time (s)			
	AR-DoS	AR-Side-Swing	AR-Switch	EMI-Saturation
LPF	2.826	1.463	1.252	1.283
DAE	2.097	2.228	1.781	1.086
CAF	1.812	2.041	2.033	2.388
MARS	Survived	Survived	Survived	Survived

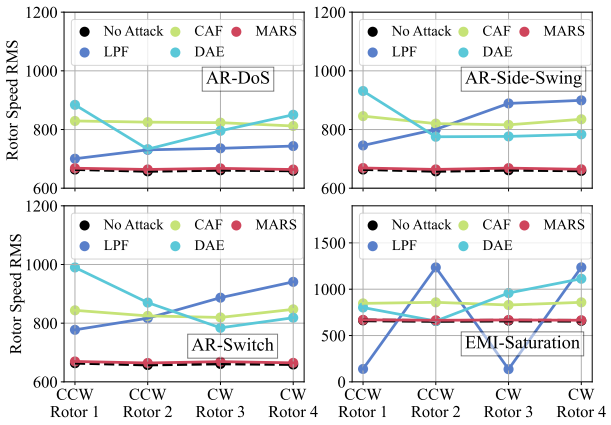

Figure 10: Average rotor speed fluctuation while hovering.

$\times$ 250 Hz sampling rate = 800k data points. In this evaluation, we focused on the *survival time* and *control smoothness* of the UAV by monitoring the time of staying in air under different attacks and the rotor speed fluctuations from the attack start to the UAV crash time. Table 2 summarizes the average survival times for the evaluated methods, and Fig. 10 shows the average rotor speed fluctuations.

As shown, **all** previous methods were not able to keep the drone in the air under IMU attacks, with CAF slightly outperforming the other two (detailed discussion is provided in App. 7.6.2). On the other hand, MARS-based control design kept the UAV flying – *MARS is capable of providing timely and reliable control outputs to survive all types of IMU attacks.* MARS successfully survived all four analyzed attacks by immediately isolating the compromised sensors and acquiring body-frame attitude information through a robust sensor fusion that utilizes rotor speed measurements. MARS overcomes the limitations of filtering-based methods, as its performance is much less dependent on the specific profile of attack vectors. As an attitude reconstruction method, it outperforms CAF by leveraging reliable rotor speed measurements, fused in the EKF to capture subtle changes in drone attitude and provide accurate control feedback.

4.5 Dynamic Mission Recovery

We further tested MARS multi-stage dynamic recovery strategy in more dynamic flight scenarios (i.e., beyond hovering).

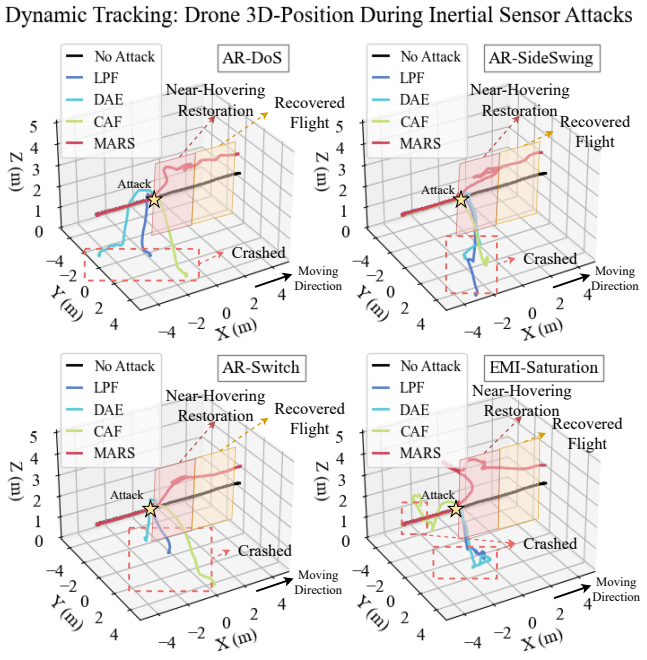

Figure 11: 3D drone position in a dynamic tracking mission under 4 IMU sensor attacks, for different recovery methods.

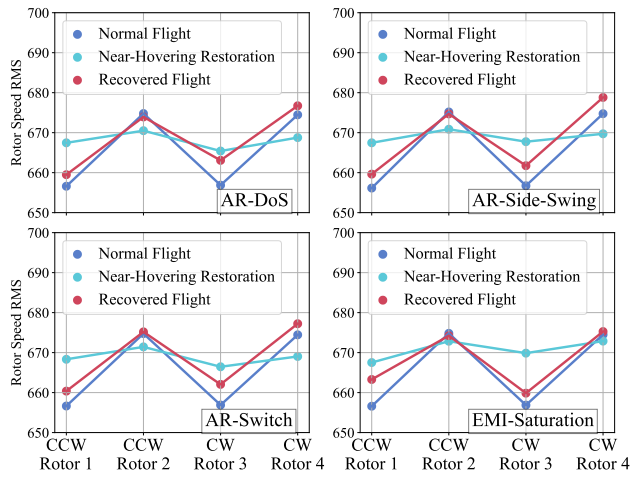

Figure 12: Average rotor speed fluctuation in RMS in the multi-stage dynamic recovery of dynamic tracking missions.

The UAV was tasked with completing a tracking mission along a 10 m trajectory on the X-axis in the Earth frame while maintaining a fixed altitude. Attacks on the inertial sensors were initiated mid-mission, and our objective was to complete the mission without crashing. Fig. 11 illustrates the 3D position of the drone in the dynamic tracking missions. Compared to the three existing methods, which all failed, MARS successfully implemented the multi-stage dynamic recovery process, which included 'braking' when the attacks were detected, restoring to a near-hovering condition, and then continuing the mission in the recovered flight mode.

Table 3: Average Tracking Error in Dynamic Missions.

Dynamic Tracking Mission		**Lateral Tracking Error (m) (RMSE)**		
Attack Vector	Recovery Method	No Attack	Recovered	Increment
AR-DoS	MARS	0.256	0.459	0.203
AR-Side-Swing	MARS	0.256	0.454	0.198
AR-Switch	MARS	0.256	0.362	0.106
EMI-Saturation	MARS	0.256	0.638	0.382

Table 4: Average Completion Time in Dynamic Tracking.

Dynamic Tracking Mission Completion Time (s)				
No Attack	AR-DoS	AR-Side-Swing	AR-Switch	EMI-Saturation
22.772	28.212	27.968	28.113	28.769

To evaluate MARS multi-stage dynamic recovery, we analyze the recovered flights from the perspective of the *control smoothness* of the UAV movement reflected by the rotor speed fluctuations, and whether the *mission was completed*, reflected by (a) the position tracking error in the lateral direction in RMSE, and (b) mission completion time. Our results indicate that MARS maintains rotor speed fluctuations at a low level, with the near-hovering restoration stage experiencing the smallest oscillation. The control smoothness of the normal flight was largely preserved in the recovered flight, with a rotor speed RMS difference of less than 10 rad/s. From the mission completion perspective, MARS achieved an average tracking error of approximately 0.5 m in the lateral direction, and the increase in completion time was below 30%, as shown in Tables 3 and 4.

Thus, *the MARS multi-stage dynamic recovery provides attack-resilience, both in terms of the control quality as well as mission completion time.* Although the braking mechanism temporarily suspends the drone's flight, the multi-stage dynamic recovery prevents significant control oscillations or UAV crashes. Also, the drone maintains control efficiency, completing the task with an acceptable tracking error.

5 Real-world Case Studies

We built a physical platform using commercially available quadrotor frames and onboard sensors, and deployed the modified MARS-PX4 autopilot. Demo videos are available at [37].

Experimental Setup (Fig. 13). We built a customized quadrotor with Pixhawk 6X flight controller and Holybro X500 V2 airframe, as well as an Intel NUC computer to generate high-level commands and record datasets. For safety concerns, we conducted experiments first in an indoor arena with VICON motion capture system (MOCAP) installed.

Initial onboard sensors included an IMU, a compass, and a barometer. To enable MARS, we added four ThunderFly TFRPM01 I2C bus-based drone tachometer sensors to the Pixhawk 6X board. Moreover, VICON MOCAP played two

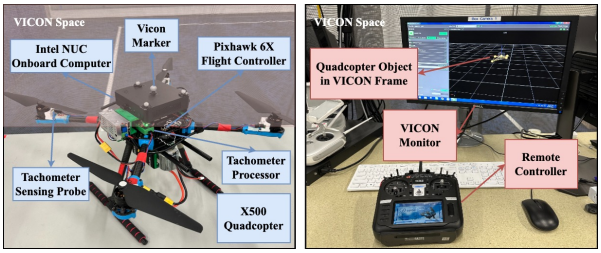

Figure 13: MARS-PX4 X500 physical platform setup.

roles in our experiments: it was used to simulate GPS measurements and compass for indoor environments at a frequency of 10 Hz, and to provide ground-truth data for analysis at 200 Hz. For MARS-based resilient control, *only MOCAP (to emulate GPS) and tachometer* measurements were utilized to generate state estimates for the drone controllers. The details on the sensing parameters (e.g., update frequency and data output), as well as the parameters of the autopilot modules are provided in App. 7.5.1 (Tables 9, 10).

MARS-PX4 Computation Overhead. We measured the computation overhead of MARS modules inside the PX4 ecosystem. On average, MARS only takes 0.261 ms to execute with minimal increase in CPU (1%) and memory usage (12 KB) (more results in App. 7.5.3 (Table 12 and Fig. 20).

5.1 Real-time Attacks and Anomaly Detection

Tachometer-based estimation of thrust and torque. In App. 7.5.4, our experimental results show that with tachometer data sampled at 200 Hz on X500, we accurately estimate thrust and torque from high quality rotor speed measurement, enabling resilient state estimate without IMUs.

MARS real-time resilient state estimate. We also evaluated the estimation accuracy of online MARS before feeding the estimates into the controller. We set the drone into hovering and focused on MARS-RSE error from the ground truth; from Fig. 14, the error level of online MARS-RSE is comparable to the one in SITL environments from Fig. 7. The results highlight the consistent MARS-RSE performance both in simulations and the real-world experiments.

Online attack generation. In our experiments, we used realistic attack vectors discussed in Section 4.3 into PX4 firmware.

Online detection results. Fig. 15 shows attack detection when the attack starts at $t_{attack} = 0$; MARS-AD identifies the IMU attacks with a significant increase in MARS estimate residual and sliding window detection rate. The attack is detected 0.035 s after it started. Table 5 shows the average detection time for 4 different attack profiles (10 attacks each) in real flights. The statistical results demonstrate MARS-AD effectiveness in both simulated and real-world scenarios.

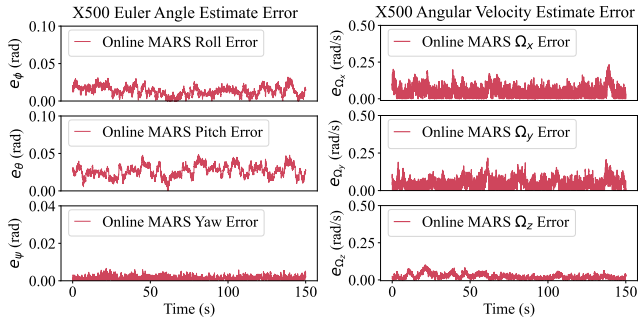

Figure 14: Online attitude and angular velocity estimate error of MARS running on the X500 quadrotor.

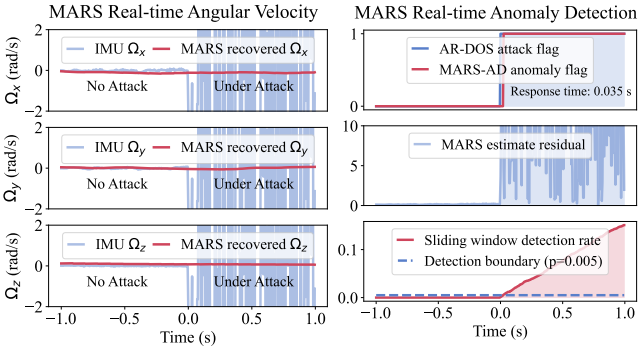

Figure 15: Real-time MARS signal recovery and anomaly detection results from the X500 quadrotor.

Table 5: MARS-AD average response time in real flights.

MARS Anomaly Detector Average Response Time (s)			
AR-DoS	AR-Side-Swing	AR-Switch	EMI-Saturation
0.030	0.065	0.026	0.024

5.2 Recovering Real-world Flights

We conducted four *real-world* flight case studies: (i) Hovering; (ii) Way-point visiting; (iii) Square tracking; (iv) Hovering under wind disturbance. In (i) experiments, a series of impulsive EMI saturation attacks of 0.5 s per event were created [38] to introduce sudden perturbations to test the responsiveness of MARS-AD and the smoothness of the controller *switching between standard and resilient modes*. In (ii) and (iii), we demonstrated the dynamic recovery capability in surviving attacks and completing missions against AR-DoS attacks. To further demonstrate MARS's robustness to dynamical changing environments, we created *wind disturbance* using a wind blower testing MARS against attacks under this challenging condition. The wind blower was set directly pointing at the drone from a distance of 3 m, and produced a wind disturbance at around 3 m/s at the hovering position (measured by a handheld anemometer).

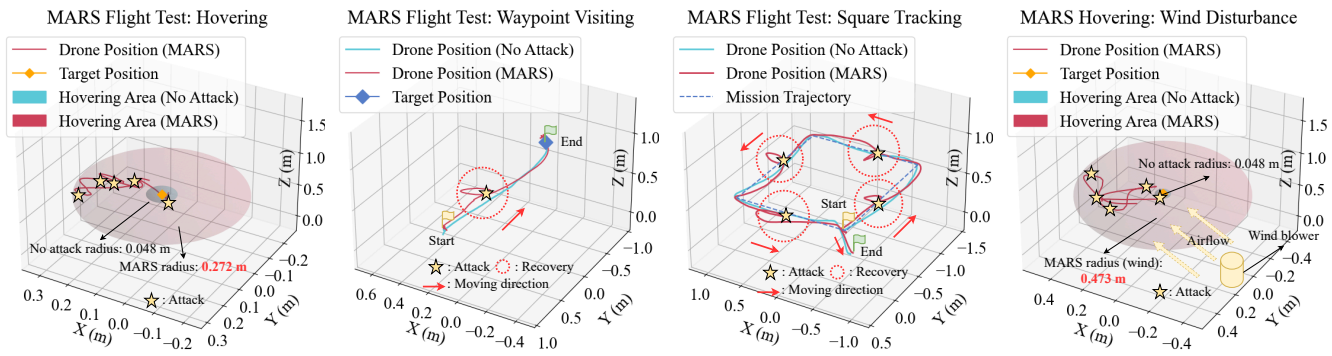

Figure 16: MARS real flight tests. Left to right: Hovering; Way-point visiting; Square tracking; Hovering under disturbance.

Table 6: MARS real-flight Euler angle fluctuations (*rad*).

Mission	Status	Roll Angle		Pitch Angle	
		Max Tilt	Std.	Max Tilt	Std.
Hovering	No Attack	0.012	0.003	0.063	0.002
	MARS	0.053	0.017	0.297	0.068
Way-point Visiting	No Attack	0.043	0.007	0.031	0.003
	MARS	0.184	0.015	0.152	0.013
Square Tracking	No Attack	0.036	0.007	0.077	0.013
	MARS	0.190	0.019	0.405	0.030
Hovering (Wind)	MARS	0.234	0.058	0.248	0.073

MARS successfully defended against attacks. Fig. 16 captures the drone trajectories, showing successful attack detection and recovery in various missions. In the hovering case, MARS detected the EMI saturation attack signals and switched to resilient control, making the drone hover near the target position, with the largest deviation at 0.272 *m*. In the dynamic missions, the multi-stage recovery strategy was effectively executed, pausing the movement of drone until attack is cleared, before continuing with the mission.

MARS minimized control performance loss through dynamic recovery. Table 6 shows the maximum tilt and the standard deviation of roll and pitch angles measured by the motion capture system. Due to the near-hovering assumption, MARS had good control performance in hovering, with the maximum tilt in pitch angle at 0.297 *rad*. The dynamic missions experienced acute behavior when MARS executed emergency braking, with the maximum tilt 0.184 *rad* in roll angle in way-point visiting mission, and 0.405 *rad* in pitch angle in square tracking case. Nevertheless, MARS maintained a low oscillation, showing stable mission execution.

MARS demonstrated robustness to dynamical environments. Fig. 16 and Table 6 show that despite the increment of position deviation to 0.473 *m*, with maximum tilts in roll and pitch angles 0.234 *rad* and 0.248 *rad* due to the wind, MARS managed to control the drone with acceptable stability. We also conducted *outdoor* tests; videos provided at [37].

6 Discussion and Conclusion

This paper introduced MARS, a model-based anomaly detection and recovery system that secures UAVs from attacks on inertial sensors. By isolating compromised IMUs and estimating thrust and torque from rotor speed measurements, a resilient state estimator reconstructs the missing body-frame attitude and angular velocity for the drone controller. We demonstrated the capability of MARS to achieve stable hovering and dynamical tracking by accurately detecting realistic attack vectors and actively executing the multi-stage dynamical flight recovery strategy. MARS is designed to be universally applicable for UAVs encountering different threats on inertial sensors. We now discuss MARS limitations and future work.

The Near-Hovering Assumption. It causes control estimate degradation when the drone experiences acute behaviors, and may lead to an increase in the estimation error. We addressed this problem by: (i) a fast AD reducing the deviation to the minimum; (ii) a temporary 'braking' mechanism pausing the movement and restoring the UAV to near-hovering; (iii) an appropriate torque compensation to facilitate the braking process. MARS aims to minimize the impact caused by inaccurate state estimates due to the deviation from near-hovering conditions under different dynamical missions. While it secures a moving drone from crashing, it sacrifices the control performance, with resilient but restricted operation. This is an intrinsic limitation of rotor speeds-based thrust and torque estimation when inertial sensors cannot be used.

Adaptability to Dynamical Environments. We demonstrated MARS robustness to wind disturbances. Such dynamic environmental changes may lead to increased estimation errors, thereby degrading control performance. To enhance MARS's adaptability to changing environmental conditions, also integrating an unknown input observer for disturbances would be beneficial; e.g., the Dryden Model [39], as in [21], enables accurate modeling of drone dynamics under wind influence. Also, learning-based system identification methods [40,41] could be used to adaptively update MARS system parameters, improving the accuracy of the resilient estimation.

Ethics Considerations

We have identified multiple stakeholders for this research: general UAV users for commercial or industrial purposes; researchers in UAV security; potential malicious parties who could misuse the identified sensor attacks. For the vulnerabilities discussed in this research, they are well-known attacks identified in prior works. Our work does not introduce new threats but focuses on mitigating the known ones. Additionally, we ensure no damage are made from our simulations and experiments, as they were conducted in controlled and isolated environments without interacting with live systems or real-world UAV users. After publication, we will reach out to work with UAV manufacturers incorporating our defenses into the latest UAV developments. Therefore, we consider it ethical and beneficial to introduce our work to the community to improve overall UAV resilience and prevent potential damage to UAV users and the general public.

Open Science

We adhere to the open science policy by making the source code, datasets and demonstrations for both simulations and real-world experiments available on our project website [37].

Acknowledgment

This work is sponsored in part by the ONR under agreement N00014-23-1-2206, AFOSR under the award number FA9550-19-1-0169, and by the NSF under NAIAD Award 2332744 as well as the National AI Institute for Edge Computing Leveraging Next Generation Wireless Networks, Grant CNS-2112562.

References

[1] Philippe Martin and Erwan Salaün. The true role of accelerometer feedback in quadrotor control. In *IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 1623–1629, 2010.

[2] Simon Castro, Robert Dean, Grant Roth, George T Flowers, and Brian Grantham. Influence of acoustic noise on the dynamic performance of mems gyroscopes. In *ASME International Mechanical Engineering Congress and Exposition*, volume 43033, pages 1825–1831, 2007.

[3] Yunmok Son, Hocheol Shin, Dongkwan Kim, Youngseok Park, Juhwan Noh, Kibum Choi, Jungwoo Choi, and Yongdae Kim. Rocking drones with intentional sound noise on gyroscopic sensors. In *24th USENIX Security Symposium*, pages 881–896, 2015.

[4] Hyungsub Kim, Rwitam Bandyopadhyay, Muslum Ozgur Ozmen, Z Berkay Celik, Antonio Bianchi, Yongdae Kim, and Dongyan Xu. A systematic study of physical sensor attack hardness. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 143–143. IEEE Computer Society, 2024.

[5] Yazhou Tu, Zhiqiang Lin, Insup Lee, and Xiali Hei. Injected and delivered: Fabricating implicit control over actuation systems by spoofing inertial sensors. In *27th USENIX Security Symposium*, pages 1545–1562, 2018.

[6] Denis Foo Kune, John Backes, Shane S Clark, Daniel Kramer, Matthew Reynolds, Kevin Fu, Yongdae Kim, and Wenyuan Xu. Ghost talk: Mitigating emi signal injection attacks against analog sensors. In *2013 IEEE Symp. on Security and Privacy*, pages 145–159, 2013.

[7] Mats G Backstrom and Karl Gunnar Lovstrand. Susceptibility of electronic systems to high-power microwaves: Summary of test experience. *IEEE Trans. on Electromagnetic Compatibility*, 46(3):396–403, 2004.

[8] Joon-Ha Jang, Mangi Cho, Jaehoon Kim, Dongkwan Kim, and Yongdae Kim. Paralyzing drones via emi signal injection on sensory communication channels. In *Netw. and Distributed Syst. Security Symp. (NDSS)*, 2023.

[9] Zhan Tu, Fan Fei, Matthew Eagon, Dongyan Xu, and Xinyan Deng. Flight recovery of mavs with compromised imu. In *2019 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 3638–3644, 2019.

[10] Hongjun Choi, Sayali Kate, Yousra Aafer, Xiangyu Zhang, and Dongyan Xu. Software-based realtime recovery from sensor attacks on robotic vehicles. In *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID)*, pages 349–364, 2020.

[11] Fan Fei, Zhan Tu, Dongyan Xu, and Xinyan Deng. Learn-to-recover: Retrofitting uavs with reinforcement learning-assisted flight control under cyber-physical attacks. In *IEEE ICRA*, pages 7358–7364, 2020.

[12] Lin Zhang, Xin Chen, Fanxin Kong, and Alvaro A Cardenas. Real-time attack-recovery for cyber-physical systems using linear approximations. In *2020 IEEE Real-Time Systems Symp. (RTSS)*, pages 205–217, 2020.

[13] Francis Akowuah, Romesh Prasad, Carlos Omar Espinoza, and Fanxin Kong. Recovery-by-learning: Restoring autonomous cyber-physical systems from sensor attacks. In *2021 IEEE 27th Int. Conf. on embedded and real-time computing systems and applications (RTCSA)*, pages 61–66, 2021.

[14] Jinseob Jeong, Dongkwan Kim, Joon-Ha Jang, Juhwan Noh, Changhun Song, and Yongdae Kim. Un-rocking drones: Foundations of acoustic injection attacks and

recovery thereof. In *Netw. and Distributed Syst. Security Symp. (NDSS)*, 2023.

[15] Patrice Joubert Doriol, Yamarita Villavicencio, Cristiano Forzan, Mario Rotigni, Giovanni Graziosi, and Davide Pandini. Emc-aware design on a microcontroller for automotive applications. In *2009 Design, Automation & Test in Europe*, pages 1208–1213, 2009.

[16] Shielding Geetha, KK Satheesh Kumar, Chepuri RK Rao, M Vijayan, and DC Trivedi. Emi shielding: Methods and materials—a review. *Journal of applied polymer science*, 112(4):2073–2086, 2009.

[17] Malek Karaim, Aboelmagd Noureldin, and Tashfeen B Karamat. Low-cost imu data denoising using savitzky-golay filters. In *2019 Int. Conf. on Comm., Signal Proc., and their Applications (ICCSPA)*, pages 1–5, 2019.

[18] Mundla Narasimhappa, Arun D. Mahindrakar, Vitor Campagnolo Guizilini, Marco Henrique Terra, and Samrat L. Sabat. Mems-based imu drift minimization: Sage husa adaptive robust kalman filtering. *IEEE Sensors Journal*, 20(1):250–260, 2020.

[19] Syed Javed Arif, Mohammad S. Jamil Asghar, and Adil Sarwar. Measurement of speed and calibration of tachometers using rotating magnetic field. *IEEE Trans. on Instrum. and Meas.*, 63(4):848–858, 2014.

[20] Nong Ye and Qiang Chen. An anomaly detection technique based on a chi-square statistic for detecting intrusions into information systems. *Quality and Reliability Eng. Int.*, 17(2):105–112, 2001.

[21] Raul Quinonez, Jairo Giraldo, Luis Salazar, Erick Bauman, Alvaro Cardenas, and Zhiqiang Lin. Savior: Securing autonomous vehicles with robust physical invariants. In *29th USENIX Security Symp.*, pages 895–912, 2020.

[22] Lorenz Meier, Dominik Honegger, and Marc Pollefeys. Px4: A node-based multithreaded open source robotics framework for deeply embedded platforms. In *IEEE ICRA*, pages 6235–6240, 2015.

[23] PX4 Development Team. Px4 official website. https://px4.io/, 2024.

[24] Mohd Khan. Quadcopter flight dynamics. *International journal of scientific & technology research*, 3(8):130–135, 2014.

[25] Paul G Savage. Strapdown inertial navigation integration algorithm design part 1: Attitude algorithms. *Journal of guidance, control, and dynamics*, 21(1):19–28, 1998.

[26] Paul G Savage. Strapdown inertial navigation integration algorithm design part 2: Velocity and position algorithms. *Journal of Guidance, Control, and dynamics*, 21(2):208–221, 1998.

[27] Yuan Xu, Xingshuo Han, Gelei Deng, Jiwei Li, Yang Liu, and Tianwei Zhang. Sok: Rethinking sensor spoofing attacks against robotic vehicles from a systematic view. In *IEEE 8th European Symp. on Security and Privacy (EuroS&P)*, pages 1082–1100, 2023.

[28] Lei Yang, ShaoBo Li, ChuanJiang Li, AnSi Zhang, and XuDong Zhang. A survey of unmanned aerial vehicle flight data anomaly detection: Technologies, applications, and future directions. *Science China Technological Sciences*, 66(4):901–919, 2023.

[29] Dingfei Guo, Maiying Zhong, and Donghua Zhou. Multisensor data-fusion-based approach to airspeed measurement fault detection for unmanned aerial vehicles. *IEEE Trans. Instrum. Meas.*, 67(2):317–327, 2017.

[30] Jian-hong Wang, Yan-xiang Wang, and Yong-hong Zhu. Bias compensation estimation in multi-uav formation and anomaly detection. *J. Control Syst. Eng*, 4:40–50, 2016.

[31] Rui Sun, Qi Cheng, Guanyu Wang, and Washington Yotto Ochieng. A novel online data-driven algorithm for detecting uav navigation sensor faults. *Sensors*, 17(10):2243, 2017.

[32] I. Jovanov and M. Pajic. Relaxing integrity requirements for attack-resilient cyber-physical systems. *IEEE Transactions on Automatic Control*, 64(12):4843–4858, Dec 2019.

[33] Cheolhyeon Kwon, Weiyi Liu, and Inseok Hwang. Analysis and design of stealthy cyber attacks on unmanned aerial systems. *Journal of Aerospace Information Systems*, 11(8):525–539, 2014.

[34] Moshe Kravchik and Asaf Shabtai. Detecting cyber attacks in industrial control systems using convolutional neural networks. In *2018 Workshop on cyber-physical systems security and privacy*, pages 72–83, 2018.

[35] Prometheus - open source autonomous drone project. https://github.com/amov-lab/Prometheus. Accessed: Nov. 13, 2024.

[36] Yalong Ban, Quan Zhang, Xiaoji Niu, Wenfei Guo, Hongping Zhang, and Jingnan Liu. How the integral operations in ins algorithms overshadow the contributions of imu signal denoising using low-pass filters. *The Journal of Navigation*, 66(6):837–858, 2013.

[37] MARS: Defending unmanned aerial vehicles from attacks on inertial sensors with model-based anomaly detection and recovery. https://sites.google.com/view/mars-uav-recovery/home.

[38] Wangli He, Feng Qian, Qing-Long Han, and Guanrong Chen. Almost sure stability of nonlinear systems under random and impulsive sequential attacks. *IEEE Transactions on Automatic Control*, 65(9):3879–3886, 2020.

[39] Department of Defense Handbook. *Flying Qualities in Piloted Aircraft*, 2012.

[40] Jeen-Shing Wang and Yen-Ping Chen. A fully automated recurrent neural network for unknown dynamic system identification and control. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 53(6):1363–1372, 2006.

[41] Wenhao Yu, Jie Tan, C Karen Liu, and Greg Turk. Preparing for the unknown: Learning a universal policy with online system identification. *arXiv preprint arXiv:1702.02453*, 2017.

[42] Sihao Sun, Angel Romero, Philipp Foehn, Elia Kaufmann, and Davide Scaramuzza. A comparative study of nonlinear mpc and differential-flatness-based control for quadrotor agile flight. *IEEE Transactions on Robotics*, 38(6):3357–3373, 2022.

[43] ICM-456XY Motion Sensor. https://invensense.tdk.com/products/motion-tracking/6-axis/icm-456xy/. Accessed: Nov. 13, 2024.

7 Appendix

7.1 Nonlinear UAV Model

The UAV physical dynamics can be captured with a standard nonlinear quadcopter model [42]. The translational motion is described by a standard Euler-Newton equation [42]

$$\ddot{\xi} = (\mathbf{R}\mathbf{f}^{\mathcal{B}} + \mathbf{f}_a)/m + \mathbf{g}, \tag{8}$$

where $\xi = \left[x, y, z\right]^T$ and m denotes the center of gravity and the total mass of the vehicle, and $\mathbf{g} = \left[0, 0, g\right]^T$ is the gravity acceleration vector; $\mathbf{f}^{\mathcal{B}} = \left[0, 0, T\right]^T$ is the force vector in the body frame for the collective thrust T applied at the center of mass; $\mathbf{f}_a$ is the aerodynamic drag force in the Earth frame $\mathbf{f}_a = \begin{bmatrix} -k_{d,x} v_x \\ -k_{d,y} v_y \\ -k_{d,z} v_z + k_h \left(v_x^2 + v_y^2\right) \end{bmatrix}$; here, $k_{d,x}, k_{d,y}, k_{d,z}, k_h$ are drag force coefficients and v_x, v_y, v_z are the earth-frame velocities.

The rotational motion can be captured as

$$\dot{\mathbf{R}} = \mathbf{R}\hat{\mathbf{\Omega}}^{\mathcal{B}}, \quad I\dot{\mathbf{\Omega}}^{\mathcal{B}} = -\mathbf{\Omega}^{\mathcal{B}} \times I\mathbf{\Omega}^{\mathcal{B}} + \boldsymbol{\tau}^{\mathcal{B}}, \tag{9}$$

where $\mathbf{\Omega}^{\mathcal{B}} = \left[\Omega_x, \Omega_y, \Omega_z\right]^T$ is the angular velocity in the body frame, $\hat{\cdot}$ denotes the operator that maps a vector in $\mathbb{R}^3$ to a skew-symmetric matrix, $I \in \mathbb{R}^{3\times 3}$ is the inertia matrix, and $\boldsymbol{\tau}^{\mathcal{B}} = \left[\tau_x, \tau_y, \tau_z\right]^T$ is the total torque vector in the body frame.

7.2 Attacks on Inertial Sensors

7.2.1 Acoustic Resonant Attack

We summarize the main properties of acoustic resonant attacks primarily include mathematical formulation, sampling drift effect, attacker manipulation, and attack requirements and effects [5].

Attack formulation: When the sensing mass is oscillating under the effect of acoustic injection, the digitized (i.e., sampled) sensor output signal $V[i]$ is represented by:

$$V[i] = A \cdot \sin(2\pi\nu \frac{i}{F_S} + \phi_0), \quad \nu = F - n \cdot F_S, \tag{10}$$

where A and ϕ_0 are the amplitude and phase of the oscillating signal, respectively; ν refers to the aliased frequency due to the Nyquist sampling theorem, where F is the frequency of the analog signal triggering the acoustic resonance, F_S is the sampling frequency, and n is the integer that describes the aliasing relationship between them.

Sampling drift: While the mathematical formulation of the acoustic attack is well-patterned, simply cutting off at the frequency $\frac{\nu}{F_S}$ is not sufficient to filter out the attack vectors. The reason is that the sampling interval of a real-time system is not perfectly maintained at $\frac{1}{F_S}$. A slight drift in the sampling rate, ΔF_S, can be amplified by n times into $\Delta\nu$ due to the aliasing effect:

$$\Delta\nu = (F - n \cdot (F_S + \Delta F_S)) - (F - n \cdot F_S) = -n \cdot \Delta F_S.$$

The typical resonant frequency of IMU sensors ranges from 20 kHz to 30 kHz, while the sampling frequency is often around 1 kHz – the above equation indicates that a shift of 0.01 Hz in the sampling rate will be amplified 100 times into 1 Hz in the induced frequency.

Attacker manipulation: The attacker is also capable of manipulating the amplitude and frequency of the acoustic resonant signals. The compromised sensor measurement is a combination of the benign signal and the acoustic resonance

$$\tilde{s}[i] = s[i] + A[i] \cdot \sin(\Phi[i]), \quad \Phi[i] = 2\pi\nu[i] \frac{i}{F_S} + \phi_0,$$

where $\tilde{s}[i]$ and $s[i]$ are the compromised and benign sensor readings from either the accelerometer or gyroscope due to the effects of acoustic resonant attacks. $A[i]$ and $\Phi[i]$ are the modulated amplitude and phase at the aliased frequency $\nu[i]$. The attacker takes advantage of amplitude tuning and phase shifting to induce different acoustic attack variations. The commonly used variations include Denial-of-Service (DoS),

Side-Swing, and Switch attacks. These variations pose significant challenges for defense mechanisms in detection, identification, and mitigation.

Attack requirement and effect: To successfully carry out an acoustic resonant attack, the attacker first identifies the target inertial sensors to determine their resonant frequencies. Then, using consumer-grade speakers or long-range acoustic devices, the attacker broadcasts an attack signal with proper modulation to strike the MEMS components of the victim sensors. Due to the strong oscillations in inertial sensor measurements caused by the resonance effect, the drone loses reliable attitude estimates within tens of milliseconds and faces the danger of losing stable control.

7.2.2 Electromagnetic Interference Attack

To successfully launch an EMI attack against a target drone [8], the attacker needs to identify the unique frequency range of the specific control board that is susceptible to electromagnetic interference and the required signal power to trigger the attack. Then, using antennas and RF generators, a physical attack path can be created to target the victim drone. The attacked communication channel could potentially cause unintended bit flips, incorrect values, sensor saturation, and temporary or persistent data loss. The corrupted data received at the control unit leads to serious oscillations in the sensor fusion module, further causing significant fluctuations in control output and rotor speeds [8].

Compared with acoustic resonant attacks, EMI attacks do not target the sensing properties of the IMUs but rather the communication link between them and the flight control unit. Unlike acoustic resonance, which can be mathematically formulated into unique patterns, EMI induces arbitrary effects where benign signals are largely overwhelmed or completely blocked due to data packet mismatches, the transmission of incorrect values, and communication losses.

7.3 Nonlinear Extended Kalman Filter (EKF)

This section summarizes the implementation of an EKF for the nonlinear quadrotor system. Starting from the continuous-time model summarized in Sec. 7.1, the state-space form of the quadrotor discrete-time dynamics is:

$$\begin{aligned} \mathbf{x}_k &= f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}) + \mathbf{w}_{k-1}, \\ \mathbf{y}_k &= h(\mathbf{x}_k) + \mathbf{v}_k, \end{aligned} \tag{11}$$

where $\mathbf{w}_k$ is the system disturbance, $\mathbf{y}_k$ is the vector of the raw sensor measurements with Gaussian noise $\mathbf{v}_k$.

EKF has a two-step process, predict step and update step. In the predict step, the state transition function f can be used to calculate the predicted state $\hat{\mathbf{x}}_{k|k-1}$ from the estimation of previous time $\hat{\mathbf{x}}_{k-1|k-1}$ and control input $\mathbf{u_k}$, and observation function h can be used to calculate the predicted covariance $\mathbf{P}_{k|k-1}$:

$$\begin{aligned} \hat{\mathbf{x}}_{k|k-1} &= f(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_k), \\ \mathbf{P}_{k|k-1} &= h(\mathbf{F}_k \mathbf{P}_{k-1|k-1} \mathbf{F}_k^T + \mathbf{Q}_k); \end{aligned} \tag{12}$$

here $\mathbf{F}_k$ is the Jacobian matrix of state transition function, $\mathbf{u_k}$ is control input, $\mathbf{P}_{k-1|k-1}$ is previous covariance estimate, $\mathbf{Q}_k$ is the covariance of process noise. The state transition and observation matrices $\mathbf{F_k}$ and $\mathbf{H_k}$ are the Jacobians of the state transition and observation functions:

$$\mathbf{F}_k = \frac{\partial f}{\partial x}|_{\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_k}, \qquad \mathbf{H}_k = \frac{\partial h}{\partial x}|_{\hat{\mathbf{x}}_{k|k-1}}. \tag{13}$$

In update step, the innovation residual $\tilde{\mathbf{s}}_k$, innovation covariance $\mathbf{S}_k$, Kalman gain $\mathbf{K}_k$, updated state estimate $\hat{\mathbf{x}}_{\mathbf{k}|\mathbf{k}}$ and updated covariance estimate $\mathbf{P}_{k|k}$ can be computed:

$$\begin{aligned} \tilde{\mathbf{s}}_k &= \mathbf{y}_k - h(\hat{\mathbf{x}}_{k|k-1}), \\ \mathbf{S}_k &= \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{R}_k, \\ \mathbf{K}_k &= \mathbf{P}_{k|k-1} \mathbf{H}_k^T \mathbf{S}_k^{-1}, \\ \hat{\mathbf{x}}_{k|k} &= \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \tilde{\mathbf{s}}_k, \\ \mathbf{P}_{k|k} &= (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k|k-1}, \end{aligned} \tag{14}$$

where $\mathbf{I}$ is the indentity matrix and $\mathbf{R}_k$ is the covariance of observation noise. From (14), we reconstruct the quaternion and body-frame angular velocity from position, heading and rotor speed sensor measurements.

7.4 Anomaly Detection Details

7.4.1 Anomaly Detection as Hypothesis Testing

Regardless of the AD type, it aims to solve a hypothesis testing problem. Given a time series sequence from sensor observation data $Y : \{Y_{k-l_0+1}, Y_{k-l_0}, ..., Y_{k-1}, Y_k\}$ of length l_0, the two hypotheses are:

$$\begin{aligned} &H_0 : \text{No attack, } Y_k \text{ was received;} \\ &H_1 : \text{Under attack, } \tilde{Y}_k \text{ was received.} \end{aligned} \tag{15}$$

Solving the hypothesis problem is to determine whether current observation data belong to the null hypothesis distribution H_0, or an alternative (but **unknown**) distribution H_1. The AD maps the current observation data into a binary classification: $D(Y_k) \to \{0, 1\}$, and the detection result is:

$$\begin{aligned} \text{True detection} &: D(Y_k) = 1, Y_k \in H_1, \\ \text{False alarm} &: D(Y_k) = 1, Y_k \in H_0. \end{aligned} \tag{16}$$

By denoting the probability of true detection as P_{TD} and false alarm as P_{FA}, we evaluate the effectiveness of ADs.

To solve the hypothesis testing problem (i.e., design an AD), we design a proper mapping for $D(Y_k) \to \{0, 1\}$.

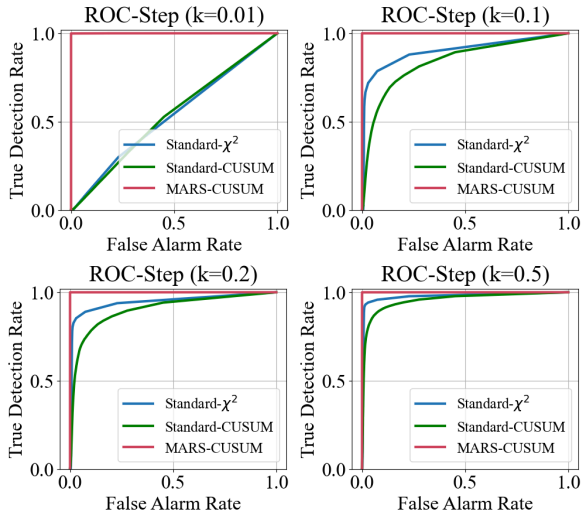

Figure 17: AD offline ROC curves with Step attacks.

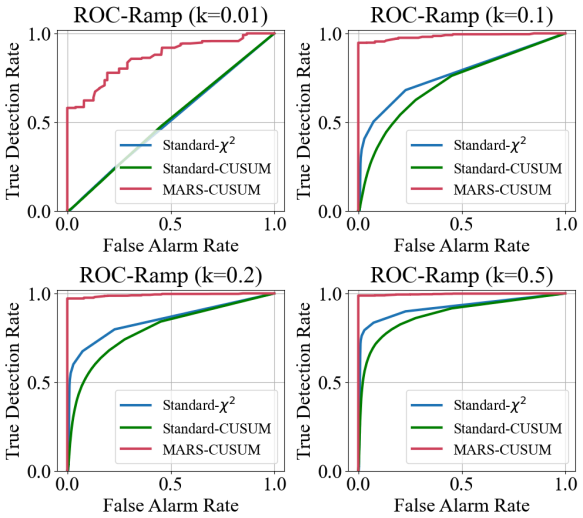

Figure 18: AD offline ROC curves with Ramp attacks.

7.4.2 Anomaly Detection for Modulated Attacks

We also evaluate the MARS-based AD, and compare it to the other benchmark detectors, against modulated attacks. Specifically, two more types of attack vectors modulated from the standard AR-DoS attacks are considered: *Step Amplitude* attack and *Ramp Amplitude* attack. The former can freely adjust the amplitude of the acoustic resonance while the latter can slowly increase the attack power over time. These more stealthy attacks compared to standard acoustic resonant attacks and EMI attacks pose a greater challenge to the ADs. By choosing different scaling factors k, which control the maximum amplitude of the attacks, Fig. 17 and 18 show that the standard estimators have significant performance drop as the attack amplitude decreases. In comparison, MARS significantly outperforms the standard ADs and provide more resiliency to the system.

7.5 MARS-PX4 Autopilot

7.5.1 System Parameters

We now summarize the MARS-PX4 system parameters used for simulation and physical world evaluations. Table 7 and 8 show the virtual sensors, estimators, and controller specifications used in the PX4 SITL environments, while Tables 9 and 10 show the sensor configurations and firmware specifications, respectively. The software and firmware architecture is illustrated in Fig. 5, except for the virtual sensing setup in SITL replaced with real-world sensors. The specifications of these autopilot modules are shown in Table 10. Specifically, the update frequency of attitude rate controller is dependent on the choice of estimators. If the IMU is not under attack, it updates at 800 Hz for best control performance; if MARS resilient control is activated, it adopts a more conservative 200 Hz update frequency in tune with the rate of rotor speed measurements to provide more stable and accurate control signals.

Table 7: PX4 SITL virtual sensors specifications.

Simulation Virtual Sensors	Update Frequency (Hz)	Data Output
Accelerometer	250	a_x, a_y, a_z
Gyroscope	250	$\Omega_x, \Omega_y, \Omega_z$
Magnetometer	250	M_x, M_y, M_z
Tachometer	250	$\omega_1, \omega_2, \omega_3, \omega_4$
GPS	20	x, y, z, v_x, v_y, v_z
Compass	20	ψ

Table 8: PX4 SITL state estimators and control specs.

Estimators & Controllers	Update Frequency (Hz)	Data Output
PX4-Standard State Estimator	250	$\hat{q}, \hat{x}, \hat{y}, \hat{z}, \hat{v}_x, \hat{v}_y, \hat{v}_z$
MARS-Resilient State Estimator	250	$\hat{q}, \hat{\Omega}, \hat{x}, \hat{y}, \hat{z}, \hat{v}_x, \hat{v}_y, \hat{v}_z$
MARS-Anomaly Detector	250	α, α_s
PX4-Position Controller	50	$q^{\text{ref}}, T^{\text{ref}}$
PX4-Attitude Controller	250	$\Omega_x^{\text{ref}}, \Omega_y^{\text{ref}}, \Omega_z^{\text{ref}}$
PX4-Attitude Rate Controller	250	$\omega_1^{\text{ref}}, \omega_2^{\text{ref}}, \omega_3^{\text{ref}}, \omega_4^{\text{ref}}$

Table 9: MARS-PX4 X500 sensing specifications.

Sensors	Update Frequency (Hz)	Data Output
IMU-Accelerometer (ICM-45686)	800	a_x, a_y, a_z
IMU-Gyroscope (ICM-45686)	800	$\Omega_x, \Omega_y, \Omega_z$
IMU-Magnetometer (BMM150)	800	M_x, M_y, M_z
Barometer (ICP20100)	400	H_{baro}
Tachometer (TFRPM01)	200	$\omega_1, \omega_2, \omega_3, \omega_4$
GPS&Compass (VICON MOCAP)	10	$x, y, z, v_x, v_y, v_z, \psi$

Table 10: MARS-PX4 software & firmware specifications.

MARS-PX4 Modules	Update Frequency (Hz)	Data Output
PX4-Standard State Estimator	200	$\hat{q},\hat{x},\hat{y},\hat{z},\hat{v}_x,\hat{v}_y,\hat{v}_z$
MARS-Resilient State Estimator	200	$\hat{q},\hat{\Omega},\hat{x},\hat{y},\hat{z},\hat{v}_x,\hat{v}_y,\hat{v}_z$
MARS-Anomaly Detector	200	α,α_s
PX4-Position Controller	100	$q^{\text{ref}},T^{\text{ref}}$
PX4-Attitude Controller	200	$\Omega_x^{\text{ref}},\Omega_y^{\text{ref}},\Omega_z^{\text{ref}}$
PX4-Attitude Rate Controller	200/800	$\omega_1^{\text{ref}},\omega_2^{\text{ref}},\omega_3^{\text{ref}},\omega_4^{\text{ref}}$

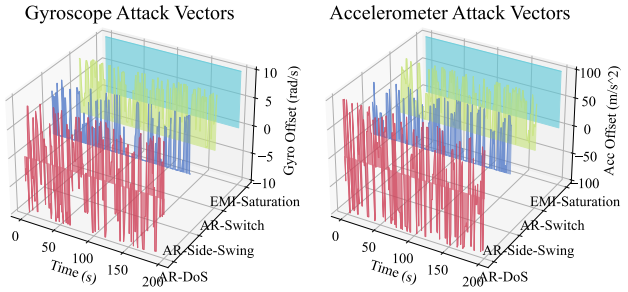

Figure 19: Illustration of the four considered inertial sensor attack profiles for gyroscope and accelerometer.

7.5.2 Inertial Attack Parameters

In the simulated environment, we use the IMU sensor profile from the ICM-456xy MEMS Motion Sensor [43], with a full detection range for the accelerometer and gyroscope of $\pm$ 4000 dps and $\pm$ 32 g, which correspond to $\pm$ 300 m/s^2 and $\pm$ 70 rad/s, respectively. We set the normal range for accelerometer and gyroscope readings at $\pm$ 100 m/s^2 and $\pm$ 10 rad/s. Thus, we set the amplitude of the acoustic resonant attack at the maximum value within the normal detection range, and the amplitude of the EMI saturation attack at the maximum value of the full range.

The resonant frequency of the drone onboard MEMS typically ranges from 20 kHz to 30 kHz, while our system sampling frequency is 250 Hz. After the signal aliasing effect, the induced frequency lies in the range of 0 to 125 Hz. To select an appropriate acoustic attack induced frequency, we chose 100 Hz and introduced a sampling drift of 500 μs to simulate attack signal dispersion in real sensors.

The attack vectors are illustrated in Fig. 19. Also, four PX4 SITL inertial attack profiles and their corresponding configurations are summarized in Table 11.

7.5.3 MARS-PX4 Computation Overhead

We measured the computation overhead of MARS modules inside the PX4 ecosystem. The memory usage, CPU load and module execution time are demonstrated in Table 12.

PX4 has a list of work queues running on CPU for efficient multi-threading. We found out activating MARS does not increase the total CPU usage across all work queues (around 47%). Within the work queue, MARS increases the CPU usage by only around 1%. The increment in memory usage is only around 12 KB. At the same time, MARS is capable of finishing online processing within 0.3 ms, which is sufficient for drone's position and attitude controllers.

Table 11: PX4 SITL inertial attack profiles with implementation details for acoustic resonant attack (AR) and electromagnetic interference attack (EMI).

Inertial Sensor Attack Profile	Amplitude ($A_{\text{acc}},A_{\text{gyro}}$)	Frequency ($F_{\text{acc}},F_{\text{gyro}}$)	Sampling Drift σ_{F_s}
AR-DoS	(100 m/s^2, 10 rad/s)	(100 Hz, 100 Hz)	500 μs
AR-Side-Swing	(100 m/s^2, 10 rad/s)	(100 Hz, 100 Hz)	500 μs
AR-Switch	(100 m/s^2, 10 rad/s)	(100 Hz, 100 Hz)	500 μs
EMI-Saturation	(300 m/s^2, 70 rad/s)	(Inf, Inf)	Not Applicable

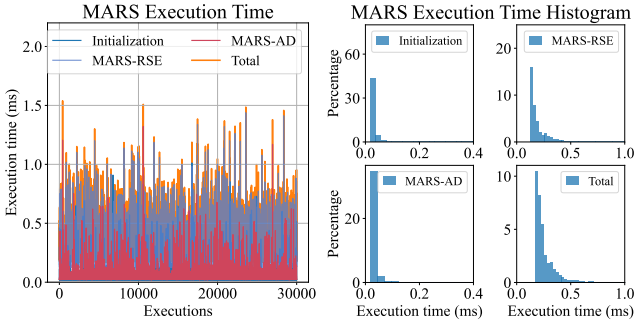

Figure 20: MARS execution times in 10000 iterations. Left: execution time plot; right: execution time distribution.

Table 12: Computation overhead of MARS-PX4.

MARS-PX4 CPU Work Queue (WQ) Usage (%)			
CPU Usage	WQ without MARS	WQ with MARS	Increment
~47	0.188	1.499	1.311
MARS-PX4 Memory Usage (KB)			
Total Memory	Usage without MARS	Usage with MARS	Increment
916.416	365.840	377.808	11.968
MARS-PX4 Average Execution Time (ms)			
Total	Initialization	MARS-RSE	MARS-AD
0.261	0.029	0.200	0.032

Fig. 20 further shows the fluctuation in MARS execution time in 10000 iterations at 200 Hz module update frequency. It focuses on three outs main MARS components: initialization with incorporating sensor data, MARS-RSE, MARS-AD, as well as the total execution time. The running time of MARS-RSE have spikes around every 10 executions because the update step of an EKF is executed when new observation data is received, which is the 10 Hz VICON-simulated GPS. As can be seen, MARS has small oscillation in the runs and the worst case scenario is below 2 ms, far less than the required sampling interval (5 ms).

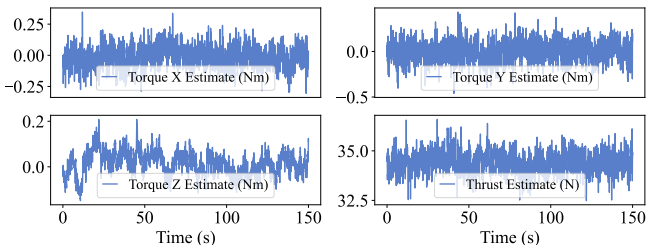

Figure 21: Online torque and thrust estimates.

7.5.4 Tachometer-based Estimation

To meet the requirement of MARS state estimator and controllers, we chose the tachometers sampling frequency at 200 Hz and recorded the measurements of all four rotors in real-time. As discussed in Sec. 3.1,we are able to estimate the UAV state by calculating the collective torque and thrust from our filtered rotor speeds and earth frame velocities. A set of proper torque compensation coefficients and bias terms for the best hovering stability is tuned experimentally as illustrated in (4). The estimated torque and thrust from the filtered rotor speeds and bias compensation are shown in Fig. 21. The high quality of the rotor speed measurements and torque/thrust estimates lays a firm foundation for the MARS-RSE as they replace the role of inertial sensors in providing attitude information to the quadrotor.

7.6 Existing IMU Attack-Recovery Methods

7.6.1 Learning-based Filtering Training Details

Deep-Auto-Encoder (DAE)-based method [14]. DAE is an emerging ML-based filtering technique that excel at capturing embedded patterns in attack profiles compared to heuristic filters. We recorded training datasets of drone hovering and waypoint-visiting missions with randomly generated reference waypoints and headings. We then built our attack recovery dataset by adding three different variations of acoustic resonant attacks, resulting in a total of 548,000 data samples in our training set. We did not include EMI attacks in the training set, as it is trivial for filtering-based methods to learn from saturated sensor measurements. Using the attack-recovery dataset, we trained two DAE models for the accelerometer and gyroscope, achieving Mean Square Error (MSE) test losses of 0.046 and 0.005, respectively.

7.6.2 Control Benchmarks Performance in Hovering

Filtering-based methods struggle with un-patterned attack signals. LPF could keep the drone stable for nearly 3 s during an AR-DoS attack but deteriorates significantly in AR-Side-Swing and AR-Switch attacks due to the faster accumulation of unfiltered in-band noise in the latter two acoustic resonant attacks. DAE does not perform better than LPF as its training involves adding noise to a clean sensor output from a normal drone flight, where the effect of the attack does not impact the drone controller. Hence, the characteristics of the attack-to-crash behaviors are not well learned. Also, DAE faces computational challenges, and cannot meet the low inference and communication time requirements. Moreover, neither LPF nor DAE can handle sensor saturation, which results in significant control fluctuations and minimal survival time. In sensor saturation, noise is overwhelmingly dominant with little useful information left to be filtered out. Similarly, filtering-based methods struggle with other un-patterned attacks, such as incorrect values caused by unexpected data loss, due to EMI attacks.

Attitude reconstruction is more robust in handling different attack profiles, but is limited by the quality of positioning sensors. Compared to filtering-based methods, CAF achieves much smaller control fluctuations and remains airborne longer, especially during EMI-Saturation attacks, where filtering-based methods suffer the most. However, the survival time of CAF decreases when the attack takes longer to be detected, even if the attack signal causes less impact. This phenomenon is attributed to the estimation logic of CAF, which relies on the quality of position sensor updates. It calculates roll and pitch angles geometrically at the frequency of position readings. However, it is currently challenging to access a position sensor with an update frequency comparable to that of drone attitude controllers. As a result, the effect of the attack takes longer to propagate to the position sensor, delaying CAF's ability to make necessary adjustments in response to the attack.