


## Article

# A Deep Learning Approach for Fault-Tolerant Data Fusion Applied to UAV Position and Orientation Estimation

Majd Saied <sup>1,\*</sup> , Abbas Mishi <sup>2</sup>, Clovis Francis <sup>3</sup> and Ziad Noun <sup>1</sup>

<sup>1</sup> Department of Electrical and Electronics Engineering, School of Engineering, Lebanese International University, Bekaa 146404, Lebanon; ziad.noun@liu.edu.lb

<sup>2</sup> Scientific Research Center in Engineering (CRSI), Faculty of Engineering, Lebanese University, Beirut 657314, Lebanon

<sup>3</sup> Laboratory of Mechanics, Surface and Materials Processing, Arts et Metiers ParisTech de Chalons en Champagne, Rue Saint Dominique, 51000 Châlons en Champagne, France; clovis.francis@ensam.eu

\* Correspondence: majd.saied@liu.edu.lb

**Abstract:** This work introduces a novel fault-tolerance technique for data fusion in Unmanned Aerial Vehicles (UAVs), designed to address sensor faults through a deep learning-based framework. Unlike traditional methods that rely on hardware redundancy, our approach leverages Long Short-Term Memory (LSTM) networks for state estimation and a moving average (MA) algorithm for fault detection. The novelty of our technique lies in its dual strategy: utilizing LSTMs to analyze residuals and detect errors, while the MA algorithm identifies faulty sensors by monitoring variations in sensor data. This method allows for effective error correction and system recovery by replacing faulty measurements with reliable ones, eliminating the need for a fault-free prediction model. The approach has been validated through offline testing on real sensor data from a hexarotor UAV with simulated faults, demonstrating its efficacy in maintaining robust UAV operations without resorting to redundant hardware solutions.

**Keywords:** sensor fusion; unmanned aerial vehicles; fault tolerance



**Citation:** Saied, M.; Mishi, A.; Francis, C.; Noun, Z. A Deep Learning Approach for Fault-Tolerant Data Fusion Applied to UAV Position and Orientation Estimation. *Electronics* **2024**, *13*, 3342. <https://doi.org/10.3390/electronics13163342>

Academic Editor: Sang Ik Han, Joobum Kim, Shiho Kim, Nurul Sarkar

Received: 3 July 2024

Revised: 17 August 2024

Accepted: 19 August 2024

Published: 22 August 2024



**Copyright:** © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Multisensor data fusion is extensively applied across various robotic mobile applications including environment mapping and sensor networks [1]. It involves the integration of data from various sensors and sources of information resulting in a more accurate description of the process of concern and to more accurate analysis than would be achieved by relying on a single sensor [2]. Data fusion is a field that encompasses numerous disciplines such as information theory, artificial intelligence and signal processing [3]. Its implementation can lead to increased data reliability, accuracy, and consistency.

Despite the clear advantages of data fusion, there exist several challenges and limitations that make their implementation difficult. One major challenge is the need for redundant sensors, which increases the risk of their failures. Additionally, there are issues associated to the used data, including the sensors' imperfection, and the specific application requirements. These factors can make it difficult to achieve accurate and reliable results [2,4,5].

The validation of data fusion techniques also presents two significant difficulties [6]. These are the following:

- The absence of a definitive reference or ground truth data for measuring the data fusion results.
- The difficulty of isolating the effect of the data fusion algorithm from other factors that can impact performance, such as sensor errors and noise.

An alternate solution to the validation challenges is to introduce fault-tolerance techniques in data fusion, which work to reduce or remove the impact of defects on the process's

performance [7]. Fault tolerance enables the system to continue operating accurately even in the presence of faults [8,9]. It is typically achieved through the implementation of fault detection and system recovery mechanisms. Fault detection identifies faults in the system, while system recovery works to correct or mitigate the effects of those faults. Together, these strategies allow the system to continue functioning correctly even when errors occur.

The use of fault-tolerant data fusion, specifically in aircraft navigation systems, has a history of over 50 years [10]. The current literature mainly focuses on duplication–comparison techniques for fault tolerance, which involve evaluating outputs from a minimum of two independent and duplicate modules that provide the same service. The duplication approach encompasses two, or more, redundancy techniques:

- One variation of the duplication method is the use of analytical models, which act as an alternative to physical sensors. A common data fusion technique used in this approach is the Kalman Filter, which employs a system model for the estimation of an observation that is redundant to the one provided by the actual sensor. The difference between the measurement estimated by the model and the one provided by the sensor is then adopted as a sign of faults. Examples of this type of duplication method can be found in the literature, such as [11–13].
- Another variation of the duplication method is the use of hardware redundancy, which involves combining multiple data sources. Unlike the analytical model-based approach, this technique is based on the evaluation of internal parameters. The work in [14] uses the temporal analysis of conflicts arising from data source fusion usage to detect malfunction. Other works like [15,16] suggest the use of dynamic and static reliability analyses of data sources. Other works discussing these techniques can be found in [17,18].

As autonomous unmanned aerial vehicles (UAVs) become more prevalent, fault-tolerant data fusion is emerging as an increasingly important requisite for a secure and trustworthy operation. UAVs are able to complete a variety of tasks in unknown conditions where human intervention is either impossible or unsafe [19]. However, these operations rely on sensors being susceptible to various faults [20,21]. As a result, it is crucial for UAVs to detect and diagnose sensor faults in order to ensure accurate state estimates. Research on multisensory fusion strategies for UAVs has been conducted for various applications such as position, velocity, and attitude estimation [22]. The real-world evaluation of these strategies on actual UAV systems was also considered in [23].

Various studies have investigated the implementation of fault tolerance in sensor fusion for UAV systems. In [24], a navigation system of UAVs was designed using a combination of height sensor measurements and a main Kalman Filter with sub-filters. A Chi-Square test was employed for fault isolation. Another study [25] developed a scheme for reliable UAV attitude estimation through the use of an Unscented Information Filter. Similar approaches were also explored in [26], comparing various Kalman Filters implemented for sensor fusion. A data fusion technique to tolerate both software and sensor failures in a quadrotor UAV was proposed in [27] using the duplication–comparison technique. A similar architecture was applied in [28] on a framework that makes use of extended Informational Kalman Filters for performing the state estimation of a quadrotor UAV and Bhattacharyya Distance for residual evaluation.

In the majority of studies examining these methods, state estimation is achieved through traditional Kalman and complementary filters. Despite their effectiveness, these traditional techniques present certain limitations [29]. Typically, attitude estimation for UAV systems is obtained through integrating the sensor measurements of an Inertial Measurement Unit (IMU). However, the correlation between attitude error and IMU error can be complex, making it challenging to establish a precise mathematical model. An alternative solution is to consider this relationship as a time series dataset, which can be effectively modeled using artificial neural networks [30].

Traditional feedforward neural networks, while powerful for many applications, are inherently limited when it comes to handling time series data due to their lack of temporal

memory. These networks process input data in isolation, treating each time point as independent and not leveraging any historical information from previous inputs. As a result, they are less effective in tasks that require understanding and predicting based on sequential data or patterns over time. To address this limitation, Recurrent Neural Networks (RNNs) were introduced, which are specifically designed to capture temporal dependencies through their recurrent structure. RNNs maintain a form of memory by looping connections that allow information to persist across multiple time steps [31,32]. This architecture enables RNNs to use information from previous time points to influence the prediction of future values, making them more suitable for sequential data tasks. However, despite their advantages, RNNs face challenges related to the gradient vanishing problem during training. This issue arises because gradients, which are used to update the network's weights, can become exceedingly small, effectively halting the learning process and preventing the network from capturing long-term dependencies in the data. To overcome this challenge, Long Short-Term Memory (LSTM) networks were developed as a specialized type of RNN. LSTMs introduce a unique architecture designed to address the gradient vanishing problem through the incorporation of special gate units. These gate units—namely the input gate, forget gate, and output gate—regulate the flow of information through the network. The input gate controls the extent to which new information is added to the memory cell, the forget gate determines which information should be discarded, and the output gate manages how the information in the memory cell is used to influence the network's predictions. The memory cell in an LSTM network functions similarly to a delay operator, providing a mechanism to retain information over extended periods. This allows the network to maintain and utilize relevant information from previous time steps effectively. By leveraging these gates, LSTMs can maintain a long-term context, which is particularly beneficial for tasks that involve complex time series data, such as predicting future values based on historical sensor readings.

This work developed a new strategy for fault-tolerant data fusion in UAV position and attitude estimation. The method is an adaptation of the duplication/comparison approach. A deep learning framework was designed using an LSTM NN for estimating the state using training data obtained from available sensor measurements. For the diagnostic layer, faults are identified and assessed through the generation and evaluation of fault indicators using the moving average (MA) metric. The moving average is a technique used to identify abnormal behavior in a system by analyzing the historical data of a certain signal or measurement. This method involves computing the mean value of a certain amount of data points over a pre-determined time frame and comparing it to the current value of the signal. Any significant deviation from the average value can signify the occurrence of a malfunction in the system. This approach is widely used in industrial settings, such as machinery or process control systems, to detect and diagnose faults in a timely manner [33]. Compared to similar works in the literature, the proposed architecture does not rely on estimating the dynamic model of the UAV, thus eliminating the potential for errors introduced by imperfect model estimates, system uncertainties, or challenging environmental conditions.

The remaining sections of this paper are organized as follows: Section 2 provides an overview of the UAV states estimation and fusion, while Section 3 outlines the design of the fault-tolerant data fusion framework. Section 4 showcases the results of an offline experiment using real data. Lastly, the paper ends with a summary in Section 5.

## 2. Preliminaries

We start by identifying two frames: the body frame ( $\mathcal{B}$ ) and the North-East-Down (NED) navigation cartesian stationary frame ( $\mathcal{N}$ ) with the orientation between them being described by the roll ( $\phi$ ), pitch ( $\theta$ ), and yaw ( $\psi$ ), the three Euler angles, and the rotating Direction Cosine Matrix is represented as:

$$\mathcal{R}_N^B = \begin{bmatrix} \cos \psi \cos \theta & -\sin \psi \cos \phi + \cos \psi \sin \theta \sin \phi & \sin \psi \sin \phi + \cos \psi \sin \theta \cos \phi \\ \sin \psi \cos \theta & \cos \psi \cos \theta + \sin \psi \sin \theta \sin \phi & -\cos \psi \sin \phi + \sin \psi \sin \theta \cos \phi \\ -\sin \theta & \cos \theta \sin \phi & \cos \theta \cos \phi \end{bmatrix} \quad (1)$$

### 2.1. UAV Attitude Equations

The UAV orientation in a three-dimensional space is predicted through the combination of gyroscope, accelerometer, and magnetometer sensors. It is represented by the three Euler angles.

The relation between the angular velocities  $p$ ,  $q$ , and  $r$  as measured by the gyroscope and the time derivatives of the Euler angles can be found using the rotation matrix derived previously, as shown below:

$$\begin{aligned} \dot{\phi} &= p + q \sin \phi \tan \theta + r \cos \phi \tan \theta \\ \dot{\theta} &= q \cos \phi - r \sin \phi \\ \dot{\psi} &= q \sin \phi \sec \theta + r \cos \phi \sec \theta \end{aligned} \quad (2)$$

To achieve an accurate estimate of the attitude, it is important to fuse the measurements from another sensor with those of the gyroscope. The latter, when integrated, may accumulate errors and lead to a drift over time, hindering its accuracy as a single unit attitude estimator. Integrating the Euler rates allows for the derivation of attitude estimates from the gyroscope readings. However, the integration process, which involves cumulative addition, results in the buildup of undesirable components in the readings.

On the other hand, the linear accelerations measured along the three orthogonal axes, denoted as  $a_x$ ,  $a_y$ , and  $a_z$ , can be employed to calculate the pitch and roll angles with the help of the accelerometer. This calculation can be performed using Equations (3) and (4), provided that external disturbances are ignored.

$$\phi_a = \tan^{-1} a_y / a_z \quad (3)$$

$$\theta_a = \tan^{-1} (-a_x / (a_y \sin \phi + a_z \cos \phi)) \quad (4)$$

The magnetometer provides measurements of magnetic field strengths  $m_x$ ,  $m_y$ , and  $m_z$ , which can be used to calculate the heading or yaw angle using the following equation:

$$\psi_m = \tan^{-1} (-a_x / m_x \cos \phi + m_y \sin \phi \sin \theta + m_z \cos \phi \sin \theta) \quad (5)$$

The frequency characteristics of the accelerometer, gyroscope, and magnetometer are complementary, and relying solely on the gyroscope for orientation estimation is not effective. To achieve precise orientation estimates, it is necessary to combine these sensors.

### 2.2. UAV Altitude and Position Equations

The accelerometer measures the accelerations relative to the stationary gravity vector, and are given by

$$\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \frac{d}{dt} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} + \mathcal{R}_N^B \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} \quad (6)$$

$\frac{d}{dt} [\dot{x} \ \dot{y} \ \dot{z}]^T$  is the velocity vector's rate of variation as seen from the stationary reference frame, and is defined as follows:

$$\frac{d}{dt} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \frac{d}{dt} \begin{bmatrix} \dot{x}_b \\ \dot{y}_b \\ \dot{z}_b \end{bmatrix} + \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} \dot{x}_b \\ \dot{y}_b \\ \dot{z}_b \end{bmatrix} \quad (7)$$

where  $\dot{x}_b$ ,  $\dot{y}_b$  and  $\dot{z}_b$  are the velocity components along the body axes.

Combining the two equations above results in the following velocity states dynamics:

$$\frac{d}{dt} \begin{bmatrix} \dot{x}_b \\ \dot{y}_b \\ \dot{z}_b \end{bmatrix} = \begin{bmatrix} r\dot{y}_b - q\dot{z}_b + a_x - g \sin \theta \\ p\dot{z}_b - r\dot{x}_b + a_y + g \sin \phi \cos \theta \\ q\dot{x}_b - p\dot{y}_b + a_z + g \cos \phi \cos \theta \end{bmatrix} \quad (8)$$

The position of a UAV can be estimated by combining data from both a Global Positioning System (GPS) receiver and an accelerometer. The GPS offers global location information, while the accelerometer measures linear acceleration in the UAV’s body frame. Through combining these two measurements, the accuracy of position estimation is increased. The accelerometer provides additional velocity and orientation information, and this combined approach is also useful in situations where the GPS signal is weak or unavailable.

### 3. Fault-Tolerance Architecture

A fault-tolerant data fusion scheme that incorporates a duplication-comparison method for detection and recovery purposes is proposed in this section. The design includes two parallel and separate branches, each implementing a data fusion block (DF1 and DF2) utilizing redundant or diversified sensors blocks ((S1, S2) and (S3, S4)) for state estimation. S1 and S3 should perform similarly, whether they are redundant or diversified sensors. The same applies to S2 and S4. An implementation of this architecture is illustrated in Figure 1. S1 (and S3) consists of an Inertial Measurement Unit (IMU) that combines a three-axis accelerometer and three-axis gyroscope. S2 (and S4) consists of a GPS module for determining the UAV’s absolute position and a barometer for measuring its altitude.

This architecture offers the capability to tolerate or detect hardware faults under the assumption that only one fault is present at a time in the system. To accommodate for multiple faults, the level of hardware redundancy should be increased. The principle of this architecture involves evaluating the outputs from two separate data fusion blocks and determining if there is any significant discrepancy between them. This difference serves as an indication of the presence of an error in the system. To diagnose the source of the error, the outputs of the sensor and the residual from the fusion are analyzed using the moving average technique. The detailed architecture applied on the UAV is shown in Figure 2.

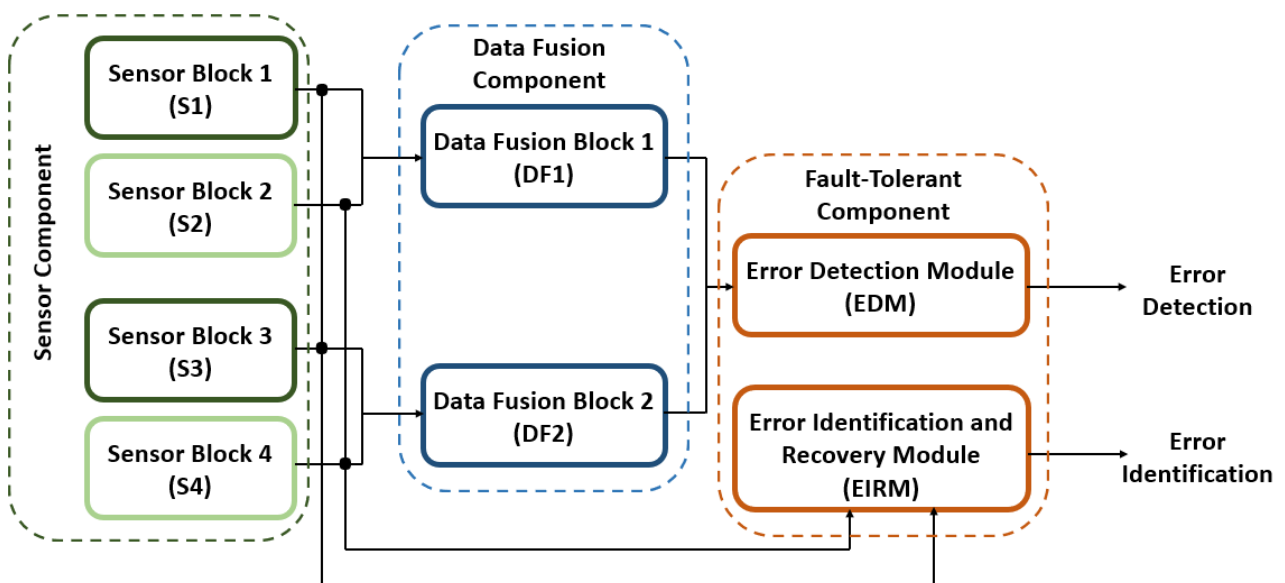


Figure 1. Fault-tolerant data fusion scheme using duplication-comparison.

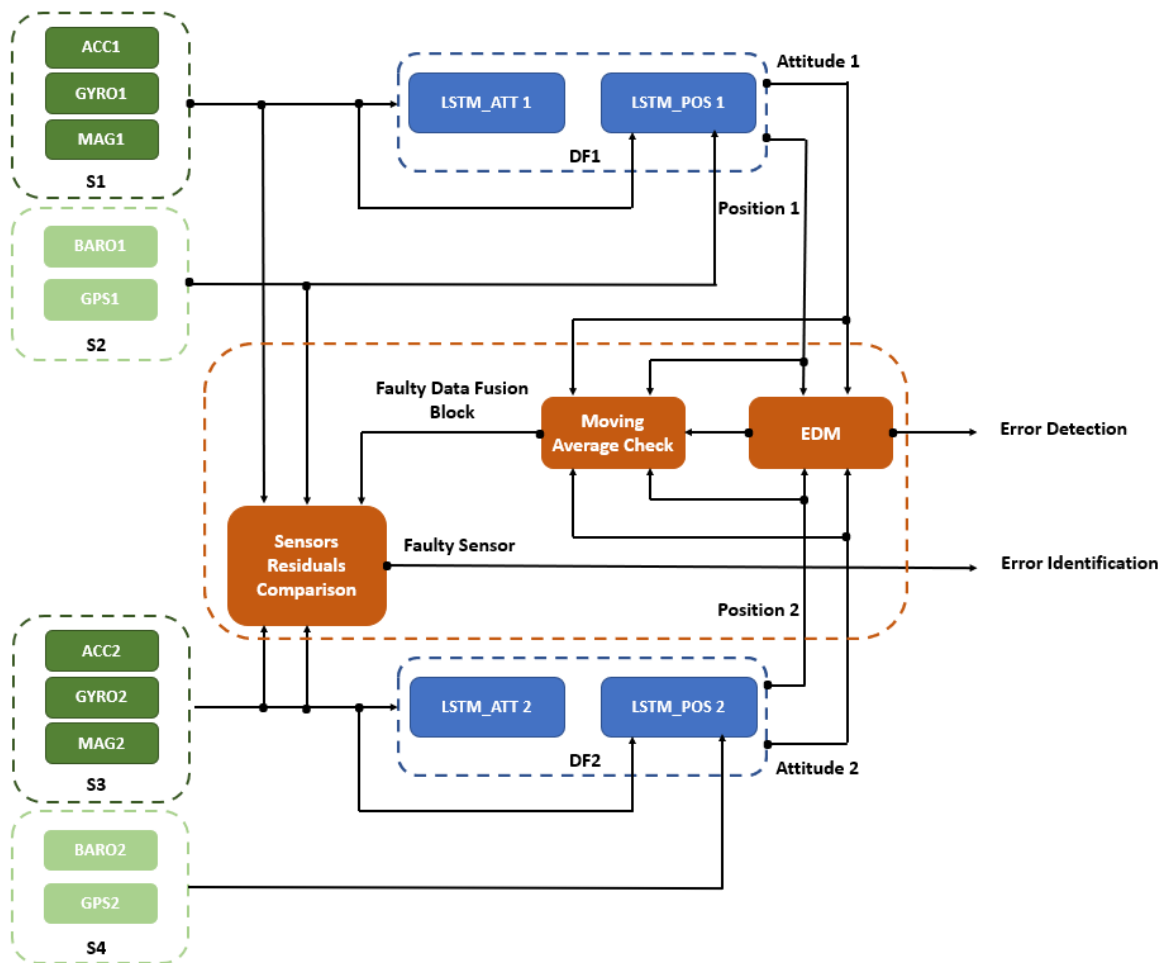


Figure 2. Implementation of fault-tolerance architecture on UAV system.

### 3.1. Data Fusion Component

An LSTM-based fusion technique is applied on data from magnetometer, accelerometer, and gyroscope sensors for the attitude prediction of a hexarotor UAV, and on accelerometer, barometer, and GPS sensors for its position and altitude prediction.

Starting with the attitude prediction, the output from three tri-axial inertial sensors are fed into a Long Short-Term Memory (LSTM) network. The nine outputs from these sensors are integrated into an input array, as depicted in Figure 3. The input to the LSTM layer includes both current and previous step measurements, resulting in a time step of 2. The model comprises two LSTM hidden layers, and a linear activation function is used for the prediction. Using a supervised learning approach, the LSTM extracts features from the inputs, consisting of Euler angles, angular speeds, and accelerations, and generates the required output. The network outputs are the estimated angles based on the trained model and sensor measurements during the prediction phase.

Yet, the position network shown in Figure 4 differs from the previous network only in the input and output layers, where seven measurements (tri-axial accelerometer, barometer altitude, GPS longitude, latitude, and altitude) constitute the input, and the UAV estimated position (longitude, latitude, and altitude) constitutes the output (prediction) layer.

A linear activation function is employed since this is a regression problem. In regression, the goal is to obtain the real output of the network, whereas in classification, the output is categorized into a specified range using an activation function.



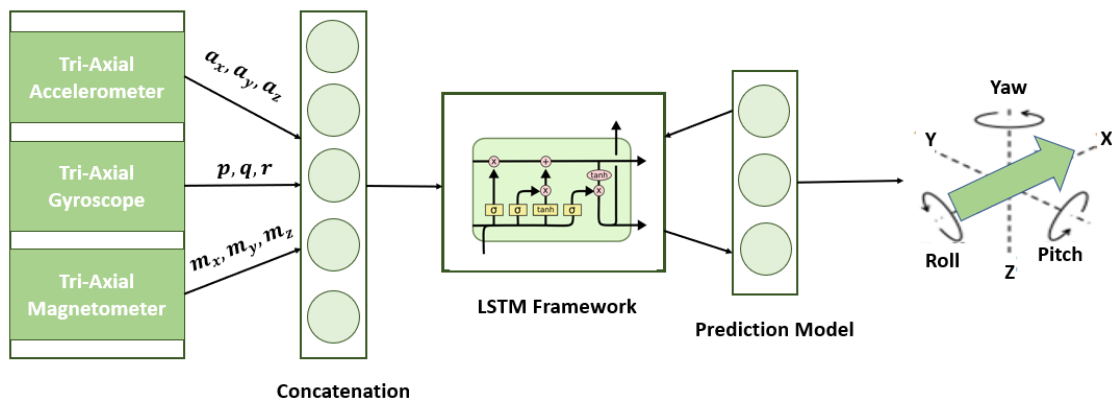


Figure 3. LSTM neural network-based data fusion for attitude estimation.

Between the two hidden layers of the LSTM architecture, a dropout layer is added. The purpose of dropout is to prevent overfitting by randomly disconnecting some of the inputs to the next layer during training, with a probability  $p$ . This makes the network architecture more diverse and eliminates the dependence of any single node on a given pattern, which leads to more robust and generalized models. The dropout technique aims to improve the testing accuracy while potentially compromising the training accuracy, and serves as a form of regularization.

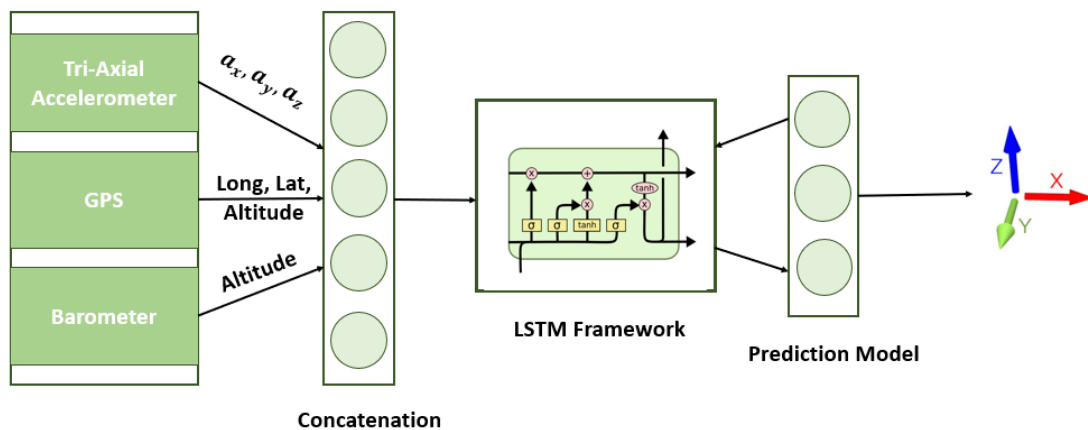


Figure 4. LSTM neural network-based data fusion for position and altitude estimation.

### 3.2. Fault-Tolerant Component

In this section, we discuss the process of detecting faults in the sensors and how the system recovers from the faults.

#### 3.2.1. Error Detection Module

As shown in Figure 1, the fault-tolerance architecture component starts with the error detection module. A comparison between the states (attitude and position) estimated by the two data fusion blocks is performed in this module using residual generation. A residual  $r$  is the difference between the estimated modeled output  $\hat{y}$  and the actual sensor output  $y$ :

$$r = y - \hat{y} \tag{9}$$

Two sets of residuals are to be computed,  $r_{ATT} = [r_{Roll} \ r_{Pitch} \ r_{Yaw}]^T$  and  $r_{POS} = [r_{Lat} \ r_{Long} \ r_{Alt}]^T$ . When at least one set of the residuals exceeds its specified threshold,  $Th_{Det(ATT)} = [Th_{Det(Roll)} \ Th_{Det(Pitch)} \ Th_{Det(Yaw)}]^T$  and  $Th_{Det(POS)} = [Th_{Det(Lat)} \ Th_{Det(Long)} \ Th_{Det(Alt)}]^T$ , an error is detected in the system. Then, the error identification

and recovery module is checked to diagnose the fault and the faulty sensor. Thresholds are fixed empirically using the trial-and-error technique.

### 3.2.2. Error Identification and Recovery Module

When a fault is detected through residual generation at time step  $n$ , the moving average algorithm is checked on all fusion block outputs to identify the faulty block. The moving average algorithm has been widely used in the literature for fault detection. For example, in [34], it was used for DC series arc detection in photovoltaic (PV) systems. It consists of the continuous calculation of the averages of a specified number  $m$  of data samples:

$$MAV = \frac{1}{m} \sum_{i=1}^m d(i) \quad (10)$$

where  $MAV$  represents the moving average to be calculated,  $m$  represents the window size over which the average is to be computed, and  $d(i)$  represents the data at the  $i$ -th time step. Two sets of moving averages are to be computed,  $MAV_{ATT(i)} = [MAV_{Roll(i)} \quad MAV_{Pitch(i)}]^T$  and  $MAV_{POS(i)} = [MAV_{Lat(i)} \quad MAV_{Long(i)} \quad MAV_{Alt(i)}]^T$ , with  $i = 1, 2$  corresponding to the first and second fusion blocks. They correspond to the attitude and position outputs of the fusion blocks.

At time step  $n$ , where an error is detected, the thresholds of the moving averages corresponding to the non-zero residual components are determined by the moving averages of the two blocks at that time step, i.e.,:

$$\begin{aligned} Th_{MAV_{ATT(i)}} &= MAV_{ATT(i)}[n] \\ Th_{MAV_{POS(i)}} &= MAV_{POS(i)}[n] \end{aligned} \quad (11)$$

with  $i = 1, 2$  corresponding to the first and second fusion blocks. From this threshold, we can fix a region  $[Th_{MAV_{ATT(i)}} - \epsilon, Th_{MAV_{ATT(i)}} + \epsilon]$  where  $\epsilon$  is considered a tolerance on this threshold and is set empirically by trial and error. The same is applied for  $Th_{MAV_{POS(i)}}$ . After a fault is detected, if the moving average of the output of a sensor block lies outside this region, we can isolate the fault in this block. However, if neither of the two blocks shows out-range values, we consider it a false alarm.

After identifying the faulty sensor block, residuals are generated between each of the two similar sensors. When a residual is non-zero, the sensor belonging to the faulty block is identified to be the faulty one. We first examine the measurements from block S1 with those of the block S3 by evaluating the distances  $D_{(S1,S3)a}$ ,  $D_{(S1,S3)b}$ , and  $D_{(S1,S3)c}$  with respect to thresholds  $Th_{13a}$ ,  $Th_{13b}$ , and  $Th_{13c}$ . These distances are defined as

$$\begin{aligned} D_{(S1,S3)a} &= \sqrt{(a_{x1} - a_{x2})^2 + (a_{y1} - a_{y2})^2 + (a_{z1} - a_{z2})^2} \\ D_{(S1,S3)b} &= \sqrt{(p_1 - p_2)^2 + (q_1 - q_2)^2 + (r_1 - r_2)^2} \\ D_{(S1,S3)c} &= \sqrt{(m_{x1} - m_{x2})^2 + (m_{y1} - m_{y2})^2 + (m_{z1} - m_{z2})^2} \end{aligned} \quad (12)$$

with  $a_{xi}$ ,  $a_{yi}$ ,  $a_{zi}$ ,  $p_i$ ,  $q_i$ ,  $r_i$ ,  $m_{xi}$ ,  $m_{yi}$ , and  $m_{zi}$  being the outputs of the Inertial Measurement Unit  $IMU_i$ . Next, we evaluate the output from sensor block S2 against that from sensor block S4 by comparing the distances  $D_{(S2,S4)a}$  and  $D_{(S2,S4)b}$  with the thresholds  $Th_{24a}$  and  $Th_{24b}$ .  $D_{(S2,S4)a}$  and  $D_{(S2,S4)b}$  are as follows:

$$\begin{aligned} D_{(S2,S4)a} &= \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \\ D_{(S2,S4)b} &= |z_1 - z_2| \end{aligned} \quad (13)$$

The  $x_i$  and  $y_i$  positions are obtained from the  $GPS_i$ , while the altitude  $z_i$  is measured by the  $Barometer_i$ . The steps of the diagnosis and fault-tolerant algorithm are outlined in Algorithm 1.



**Algorithm 1: Fault-Tolerant Algorithm**


---

```

Data:  $Th_{Det(ATT)}$ ,  $Th_{Det(POS)}$ ,  $Th_{13a}$ ,  $Th_{13b}$ ,  $Th_{13c}$ ,  $Th_{24a}$ ,  $Th_{24b}$ 
begin
  if ( $r_{(ATT)} > Th_{Det(ATT)}$ ) || ( $r_{(POS)} > Th_{Det(POS)}$ ) then
    /* A fault is detected at sample  $n$  */;
    /* Compute  $Th_{MAV_{ATT(1)}}$ ,  $Th_{MAV_{ATT(2)}}$ ,  $Th_{MAV_{POS(1)}}$ ,  $Th_{MAV_{POS(2)}}$  */ if
      ( $MAV_{ATT(1)} > Th_{MAV_{ATT(1)}}$ ) || ( $MAV_{POS(1)} > Th_{MAV_{POS(1)}}$ ) then
        /* The fault is in the first branch */;
        if ( $D_{(S1,S3)a} > Th_{13a}$ ) || ( $D_{(S1,S3)b} > Th_{13b}$ ) || ( $D_{(S1,S3)c} > Th_{13c}$ ) then
          | S1 is faulty: isolate DF1;
        else
          if ( $D_{(S2,S4)a} > Th_{24a}$ ) || ( $D_{(S2,S4)b} > Th_{24b}$ ) then
            | S2 is faulty: isolate DF1;
          else
            /* cannot identify the faulty sensor, keep running without
            modifications */
          else
            if ( $MAV_{ATT(2)} > Th_{MAV_{ATT(2)}}$ ) || ( $MAV_{POS(2)} > Th_{MAV_{POS(2)}}$ ) then
              | /* The fault is in the second branch */;
            else
              /* cannot identify the faulty branch: keep running without
              modifications */;
              if ( $D_{(S1,S3)a} > Th_{13a}$ ) || ( $D_{(S1,S3)b} > Th_{13b}$ ) || ( $D_{(S1,S3)c} > Th_{13c}$ ) then
                | S3 is faulty: isolate DF2;
              else
                if ( $D_{(S2,S4)a} > Th_{24a}$ ) || ( $D_{(S2,S4)b} > Th_{24b}$ ) then
                  | S4 is faulty, Isolate DF2;
                else
                  /* cannot identify the faulty sensor: keep running without
                  modifications */
                else
                  /* No fault is detected */
                end
            end
          end
        end
      end
    end
  end

```

---

**4. Experimental Results and Analysis**

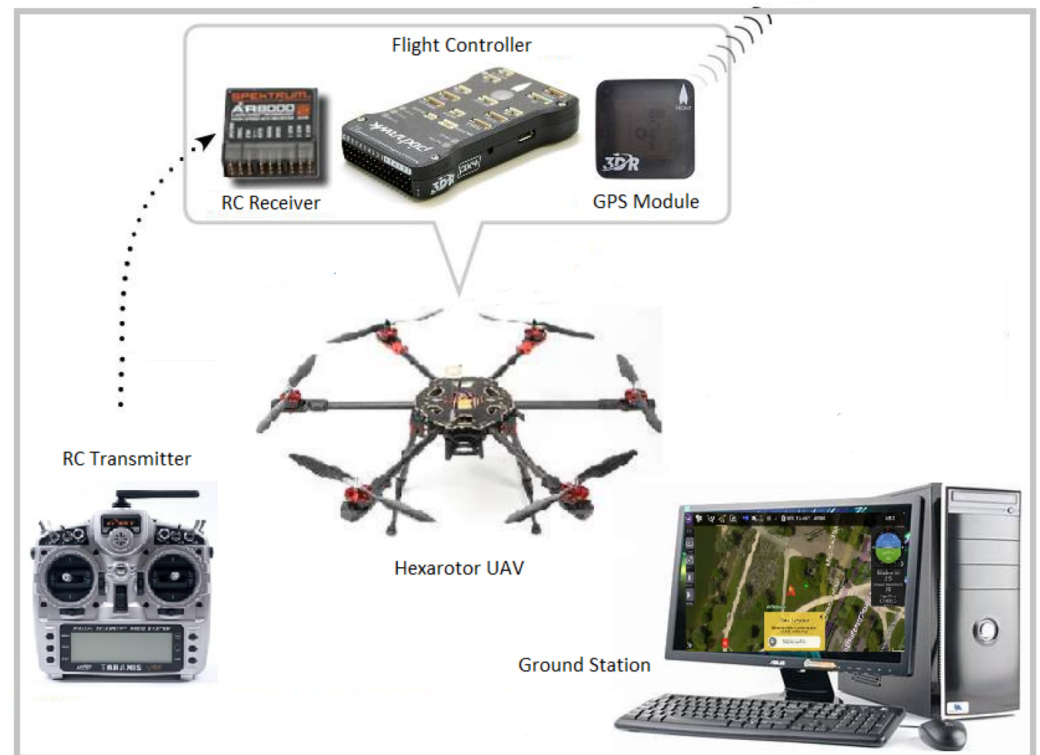
To validate the performance of the outlined approach, experiments with fault injection were conducted on a hexarotor UAV, and the results are analyzed in depth.

**4.1. System Architecture and Setup**

The effectiveness of the fault-tolerant architecture is demonstrated through offline experimental validation using real data and fault injection. An outdoor test environment was created for data acquisition (Figure 5). The experimental UAV is a Tarot hexarotor equipped with a pixhawk flight controller and various sensors such as an InvenSense ICM-20689 tri-axial Inertial Measurement Unit (IMU1) (InvenSense TDK Group, Shanghai, China), a Bosch BMI055 Inertial Measurement Unit (IMU2) (Bosch, Shanghai, China), two Ublox NEO-M8N GPS (u-blox, Shanghai, China) modules, and two MS5611-01BA03 (TE Connectivity, Shanghai, China) barometers.

The hexarotor maintains communication with both the ground station and the RC (Remote Control) transmitter. This dual communication setup serves different purposes to enhance the control and monitoring of the hexarotor during its flight operations. The ground station allows for more extensive and sophisticated control, enabling the execution

of complex flight paths, autonomous missions, and data transmission for analysis. On the other hand, the RC transmitter provides a direct and immediate control link, allowing for quick responses and manual intervention in case of emergencies or unforeseen situations. This redundant communication ensures a reliable and flexible connection with the hexarotor, enhancing its safety and versatility in various flight scenarios.



**Figure 5.** Overview of the test environment.

#### 4.2. Dataset Description

To evaluate the practicality and effectiveness of the proposed framework, two distinct fault conditions were developed. As a first fault scenario, an additive fault was simulated on the output of the magnetometer of the first fusion block (Mag1). This type of fault occurs when the magnetometer is not calibrated for hard iron and soft iron biases. Hard iron biases are a permanent offset in the magnetic field at the current location, while soft iron biases are non-permanent biases caused by nearby electronic and magnetic fields. To minimize this issue, it is important to calibrate the magnetometer beforehand.

The second fault was a freeze fault simulated on the Gyroscope of the second fusion block. This fault refers to a malfunction or failure in the gyroscope sensor that causes it to become frozen or unresponsive.

When designing experimental data collection environments, faults representative of those that could be encountered in a real UAV setting were simulated.

Different datasets were collected using the experimental configurations described earlier. They were composed of accelerometer, gyroscope, magnetometer, GPS, and barometer sensor measurements. Data were split for training and testing by the following, known splitting rule:  $(0.7 - 0.8) \times$  data for training and  $(0.2 - 0.3) \times$  data for testing. Table 1 below shows a description of the attitude and position real-data characteristics.

**Table 1.** Attitude and position real-data description.

Attitude		Position	
-	Size: 13,166	-	Size: 13,166
-	Standard Deviation:	-	Standard Deviation:
Data	■ Roll: 0.041880 rad = 2.4°	■	Lat: 0.000058 m
	■ Pitch: 0.02582 rad = 1.5°	■	Long: 0.000086 m
	■ Yaw: 0.108069 rad = 6.2°	■	Alt: 1.101917 m

#### 4.3. Data Fusion Performance

Before feeding the data into the LSTM model, several preprocessing steps were undertaken to ensure their suitability for training. The raw sensor data collected from the UAV were first cleaned to address any missing values and reduce noise through filtering techniques. The data were then normalized using Min-Max scaling to bring all features into a [0, 1] range, which is essential for the effective training of the LSTM model. Subsequently, the data were reshaped into sequences with a number of time steps of 2. The LSTM network used for attitude data fusion was configured with the following parameters:

- Input layer of shape (2,9) with 2 being the number of time steps and 9 the number of input features.
- LSTM layer of shape (2,128) with 128 being the number of units.
- Batch normalization layer of shape (2,128). Batch normalization is applied to each feature, which helps in stabilizing and accelerating training.
- Dropout Layer of shape (2,128). It applies dropout with a rate of 0.25 to the LSTM outputs. This helps in regularization by randomly dropping 25% of the units during training.
- LSTM layer of 128 units. This layer processes the output from the previous layer and reduces the sequence dimension, outputting a single vector of 128 features for each sample.
- Dense layer that produces three outputs. The activation function here is linear, suitable for regression or multi-class classification depending on the loss function used.
- Batch normalization layer that normalizes the outputs of the dense layer. Batch normalization is applied to the final layer outputs.

The LSTM network used for position and altitude estimation exhibits the same characteristics but with seven input features.

The root mean squared error (RMSE) was considered a metric for evaluating the deep learning-based data fusion. For the attitude estimation, the following errors were computed:

$$\begin{aligned}
 RMSE_{Roll} &= \sqrt{\frac{1}{n} \sum_{i=1}^n (\phi_i - \hat{\phi}_i)^2} \\
 RMSE_{Pitch} &= \sqrt{\frac{1}{n} \sum_{i=1}^n (\theta_i - \hat{\theta}_i)^2} \\
 RMSE_{Yaw} &= \sqrt{\frac{1}{n} \sum_{i=1}^n (\psi_i - \hat{\psi}_i)^2}
 \end{aligned} \tag{14}$$

For the position estimation, the following errors on latitude, longitude, and altitude were computed:

$$\begin{aligned}
 RMSE_{Lat} &= \sqrt{\frac{1}{n} \sum_{i=1}^n (Lat_i - \hat{Lat}_i)^2} \\
 RMSE_{Long} &= \sqrt{\frac{1}{n} \sum_{i=1}^n (Long_i - \hat{Long}_i)^2} \\
 RMSE_{Alt} &= \sqrt{\frac{1}{n} \sum_{i=1}^n (Alt_i - \hat{Alt}_i)^2}
 \end{aligned} \tag{15}$$

The results of attitude data fusion in the fault-free case are shown in Figure 6.

To evaluate the performance of the Long Short-Term Memory (LSTM) network for data fusion in comparison to the Extended Kalman Filter (EKF), we analyzed two distinct attitude datasets characterized by low and high dynamic motion considered in [35]. The results are illustrated in Figures 7 and 8 and Table 2. The analysis reveals that the LSTM

outperforms the EKF, particularly in scenarios involving high dynamic motion. This is evident from the superior accuracy of the LSTM in tracking and predicting attitude changes under high-dynamic motion scenarios. During high dynamic motion, LSTMs can leverage their ability to adapt and learn from varying conditions to provide more robust performance. They can handle abrupt changes and complex patterns in the data more effectively than traditional models. However, in high dynamic scenarios, the EKF's performance can degrade if the system dynamics are highly nonlinear or if the process and measurement noise assumptions are not accurate. The linearization steps in EKF can lead to significant errors under such conditions. In addition, LSTMs have the ability to remember long-term dependencies thanks to their gating mechanisms. This is crucial in data fusion where past states influence future states over long periods, allowing LSTMs to better capture and utilize historical information. The performance of the LSTM network was also compared to that of the Gated Recurrent Units (GRU) network when applied to data fusion. The results show that the LSTM outperforms also the GRU. In fact, the LSTMs' architecture including separate memory cells and additional gates can make the LSTM network more effective for the data fusion task as it involves complex long-term dependencies.

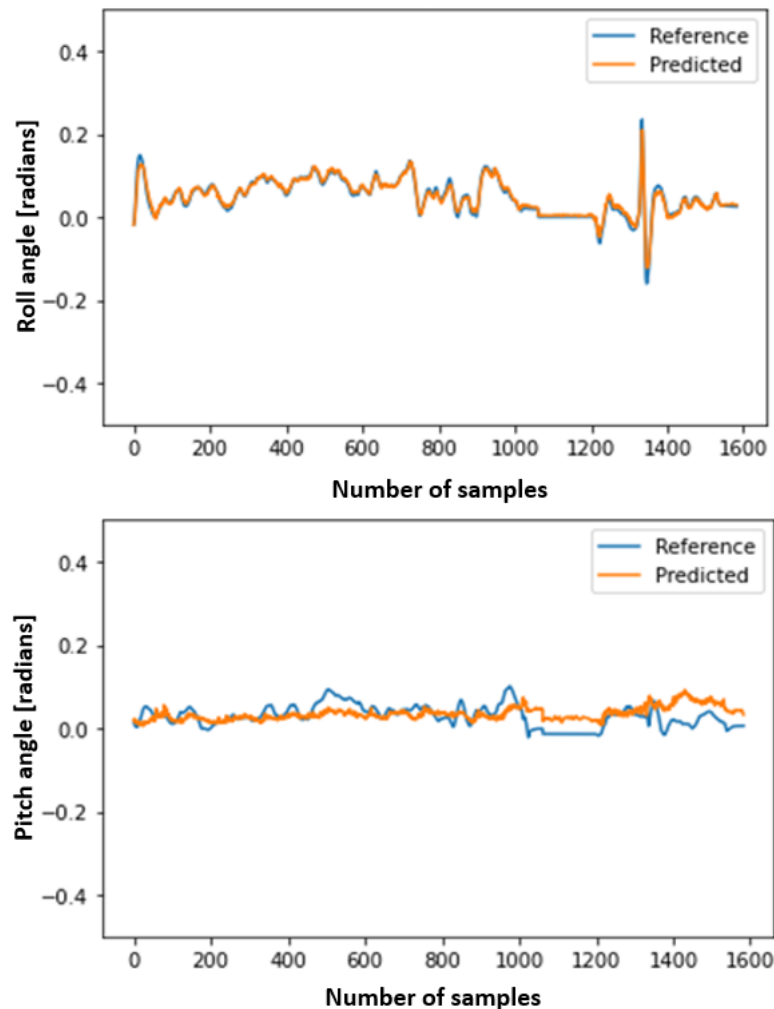


Figure 6. Cont.

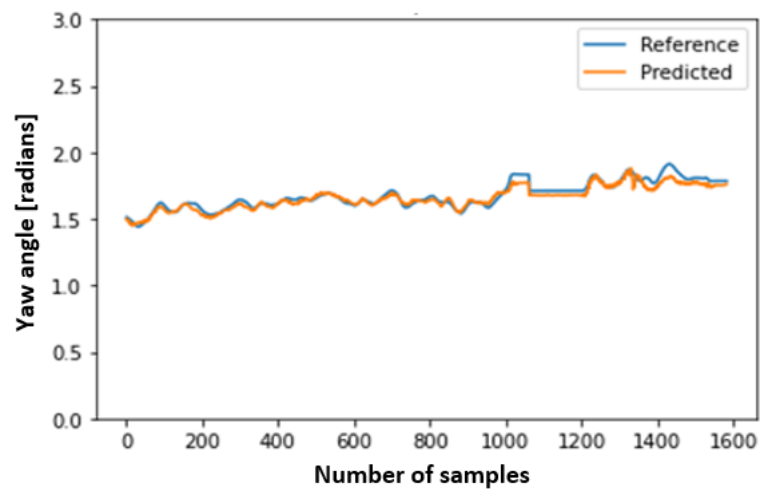


Figure 6. Data fusion results on fault-free case: attitude states.

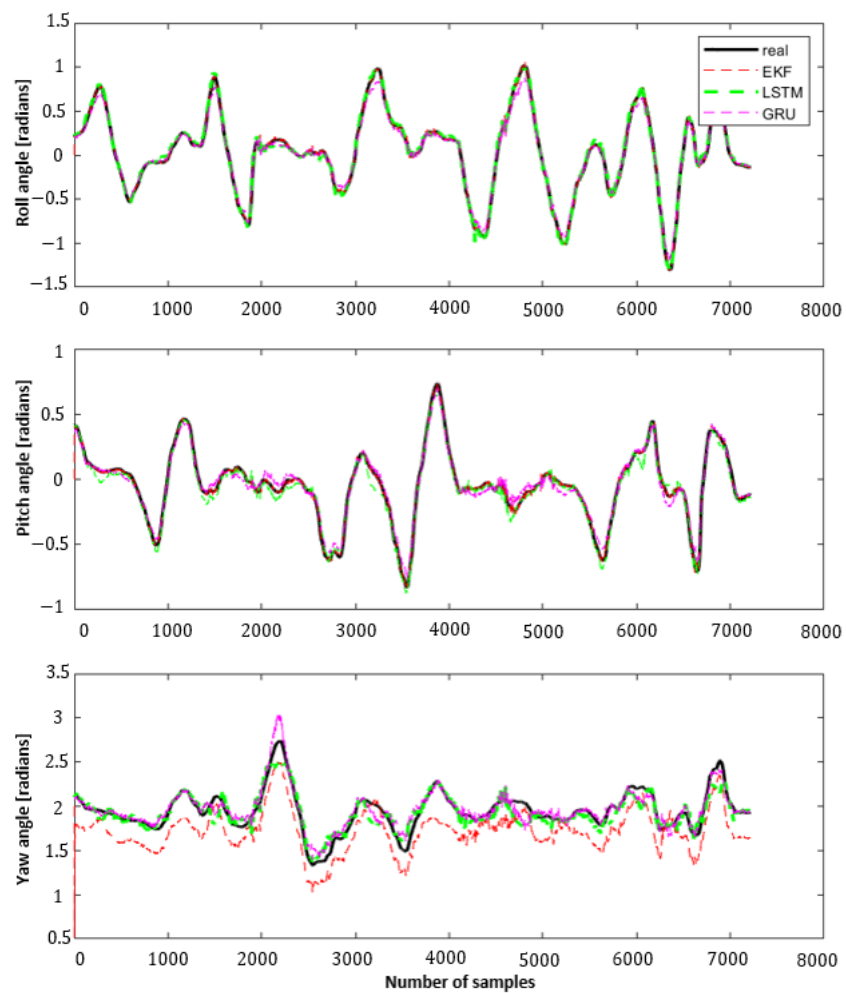


Figure 7. Data fusion results on fault-free case during low dynamic motion: attitude states.

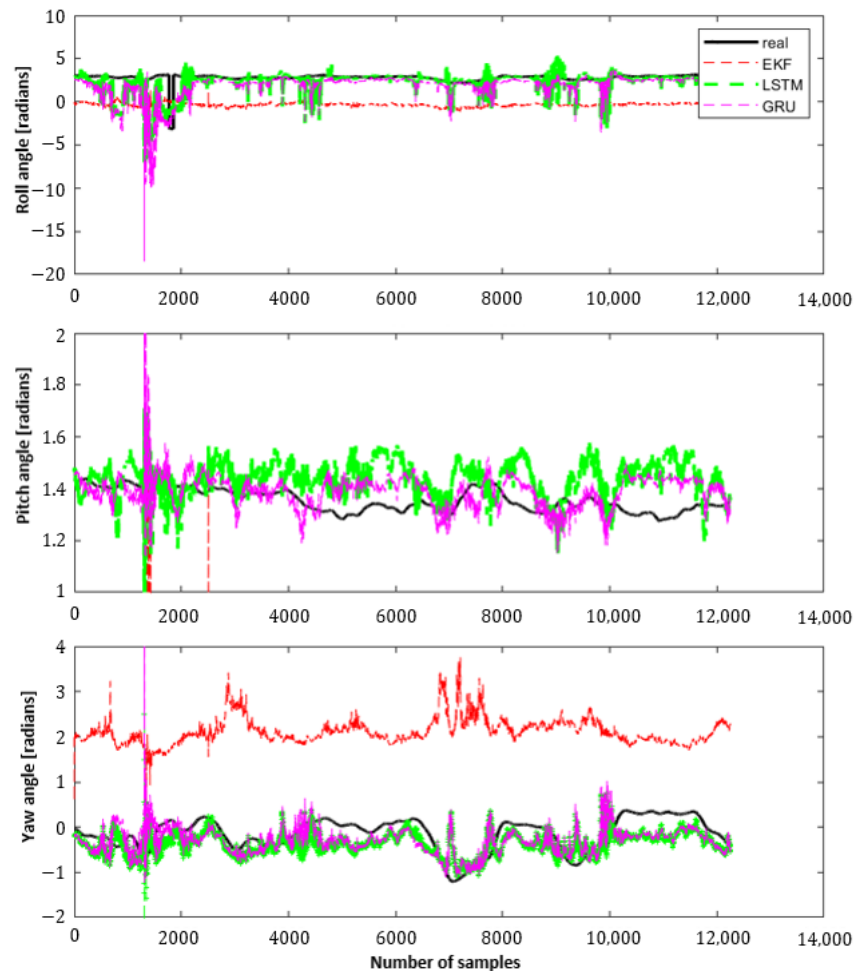
**Table 2.** Data fusion performance comparison of different techniques using root mean squared error (RMSE) metric.

		Roll RMSE	Pitch RMSE	Yaw RMSE
Low dynamic motion	EKF	0.0210	0.0153	0.2435
	LSTM	0.037	0.044	0.075
	GRU	0.055	0.043	0.079
High dynamic motion	EKF	3.14	2.70	2.34
	LSTM	1.39	0.13	0.37
	GRU	1.86	0.14	0.42

4.4. Fault-Tolerance Performance

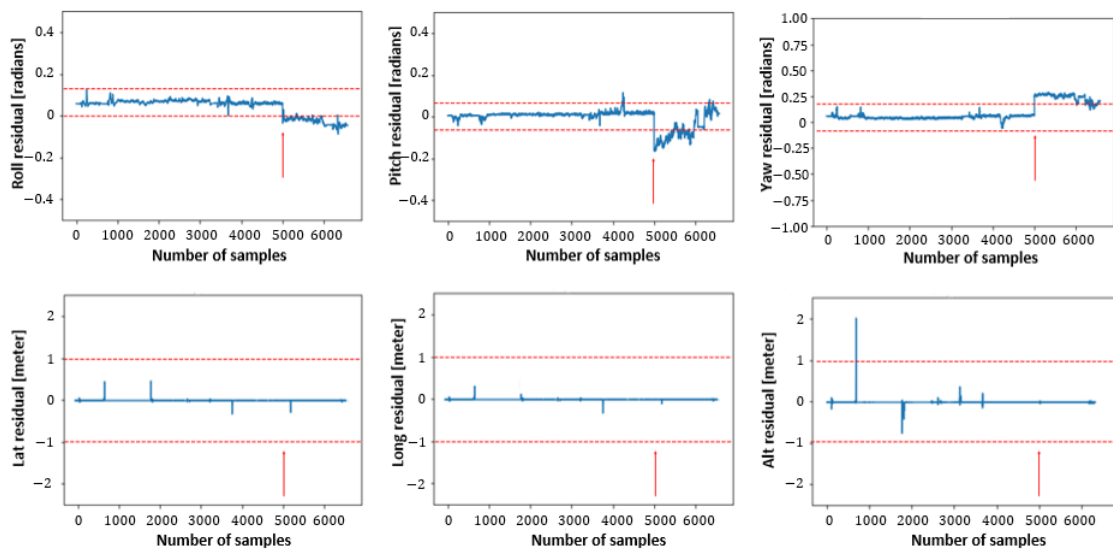
We then examined the effects of two fault injections: an additive fault on the first magnetometer sensor and a freeze fault on the second gyroscope.

At sample 5000, which corresponds to time  $t = 3.34$  s, an additive fault of value +20 is added on the magnetometer’s three-axis measurements. The attitude and position residuals between the two fusion blocks are visualized on Figure 9, noting that the fault injection time is represented by the red arrow and the thresholds are represented by the horizontal red lines. This figure shows that only the attitude residuals exceed the predefined thresholds, which indicates that a fault is detected in the attitude network.



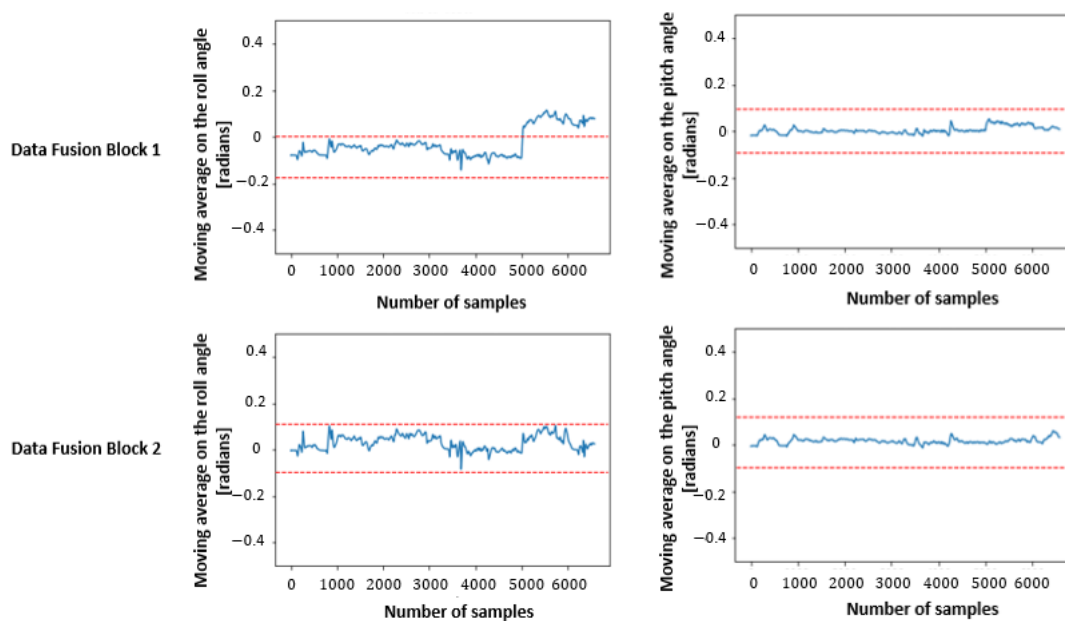
**Figure 8.** Data fusion results on fault-free case during high dynamic motion: attitude states.





**Figure 9.** Residuals after an additive fault on Mag1 (**upper** figures for Block 1, **lower** figures for Block 2). The vertical red arrows indicate the fault injection time. The horizontal red lines represent the thresholds.

Figure 10 shows the moving averages of the two data fusion blocks implemented with a window size  $m = 1000$ . The thresholds are calculated using Equation (11) with  $\epsilon = 0.1$  to check which of the two blocks shows out-region samples. It is clear that the first branch is the faulty one.



**Figure 10.** Attitude moving averages after an additive fault on Mag1. The horizontal red lines represent the thresholds.

A similar reasoning for the residuals shown in Figure 11 enabled us to determine that the magnetometer in the first branch is the faulty sensor.

A second fault was simulated on the gyroscope measurements of the S4 sensor block. The freeze starts at sample 5000, corresponding to time  $t = 3.34$  s. The difference between the outputs of the data fusion blocks after the occurrence of this fault is shown in Figure 12. A procedure similar to that above is then followed for the identification of the faulty sensor.

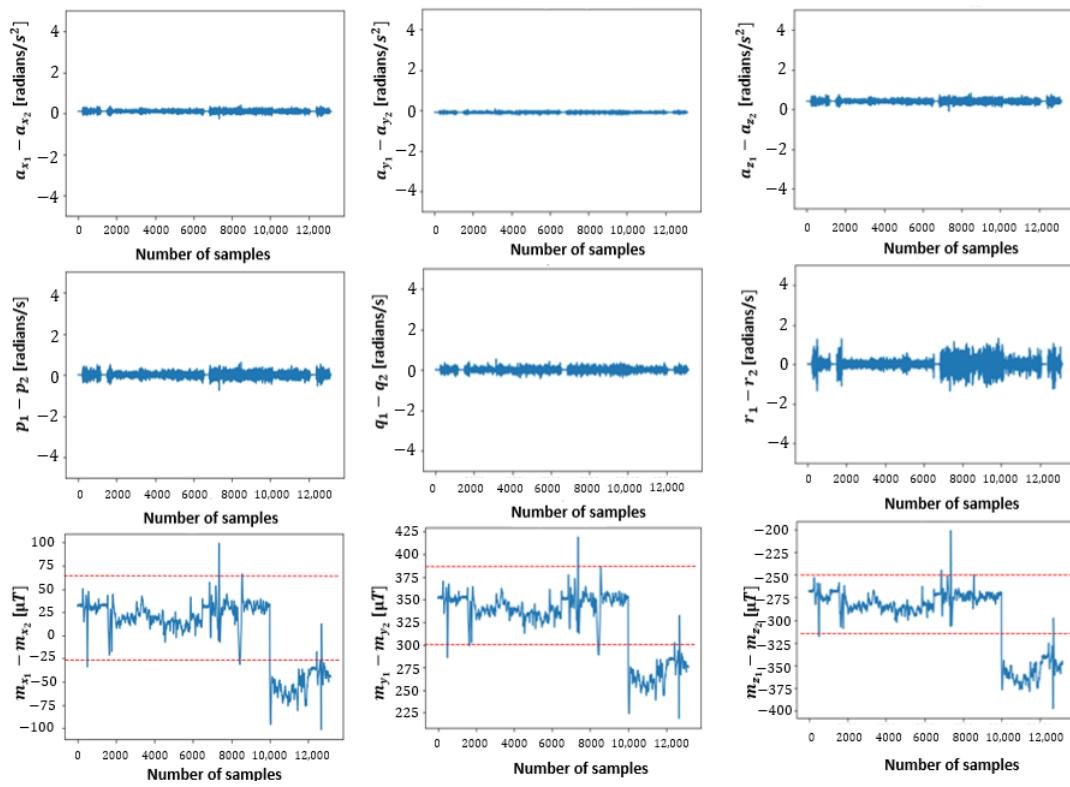


Figure 11. The residual tests after Mag1 fault. The horizontal red lines represent the thresholds.

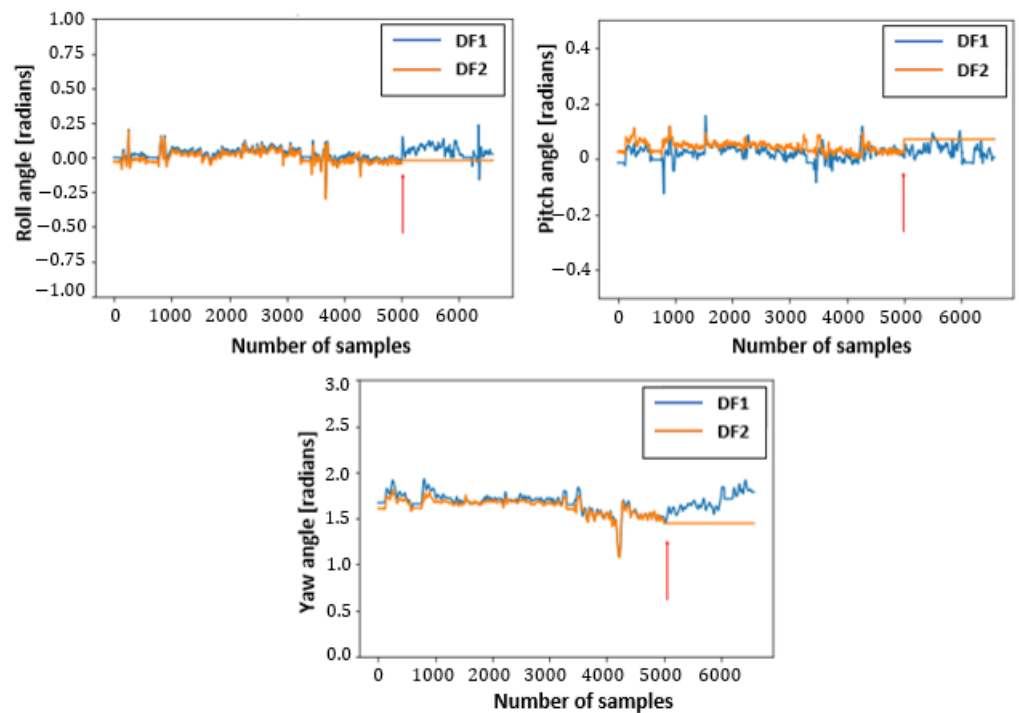


Figure 12. The outputs of the fusion blocks after Gyro2 fault. The vertical red arrows indicate the fault injection time.

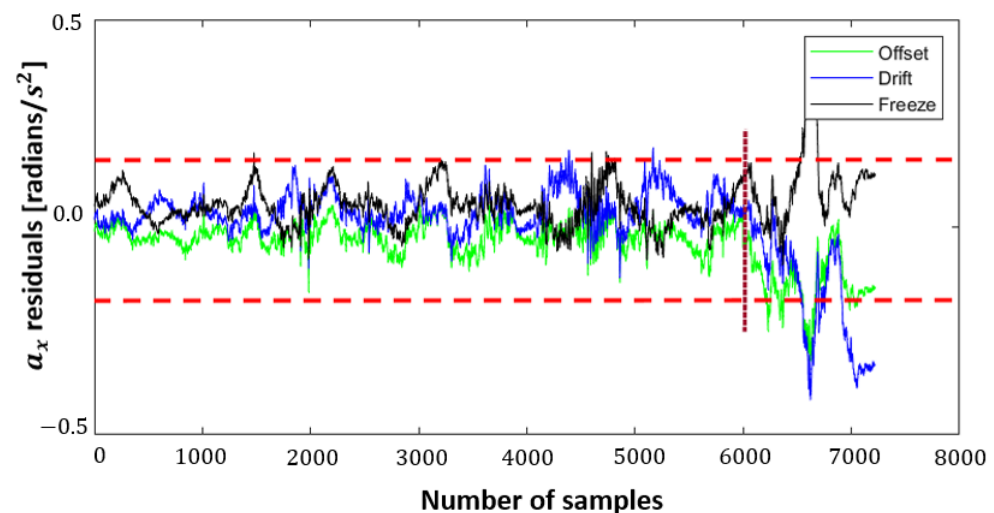
Table 3 shows the RMSEs of the different states obtained for the LSTM fusion.

**Table 3.** Attitude and position estimation results using LSTM fusion.

	Attitude		Position
	- Loss: 0.000416		- Loss: 0.00606
	- RMSE:		- RMSE:
Validation	■ Roll: 0.00751 rad = 0.4°		■ Lat: 0.0077 m
	■ Pitch = 0.0287 rad = 1.6°		■ Long = 0.005 m
	■ Yaw = 0.0326 rad = 1.86°		■ Alt = 0.94 m

Sensor faults can be categorized into various types, including offset, drift, and freeze. Each type represents a different way that a sensor's performance can be impaired, leading to different effects on residuals, as shown in Figure 13. Offset, drift, and freeze faults were simulated on the accelerometer measurement. Offset faults are easy to detect in a shorter detection time because they produce a consistent error pattern. However, drift and freeze faults require longer detection times due to their gradual nature. The LSTM network proves highly effective in detecting these various types of sensor faults due to its ability to model complex temporal dependencies and capture intricate patterns within time-series data. The LSTM's architecture is particularly well suited for identifying offset faults, as it can learn to recognize consistent deviations from expected patterns and flag persistent shifts in sensor readings. For drift faults, the LSTM is well suited in detecting gradual, progressive changes over time, leveraging its memory capabilities to identify trends and subtle variations that deviate from historical norms. By maintaining long-term dependencies, the LSTM can discern slowly evolving patterns indicative of drift. In the case of freeze faults, the LSTM's capacity to track temporal changes allows it to spot discrepancies when the sensor output fails to vary despite changes in the true values. This enables the network to identify periods where the residuals diverge significantly from expected behavior due to the sensor's inability to adapt to new data.

Thresholds in this study were fixed empirically; they were determined based on observed data to balance the trade-offs between detecting faults and avoiding false alarms.

**Figure 13.** Effects of sensor fault types on residuals. The horizontal red lines represent the thresholds.

#### 4.5. Discussion

This architecture can easily be modified to isolate software faults, as previously discussed in [6]. These faults are typically caused by human error during the development stage. However, the detection of software faults is beyond the scope of this study.

The architecture proposed in this paper offers several benefits when compared to other similar approaches for unmanned aerial vehicles. One of the key advantages is that it does not rely on a dynamic model for sensor fault isolation like [27], which requires tuning the

model parameters following each failure to prevent drift in model accuracy. In addition, compared to the architecture in [24], the proposed architecture requires fewer redundant sensors and data fusion blocks. Traditional fault-tolerant methods, such as weighted-mean or majority-voting, which use a minimum of three redundant or diversified sensors as in [24], remain widely implemented in aviation systems. Yet, these methods are costly and could lead to a substantial weight increase in the system. Finally, unlike the other architectures based on Kalman Filters [28], this architecture uses a deep learning framework for data fusion. It was proven in [36] that the LSTM fusion method's estimated states do not show cumulative divergence error, unlike those obtained from a standard Kalman Filter. This comparison is summarized in Table 4.

**Table 4.** Comparison of the proposed architecture with similar works in the literature.

Technique	Characteristic
<ul style="list-style-type: none"> <li>Architecture based on Extended Kalman Filter and voting system [27]</li> </ul>	<ul style="list-style-type: none"> <li>Need of an identified analytical model</li> <li>Need to tune the parameters of the model following each failure to prevent model drift</li> </ul>
<ul style="list-style-type: none"> <li>Architecture based on hardware redundancy [24]</li> </ul>	<ul style="list-style-type: none"> <li>Need of at least three diversified or redundant sensors for each state</li> </ul>
<ul style="list-style-type: none"> <li>Architecture based on an informational approach [28]</li> </ul>	<ul style="list-style-type: none"> <li>Existence of a cumulative divergence error due to the use of a Kalman Filter</li> </ul>

This fault-tolerance technique for data fusion in UAVs not only contributes to enhancing the reliability and efficiency of UAV operations but also holds considerable promise for broader impacts in the industry. The reduced dependency on redundant hardware would make UAVs more cost-effective and efficient, which leads to substantial cost savings in both the manufacturing and maintenance processes.

## 5. Conclusions

In this paper, a fault-tolerant multi-sensor fusion strategy based on a deep learning framework was proposed for a UAV system. The data fusion was performed using an LSTM network, which presents many advantages compared to the traditional Kalman Filter. However, its performance was tested when the UAV undergoes small displacements. This paper does not address dynamic motions, which should be examined in future research. On the other hand, the fault diagnosis steps were formulated using residual generation between the outputs of the fusion blocks and the evaluation of the moving averages of these outputs. The moving average used in the residual tests offers numerous benefits in comparison to other methods due to its simplicity since it does not require complex mathematical models or algorithms to implement, as well as adaptability, as it can adapt to changes in the system over time, as the average is recalculated over a sliding window of recent measurements. In addition, moving average can detect small changes in the data, which allows for the early detection of faults. However, this study does not account for the optimization of the thresholds used for fault detection or their effects on detection delay, which should be explored in future research.

**Author Contributions:** Conceptualization, M.S., A.M., C.F., and Z.N.; validation, A.M.; formal analysis, M.S., C.F., and Z.N.; writing—original draft preparation, M.S.; writing—review and editing, C.F.; supervision, C.F. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Data Availability Statement:** The original data presented in the study are openly available at <https://github.com/MA-Saied/UAV-Dataset.git>, accessed on 7 August 2024.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## Abbreviations

The following abbreviations are used in this manuscript:

UAV	Unmanned Aerial Vehicle;
MA	Moving Average;
LSTM	Long Short-Term Memory;
IMU	Inertial Measurement Unit;
RNN	Recurrent Neural Network;
NN	Neural Network;
NED	North-East-Down;
GPS	Global Positioning System;
DC	Direct Current;
PV	Photovoltaic;
RC	Remote Control;
RMSE	Root Mean Squared Error.

## References

- Durrant-Whyte, H.; Henderson, T.C. Multisensor Data Fusion. In *Handbook of Robotics*; Siciliano, B., Khatib, O., Eds.; Springer: Berlin/Heidelberg, Germany, 2008; pp. 585–610.
- Khaleghi, B.; Khamis, A.; Karray, F.O.; Razavi, S.N. Multisensor data fusion: A review of the state-of-the-art. *Inf. Fusion* **2013**, *14*, 28–44. [[CrossRef](#)]
- Hall, D.; Llinas, J. An introduction to multisensor data fusion. *Proc. IEEE* **1997**, *85*, 6–23. [[CrossRef](#)]
- Smets, P. Analyzing the combination of conflicting belief functions. *Inf. Fusion* **2007**, *8*, 387–412. [[CrossRef](#)]
- Zhu, Y.; Song, E.; Zhou, J.; You, Z. Optimal Dimensionality Reduction of Sensor Data in Multisensor Estimation Fusion. *IEEE Trans. Signal Process.* **2005**, *53*, 1631–1639.
- Bader, K.; Lussier, B.; Schon, W. A fault tolerant architecture for data fusion: A real application of Kalman filters for mobile robot localization. *Robot. Auton. Syst.* **2017**, *88*, 11–23. [[CrossRef](#)]
- Du, B.; Shi, Z.; Song, J.; Wang, H.; Han, L. A Fault-Tolerant Data Fusion Method of MEMS Redundant Gyro System Based on Weighted Distributed Kalman Filtering. *Micromachines* **2019**, *10*, 278. [[CrossRef](#)]
- Darvishi, H.; Ciunzo, D.; Eide, E.R.; Salvo Rossi, P. Sensor-Fault Detection, Isolation and Accommodation for Digital Twins via Modular Data Driven Architecture. *IEEE Sens. J.* **2021**, *21*, 4827–4838. [[CrossRef](#)]
- Liu, L.; Han, G.; He, Y.; Jiang, J. Fault-Tolerant Event Region Detection on Trajectory Pattern Extraction for Industrial Wireless Sensor Networks. *IEEE Trans. Ind. Inform.* **2020**, *16*, 2072–2080. [[CrossRef](#)]
- Allerton, D.J.; Jia, H. A Review of Multisensor Fusion Methodologies for Aircraft Navigation Systems. *J. Navig.* **2005**, *58*, 405–417. [[CrossRef](#)]
- Simanek, J.; Kubelka, V.; Reinstein, M. Improving multi-modal data fusion by anomaly detection. *Auton. Robot.* **2015**, *39*, 139–154. [[CrossRef](#)]
- Kellalib, B.; Achour, N.; Coelen, V.; Nemra, A. Towards simultaneous localization and mapping tolerant to sensors and software faults: Application to omnidirectional mobile robot. *Proc. Inst. Mech. Eng. Part I J. Syst. Control Eng.* **2021**, *235*, 269–288. [[CrossRef](#)]
- Sadeghzadeh-Nokhodberiz, N.; Poshtan, J. Distributed Interacting Multiple Filters for Fault Diagnosis of Navigation Sensors in a Robotic System. *IEEE Trans. Syst. Man Cybern. Syst.* **2017**, *47*, 1383–1393. [[CrossRef](#)]
- Ricquebourg, V.; Delafosse, M.; Delahoche, L.; Marhic, B.; Jolly-Desodt, A.; Menga, D. Fault Detection by Combining Redundant Sensors: A Conflict Approach within the TBM Framework. In *Proceedings of the Cognitive Systems with Interactive Sensors*. 2007. Available online: [https://www.researchgate.net/publication/228538641\\_Fault\\_Detection\\_by\\_Combining\\_Redundant\\_Sensors\\_a\\_Conflict\\_Approach\\_Within\\_the\\_TBM\\_Framework](https://www.researchgate.net/publication/228538641_Fault_Detection_by_Combining_Redundant_Sensors_a_Conflict_Approach_Within_the_TBM_Framework) (accessed on 2 July 2024).
- Delmotte, F.; Gacquer, G. Detection of defective sources with belief functions. In *Proceedings of the International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, Torremolinos, Malaga, Spain, 22–27 June 2008.
- Delmotte, F. Detection of defective sources in the setting of possibility theory. *Fuzzy Sets Syst.* **2007**, *158*, 555–571. [[CrossRef](#)]
- Li, S.-Q.; Zhang, S.-X. A congeneric multi-sensor data fusion algorithm and its fault-tolerance. In *Proceedings of the International Conference on Computer Application and System Modeling*, Taiyuan, China, 22–24 October 2010.
- Allerton, D.J.; Jia, H. Distributed data fusion algorithms for inertial network systems. *IET Radar Sonar Navig.* **2008**, *2*, 51–62. [[CrossRef](#)]
- Silvagni, M.; Tonoli, A.; Zenerino, E.; Chiaberge, M. Multipurpose UAV for search and rescue operations in mountain avalanche events. *Geomat. Nat. Hazards Risk* **2017**, *8*, 18–33. [[CrossRef](#)]
- Saied, M.; Mahairy, T.; Francis, C.; Shraim, H.; Mazeh, H.; El Rafei, M. Differential Flatness-Based Approach for Sensors and Actuators Fault Diagnosis of a Multirotor UAV. *IFAC-Pap.* **2019**, *52*, 831–836. [[CrossRef](#)]
- Avram, R.; Zhang, X.; Muse, J. Quadrotor Sensor Fault Diagnosis with Experimental Results. *J. Intell. Robot. Syst.* **2017**, *86*, 115–137. [[CrossRef](#)]

22. Sabatini, R.; Ramasamy, S.; Gardi, A.; Salazar, L.R. Low-cost Sensors Data Fusion for Small Size Unmanned Aerial Vehicles Navigation and Guidance. *Int. J. Unmanned Syst. Eng.* **2013**, *1*, 16–47. [[CrossRef](#)]
23. Garcia, J.; Molina, J.M.; Trincado, J. Real evaluation for designing sensor fusion in UAV platforms. *Inf. Fusion* **2020**, *63*, 136–152. [[CrossRef](#)]
24. Geng, K.; Chulin, N.A. Applications of multi-height sensors data fusion and fault-tolerant Kalman filter in integrated navigation system of UAV. *Procedia Comput. Sci.* **2017**, *103*, 231–238. [[CrossRef](#)]
25. Gu, Y.; Gross, J.N.; Rhudy, M.B.; Lassak, K. A Fault-Tolerant Multiple Sensor Fusion Approach Applied to UAV Attitude Estimation. *Int. J. Aerosp. Eng.* **2016**, *2016*, 6217428. [[CrossRef](#)]
26. Gross, J. Sensor Fusion Based Fault-Tolerant Attitude Estimation Solutions for Small Unmanned Aerial Vehicles. Ph.D. Thesis, Degree-West Virginia University, Morgantown, WV, USA, 2011.
27. Hamadi, H. Fault-Tolerant Control of a Multicopter Unmanned Aerial Vehicle under Hardware and Software Failures. Ph.D. Thesis, University of Technology of Compiègne, Compiègne, France, 2020.
28. Saied, M.; Tabikh, A.R.; Francis, C.; Hamadi, H.; Lussier, B. An Informational Approach for Fault Tolerant Data Fusion Applied to a UAV's Attitude, Altitude, and Position Estimation. *IEEE Sens. J.* **2021**, *21*, 27766–27778. [[CrossRef](#)]
29. Gültekin, O.; Cinar, E.; Ozkan, K.; Yazıcı, A. Multisensory data fusion-based deep learning approach for fault diagnosis of an industrial autonomous transfer vehicle. *Expert Syst. Appl. Int. J.* **2022**, *200*, 117055. [[CrossRef](#)]
30. Kolanowski, K.; Hwietlicka, A.; Kapela, R.; Pochmara, J.; Rybarczyk, A. Multisensor data fusion using Elman neural networks. *Appl. Math. Comput.* **2018**, *319*, 236–244. [[CrossRef](#)]
31. Hochreiter, S.; Schmidhuber, J.S. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [[CrossRef](#)]
32. Gers, F.; Schmidhuber, J.; Cummins, F. Learning to forget: Continual prediction with LSTM. *Neural Comput.* **2000**, *12*, 2451–2471. [[CrossRef](#)]
33. Zhao, Y.; He, X.; Zhang, J.; Ji, H.; Zhou, D.; Pecht, M.G. Detection of intermittent faults based on an optimally weighted moving average T2 control chart with stationary observations. *Automatica* **2021**, *123*, 109298. [[CrossRef](#)]
34. Kim, J.; Kwak, S. DC Series Arc Detection Algorithm Based on Adaptive Moving Average Technique. *IEEE Access* **2021**, *9*, 94426–94437. [[CrossRef](#)]
35. Narkhede P.; Walambe R.; Poddar S.; Kotecha, K. Incremental learning of LSTM framework for sensor fusion in attitude estimation. *PeerJ Comput. Sci.* **2021**, *7*, e662. [[CrossRef](#)] [[PubMed](#)]
36. Guang, X.; Gao, Y.; Li, G. IMU Data and GPS Position Information Direct Fusion Based on LSTM. *Sensors* **2021**, *21*, 2500. [[CrossRef](#)]

**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.