Real-Time Sensor Fault Detection in Drones: A Correlation-Based Algorithmic Approach

Inbal Roshanski
Department of Software and Information Systems Engineering, Ben Gurion University of the Negev, Beersheba, Israel

Magenya Roshanski
CTO, Maris, Israel

Meir Kalech
Department of Software and Information Systems Engineering, Ben Gurion University of the Negev, Beersheba, Israel

Abstract

Drones, or unmanned aerial vehicles (UAVs), are becoming increasingly vital across various industries, where their reliable operation is crucial for safety and efficiency. Ensuring this reliability requires the early detection of sensor-related faults, which are critical for maintaining the performance and safety of UAVs. This study addresses this challenge by leveraging real-world data from an Aero-Sentinel Military UAV Sentinel G2 quadcopter. The data was collected through a collaboration with Maris-Tech Ltd, using their advanced Mercury Nano system to capture detailed communication between the drone and its control unit. A set of correlation-based algorithms was developed and evaluated, specifically tailored to address the unique complexities of drone sensor data, which is often influenced by environmental factors. Among the algorithms tested, two novel methods emerged as particularly effective, demonstrating significant improvement compared to previous methods, in fault detection accuracy. These methods, designed to accurately identify and predict sensor malfunctions, offer a robust solution for enhancing the reliability and safety of UAV operations.

2012 ACM Subject Classification Hardware → Failure prediction; Hardware → Sensors and actuators; Computing methodologies → Modeling methodologies

Keywords and phrases Drones, Sensor Fault Detection, Correlation-Based Algorithms, Sensor Data Analysis, Anomaly Detection, Data-Driven Fault Detection

Digital Object Identifier 10.4230/OASIcs.DX.2024.17

1 Introduction

Drones, also known as unmanned aerial vehicles (UAVs), have seen a significant rise in their utilization across various industries such as agriculture, construction, and transportation. They are employed for an array of tasks, including aerial surveillance, data collection, and cargo delivery, functioning effectively in diverse environments ranging from remote areas to densely populated urban centers. Despite their growing adoption and versatility, drones are susceptible to faults, which can result in accidents and potential damage to the equipment. This makes fault detection and prediction a critical area of research in the realm of UAV technology.

Drones rely on data collected from a multitude of sensors, such as cameras, GPS, accelerometers, and barometers. This sensor data is necessary for controlling the drone's movements and executing various tasks. Typically, the data collected by these sensors is in the form of time-series, meaning it is recorded over time at varying frequencies. Analyzing time-series data presents significant challenges due to its inherent noisiness, irregularity, and the potential presence of gaps, often caused by network failures or other technical issues.

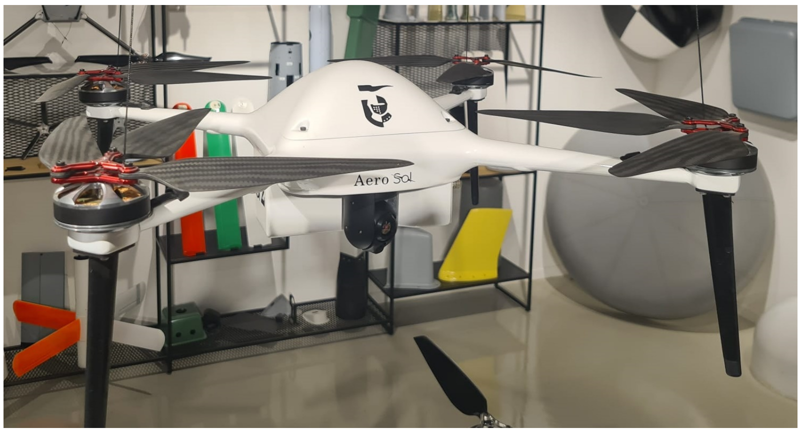


Figure 1 Aero-Sentinel Military UAV Sentinel G2 quadcopter drone.

Drones are vulnerable to several types of faults, including sensor malfunctions, structural failures, and communication errors within the control unit. These faults can be triggered by numerous factors, such as adverse weather conditions, mechanical wear and tear, or human error. Identifying and predicting these faults before they manifest is crucial to preventing accidents and ensuring the safety of both the drone and the surrounding environment.

Although many studies have been conducted on drone fault prediction, few have addressed the challenge of detecting different types of faults while accounting for data from a wide array of sensors sampled at varying frequencies. For instance, works such as [5] and [14] focus primarily on individual sensor faults or specific fault types, without considering the complexities introduced by multiple sensors. Additionally, most existing research, such as [10] and [5], primarily involves experiments conducted using simulated data or controlled laboratory environments. This leaves a gap in evaluating fault detection methods under real-world conditions, where diverse sensor inputs and environmental factors come into play. One of the significant advancements of our research is addressing situations with real-world conditions. We conduct our experiments on data collected from actual drones operating in real-world environments, providing a more practical and robust evaluation of fault detection methodologies, particularly focusing on sensor-related faults.

A crucial aspect of our research is the collaboration with Maris [1], a company specializing in Intelligent Video Surveillance and Analytics. Maris, a pioneer in high-performance AI edge video and analytics technologies, played a key role in enabling the collection of real-world data. Utilizing their Mercury Nano system [2] – an advanced, dual-channel low-power encoder designed to integrate with a wide range of platforms – we recorded communication between an Aero-Sentinel Military UAV Sentinel G2 quadcopter drone [3] and its control unit, as shown in Figure 1. This process captured detailed sensor data influenced by various environmental factors, resulting in a dataset that encompasses approximately 17 hours of flight data collected under diverse conditions.

Our first contribution lies in the creation and presentation of this unique dataset derived from real-world drone flights. Unlike simulated or laboratory-generated data commonly used in existing studies, our dataset includes genuine environmental noise and other real-world

[1] `https://www.maris-tech.com/`
[2] `https://www.maris-tech.com/solutions/mercury/`
[3] `https://www.aero-sentinel.com/military-drones/military_drone_sentinel_g2/`

variables, offering a comprehensive and authentic foundation for evaluating and validating fault detection models. This dataset addresses a significant gap in the literature by reflecting the complexities inherent in drone operations.

Our second contribution is the development of a reliable fault prediction model capable of real-time detection of sensor failures in operational drones. To achieve this, we present a sensor-based approach for Fault Detection specifically tailored for drones. This approach builds upon the methodology initially introduced in [11], which combines data-driven and model-based techniques to enhance fault detection capabilities. We have adapted and refined this methodology to address the unique challenges associated with drone operations by implementing both the basic Sensor Fault Detection and Diagnosis (Simple-SFDD) and an extended version that incorporates additional features for improved performance (Extended-SFDD). In addition to these, we have developed two more versions of our own. The Combined-SFDD version combines elements of both the basic and Extended-SFDD methods, aiming to leverage the strengths of each. The Knowledge-Based-SFDD version introduces a novel variation of this algorithm, which integrates modeling to learn and predict the normal behavioral patterns of drones through data-driven analysis. These enhancements collectively contribute to more accurate and efficient detection of sensor-related faults in drone systems.

Our experiments on this dataset demonstrate the robustness and effectiveness of our approach. The results indicate that the fourth variation of our algorithm, which is based on sensor behavior learning, achieves notable improvements. Specifically, this variation shows superior performance in both True Positive Rate (TPR) and False Positive Rate (FPR), effectively bridging the gap in fault detection accuracy.

2 Related Work

Unmanned Aerial Vehicles (UAVs) are aircraft that operate without a human pilot, crew, or passengers on board. UAVs can be controlled remotely by a human operator, referred to as Remotely Piloted Aircraft (RPA), or they can operate with varying levels of autonomy, ranging from autopilot assistance to fully autonomous flight with no human intervention. UAVs are categorized based on various criteria, including the number of propellers, their configurations, and construction types, which include rotary wing, tilt-rotor, fixed wing, and flapping wing designs. Our study focuses specifically on quadrotor drones, which are UAVs equipped with four rotors and are commonly used due to their stability and maneuverability [19]. Drones, a subset of UAVs, are versatile, robot-like aircraft that can range in size from as large as a full-scale aircraft to as small as the palm of your hand.

Drones are now employed in a wide range of roles and responsibilities [3]. Over time, drones have evolved to serve various purposes, including monitoring climate change, conducting search and rescue operations during natural disasters, photography, cinematography, and delivering commodities. Drones are also heavily involved in research that addresses diverse tasks such as path planning, communication networks, autonomous control, and more [18]. These tasks are critical for enhancing drone capabilities, making them more intelligent, but they also increase the complexity of both their software and hardware – thereby making them more susceptible to faults. Given that a drone malfunction today can have serious consequences, the importance of fault detection has become paramount.

The topic of fault prediction and detection is vast and crucial for maintaining operational systems. It aids professionals in managing components or systems and even in prioritizing testing procedures. In the field of UAVs, several studies have endeavored to address this

challenge. UAV failures are typically classified into two main categories: actuator faults and sensor faults. Although there have been studies focused on damage to circuit boards [21], the drone's frame [2], or the surface of the flying vehicle itself [1], these areas are not the primary focus of most research. In our study, we will concentrate specifically on the challenge of identifying faults using the drones' sensors.
The most current survey on the topic was conducted by Puchalski et al., who categorize the available methodologies in the field into two groups [16], model-based and data-driven.

2.1 Model-Based

Model-based methods involve mathematical models that describe and predict the behavior of a system using a set of equations and algorithms. These models are developed based on assumptions and hypotheses derived from expert knowledge, and they are commonly employed to simulate and assess the system's performance under various scenarios [20]. Currently, model-based methods are the most widely used approach for detecting faults in unmanned aerial vehicles.
Herdjunanto et al. [9] were among the first to apply a model-based fault detection technique to an unmanned quadrotor. Their goal was to address actuator fault signal isolation during hovering motion. Their work focuses on isolating actuator fault signals using a detection filter, capable of managing various fault types without the need for adjusting filter settings. By modeling the quadrotor and simulating specific fault signals, they demonstrate effective fault detection through a virtual actuator approach.
The Kalman filter is a mathematical algorithm for estimating system states from noisy observations. Classified as a model-based method, the Kalman filter employs a mathematical model to describe and predict system behavior, making it a valuable tool in UAV fault detection research. [13]. Several studies have utilized the Kalman filter for this purpose. For instance, [22] applied an adaptive two-stage extended Kalman filter to detect sensor faults in UAVs, successfully modeling quadrotor kinematics and testing various fault scenarios using the Quanser Qball-X4 model. Their results confirmed the filter's effectiveness. The extended Kalman filter has also been utilized by [23] and [6] for similar applications.

In their study, Zhong et al. [23] proposed an actuator fault detection and diagnosis method specifically for quadrotors. This method is based on a linearized dynamic UAV model and a decomposed adaptive augmented state Kalman filter. Through simulation experiments, they demonstrated the effectiveness of their approach in accurately detecting actuator faults. Similarly, Demircan et al. [6] explored the capability of a nonlinear extended Kalman filter to detect aileron locking in fixed-wing aircraft. To estimate the state of the roll rate, which is directly influenced by the aileron, they employed an extended Kalman filter and conducted simulations using MATLAB. Their results indicated that the faults were successfully detected. Further advancing the use of Kalman filtering in UAV fault detection, Hamadi et al. [8] applied both the basic and extended versions of the Kalman filter to address faults in quadrotors. Their approach targeted faults caused by hardware sensor issues (e.g., GPS, IMU, and magnetometer) as well as software errors, including faults within the Kalman filters themselves or incorrect parameter settings.

Fu et al. [7] introduced a singular Markov switching system for detecting sensor and actuator faults in quadrotors. They developed an adaptive observer and validated their approach through simulations.
Maqsood et al. [14] proposed a system for detecting faults in angular rate sensors. Their approach predicts faults using a chain differentiator integrator and enhances detection accuracy with a modified high-gain observer, incorporating a sliding mode effect to improve

detection and reduce overshoot and chattering. They simulated both gradual and abrupt faults, comparing their method to traditional detection techniques. The results showed a clear advantage over other nonlinear strategies.

The Model-Based approach, while effective, faces several challenges that limit its reliability and practical application. One major issue is its reliance on expert knowledge for model building and validation, which can be difficult or impossible to obtain in some cases, reducing the model's usefulness. Additionally, the complexity of the mathematical models involved makes them challenging to understand, interpret, and implement. These models often require specialized software and expertise, making the Model-Based approach both time-consuming and difficult to apply in practice.

2.2 Data-Driven

These models are constructed using data rather than relying on expert knowledge. Typically, machine learning or statistical methods are employed to automatically uncover patterns and correlations within the data. Data-driven models offer several advantages over model-based approaches. They can capture complex and nonlinear patterns that may be challenging or impossible for model-based methods to represent. Additionally, data-driven models can evolve and improve over time as new data becomes available, thereby enhancing the model's accuracy and reliability [15].

Chen et al. [5] proposed a data-driven model using a backpropagation neural network (BPNN) optimized by a genetic algorithm. They trained the model with pitch rate data from the speed gyroscope during UAV flight. The sensor signal was first divided into eight frequency bands using a three-layer wavelet packet, and the energy feature vector of each band served as input to the neural network. Their MATLAB simulations, which tested various sensor faults, demonstrated that this method outperformed the standard BPNN in terms of accuracy and error. Continuing their research, Chen et al. [4] introduced an enhanced sensor fault detection method using the same wavelet packet and BPNN framework, but with adaptive fireworks to improve local search capability and algorithm convergence. This approach also utilized a distributed mechanism for parallel information sharing and multi-scale analysis via wavelet transform. Compared to the initial method, the adaptive fireworks algorithm showed superior classification efficiency, faster convergence, shorter runtime, and improved global search ability.

Iannace et al. [10] employed sound analysis to detect faults in quadrotor propeller blades. A microphone placed 1.2 meters from the drone captured the rotor noise, with strips of paper tape attached to the blade surface to simulate faults. Frequency analysis was then used to extract 31 features from the recorded sounds, which were used to train a feed-forward multilayer neural network. The authors reported high accuracy in fault detection, though the method's main limitation is that it can only be tested in indoor conditions.

Sadhu et al. [17] developed a method for real-time fault detection and identification using deep Convolutional and Long Short-Term Memory Neural Networks. They collected data from accelerometer, gyroscope, and magnetometer sensors, dividing it into timestep windows of 100, 50, and 25. Initial experiments on a small drone (CrazyFlie 2.0) produced poor results, leading them to conduct further tests with a larger drone using Microsoft's AirSim simulator. These later experiments demonstrated good accuracy, as reported by the authors. However, a significant drawback is that the method was only tested in a simulated environment, raising concerns about its effectiveness in real-world scenarios.

Data-driven models, while powerful, have certain limitations. They often require large datasets for training and validation, and they can be sensitive to noise and outliers. While these models are valuable for modeling and evaluating complex systems, they must be used with caution, considering their limitations.

3 Background and Problem Definition

Drones are typically equipped with a variety of sensors capable of measuring different physical quantities and converting them into signals that can be interpreted by the onboard computer systems. These sensors are essential for monitoring the drone's status and its interaction with the environment.

To formalize this, let $\mathcal{S} = \{s_1, s_2, \ldots, s_n\}$ represent the set of sensors deployed in a drone. In this work, we specifically experimented with the following sensors: accelerometer, magnetometer, gyroscope, barometer, and temperature sensors.

▶ **Definition 1** (Sensor). *Let $s_i \in \mathcal{S}$ denote a sensor within the drone's sensor set $\mathcal{S}$, s_i, reflects distinct observations or readings about the environment or the drone itself.*

These sensor readings are paramount, forming the foundational data points required for evaluating the reliability and functionality of the sensors, and consequently, ensuring the optimal performance of the drone. Each sensor provides a stream of data over time, which we can represent as an observation vector. This vector captures the sensor's readings within a specific time frame.

▶ **Definition 2** (Observation Vector). *For each sensor s_i, the collected values or readings over a period form an observation vector, $O^{s_i}_{j,j+t} = [o^{s_i}_j, o^{s_i}_{j+1}, \ldots, o^{s_i}_{j+t}]$, where each element represents an individual observation of sensor s_i from j to $j+t$.*

When considering all sensors together, we can aggregate their observation vectors over the same time frame into what we refer to as Window Observation Vectors. This gives a comprehensive snapshot of the drone's sensor readings during a particular interval.

▶ **Definition 3** (Window Observation Vectors). *We denote $O_{j,j+t}$ as the Window Observation Vectors, which consolidates all individual sensor observation vectors within a window time frame from j to $j+t$. Formally expressed as $O_{j,j+t} = \{O^{s_1}_{j,j+t}, O^{s_2}_{j,j+t}, \ldots, O^{s_n}_{j,j+t}\}$.*

Understanding the relationships between different sensors is crucial for fault detection. Sensors that are correlated may exhibit patterns that can indicate potential issues.

▶ **Definition 4** (Correlated Sensors). *Let $s_i, s_f \in \mathcal{S}$. The predicate $\sigma(s_i, s_f) = --xttrue$ asserts that sensors s_i and s_f are correlated, implying that their respective Observation Vectors exhibit a correlated relationship.*

In our implementation, we computed the correlation between sensors using Pearson correlation calculations, which identify meaningful relationships between sensor readings.

Observation Vectors can exhibit various patterns, where a pattern is defined as a specific trend or behavior in the sensor readings over a period of time. For instance, a "constant" pattern signifies that the sensor readings do not change over time. We categorize these patterns into a set $\mathcal{P}$, which includes patterns like "constant," "drift," and "abrupt."

▶ **Definition 5** (Pattern Recognition). *Let the function $\pi : O \to \mathcal{P}$ signify that a given Observation Vector $O^{s_i}_{j,j+t}$ exhibits a pattern $p \in \mathcal{P}$, where $\mathcal{P}$ is the set of all possible patterns.*

For example, the predicate $\pi(O^{s_i}_{j,j+t})$ = "constant" if all values in the Observation Vector of sensor s_i within the time window j to $j+t$ are identical. Similarly, $\pi(O^{s_i}_{j,j+t})$ = "drift" if the derivative of the values in the Observation Vector shows a consistent increase or decrease beyond a certain point in time. Finally, $\pi(O^{s_i}_{j,j+t})$ = "abrupt" if the Observation Vector demonstrates a sudden and significant change in values, indicating a potential abrupt in the sensor readings.

In our research, we focus on detecting faults characterized by three primary patterns: abrupt, drift, and constant behaviors. These patterns often raise suspicion of malfunctions and can serve as indicators of potential sensor issues. In contrast, other patterns, such as fluctuations with varying intensities, are typically indicative of normal sensor operation. These normal patterns are explored further in the article in the evaluation section. Any sensor behavior can be considered a type of pattern, and while most are associated with normal drone function, recognizing and categorizing these patterns is essential for identifying potential malfunctions.

With these concepts in place, we can now define the Fault Detection Problem, which is central to our work. The goal is to determine whether any sensor within the drone is exhibiting patterns indicative of a fault.

▶ **Definition 6** (Fault Detection Problem). *Given the set of sensors $\mathcal{S}$ and their corresponding Window Observation Vectors $O_{j,j+t}$, the fault detection mechanism aims to determine whether sensor $s_i \in \mathcal{S}$ is exhibiting a pattern $x \in \mathcal{P}$ indicative of a fault.*

Accurately identifying correlated sensors, as represented by the predicate $\sigma(s_i, s_f)$ and recognizing predefined patterns within the sensor Observation Vectors, denoted by the predicate $\pi(O^{s_i}_{j,j+t}, p)$ are key components in approaches that rely on correlation-based fault detection. A robust solution must carefully analyze these correlations and patterns to differentiate genuine faults from innocuous variations or sensor noise, ensuring the reliability and safety of drone operations.

3.1 The sensor-based fault detection

Previous studies [11] proposed leveraging sensor correlations to identify faults in drone systems. The idea is to examine whether a sensor is displaying abnormal patterns, and then determine if this abnormality disrupts its usual correlation with other sensors. This brings us to the discussion of two significant methodologies:

Algorithm 1 Simple-SFDD.

```
   Input: sensor data stream $D$, window size $W$, $threshold$
   Output: faulty sensor identification $F$
1  for each sliding window do
2      for each pair of sensors $(i, j)$ do
3          Calculate Pearson correlation coefficient $\rho_{i,j}$ from first half of the window;
4          if $|\rho_{i,j}| > threshold$ then
5              Mark sensors $i$ and $j$ as correlated;
6          end
7      end
8  end
9  for each sensor $i$ in the second half do
10     if $i$ has a pattern of a fault then
11         if no other correlated sensor in the first half of the window has the same pattern
            then
12             Mark sensor $i$ as faulty;
13         end
14     end
15 end
```

Simple-SFDD

The first strategy is named *Simple-SFDD* [12], and it operates in real-time. This method begins each flight with the assumption that all sensors are functioning correctly. It uses the initial portion of each sliding window to analyze the relationships between every pair of sensors, utilizing Pearson correlation to identify clear and linear relationships. A predetermined threshold is used to determine whether the sensors are indeed correlated. In the latter portion of the window, each sensor is assessed for abnormal patterns such as drift, abrupt, or constant readings. If a sensor exhibits such patterns, we look for another sensor that was correlated with it in the initial portion and is displaying the same abnormal pattern. If no such correlated sensor is found, the original sensor is flagged as faulty. This approach enables the timely detection of sensor faults, allowing for immediate corrective action and preventing potential damage. A detailed algorithm can be found in Algorithm 1.

The *Simple-SFDD* method analyzes correlations within a sliding window to identify temporal correlations. However, it has two key limitations. First, normally uncorrelated sensors may temporarily appear correlated, leading to false negatives. Second, sensors expected to be correlated may occasionally show a lack of correlation, resulting in false positives and incorrect fault reports. Additionally, the method's requirement for correlated sensors to exhibit identical patterns is overly strict. A more flexible approach that allows for varying patterns could improve the heuristic's versatility and applicability.

Extended-SFDD

The observed limitations of the Simple method paved the way for the development of an enhanced approach, referred to as *Extended-SFDD* [11] builds on its predecessor by integrating both online and offline components. The offline phase includes several key steps: First, feature extraction adds a virtual sensor s_d for each sensor s_i created from the differentials of s_i's raw readings. Next, the correlation detection process is conducted offline, using a fault-free historical record. This step evaluates every pair of sensors for correlation throughout the entire flight, ensuring that only correlations sustained over the full operation are considered, thus filtering out insignificant temporal discrepancies.

This approach allows for a more flexible fault detection heuristic by accommodating diverse patterns that represent normal behavior. For correlated sensor pairs, a tuple set is constructed offline, capturing the various patterns observed during normal flights. For example, if a sensor typically shows a 'drift', the correlated sensor's common patterns during normal operation – such as small fluctuations – are included in the set. These patterns are mapped to the corresponding dynamic window size, meaning that for a specific time window (e.g., 10 seconds), the system expects to see the 'drift' in one sensor and corresponding behavior, such as fluctuations, in the correlated sensor.

In the online phase, each time window during the flight is carefully analyzed. The focus is on identifying sensors that are now exhibiting patterns suggestive of a fault. If such patterns are detected, the system checks whether a correlated sensor is showing an unrecognized, abnormal pattern. For instance, if sensor X shows an abrupt pattern and a correlated sensor Y shows slight fluctuations, the system will verify whether this combination has been observed during normal flight conditions. If this combination of patterns has been seen before, the current operational state is considered normal, and no fault is flagged. However, if this combination is new or abnormal, it signifies a potential fault.

The *Extended-SFDD* method, while innovative, has its limitations. A key challenge is the requirement for sensors to maintain correlation throughout the entire flight, which can lead to the oversight of intermittently correlated sensors. While lowering the correlation threshold might address this, it risks falsely identifying uncorrelated sensors as correlated, which hinders accurate fault detection.

A more complex issue arises from the way sensors exhibit patterns in normal data. For example, a sensor displaying a drift pattern might normally correlate with another sensor that shows all possible patterns alongside the drift. As a result, if sensor X shows a drift pattern, it might never be flagged as a fault because there will always be a corresponding sensor Y exhibiting some pattern that has been observed before in normal conditions. This means that sensor X's drift will always appear normal, even if it indicates a fault, because the correlation with sensor Y's varied patterns has already been established as standard. This situation underscores the overly generalized nature of the fault detection heuristic, highlighting the need for refinement to accurately detect faults.

In light of these challenges, and the increased complexity within the domain of drones, we developed the methodologies presented in this article. Two new algorithms, *Combined-SFDD* and *Knowledge-Based-SFDD*, were created, both utilizing correlation detection. These methodologies will be discussed in the ensuing section.

4 Methodology

In this paper, we introduce two innovative methods for detecting faults in drones, with a focus on analyzing correlations within their operational components. These methods are crafted to address the shortcomings of previous correlation-based approaches, which have been found lacking in certain scenarios. By combining offline analysis with real-time monitoring, and leveraging a data-driven knowledge-based system, our methods offer a comprehensive and more effective solution. These approaches not only tackle the challenges faced by earlier models but also excel in situations where the original methods struggle due to the complex nature of data patterns and distributions.

4.1 Combined-SFDD

The first methodology, termed *Combined-SFDD* is a hybrid approach that integrates the strengths of both the Simple and Extended-SFDD methods. The goal is to leverage the advantages of each, leading to a more accurate and reliable fault detection system. This method operates in two phases: the offline phase and the online phase.

Offline Phase. The process begins with the execution of the Extended-SFDD method, where the following steps are performed offline:

1. For each identified feature, an additional feature is generated by calculating the differences between consecutive values of the original feature.
2. Sensors are correlated based on data from the entire flight duration, rather than within a temporary window, to identify sustained correlations over time.
3. For sensors that are correlated during normal flight conditions, a set of coexisting patterns is established and stored for reference.

Online Phase. In the online phase, flight data is continuously analyzed within a moving time window. Sensors are monitored for any patterns indicative of potential faults, such as constant, abrupt, or drift behaviors. If a sensor shows a suspicious pattern, the system first checks whether a correlated sensor displays a pattern that has been previously identified as normal based on the offline analysis. If such a correlated sensor is found, the system considers the situation normal and no fault is flagged.

However, if no correlated sensor with a normal pattern is identified, the Combined-SFDD approach does not stop. It then applies the Simple-SFDD method to the same time window as an additional step. This involves examining the correlations observed in the first half of the time window and verifying whether these correlations persist in the second half. If the correlation fails to hold, a fault is declared.

This dual approach ensures that if the Extended-SFDD method fails to detect a fault, the Simple-SFDD method is used as a secondary check to identify issues that may have been missed. While this thorough process may slightly increase the likelihood of false positives, it significantly enhances the detection of true positives, ensuring that potential faults are not overlooked, especially in complex and varied operational scenarios.

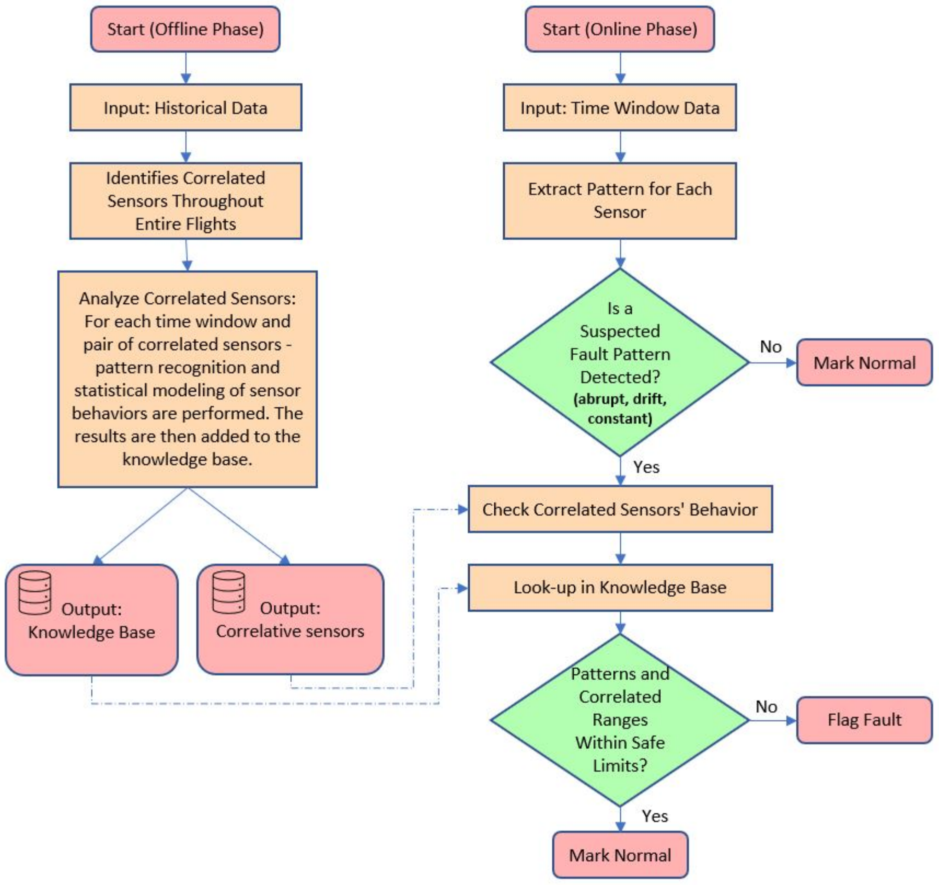

Start (Offline Phase)
Input: Historical Data
Identifies Correlated Sensors Throughout Entire Flights
Analyze Correlated Sensors: For each time window and pair of correlated sensors - pattern recognition and statistical modeling of sensor behaviors are performed. The results are then added to the knowledge base.
Output: Knowledge Base
Output: Correlative sensors
Start (Online Phase)
Input: Time Window Data
Extract Pattern for Each Sensor
Is a Suspected Fault Pattern Detected? (abrupt, drift, constant)
No
Mark Normal
Yes
Check Correlated Sensors' Behavior
Look-up in Knowledge Base
Patterns and Correlated Ranges Within Safe Limits?
No
Flag Fault
Yes
Mark Normal

Figure 2 Flowchart of the Knowledge-Based-SFDD Algorithm. This diagram illustrates the process of offline and online phases for detecting sensor faults in real-time.

4.2 Knowledge-Based-SFDD

The *Knowledge-Based-SFDD* strategy introduces an advanced method that merges elements from Extended-SFDD with sophisticated knowledge-based system techniques. This approach is designed to deliver a deeper and more precise analysis of sensor behavior, providing a robust framework for fault detection.

Offline Phase. In the offline phase, Knowledge-Based-SFDD goes far beyond the basic steps outlined in the Combined-SFDD approach. This phase involves an exhaustive learning process that thoroughly analyzes all available data from previous flights, focusing on each sensor individually to understand its "normal" behavior and patterns. The aim is not merely to identify existing templates but to refine and model these templates within defined parameters.

For example, instead of simply noting that sensor X exhibits drifting while sensor Y shows slight fluctuations, Knowledge-Based-SFDD details the behavior more precisely: sensor X typically drifts at a slope between -5 and 5, while sensor Y fluctuates within a range of 2 to 4. By modeling the behavior of each sensor in this way, the system defines what constitutes normal and abnormal patterns, considering both the type of pattern and the specific range of values associated with it.

This meticulous modeling process allows for a comprehensive understanding of each sensor's behavior, distinguishing between normal variations and true anomalies. All identified patterns and correlations, along with their precise parameters, are compiled into an extensive knowledge base. This repository serves as a reference for real-time analysis, enabling the system to distinguish between normal operational variations and genuine anomalies effectively.

Online Phase. In the online phase, Knowledge-Based-SFDD continuously monitors real-time flight data within a moving time window, meticulously analyzing sensor readings for patterns that might indicate potential faults, such as constant values, abrupt changes, or unusual drifts. When a suspicious pattern is detected, the system references the comprehensive knowledge base developed during the offline phase to determine whether a correlated sensor exhibits a corresponding normal pattern within the expected parameter ranges. This assessment is not limited to merely recognizing patterns; it involves a detailed evaluation of whether the detected patterns fall within the predefined safe operational limits. If a correlated sensor is found that displays a recognized and acceptable pattern within these ranges, the system considers the situation normal, and no fault is declared. However, if no such correlated sensor is identified, or if the patterns observed fall outside the established safe ranges, the system conclusively flags the sensor as faulty. This rigorous process ensures a more accurate and reliable fault detection, minimizing false positives while effectively identifying genuine anomalies.

In summary, the process of Knowledge-Based-SFDD, including its offline and online phases, is visually represented in Figure 2. This flowchart provides a clear depiction of how the algorithm models sensor behavior, checks correlated patterns, and makes real-time decisions based on predefined parameters.

Benefits of Knowledge-Based-SFDD: The precision of Knowledge-Based-SFDD offers several critical advantages:

- **Enhanced Accuracy:** By defining normal behavior with specific parameters, Knowledge-Based-SFDD significantly reduces the risk of false positives and negatives, ensuring that only genuine faults are flagged.
- **Comprehensive Analysis:** This approach provides a holistic understanding of sensor interactions, considering both the patterns and their permissible ranges, which leads to more informed and accurate fault detection.
- **Adaptability:** The model adapts to the unique characteristics of each sensor, offering a tailored analysis that can accommodate various operational scenarios and sensor configurations.

By modeling the entirety of each sensor's behavior, Knowledge-Based-SFDD offers a nuanced and highly accurate fault detection system. This method not only identifies deviations from expected patterns but also assesses the severity and relevance of these deviations in real-time. The result is a robust and reliable system that enhances the safety and efficiency of drone operations, ensuring that anomalies are detected early and accurately.

5 Evaluation

Next, we precisely define the experiments we conducted to validate our methodologies. This will include detailed descriptions of the test scenarios, the datasets used, and the specific metrics employed to assess performance. Following that, in the Results chapter, we will present and analyze the outcomes of these experiments, demonstrating the effectiveness and superiority of our proposed methods, Combined-SFDD and Knowledge-Based-SFDD, in accurately detecting faults under various conditions.

5.1 Data set

One of the primary contributions of this study is the use of real-world data, collected in real-time during the communication between a drone and its control unit. This data is subject to various external factors such as weather conditions, heat, and humidity, which can significantly influence the drone's behavior and sensor readings. We have taken meticulous care to ensure the accuracy and reliability of the data used in our experiments, thereby guaranteeing that the results obtained are robust and applicable to real-world scenarios.

Collecting such data presents significant challenges. To overcome these obstacles, we partnered with Maris, a company specializing in Intelligent Video Surveillance and Analytics. Maris provided us with the technical expertise and tools necessary for capturing high-quality, real-world data.

For our study, we used drones from Aero Sentinel, a leading manufacturer in the tactical UAV industry, known for its high-performance military drones used in various applications, including combat, Homeland Security (HLS), police operations, and civilian use. We leveraged a substantial amount of raw flight data collected from these actual Aero Sentinel drones. This collaboration allowed us to gather a comprehensive dataset that includes detailed sensor data recorded during real-world drone operations under various environmental conditions.

The data was gathered during controlled flights, with sensor data recorded and stored in files, each containing approximately 35 minutes of flight time. A total of 30 flights were conducted, resulting in about 17 hours of documented flight data. To maintain consistency, all flights were conducted using the same drone model: Aero Sentinel Military UAV Sentinel G2 [4]). This extensive dataset comprises approximately 611,000 records, underscoring the substantial volume of data collected for analysis and model training.

The controlled flights involved testing a variety of real-world flight scenarios to capture diverse operational conditions. These included hovering at different altitudes under varying wind speeds, flights at multiple velocities, and takeoff and landing maneuvers at different angles. By incorporating such a range of flight conditions, we were able to simulate the types of environments a drone might encounter in practice, ensuring that the dataset reflected a comprehensive spectrum of sensor behaviors.

[4] https://www.aero-sentinel.com/military-drones/military_drone_sentinel_g2/

Table 1 Summary information for the sensors.

Sensor	Measure	Description	Units
Accelerometer	Xacc	X acceleration	int
	Yacc	Y acceleration	int
	Zacc	Z acceleration	int
Magnetometer	Xmag	X Magnetic field	int
	Ymag	Y Magnetic field	int
	Zmag	Z Magnetic field	int
Gyroscope	Xgyro	Angular speed around X axis	int
	Ygyro	Angular speed around Y axis	int
	Zgyro	Angular speed around Z axis	int
Barometer	Press abs	Absolute pressure	hPa
	Temperature	Absolute pressure temperature	cdegC

The collected data encompasses readings from all of the drone's sensors, recorded at varying sampling intervals for each sensor. Our study specifically focused on sensors providing information about gyroscope, accelerometer, magnetometer, barometer, and drone temperature. Detailed information about the available sensors is provided in Table 1.

The dataset we created, with the help of capturing and translating packets transmitted over the network during real flights, represents a significant contribution of this study. Developing this dataset was a challenging process, involving the meticulous collection and interpretation of data under diverse environmental conditions. After extensive pre-processing operations, this unique dataset is now available for use, offering a rich and authentic resource that is crucial for evaluating and improving fault detection methods in drones. Its real-world applicability makes it an invaluable tool for advancing research and ensuring that our findings are relevant and effective in practical scenarios.

The entire dataset, including the original data and the data post-fault injection, is available in the parent Git repository. Access is via the following link:
`https://github.com/inbalros/SensorFaultDetection_Correlation`

Data Collection and Processing

Sensor data was collected from the communication between the drone and its control unit, recorded in a binary LOG file. Using the Mavlink v1 protocol, relevant sensor data packets were identified and processed, with timestamps provided every 0.1 seconds to create a time-series dataset for fault prediction.

Due to the high data collection frequency, a large volume of records was generated, requiring significant processing time. To optimize this process, we explored whether resampling the data at longer intervals, specifically half a second and one second, could still maintain accuracy while reducing the dataset size. Although this approach decreased the number of records, initial experiments showed that the algorithm's performance was less effective compared to using the full dataset at the 0.1-second frequency.

After collecting the data, we encountered two challenges in creating a usable tabular dataset. The first challenge involved handling multiple samples recorded from certain sensors within the same time interval. The second issue was dealing with missing samples from other sensors during certain time intervals.

To solve the multiple samples problem, we implemented two methods:

1. Avg_{comb} **(Averaging Combination):** This method averages the values when multiple samples exist within a time window and fills in missing values accordingly.
2. $Last_{comb}$ **(Last Value Combination):** This method takes the last value in cases of multiple samples within a time window and completes any missing values.

Preliminary experiments indicated that the $Last_{comb}$ method yielded the best results.

To address the missing values issue, four strategies were employed for imputing the missing data:

1. $Last_{missing}$ **(Last Value Imputation):** This method fills in missing values with the last available value for that sensor. If no prior value exists, a default value is assigned.
2. $SMA_{missing}$ **(Simple Moving Average):** A weighted moving average is calculated using the last 10 samples from the same sensor, with statistical prediction applied to estimate the missing value. If fewer than 10 samples are available, a default value is used.
3. $LR_{missing}$ **(Linear Regression):** Linear regression is applied by gathering all available samples of the sensor up to the missing point and predicting the missing value based on this data.
4. $ARIMA_{missing}$ **(Autoregressive Integrated Moving Average):** ARIMA, a time series forecasting model, is used to predict missing values. This model requires careful parameter configuration, achieved through an offline grid search to identify the best-performing ARIMA model for each sensor. The chosen ARIMA model is then used to predict missing values based on all available sensor data up to that point.

Extensive experiments demonstrated that the $Last_{missing}$ method consistently provided the best results in terms of accuracy and reliability.

Simulating Sensor Faults in Flight Data

A critical challenge in this research is the lack of real fault data, as the available dataset only contains normal flight information. To address this, we employed simulation techniques to generate artificial sensor faults, which were then integrated into the normal flight data. This approach allows for effective training and evaluation of our fault detection models.

An automated simulation tool was developed to generate data files containing simulated faults, providing a controlled environment for testing and refining our models. The types of faults simulated include: (1) *Constant Faults:* Represent a persistent deviation from normal sensor readings. (2) *Drift Faults:* Describe a gradual change in sensor readings over time. (3) *Abrupt Faults:* Indicate sudden and significant changes in sensor readings, which can be expressed as sharp increases or decreases.

To ensure realistic simulations, expert insights from the Maris and Aero-Sentinel companies were combined with thorough statistical analysis of the sensor data. This analysis helped establish the range of values each sensor can sample, including averages and standard deviations, which served as guidelines for creating realistic fault simulations. The simulated faults were carefully designed to stay within the possible value ranges for each sensor.

The following ranges were defined for each type of fault:

1. constant:
   - Fixed to the first instance value within the current time window.
   - Fixed to the maximum possible sensor value.
   - Fixed to the minimum possible sensor value.
   - Fixed to 0, simulating a scenario where the sensor stops receiving information (if 0 is a possible value for that sensor).

2. drift:
   - A decrease from the first value in the time window, following the steepest negative slope observed for that sensor.
   - An increase from the first value in the time window, following the steepest positive slope observed for that sensor.
3. abrupt:
   - A sudden increase from the initial value in the time window, based on the largest positive jump the sensor is capable of.
   - A sudden decrease from the initial value in the time window, based on the largest negative drop the sensor is capable of.

To ensure comprehensive testing, the normal flight data was segmented into slightly larger time windows than those used in the fault detection algorithm. Three distinct data files were created, each containing a different type of fault: constant, abrupt, and drift. Faults were systematically introduced into every second time window, covering all sensors and representing all possible values of the specific fault type for each file. In total, 90 files (3 types of faults $\times$ 30 different scenarios) were generated, each containing various faults across different sensors and flight conditions.

5.2 Competing Approaches and Pattern Analysis

To evaluate the effectiveness of our correlation-based algorithm, several different approaches were tested:

1. **Simple-SFDD:** This method, based on the work of Khalastchi et al. [11], uses a straightforward approach to detect faults by analyzing sensor correlations within a dynamic time window. Further details can be found in the Background section (3.1).
2. **Extended-SFDD:** An enhancement of the Simple-SFDD, based on the work of Khalastchi et al. [11], this method introduces offline analysis to refine correlation detection, making it more robust. It is also detailed in the Background section (3.1).
3. **Combined-SFDD:** Building on the principles of Simple and Extended-SFDD, this method integrates both approaches to improve fault detection accuracy and is specifically adapted for drone operations. This method is discussed in detail in the Methodology section (4).
4. **Knowledge-Based-SFDD:** An advanced method that incorporates knowledge-based system techniques to model and analyze sensor behavior, offering enhanced fault detection capabilities. This approach is also detailed in the Methodology section (4).

5.3 Experimental Process and Setup

To evaluate the effectiveness of our fault prediction model, we conducted a series of experiments using K-fold cross-validation, with K set to 3. In each experiment, 20 flight files were randomly selected for training, while 10 flight files were designated for testing. These test files were drawn from the simulated fault files generated during the fault simulation phase, with each test file corresponding to three different simulated fault scenarios, resulting in a total of 30 variations of test files.

The training data plays a crucial role in the offline phases of the correlation-based algorithms, allowing the models to learn and adapt. In contrast, the testing data, used for the online phase only, (the Simple-SFDD approach, which operates solely online) did not utilize the training files within its fold.

For correlation-based detection, we applied a Pearson correlation coefficient of 40% between two sensors to identify meaningful relationships and detect faults.

Additionally, during the Knowledge-Based-SFDD algorithm, we managed the Knowledge Base to contain the modeling of patterns that were suspected to indicate faults, such as abrupt and drift patterns. The Knowledge-Based approach not only identifies patterns but also models their specific values, such as the magnitude of the jump in abrupt patterns and the slope in drift patterns, against the maximum possible values observed for two correlated sensors. This modeling is context-specific. Instead of comparing a sensor's behavior across its entire history, we analyze how a pattern like a drift behaves when a correlated sensor exhibits a particular pattern.

Moreover, we tested different sizes of time windows across all four algorithms: 6 seconds, 10 seconds, 14 seconds, and 20 seconds. The number of instances varied depending on the data representation method used. For raw data, where each record represents 0.1 seconds, a 10-second window contains 100 records.

5.4 Evaluation Metrics

In addressing the problem outlined in this article, the most relevant metrics for evaluating the success of our fault detection algorithms are the True Positive Rate (TPR) and the False Positive Rate (FPR).

True Positive Rate (TPR). This metric measures the proportion of time windows where the algorithm correctly identified a fault when one was actually present. It is calculated as the number of true positives (correctly identified faults) divided by the total number of actual faults ($TP/(TP+FN)$). Importantly, a true positive (TP) is only considered valid if the fault is detected in the correct sensor. If the algorithm detects a fault but incorrectly diagnoses it in a different sensor, this is not counted as a correct detection. Thus, TPR reflects the algorithm's success rate in accurately detecting both the fault and the affected sensor. Essentially, TPR reflects the algorithm's success rate in detecting genuine faults.

False Positive Rate (FPR). This metric indicates the proportion of time windows where the algorithm incorrectly flagged a fault when none existed. It is calculated as the number of false positives (incorrectly identified faults) divided by the total number of windows where no fault was present ($FP/(FP+TN)$). In addition, FPR includes cases where a fault is present, but the algorithm incorrectly identifies the wrong sensor. So, even if a fault exists, if the diagnosis points to the wrong sensor, this is counted as a false positive, highlighting the algorithm's error rate in both detecting non-existent faults and misdiagnosing sensor faults.

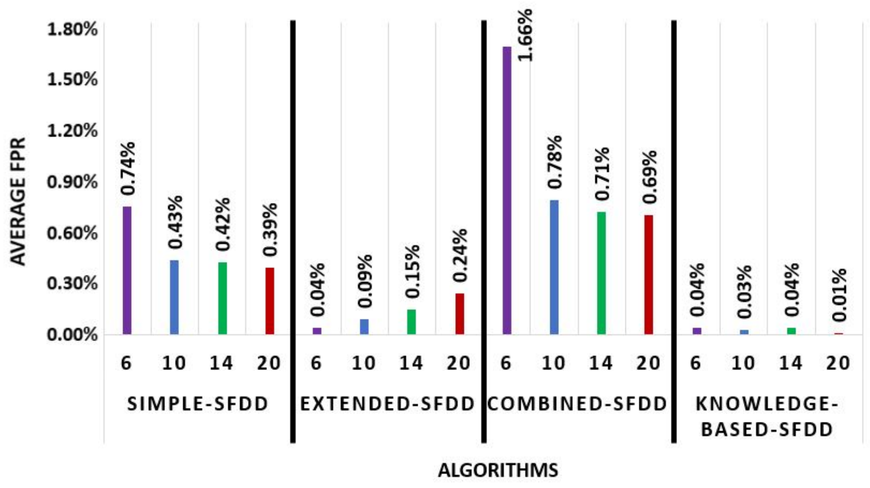
In addition to these primary metrics, we also considered using Precision and Recall. Recall, is identical to TPR, measuring the accuracy of the algorithm in identifying actual faults. Precision, evaluates how many of the time windows identified as faulty by the algorithm were indeed faulty, out of all the time windows flagged as faulty, whether correctly or not. However, Precision may be less relevant in our context due to the significant imbalance between the number of time windows without faults and those with faults.

6 Results

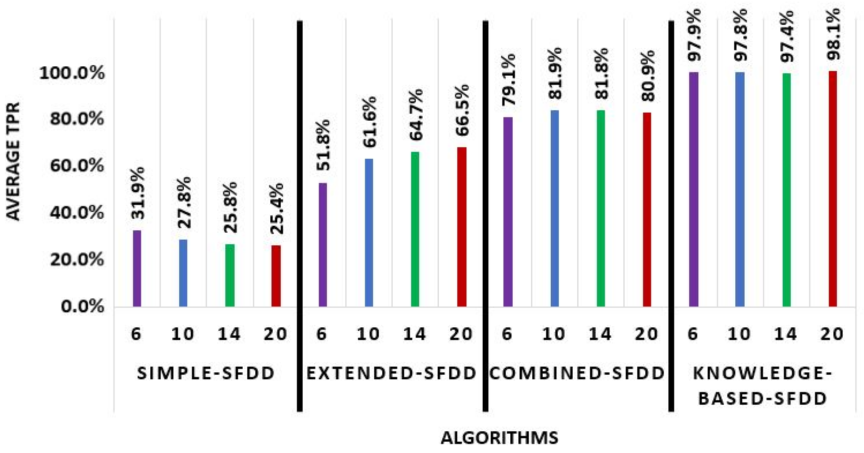
In the following figures 3a, 3b, we present a comparison of the four algorithms – Simple-SFDD, Extended-SFDD, Combined-SFDD, and Knowledge-Based SFDD – using two key metrics: Average False Positive Rate (FPR) and Average True Positive Rate (TPR). These graphs visually demonstrate how each algorithm performs across various time window sizes (6, 10, 14, and 20 seconds), providing a clearer illustration of the algorithm's effectiveness.

Figure 3a depicts the Average FPR for each algorithm. A lower FPR indicates fewer instances of false fault detection, where a malfunction is falsely identified. It can be observed that Knowledge-Based SFDD consistently outperforms other methods, achieving the lowest FPR across all window sizes, while Combined-SFDD demonstrates the highest FPR values.

Figure 3b shows the Average TPR, which reflects the accuracy of each algorithm in detecting genuine faults. A higher TPR suggests better fault detection. Once again, Knowledge-Based SFDD stands out with the highest TPR in all configurations, while Simple-SFDD yields the lowest TPR.



(a) Average False Positive Rate (FPR).



(b) Average True Positive Rate (TPR).

Figure 3 Comparison of FPR and TPR for Simple-SFDD, Extended-SFDD, Combined-SFDD, and Knowledge-Based SFDD across different time windows (6, 10, 14, and 20 seconds).

The results show that our Knowledge-Based-SFDD algorithm consistently delivers excellent performance, achieving an average TPR of 98%, with an impressively low FPR of just 0.03%. This indicates that the Knowledge-Based-SFDD not only accurately predicts malfunctions but also minimizes the instances of false alarms – where a fault is indicated, but none exists. The model's algorithm systematically outperforms the other approaches across all configurations, clearly demonstrating its robustness and reliability.

The Simple-SFDD, which operates exclusively online, achieves the lowest success rates in fault detection. Its performance is hampered by its reliance on short-term correlations, which often fail to capture the complexity of sensor interactions over time. Consequently, it struggles with both lower TPR and a higher incidence of false positives. Expanding the window size does little to alleviate these issues, as the difficulty in detecting correlations within larger windows further complicates fault detection.

The Extended-SFDD improves upon the Simple approach by incorporating offline analysis, yet it still falls short in several areas. The primary limitation here is that the presence of multiple potential patterns for each sensor makes it difficult to pinpoint specific faults. However, as the window size increases, the algorithm gains confidence in identifying patterns, resulting in a gradual improvement in TPR.

The Combined-SFDD, which integrates elements of both the Simple and Extended approaches, shows a notable improvement in fault detection. In this method, if the Extended-SFDD does not detect a fault, the algorithm applies the Simple-SFDD, but with additional features such as differentials. This enhanced version of the Simple-SFDD allows it to detect more faults than the standard Simple approach, as it now takes into account the extra features. However, this comes at the cost of a higher FPR, indicating that while the Combined-SFDD is more sensitive to faults, it is also more prone to false positives due to the broader feature set being examined.

The Knowledge-Based-SFDD emerges as the most effective algorithm across all configurations. It consistently delivers the highest TPR and the lowest FPR, making it the best choice for reliable fault detection. The model's ability to learn and model complex sensor behaviors, combined with the increased confidence provided by larger window sizes, ensures that it excels in both detecting true faults and avoiding false alarms.

7 Conclusions and Future Work

Drones, or unmanned aerial vehicles (UAVs), play an increasingly important role across various industries, making their reliability and safety-critical. A major challenge in maintaining drone operations is the detection and diagnosis of sensor faults, which can significantly impact performance. In this study, we focused on detecting these sensor-related faults to enhance the safety and efficiency of drones. Through a crucial collaboration with Maris, a pioneer in AI edge video and analytics technologies, we gathered a comprehensive dataset from real drones. Utilizing Maris's advanced Mercury Nano system, we recorded the communication between an Aero-Sentinel Military UAV Sentinel G2 quadcopter drone and its control unit, capturing detailed sensor data during operation. This partnership was instrumental in providing the high-quality data that formed the foundation of our research. We developed and tested correlation-based algorithms, including two new methods – Combined-SFDD and Knowledge-Based-SFDD specifically designed to address the complexities of drone sensor data. Our experiments demonstrated the clear superiority of these algorithms, particularly the Knowledge-Based-SFDD, which achieved a 98% True Positive Rate (TPR) and a low False Positive Rate (FPR) of 0.03%. These results highlight the effectiveness of our approach in accurately detecting faults while minimizing false alarms.

Looking forward, several avenues for enhancing and expanding this research can be explored. One promising direction is the integration of audio data into the fault detection process. By incorporating sound analysis, we can potentially identify and diagnose faults that may not be detectable through sensor data alone, further improving the accuracy and reliability of the detection system. Another key area for future work is the exploration and comparison of deep learning algorithms for detecting structural faults in drones. While our current study focused on sensor-related faults, structural faults – such as those affecting the drone's frame or mechanical components – pose significant risks to drone operations. Leveraging deep learning techniques could provide more advanced and effective methods for identifying these types of faults, thereby enhancing the overall safety and performance of drone systems.

References

1 Mogens Blanke and Søren Hansen. Towards self-tuning residual generators for uav control surface fault diagnosis. In *2013 Conference on Control and Fault-Tolerant Systems (SysTol)*, pages 37–42. IEEE, 2013.

2 Mark Bowkett, Kary Thanapalan, and Ewen Constant. Failure detection of composites with control system corrective response in drone system applications. *Computers*, 7(2):23, 2018. `doi:10.3390/COMPUTERS7020023`.

3 Mingzhang Chen, Xuancheng Zhang, Xiaoshuang Xiong, Fanfei Zeng, and Wuhao Zhuang. Transformer: A multifunctional fast unmanned aerial vehicles–unmanned surface vehicles coupling system. *Machines*, 9(8):146, 2021.

4 Yuepeng Chen, Xia Hu, Qingyong Zhang, and Cong Zhang. Research on multi-classification method of uav sensor fault based on wavelet entropy and afwa-bp neural network. In *2017 32nd Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pages 837–842. IEEE, 2017.

5 Yuepeng Chen, Cong Zhang, Qingyong Zhang, and Xia Hu. Uav fault detection based on ga-bp neural network. In *2017 32nd Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pages 806–811. IEEE, 2017.

6 Merve Demircan and Coşku Kasnakoğlu. Aileron locking fault detection based on extended kalman filter for uav. In *Proceedings of the 3rd International Conference on Vision, Image and Signal Processing*, pages 1–6, 2019.

7 Xingjian Fu and Xinyao Geng. Fault estimation and robust fault-tolerant control for singular markov switching systems with mixed time-delays and uav applications. *Journal of Control Engineering and Applied Informatics*, 23(2):53–66, 2021.

8 Hussein Hamadi, Benjamin Lussier, Isabelle Fantoni, and Clovis Francis. Data fusion fault tolerant strategy for a quadrotor uav under sensors and software faults. *ISA transactions*, 2022.

9 Samiadji Herdjunanto. Actuator fault signal isolation of an unmanned aerial vehicle (uav) quadrotor based on detection filter. In *2016 8th International Conference on Information Technology and Electrical Engineering (ICITEE)*, pages 1–6. IEEE, 2016.

10 Gino Iannace, Giuseppe Ciaburro, and Amelia Trematerra. Fault diagnosis for uav blades using artificial neural network. *Robotics*, 8(3):59, 2019. `doi:10.3390/ROBOTICS8030059`.

11 Eliahu Khalastchi and Meir Kalech. A sensor-based approach for fault detection and diagnosis for robotic systems. *Autonomous Robots*, 42:1231–1248, 2018. `doi:10.1007/S10514-017-9688-Z`.

12 Eliahu Khalastchi, Meir Kalech, and Lior Rokach. Sensor fault detection and diagnosis for autonomous systems. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, pages 15–22, 2013. URL: `http://dl.acm.org/citation.cfm?id=2484927`.

13 Qiang Li, Ranyang Li, Kaifan Ji, and Wei Dai. Kalman filter and its application. In *2015 8th International Conference on Intelligent Networks and Intelligent Systems (ICINIS)*, pages 74–77. IEEE, 2015.

14 Hamid Maqsood, Muhammad Taimoor, Zahid Ullah, Nihad Ali, and Muhammad Sohail. Novel sensor fault detection and isolation for an unmanned aerial vehicle. In *2021 International*

Bhurban Conference on Applied Sciences and Technologies (IBCAST), pages 486–493. IEEE, 2021.

15 Francisco J Montáns, Francisco Chinesta, Rafael Gómez-Bombarelli, and J Nathan Kutz. Data-driven modeling and learning in science and engineering. *Comptes Rendus Mécanique*, 347(11):845–855, 2019.

16 Radosław Puchalski and Wojciech Giernacki. Uav fault detection methods, state-of-the-art. *Drones*, 6(11):330, 2022.

17 Vidyasagar Sadhu, Saman Zonouz, and Dario Pompili. On-board deep-learning-based unmanned aerial vehicle fault cause detection and identification. In *2020 ieee international conference on robotics and automation (icra)*, pages 5255–5261. IEEE, 2020. `doi:10.1109/ICRA40945.2020.9197071`.

18 Hazim Shakhatreh, Ahmad H Sawalmeh, Ala Al-Fuqaha, Zuochao Dou, Eyad Almaita, Issa Khalil, Noor Shamsiah Othman, Abdallah Khreishah, and Mohsen Guizani. Unmanned aerial vehicles (uavs): A survey on civil applications and key research challenges. *Ieee Access*, 7:48572–48634, 2019. `doi:10.1109/ACCESS.2019.2909530`.

19 Hassan Shraim, Ali Awada, and Rafic Youness. A survey on quadrotors: Configurations, modeling and identification, control, collision avoidance, fault diagnosis and tolerant control. *IEEE Aerospace and Electronic Systems Magazine*, 33(7):14–33, 2018.

20 Venkat Venkatasubramanian, Raghunathan Rengaswamy, Kewen Yin, and Surya N Kavuri. A review of process fault detection and diagnosis: Part i: Quantitative model-based methods. *Computers & chemical engineering*, 27(3):293–311, 2003. `doi:10.1016/S0098-1354(02)00160-6`.

21 Benkuan Wang, Datong Liu, Yu Peng, and Xiyuan Peng. Multivariate regression-based fault detection and recovery of uav flight data. *IEEE Transactions on Instrumentation and Measurement*, 69(6):3527–3537, 2019. `doi:10.1109/TIM.2019.2935576`.

22 Yujiang Zhong, Wei Zhang, and Youmin Zhang. Sensor fault diagnosis for unmanned quadrotor helicopter via adaptive two-stage extended kalman filter. In *2017 international conference on sensing, diagnostics, prognostics, and control (SDPC)*, pages 493–498. IEEE, 2017.

23 Yujiang Zhong, Youmin Zhang, Wei Zhang, Junyi Zuo, and Hao Zhan. Robust actuator fault detection and diagnosis for a quadrotor uav with external disturbances. *IEEE Access*, 6:48169–48180, 2018. `doi:10.1109/ACCESS.2018.2867574`.