A Comprehensive Study on Drones Resilience in the Presence of Inertial Measurement Unit Faults

Anamta Khan, Naghmeh Ivaki, Henrique Madeira
CISUC, Department of Informatics Engineering, University of Coimbra, Portugal
anamta@dei.uc.pt, naghmeh@dei.uc.pt, henrique@dei.uc.pt

***Abstract*—Unmanned aerial vehicles (UAVs) have gained immense popularity for their versatility and diverse applications. However, this increased usage has raised concerns about the safety and security of UAVs, emphasizing the critical role of their Inertial Measurement Units (IMUs) in ensuring accurate orientation and position data. IMU faults, including both Accelerometer faults and Gyrometer faults, can lead to severe consequences, such as mission failures, collisions, or loss of control. This study addresses the need to enhance UAV resilience in urban airspace by exploring the impact of various IMU faults. A comprehensive fault model is introduced in this paper, covering a range of faults from hardware malfunctions to external attacks. Through extensive fault injection experiments in a simulated environment, the study assesses the effects of different fault types and durations on mission outcomes, providing valuable insights for developing resilient and fault-tolerant UAV systems. Evaluation metrics, including inner and outer bubble violations, missions completed, flight duration, and distance traveled, offer a comprehensive understanding of IMU fault impacts in dynamic operational scenarios. Results reveal that longer injection durations, particularly at 30 seconds, increase bubble violations and significantly reduce mission completion rates. Accelerometer faults, such as "Accelerometer Freeze" and "Accelerometer Random" exhibit reduced mission completion rates of 42.5% and 5%, respectively. Gyrometer faults, especially "Gyrometer Minimum" and "Gyrometer Random" lead to the lowest mission completion rates (2.5%). Additionally, IMU faults (where the fault affects both the Accelerometer and Gyrometer), notably "IMU Minimum", "IMU Freeze", and "IMU Random" result in complete mission failures, highlighting the importance of understanding specific fault characteristics. These insights can contribute to developing fault tolerance mechanisms and resilient UAV systems in complex and dynamic environments.**

***Index Terms*—UAVs, Flight Controller, IMU, Accelerometer, Gyrometer, Extended Kalman Filter (EKF), Fault Injection**

I. Introduction

Drones, especially autonomous Unmanned aerial vehicles (UAVs), have become increasingly popular in recent years for various applications such as delivery, search and rescue, surveying and inspection, precision agriculture, aerial photography, and hobbies [1]. These vehicles rely on onboard sensors, like GPS, barometer, and Inertial Measurement Unit (IMU), to provide essential feedback to the flight controller regarding Geo-location, attitude, position, air data, and other necessary flight information.

IMU is particularly crucial for providing attitude and position feedback to the flight controller. IMU measures a drone's acceleration, rotation, and orientation using a combination of accelerometers and gyroscopes [2], [3]. In some cases, a magnetometer is also included, serving as the drone's compass. However, for this study, we do not consider the magnetometer. IMU is able to measure motion by converting the detected inertia, which are forces created due to an object's resistance to changing direction, into output data that describes the motion of the object. This data is then used by some other system, for example, to control a vehicle [4]. Therefore, IMUs are crucial in maintaining stability and providing accurate directional information for unmanned aerial systems (UAS).

However, recent studies have shown vulnerabilities in commonly used IMUs, exposing drones and their systems to security threats rooted in their physical characteristics [5]. Ensuring the reliability of drones is crucial in U-space (a European initiative for Unmanned traffic management system (UTM)) [6], [7], where adherence to separation minima (also known as "Bubble", a virtual safety volume around the drone throughout the mission that the drone should follow for a safe and conflict-free flight) is the primary risk metric [8], [9]. This paper presents a simulation-based approach to introduce faults into IMUs, primarily focusing on one of the most widely used flight controllers, PX4[1], and a highly accepted drone simulator, Gazebo[2]. We systematically assess the impact of faults on the safety and reliability of the drone's overall system. By utilizing controlled fault scenarios in a simulated environment, we aim to identify potential vulnerabilities, contributing to the ongoing efforts to enhance the safety and resilience of the drone's overall systems and, consequently, the operations in U-space.

In summary, the paper makes the following contributions:

1) Presents a Fault model that specifies potential vulnerabilities present in IMUs of UAVs.
2) Proposes a 2-layered Bubble system helping to detect and avoid conflicts and collisions and mitigate IMU errors and failures.
3) Defines metrics to measure and comprehend the impact of each fault.
4) Integrates the fault injector into the PX4 flight controller.
5) Conduct simulated flights with and without injected faults, then analyze the results based on predefined metrics to assess fault severity and evaluate the flight controller's response.

[1]https://px4.io/
[2]https://gazebosim.org/home

2158-3927/24/$31.00 ©2024 IEEE
DOI 10.1109/DSN58291.2024.00040

II. Related Work

In order to investigate the impact of faults in UAVs, researchers, in [10], utilized a simulation-based environment to evaluate three subsystems (Navigation, GPS, and Transmission) by injecting two types of faults (failure and signal strength), resulting in a total of six faults. Their objective was to identify mitigation strategies and establish recovery mechanisms. Another related study [11] conducted using Simulink identified two common challenges: i) insufficient historical fault samples for comprehensive UAV fault model coverage and ii) the impracticality of identifying and studying faults through test flights. Our paper addresses both challenges by utilizing simulations for fault injection and incorporating a comprehensive IMU fault model to study the impact. Additionally, in [12], the authors explored fault impacts and validated simulation environments. They developed a custom fault injection simulator, verifying its accuracy and demonstrating that fault injection results in simulation models that closely mirror real-world events, a methodology also applied in our research.

The Extended Kalman Filter (EKF) algorithm is responsible for estimating UAV's position, velocity, and orientation [13]. The importance of EKF in the navigation of UAVs demands that it be highly reliable and able to withstand faults to ensure that the mission can continue and be completed even in the presence of faults. However, in our previous studies [14], [15], we evaluated the reliability of the PX4 flight controller under different abnormal GPS conditions, such as GPS malfunctions and interference in GPS data. We used simulation-based fault injection to emulate these conditions and monitored the UAV's behavior to evaluate the flight controller's reliability.

Similar to GPS, the IMU is a crucial component for providing precise flight data, ensuring stability, and optimizing flight performance. Given its crucial role, the IMU is essential for the safety assurance of drone operations. The significance of reliable flight data cannot be overstated, as it is essential for accurately guiding the drones' movements. [16], [4].

While the resilience of IMUs has not received significant attention from the UAV and drone research community, there are some available studies, particularly focusing on the vulnerability of IMUs to malicious attack scenarios. For example, gyroscopic sensors can be attacked with intentional sound noise, causing drones to rock [2]. Acoustic injection attacks can also be used to wage doubt on the integrity of accelerometers [3]. These studies highlight the importance of evaluating the IMU in simulation-based faulty conditions, as proposed in this paper, to assess the impact of such attacks on the safety, dependability, and reliability of drones and their systems.

Furthermore, some researchers have emphasized the significance of stress testing and performance analysis of orientation estimation algorithms [17], [18]. This further highlights the need for evaluating the IMU in faulty conditions to assess its impact on the performance of orientation estimation algorithms (i.e. Extended Kalman Filter (EKF) in our study) and the overall system reliability.

III. Approach and Experimental Setup

The fault model developed in this study includes various fault types detailed in Table I. We simulate these faults to assess their impact and tolerance by the Extended Kalman Filter (EKF) and flight controller. The experiments involve injecting faults into 10 missions in a Gazebo and PX4 simulated environment. Evaluation metrics, such as inner and outer bubble violations (see sub-section D), missions completed, flight duration, and distance traveled, provide a comprehensive analysis of IMU fault impact on drone performance and safety.

A. Fault Model

In this section, we present a **fault model** for IMUs used in drones. To create the fault model, we surveyed the known faults that may happen in an IMU.The identified faults, along with their descriptions, references, and the way they can be coded/represented through fault injection, are listed in Table I. The proposed fault model is then used to simulate different fault scenarios to evaluate the impact and the tolerance of these faults by the EKF and flight controller.

The identified faulty scenarios used in the fault injection can be listed as 1) Fixed values (a Random constant value), 2) No updates/Zeros (Zeros as output), 3) Freeze values (same previous value from the point the injection started), 4) Random value (Random in range values), 5) Min value (the minimum allowed value that is in negative), 6) Max value (The maximum allowed value), and 7) Noise (A not so drastic random value added/subtracted to the current value). Table I provides references supporting all these types of faulty scenarios.

B. Experiment Design

In this work, we used **10 missions** from a scenario designed for running the experiments from a U-space perspective [6]. This scenario is framed in an area of high-density controlled air traffic in the urban center of a Valencia, Spain. The simulated zone spans 25 km^2 with a height restriction of 60 feet. In this specific scenario, each drone is equipped with distinct specifications, including varying payloads and velocities (i.e., 2 drones of 5km/h, 1 drone of 10km/h, 3 drones of 12km/h, 3 drones of 14km/h, and 1 drone of 25km/h). These missions have different directions; some go from North to South, East to West, or vice versa. Also, 4 of the missions include turning points.

The experiment involved injecting the defined **7 faults types**, each tested over **4 durations** of 2, 5, 10, and 30 seconds injections. Fault instances were introduced strategically at the 90-second mark after take-off, which placed the injection within the mission, whether midway between waypoints, the fault being injected into turning points, or just before or after reaching the waypoint. For each fault type, 3 test cases were explored: Accelerometer, Gyrometer, and the entire IMU (i.e., Accelerometer and Gyrometer together), resulting in a comprehensive 3 experiments per 7 fault types, leading to 21 experiments per duration. In total, there were 850 cases for study, comprising 10 drone missions with 21

TABLE I: Fault Model for IMUs Used in Drones.

Fault	Description	Can be represented by	References
Instability	This fault is caused by random values and can be due to factors like radiation or temperature	Random values	[19], [20], [21], [22]
Bias error	This fault is caused by noise and can happen due to factors like old sensors or temperature	Noise	[19], [22], [23], [24]
Gyro drift	This fault is a constant error in measurement and can be caused by factors like old sensors, noise, or bias due to temperature	Noise	[19], [20], [25], [26]
Acc drift	This fault is a constant error in measurement and can be caused by factors like old sensors, noise, or bias due to temperature	Noise	[19], [20], [27], [28]
Constant output	This fault is caused by a lag in updating and getting the same frozen values constantly	Freeze values	[19]
Damaged IMU	This fault occurs when the IMU has been damaged due to old age or external factors, causing failure in all IMU sensors	No updates/zeros	[29], [30]
Gyro failure	This fault occurs when the gyro sensor has been damaged or has failed	No updates/zeros	[30], [31], [32], [33]
Acc failure	This fault occurs when the acc sensor has been damaged or has failed	No updates/zeros	[30], [31], [34]
Acoustic attack	This fault occurs when the drone is attacked by powerful broadband pulsed or Continuous Wave (CW) acoustic energy or by narrowband CW. It can cause the drone to lose control and crash	Random values	[35], [36]
False data injection	This fault occurs when fake series of data are injected	Fixed values	[37], [38], [39]
Physical isolation	This fault occurs when one or all sensors are attacked to stop responding	No updates/zeros	[40]
Hardware trojan	This fault occurs when the electronic hardware is modified (e.g., tampering with the hardware circuit, resizing the logic gate, etc.)	Fixed values	[41]
Malicious software	This fault occurs when the Ground Control Station and the Flight Controller are prone to malicious software. It can lead to the loss of sensitive data and control of the operated UAV system	Zeros/Random Values	[35]
OS system attack	This fault occurs when potential attacks against civilian or military missions happen through the Flight Controller's system software	Min/Max/Fixed values	[42]

faulty experiments across 4 durations (21x10x4 = 840), plus 10 gold experiments (the experiments without faults as reference trajectories).

C. Experimental Environment

To execute our experiments, we utilized a simulation environment, shown in Figure 1, equipped with all the necessary components for executing fault injection campaigns, logging flight data, and generating trajectories for multiple UAVs. Within this simulated platform, each component was developed, deployed, and operated in a virtualized environment using VMware ESXi.

The fault injection campaigns were managed by a dedicated tool, which was responsible for introducing predefined faults into the UAVs' flight controller (i.e., by corrupting sensor data output) or the communication network (though the latter was not utilized in this study).

Operating within this simulated environment with Gazebo and PX4, multiple UAVs were deployed, and their behaviors were simulated using dedicated machines. Additionally, a tracking system comprising a tracker, core brokers, and edge brokers was deployed to facilitate communication with U-space for future studies. The platform records all flights, capturing data from both fault-injected and fault-free scenarios.

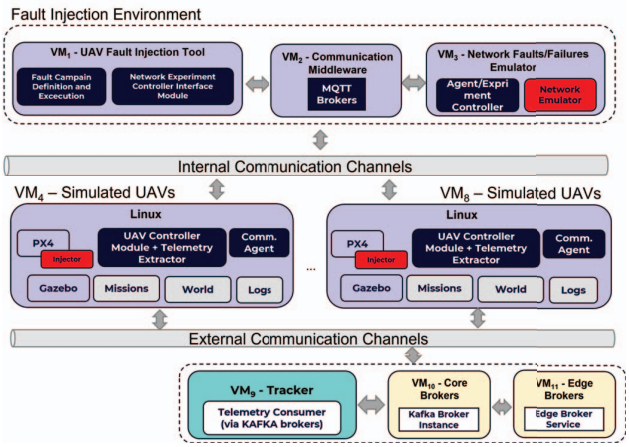

Fig. 1: Detailed View of the Fault Injection Environment.

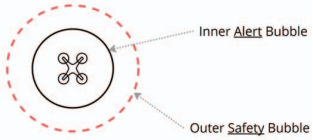

Fig. 2: Visualization of the virtual Bubble layers.

D. Evaluation Metrics

We propose a two-layered Bubble concept for U-space [6], consisting of a static inner alert bubble and a dynamic outer safety bubble, as can be seen in Figure 2. The outer safety Bubble is akin to the concept of separation minima in U-space, and our study also suggests a possible formula for the outer bubble that can serve as a separation minima. The inner bubble is designed to detect and alert issues in the drone, functioning as an alert bubble to notify the pilot and U-space of potential threats. The proposed concept aims to enhance separation management in U-space, particularly for unmanned air systems (UAS) flying in very low-level airspace (VLL). Moreover, besides the bubble violations, 3 more metrics are considered in this study and are defined as follows:

*1) **Number of Inner Bubble Violations**:* The inner bubble size is a static calculation, as shown in Equation 1. In this equation, all values are first converted into meters. D_o represents the dimensions of the drone, encompassing the wingspan. D_s reflects the recommended safety distance provided by the manufacturer, while D_m indicates the maximum distance the drone can cover at its top speed between two tracking instances. It is important to note that D_m is constant throughout the mission but varies across drones and missions. This value does not necessarily come from the gold run but is obtained from the drone's specifications. To determine the inner bubble size, we choose the greater value between D_s and D_m and add it to D_o in the formula. This approach ensures that the inner bubble calculation considers the appropriate distance, accounting for either the manufacturer's safety recommendation or the drone's maximum travel distance.

$$Bubble_{inner} = D_o + max(D_s, D_m) \quad (1)$$

*2) **Number of Outer Bubble Violations**:* Each drone is assigned a separation volume around it during a flight mission according to separation minima and its dimensions, which can be called Bubble (space around a drone at a specific position) or Volume (space around a drone along with a mission).

The definition of Separation Minima (Bubble's or Volume's size) depends on many factors [43] according to U-space, including i) The size and weight of the drone; ii) The instantaneous velocity of the drone; iii) Performance of the drone navigation system; iv) Performance of communication with UTM/U-space; v) Performance of the UTM/U-space surveillance function; vi) Risk of aggravating the hazard of any collision coming from features of the drone (e.g., high mass, flammable fuel), its cargo or the presence of passengers; vii) The risk of aggravating the hazard of any collision associated with the ground being overflown or other airspace users flying below; viii) The weather conditions. However, there is no predefined separation minima (Bubble's radius) formula provided for the drones in urban airspace. Thus, we need to define it for the drones included in this study.

As shown in Formula 2, represents the calculation of the dynamic outer bubble, $S_a(t_n)$ represents the current Airspeed, $S_a(t_{n-1})$ represents the previous Airspeed, $D(t_{n-1})$ represents the last distance covered by the drone, which represents the current speed of the drone. Utilizing these parameters enables the computation of the anticipated distance to be covered at time t_n.

$$D(t_n) = D(t_{n-1}) * \frac{S_a(t_n)}{S_a(t_{n-1)}} \quad (2)$$

The relative alteration in the anticipated distance covered (expressed in Formula 2), caused by the alteration in the Airspeed, allows us to compute the additional space allocation required for each drone beyond the inner bubble radius. It is crucial, however, to ensure that the inner bubble radius consistently remains the minimum value.

Finally, as can be seen in Formula 3, to calculate the radius of the outer bubble at time t, the result of the previous calculation is first multiplied by $Bubble_{inner}$ and then by R, which is representing the risk associated with flying in the given airspace conditions. Indeed, R is an outcome computed by multiple factors, encompassing current airspace density, weather conditions, communication quality, U-space surveillance performance, and the age of the drone. R should have a value equal to or higher than 1. In this study, to simplify the environmental conditions, we opted for a value of 1 for R.

$$Bubble_{outer}(t) = R * (Bubble_{inner} * Max(1, D(t_n)) \quad (3)$$

*3) **Missions Completed**:* This metric shows the percentage of missions completed (i.e., nor crashed neither failsafe is enabled) out of all faulty missions executed. As total of 640 faulty experiments were executed, this metric is crucial to identify the risk of the specific fault leading to mission completion.

*4) **Flight Duration**:* This metric holds significant importance, not only because precise timing is essential for managing drone flights effectively within UTM/U-space but also due to the limited battery capacity of small drones. Any impact on flight duration can directly affect the drone's ability to fulfill its designated mission. In critical situations, such as time-sensitive missions like blood delivery or firefighting, this impact can have detrimental consequences.

In our study, we calculate this metric by subtracting the take-off start time from the time the vehicle successfully lands and disarms. In instances where the drone crashes, the crash time is recorded as the endpoint.

*5) **Distance Traveled**:* Similar to flight duration, the distance a drone is expected to cover in a flight, measured through the estimated position, which is the output from the EKF, is a crucial metric. It helps gauge the effectiveness of EKF and flight controllers in filtering anomalies caused by IMU. This metric is calculated by summing the differences between the positions of drones as estimated by the EKF.

IV. Results and Analysis

The results of the fault injection experiments with 10 simulated UAVs, 4 injection durations, and 21 fault types, provide valuable insights into the relationship between fault types, injection durations, and UAV mission outcomes. The experiments showed that longer injection durations generally lead to more severe consequences, but the specific behaviors associated with each fault type play a crucial role in determining the extent of their impact. The findings serve as a foundation for developing resilient UAV systems, capable of mitigating specific faults and ensuring mission success across diverse scenarios.

A. Injection Duration Analysis

We first examined the impact of different injection durations on mission outcomes, as summarized in Table II, where each row represents the average of all missions for all faults, grouped by injection duration (i.e., 210 missions average for each injection duration and 10 missions average for the Gold runs). The table is sorted by mission completion percentage. The reference/baseline mission, represented by the "Gold Run", which covered a distance of 3.65 km in 491.26 seconds on average (of the 10 missions). However,

as the injection duration increased, a noticeable trend can be seen. Longer durations resulted in more inner and outer bubble violations, and leading to a notable decline in mission completion percentages. For instance, the 30-second injection duration exhibited the highest average inner and outer bubble violations, with a mission completion rate of 10.47%.

B. Fault-Specific Analysis

Table III provides a detailed breakdown of fault injection experiments, categorizing faults based on their types and showing the average of all missions and for all durations of injection (i.e., 40 missions average for each fault type and 10 missions average for the Gold runs). The table is sorted by mission completion percentage for each component. The various fault types show a comparable impact, with some faults causing a significant impact to the extent that no drones can successfully complete the mission, results also seem to vary based on the type of the fault. Whereas, some faults permit the completion of over 60% of missions.

1) Accelerometer Faults:

- **Acc Freeze and Acc Random:** These faults caused fewer violations, but mission completion rates of 42.5% and 5%, respectively. This could be because these faults interfere with the hardware of the accelerometer, causing the UAV to be unable to return to its normal state.
- **Acc Zeros and Acc Noise:** In contrast, these faults showed relatively high inner and outer bubble violations, and mission completion rates of 67.5% and 60%, respectively. For these faults, the increased violations correlated with a high mission success, as the drones in most missions, deviated from the trajectories but were able to stabilize.

A visual example of Accelerometer faults is shown in Figure 3, where a fault "Fixed value" (in which a random but constant value is injected) is introduced into the accelerometer of the fastest drone (drone with 25km/h) at the midpoint between two waypoints. It can be seen that the drone went off its trajectory and later crashed.

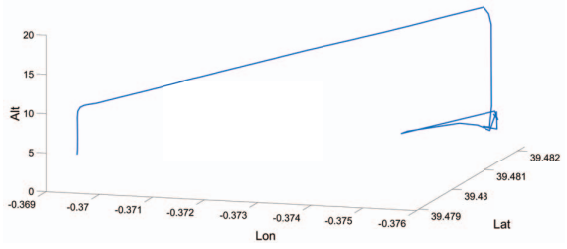

Fig. 3: Random Value injected in Acc for 30 sec - crash.

2) Gyrometer Faults:

- **Gyro Max and Gyro Random:** These faults exhibited the lowest inner and outer bubble violations, with a mission completion rate of 2.5% each. These faults also seem to impact the hardware more, causing the drone to crash or fail-safe immediately.
- **Gyro Freeze and Gyro Noise:** In contrast, these faults resulted in higher violations, with mission completion rates of 15% and 10%, respectively.

An example to show the behaviour of injecting fault in Gyrometer is presented in Figure 4, where Random Values are injected for 30 seconds just before a waypoint. While the drone was able to reach the waypoint, it could not stabilize itself for the turn and had to enable failsafe.

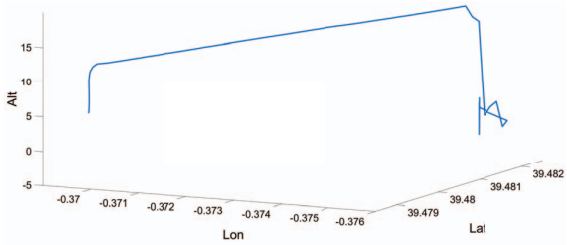

Fig. 4: Random Values injected in Gyro for 30 sec - failsafe.

3) Inertial Measurement Unit (IMU) Faults:

- **IMU Min, IMU Freeze, and IMU Random:** These faults resulted in a complete mission failure, even when faults were injected for only 2 seconds, with a 0% completion rate. The severity of these faults emphasizes their potential to compromise the overall system.
- **Other IMU Faults:** The remaining IMU faults, including IMU Max, IMU Zeros, IMU Noise, and IMU Fixed Value, exhibited varying degrees of violations and mission completion rates. Understanding the specific characteristics that each fault is crucial for developing strategies for fault mitigation.

Figure 5 shows the impact of injecting Random Values in the whole IMU (i.e. Accelerometer and Gyrometer together) for 30 seconds, and a few seconds before a waypoint. The drone crashes very quickly and seems be be very forcefully, this could be because both the accelerometer and gyrometer were not responding to stabilize the drone.

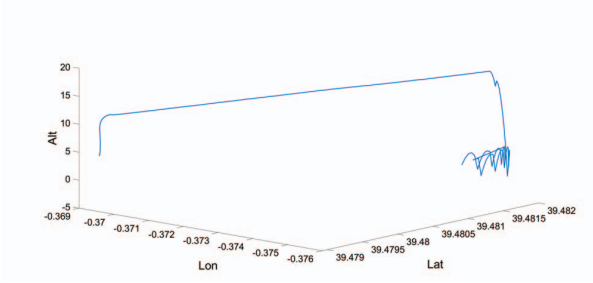

Fig. 5: Random Values injected in IMU for 30 sec - crash.

C. Mission Failure and Failsafe Rates

Digging deeper into the metric of missions completed, Table IV shows the average mission failure rates for each injection duration as well as each component (i.e., Accelerometer, Gyrometer, or both together (IMU)) in which the faults are

TABLE II: Average summary of all missions for all faults, grouped by injection duration.

Injection Duration	Inner Bubble Violations (#)	Outer Bubble Violations (#)	↓ Missions Completed (%)	Duration (sec)	Distance (km)
Gold Run	0	0	100%	491.26	3.65
2 seconds	18.30	17.81	20%	188.87	0.98
5 seconds	20.16	16.79	15.23%	146.07	0.81
10 seconds	20.97	19.16	11.42%	151.90	0.69
30 seconds	24.47	21.65	10.47%	154.70	0.75

TABLE III: Average summary of all missions and for all durations of injection, grouped by fault.

Injection Type	Inner Bubble Violations (#)	Outer Bubble Violations (#)	↓ Missions Completed (%)	Duration (sec)	Distance (km)
Gold Run	0	0	100%	491.26	3.65
Acc Zeros	23.36	17.5	67.5%	338.67	2.45
Acc Noise	25.23	13.48	60%	306.11	2.22
Acc Freeze	23.40	15.82	42.5%	244.09	1.80
Acc Random	20.13	16.34	5%	110.76	0.55
Acc Min	20.57	24.25	5%	137.18	0.51
Acc Max	41.32	35.32	2.5%	103.35	0.73
Acc Fixed Value	40.30	36.51	2.5%	103.99	0.75
Gyro Zeros	18.88	18.15	40%	223.21	1.20
Gyro Fixed Value	17.51	15.90	17.5%	159.57	0.49
Gyro Freeze	19.11	21.5	15%	145.92	0.98
Gyro Noise	16.01	20.67	10%	156.43	0.52
Gyro Random	16.75	16.36	2.5%	169.28	0.47
Gyro Max	16.32	14.13	2.5%	135.50	0.44
Gyro Min	19.73	14.86	0%	104.41	0.47
IMU Max	14.19	17.34	17.5%	212.30	0.46
IMU Zeros	18.17	16.55	2.5%	104.43	0.52
IMU Noise	21.19	17.61	2.5%	143.73	0.48
IMU Random	16	15.03	2.5%	104.66	0.53
IMU Fixed Value	15.67	14.28	2.5%	110.45	0.53
IMU Min	18.63	17.61	0%	155.08	0.46
IMU Freeze	18.03	16.71	0%	98.93	0.46

TABLE IV: Mission failure analysis.

Injection Type	Total Missions Failed (%)	Crash (%)	Failsafe (%)
Gold Run	0%	0%	0%
2 seconds	80%	73%	27%
5 seconds	84.77%	73%	27%
10 seconds	88.58%	70%	30%
30 seconds	89.53%	34%	66%
Acc	73.22%	77.2%	22.8%
Gyro	87.5%	63.1%	36.9%
IMU	96.08%	47.2%	52.8%

injected. It also provides percentages for crashes and failsafe activation for the missions that failed.

As illustrated in Table IV, when faults are injected for 2 seconds, the mission failure rate is 80%, with a significant 73% resulting in crashes. However, as the **duration of faults extends to 30 seconds, the mission failure rate increases to approximately 90%**.

Moreover, the Accelerometer, Gyrometer, and IMU exhibit varying outcomes in terms of failure, crash, and failsafe rate. Injecting faults into Accelerometer results into a mission failure rate of 73.22%, while injecting faults into gyrometer results into an increased failure rate of 87.5%. And when the faults are injected in both components (faults into the entire IMU), it poses the most formidable threat to mission success, with a failure rate exceeding 96% and a crash rate of 47.2%.

In addition to the mission failure and crashes, it is crucial to examine the role of failsafe activations by the flight controller. The correlation between fault duration and failsafe activation rates, shown in Table IV, indicates an increase in failsafe activation as the injection duration increases. For instance, when faults are injected for shorter durations, such as 2 seconds, the failsafe activation rate is only 27%. However, as fault duration extends to 30 seconds, the failsafe activation rate increases to 66%, indicating that the **failsafe mechanism takes longer to activate**.

Furthermore, the **failsafe activation appears to work differently when faults are injected into different components**. For Accelerometer, failsafe activation is 22.8% and for Gyrometer is 36.9%. In case of Accelerometer, the failsafe detection thresholds are not defined in flight controller, relying instead on the factors such as vehicle specifications and airspeed. In contrast, for Gyrometer, the default failsafe detection threshold is set at 60deg/s (configurable in the flight controller settings). Regarding the IMU, failsafe is activated in 52.8% of instances, as it can be triggered by either of the thresholds [44].

By default (although these settings are configurable, we have maintained default settings for simplicity in this study), the failsafe module initially attempts isolation by deactivating the primary sensor and activating redundant sensors (typically two more redundant sensors, whereas in this study, the fault is assumed to affect all redundant sensors) before enabling failsafe. In our study, **failsafe takes a minimum of 1900ms**; however, the exact time is not defined and may depend on multiple factors such as sensor calibration and sensor health.

This different impact of fault injection in these components (i.e. Accelerometer, Gyrometer and IMU) can be because of their roles in modulating drone navigation and control systems. Acceleration faults, disrupt velocity estimates and trajectory calculations, making drones to deviate from trajectory. While these faults can lead to compromised navigation, their effects may be less pronounced compared to faults in orientation measurements provided by Gyrometer. As a result, failsafe activation triggered by acceleration faults occur at a relatively

lower rate, also reflected by the lower mission failure.

In contrast, Gyrometer faults impacts orientation measurements, thereby hindering stability and control responsiveness. This can lead to more immediate and severe deviations from the intended trajectory and also impacts the sensor calibration (which for Gyrometer is very crucial) as also reflected by a high overall failure rate. This could also be resulting in higher failsafe activation in comparison to Accelerometer.

IMU faults, representing faults in both Acceleration and Gyrometer, shows the most profound implications for mission success, because of it's high influence on navigational and control systems. The combined impact of faulty Accelerometer and Gyrometer amplifies the risk of mission failure, as it compromises both navigation and stability control. Consequently, IMU faults has the highest rate of failsafe activation, because of the high risk involved as it also shows to have the highest failure rate associated.

D. Discussion

These are some crucial observations from the results, emphasizing the need to enhance IMU and flight controller resilience in urban UAV operations:

- **IMU Resilience in Urban UAV Operations:** The varying impacts of different IMU faults on mission outcomes highlight the need for a thorough understanding of each fault characteristics and impact to develop effective strategies for fault mitigation, especially software-based mitigation techniques in addition to hardware redundancies.
- **Critical Role of IMUs in Stability:** Fault scenarios like "IMU Min", "IMU Freeze", and "IMU Random" led to complete mission failures, emphasizing the critical role IMUs play in the overall system stability. These faults simulated scenarios where IMUs would experience real abnormal conditions, such as system attacks, constant output, instability acoustic attacks, and malicious software, resulting in the UAVs losing control or crashing.
- **Impact of Fault Duration:** Prolonged fault injection, especially at durations of 10 and 30 seconds, intensifies the severity of IMU faults, leading to increased inner and outer bubble violations crucial for maintaining safe flights and significantly reducing mission completion rates.
- **Need for Detection and Resilience Mechanisms:** This study emphasizes the importance of developing inherently resilient flight controllers capable of withstanding abnormal conditions in IMUs or other critical components like GPS. Effective fault detection and correction mechanisms, particularly in Extended Kalman Filters (EKFs) and flight controllers, are essential to mitigate the impact of prolonged faults, as the overall failsafe activation rate was 62.5% for the failed missions, which still make the IMU very prone to crashes.
- **Criticality of Gyrometer in UAV Safety:** Gyrometer faults, with their high failure rate of 87.5%, shows that the Gyrometer plays a more critical role in UAV Safety in comparison with Accelerometer. Faults in Gyrometer directly impacts the UAV's capability to sustain stable flight, resulting in more immediate and severe deviations from the intended trajectory or in many cases a crash.
- **Criticality of Accelerometer in UAV safety:** Accelerometer faults on the contrary have both higher Inner and Outer Bubble violations. This could be because Accelerometer can push the drones outside the assigned bubbles relatively faster than Gyrometer faults can. This shows that from a U-space perspective, Accelerometer might be more critical for safety in presence of faults, as it may cause collisions with other UAVs.

In addition to above, an interesting observation is related to the fault injection duration within the range of 0-2 seconds, which should be further explored, as **80% of the missions failed when the faults were injected only for 2 seconds**. This also leads to the importance of quick detection and tolerance techniques. Also surprisingly, **Zeros were better handled by the flight controller in comparison with the Min and Max values**, although all three are constant fixed value injections, when Min value was injected in Gyro for even 2 seconds, the drones were never able to complete the mission.

V. Threats to Validity and Future Works

While our experiments offer valuable insights into IMU fault impacts on UAVs, several limitations must be acknowledged. The accuracy of fault simulations relies on the realism of the fault model and on the coverage of such a set of fault types, potentially missing unexplored fault scenarios. Moreover, the study's focus on fault injection and simulation neglects potential hardware variations in commercial UAV systems (for example, the age and weight of different components). Therefore, while our findings are insightful, it is crucial to be cautious about applying them directly to real-world situations. Further research can be performed in real-world experiments to validate our findings.

Moreover, this study opens the door for future research to conduct in-depth mathematical evaluations of the flight controllers and EKF, aiming to better understand the effects caused by these faults and vulnerabilities and define more effective mitigation strategies.

VI. Conclusion

Unmanned aerial vehicles (UAVs) have become increasingly popular in recent years due to their versatility and wide range of applications. However, as the use of UAVs becomes more widespread, concerns about their safety and security have also increased. This research has investigated the resilience of drones against Inertial Measurement Unit (IMU) faults in simulation. Our fault model covered a large spectrum of faults, from hardware failures to software vulnerabilities to external attacks, and extensive fault injection experiments were conducted on 10 drone missions.

This comprehensive analysis of fault injection experiments highlights the intricate relationship between fault types, injection durations, and UAV mission outcomes. Longer durations generally lead to more severe consequences, but the specific behaviors associated with each fault type play a pivotal role in determining the extent of the impact. These findings provide valuable insights for developing resilient UAV systems capable of mitigating specific faults and ensuring mission success in diverse scenarios.

REFERENCES

[1] I. H. Beloev, "A review on current and emerging application possibilities for unmanned aerial vehicles," *Acta Technologica Agriculturae*, vol. 19, no. 3, pp. 70–76, 2016. [Online]. Available: https://doi.org/10.1515/ata-2016-0015

[2] Y. Son, H. Shin, D. Kim, Y. Park, J. Noh, K. Choi, J. Choi, and Y. Kim, "Rocking drones with intentional sound noise on gyroscopic sensors," in *Proceedings of the 24th USENIX Conference on Security Symposium*, ser. SEC'15. USA: USENIX Association, 2015, p. 881–896.

[3] T. Trippel, O. Weisse, W. Xu, P. Honeyman, and K. Fu, "Walnut: Waging doubt on the integrity of mems accelerometers with acoustic injection attacks," in *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*, 2017, pp. 3–18.

[4] S. Harris. Inertial measurement unit (imu) – an introduction. [Online]. Available: https://www.advancednavigation.com/tech-articles/inertial-measurement-unit-imu-an-introduction/

[5] F. Fei, Z. Tu, R. Yu, T. Kim, X. Zhang, D. Xu, and X. Deng, "Cross-layer retrofitting of uavs against cyber-physical attacks," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 550–557.

[6] E. D. team. What is u-space. [Online]. Available: https://www.easa.europa.eu/en/what-u-space

[7] T. Tan and Z. He, *A Survey on the Regulation Development of Civil Unmanned Aircraft System*. Association for Computing Machinery, 2020.

[8] T. Dubot and A. Joulia, *Towards U-space conflict management services based on 4D protection bubbles*. [Online]. Available: https://arc.aiaa.org/doi/abs/10.2514/6.2021-2347

[9] C.-H. J. Wang, C. Deng, and K. H. Low, "Parametric study of structured utm separation recommendations with physics-based monte carlo distribution for collision risk model," *Drones*, vol. 7, no. 6, 2023. [Online]. Available: https://www.mdpi.com/2504-446X/7/6/345

[10] V. Kumar Chandhrasekaran and E. Choi, "Fault tolerance system for uav using hardware in the loop simulation," in *4th International Conference on New Trends in Information Science and Service Science*, 2010.

[11] S. Gong, S. Meng, B. Wang, and D. Liu, "Hardware-in-the-loop simulation of uav for fault injection," in *2019 Prognostics and System Health Management Conference (PHM-Qingdao)*, 2019.

[12] J. Wen, Ji, H. Wang, M. Zhang, D. Li, and J. Wu, "Design of a real-time uav fault injection simulation system," in *IEEE Int. Conf. Unmanned Systems (ICUS)*, 2019.

[13] G. Mao, S. Drake, and B. D. O. Anderson, "Design of an extended kalman filter for uav localization," in *2007 Information, Decision and Control*, 2007, pp. 224–229.

[14] A. Khan, C. A. C. Jiménez, M.-P. Pablo, N. Ivaki, J. V. B. Tejedor, and H. Madeira, "Assessment of the impact of u-space faulty conditions on drones conflict rate," in *Computer Safety, Reliability, and Security*, M. Trapp, F. Saglietti, M. Spisländer, and F. Bitsch, Eds. Cham: Springer International Publishing, 2022, pp. 237–251.

[15] A. Khan, N. Ivaki, and H. Madeira, "Are uavs' flight controller software reliable?" in *2022 IEEE 27th Pacific Rim International Symposium on Dependable Computing (PRDC)*, 2022, pp. 194–204.

[16] P. Institute. Drone imu calibration explained. [Online]. Available: https://pilotinstitute.com/drone-imu-calibration-explained/

[17] G. Betta, D. Capriglione, M. Carratù, M. Catelani, L. Ciani, G. Patrizi, A. Pietrosanto, and P. Sommella, "Stress testing for performance analysis of orientation estimation algorithms," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–12, 2022.

[18] M. Jeerage, "Reliability analysis of fault-tolerant imu architectures with redundant inertial sensors," *IEEE Aerospace and Electronic Systems Magazine*, vol. 5, no. 7, pp. 23–28, 1990.

[19] R. C. Avram, X. Zhang, J. Campbell, and J. Muse, "Imu sensor fault diagnosis and estimation for quadrotor uavs," *IFAC-PapersOnLine*, vol. 48, no. 21, pp. 380–385, 2015, 9th IFAC Symposium on Fault Detection, Supervision andSafety for Technical Processes SAFEPROCESS 2015.

[20] S. Wang, X. Zhan, Y. Zhai, and B. Liu, "Fault detection and exclusion for tightly coupled GNSS/INS system considering fault in state prediction," *Sensors (Basel)*, vol. 20, no. 3, Jan. 2020.

[21] C. Berbra, S. Gentil, and S. Lesecq, "Identification of multiple faults in an inertial measurement unit," 11 2009.

[22] vectornav. Specifications and error budgets. [Online]. Available: https://www.vectornav.com/resources/inertial-navigation-primer/specifications--and--error-budgets/specs-imuspecs

[23] R. C. Avram, X. Zhang, J. Campbell, and J. Muse, "Imu sensor fault diagnosis and estimation for quadrotor uavs," *IFAC-PapersOnLine*, vol. 48, no. 21, pp. 380–385, 2015, 9th IFAC Symposium on Fault Detection, Supervision andSafety for Technical Processes SAFEPROCESS 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896315016857

[24] R. Avram, X. Zhang, J. Campbell, and J. Muse, "Imu sensor fault diagnosis and estimation for quadrotor uavs," *IFAC-PapersOnLine*, vol. 48, pp. 380–385, 12 2015.

[25] Sanjay and A. S. ev3lessons. Introduction to gyro sensor and drift. [Online]. Available: https://ev3lessons.com/en/ProgrammingLessons/advanced/GyroDrift.pdf

[26] I. Beavers. The case of the misguided gyro. [Online]. Available: https://www.analog.com/en/analog-dialogue/raqs/raq-issue-139.html

[27] J. Wen, H. Yao, Z. Ji, B. Wu, and M. Xia, "On fault diagnosis for high-g accelerometers via data-driven models," *IEEE Sensors Journal*, vol. 21, no. 2, pp. 1359–1368, 2021.

[28] modalshop. What's wrong with my accelerometer? [Online]. Available: https://www.modalshop.com/calibration/learn/accelerometers/what-bad-what-wrong

[29] A. K. Sikder, G. Petracca, H. Aksu, T. Jaeger, and A. S. Uluagac, "A survey on sensor-based threats and attacks to smart devices and applications," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 2, pp. 1125–1159, 2021.

[30] R. G. Franceschini, J. Liu, and S. Amin, "Damage estimation and localization from sparse aerial imagery," in *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2021, pp. 128–134.

[31] S. W. L. M. Alwateer and A. M. Zuchowicz, "Drone services: issues in drones for location-based services from human-drone interaction to information processing," *Journal of Location Based Services*, vol. 13, no. 2, pp. 94–127, 2019.

[32] Y. Liu, H. Ge, Z. Fan, X. Chen, F. Huang, Y. Wang, Z. Hou, and G. Bai, "Failure analysis and experimental validation of mems gyro under random vibration condition," in *2019 Prognostics and System Health Management Conference (PHM-Qingdao)*, 2019, pp. 1–6.

[33] J. Hong, Y. Youtang, and Y. Weiqin, "Exploration on failure diagnosis of the electrostatic suspended gyro," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 33, no. 1, pp. 319–322, 1997.

[34] R. Cao, Y. Chen, and R. Kang, "Failure mechanism analysis of quartz accelerometer under vibration condition," in *2011 Prognostics and System Health Managment Confernece*, 2011, pp. 1–5.

[35] Y. Mekdad, A. Aris, L. Babun, A. E. Fergougui, M. Conti, R. Lazzeretti, and A. S. Uluagac, "A survey on security and privacy issues of uavs," *Computer Networks*, vol. 224, p. 109626, 2023.

[36] A. I. Kalinkin and I. S. Kholopov, "The investigation of the p2p yaw error in the presence of acoustic clutters," in *2020 9th Mediterranean Conference on Embedded Computing (MECO)*, 2020, pp. 1–4.

[37] A. Sabra, H. Wridan, N. M. Alkhatani, and F. Al-Harby, "Description of security impact of drones challenges and opportunities," in *2018 21st Saudi Computer Society National Computer Conference (NCC)*, 2018, pp. 1–5.

[38] C. Kumar and S. Mohanty, "Current trends in cyber security for drones," in *2021 International Carnahan Conference on Security Technology (ICCST)*, 2021, pp. 1–5.

[39] P. Dash, M. Karimibiuki, and K. Pattabiraman, "Out of control: stealthy attacks against robotic vehicles protected by control-based techniques," in *Proceedings of the 35th Annual Computer Security Applications Conference*, ser. ACSAC '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 660–672.

[40] E. D'Amato, M. Mattei, A. Mele, I. Notaro, and V. Scordamaglia, "Fault tolerant low cost imus for uavs," in *2017 IEEE International Workshop on Measurement and Networking (M&N)*, 2017, pp. 1–6.

[41] E. Chen, J. Kan, B.-Y. Yang, J. Zhu, and V. Chen, "Intelligent electromagnetic sensors for non-invasive trojan detection," *Sensors*, vol. 21, no. 24, 2021.

[42] S. Iqbal, "A study on uav operating system security and future research challenges," in *2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC)*, 2021, pp. 0759–0765.

[43] C. Barrado, M. Boyero, L. Brucculeri, G. Ferrara, A. Hately, P. Hullah, D. Martin-Marrero, E. Pastor, A. P. Rushton, and A. Volkert, "U-space concept of operations: A key enabler for opening airspace to emerging low-altitude operations," *Aerospace*, vol. 7, no. 3, 2020.

[44] (2024, 02) Px4 parameter reference. [Online]. Available: https://docs.px4.io/main/en/advanced_config/parameter_reference.html