

Article

Sensor Faults Isolation in Networked Control Systems: Application to Mobile Robot Platoons

Wijaya Kurniawan ^{1,*}  and Lorinc Marton ^{2,†} 

¹ Department of Electrical Engineering and Information Systems, University of Pannonia, 8200 Veszprem, Hungary

² Department of Electrical Engineering, Sapiientia Hungarian University of Transylvania, 547367 Corunca, Romania; martonl@ms.sapiientia.ro

* Correspondence: wijaya.kurniawan@virt.uni-pannon.hu

† These authors contributed equally to this work.

Abstract: In networked control systems, sensor faults in a subsystem have a major influence on the entire network as the fault effect reaches the other subsystems through the network interconnections. In this paper, a fault diagnosis-oriented model is proposed for linear networked control systems that can be applied to the robotics platoon. In addition, this model can also be used to design distributed Unknown Input Observers (UIO) in each subsystem to accomplish weak sensor faults isolation by treating the network disturbances and fault propagation through the network as unknown inputs. A case study was developed in which the subsystems were represented by robots that are connected in a wireless communication-based leader-follower scheme. The simulation results show that the model successfully reproduces the expected behaviour of the robotics platoon in the presence of sensor faults. Furthermore, weak sensor faults isolation is also achieved by observing the residual signals produced by the UIOs in each of the subsystems.

Keywords: Unknown Input Observer; mobile robot platoons; weak fault isolation; sensor faults; networked systems



Citation: Kurniawan, W.; Marton, L. Sensor Faults Isolation in Networked Control Systems: Application to Mobile Robot Platoons. *Sensors* **2021**, *21*, 6702. <https://doi.org/10.3390/s21206702>

Academic Editors: Levente Tamás and Andras Majdik

Received: 18 August 2021

Accepted: 7 October 2021

Published: 9 October 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 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

The theory of cooperative robotics deals with such groups of robots that communicate with each other to achieve some common goals. Such systems have always been an important topic to be researched because they can be applied to perform challenging tasks such as space exploration, domestic help, healthcare, military operations, and mobile Wireless Sensor Networks (WSN) [1]. In this era of Industry 4.0, robot platoons becomes more popular and have an increasing demand in the modern industrial environment because they can solve tasks that can not be done by only one robot [2]. Multiple robots can cooperate in accomplishing predefined tasks, such as goods transportation in the warehouses or monitoring and surveillance of production systems. However, there are many safety-critical issues to be handled here in the area of communication and control of these systems [3,4] for example the fault isolation problem for each robot in the platoons.

To achieve a common goal, besides local measurement from its attached sensors, each robot in the multi robotic systems also needs information from the others through a communication network. Because sensor faults on a robot can propagate through the network, fault isolation problems must be acknowledged so that these faults will not affect the others and disturb the coordination. There are many approaches to solve the fault diagnosis problem in control system [5]. A book by Varga et al. discusses fault detection and isolation from a computational perspective [6]. In the case of a model-based approach, a simple linear model is proposed to detect and isolate faults in electrical grids by Pozna et al. [7]. Gertler et al. proposed the design of dynamic parity relations for fault detection and isolation [8]. The optimization-based approach usually using either H_2 or

H_∞ has also been done by Song et al. and Niemann et al. [9,10]. Menke et al. and White et al. used the Kalman filter for fault estimation in aerospace systems [11,12]. There is also a book by Simani et al. explaining how to undertake fault diagnosis based on system identification techniques [13]. The current emerging trend in Artificial Intelligence (AI) has also been exploited by Michail et al, Nasiri et al, and Xiao et al. to undertake fault detection and isolation [14–16]. In a recent article, Li et al. have also made a summary about many methods which have been researched to solve this fault detection and identification problems [17].

Other than those already mentioned, Unknown Input Observer (UIO) is a popular approach in a model-based fault diagnosis because it can decouple the influence of unknown inputs on state estimation. In [18], it was shown that weak faults isolation can be achieved using this UIO. Weak faults isolation means that the faults in different sensors can be isolated when no simultaneous fault occurs. Residual generators are built from these UIOs for fault diagnosis purposes which are sensitive to some groups of faults but insensitive to others. Chakrabarty et al. have further explored the implementation of this UIO to accommodate bounded exogenous inputs and delayed measurements [19,20]. Xu et al. combined UIO with set-theoretic methods for robust fault detection purpose [21]. Sensor faults detection on a UAV using UIO has also been researched by Zuo et al. [22].

Because the network contributes to many unpredicted unknown inputs such as disturbance and fault propagation, this unknown inputs decoupling feature becomes more important in networked control systems. Thus, much research to implement UIO in this networked environment has been done. Taha et al. investigated state estimation conducted by UIO which is connected to an observed plant via network [23]. UIO for interconnected second-order system investigation has also been done by Shames et al. [24]. Chen et al. and Chakrabarty et al. explored the design of UIO for a class of interconnected non-linear systems [25,26].

In terms of fault diagnosis in a networked environment, Liu et al. and Shames et al. explored the utilization of UIO to detect a faulty agent in a multi-agent system [27,28]. Zhang et al. investigated how to do either actuator or sensor faults identification for each agent using a global UIO in networked control systems [29].

To take this research trend further, this paper proposes the design of a bank of local UIOs to detect and isolate faulty sensors in each subsystem of the robotics platoon, not just detecting faulty robots. The research questions we are going to answer are:

- How can we develop such sensor faults isolation method for each robot of the platoons using only conventional sensors (GPS, velocity sensor, radar) and network communication?
- How to compensate faults propagation effect in the communication network from one robot to the others so that it does not affect the sensor faults isolation process?
- How to implement the sensor faults isolation in a distributed way?

We do it by extending the UIO-based sensor faults isolation problem into networked control systems. This local faults isolation scheme has advantages in terms of scalability features compared to a global UIO. The concerned networked control systems for the implementation of these banks of UIOs are a controlled robots platoon inspired by Adaptive Cruise Control (ACC) system [30]. The proposed approach ensures the isolation of faults in sensors that are critical for safe platooning.

The new contributions of this paper are as follows:

1. We have developed a model for sensor faults diagnosis purposes for the robotics platoon.
2. We have proposed a general UIO-based distributed sensor faults isolation approach for a network of linear control systems.
3. Based on these two results, we have solved the weak sensor faults isolation problem in such robotics platoon that use conventional sensors: GPS-based localization; velocity sensor; and radar sensor.

The organization of this paper will be as follow: In Section 2, the existing Unknown Input Observer method for sensor faults isolation is recalled. In Section 3, sensor faults propagation and the proposed distributed sensor faults isolation in networked control systems using UIO are explained. In Section 4, the proposed modelling of a controlled robots platoon is discussed both in the fault-free case and faulty case. In Section 5, the implementation of weak sensor faults isolation on the robotics platoon is presented. In Section 6, the simulation results of sensor faults isolation in 5 robots moved together as a platoon in a leader-follower control scheme are discussed. Finally, conclusions and suggestions for future works are given in Section 7.

2. UIO-Based Sensor Faults Isolation

This section recalls general notions about UIO-based estimation and faults isolation, see [18] for details.

2.1. General Model of Unknown Input Observer (UIO)

Consider a fault free linear system injected with unknown inputs. In addition, also consider for the sake of convenience but without loss of generality, that the system's outputs are not directly affected by the inputs. Hence, the fault free model can be written in the form of:

$$\begin{aligned}\dot{\mathbf{x}}(t) &= A\mathbf{x}(t) + B\mathbf{u}(t) + E\mathbf{d}(t) \\ \mathbf{y}(t) &= C\mathbf{x}(t)\end{aligned}\quad (1)$$

where $\mathbf{x}(t) \in \mathbb{R}^n$ is the state vector, $\mathbf{y}(t) \in \mathbb{R}^m$ is the measured output vector, $\mathbf{u}(t) \in \mathbb{R}^r$ is the known control input vector, $\mathbf{d}(t) \in \mathbb{R}^q$ is the unknown inputs (or disturbance) vector. A, B, C, and E are known matrices with appropriate dimension.

An Unknown Input Observer (UIO) is a special observer, which despite of the presence of some unknown inputs $\mathbf{d}(t)$, it still produces a state estimation $\tilde{\mathbf{x}}$ that converges to the real one if certain conditions hold. The state-space equation for this UIO's state $\mathbf{z}(t)$ is:

$$\begin{aligned}\dot{\mathbf{z}}(t) &= F\mathbf{z}(t) + T B\mathbf{u}(t) + K\mathbf{y}(t) \\ \tilde{\mathbf{x}}(t) &= \mathbf{z}(t) + H\mathbf{y}(t)\end{aligned}\quad (2)$$

where $\tilde{\mathbf{x}}(t) \in \mathbb{R}^n$ is the state estimation vector and $\mathbf{z}(t) \in \mathbb{R}^n$ is the UIO's internal state vector.

By choosing the matrices F , T , K , and H to satisfy:

$$\begin{aligned}HCE &= E \\ T &= I - HC \\ F &= A - HCA - K_1C \\ K_2 &= FH \\ K &= K_1 + K_2\end{aligned}\quad (3)$$

with I is the Identity Matrix, then $\lim_{t \rightarrow +\infty} (\mathbf{x} - \tilde{\mathbf{x}}) = \mathbf{0}$.

The necessary conditions so that Equation (3) can be satisfied are:

1. $\text{rank}(CE) = \text{rank}(E)$
 2. $(C, A - HCA)$ is a detectable pair
- (4)

The first condition is to ensure that H matrix in Equation (3) exists. It also indicates that the maximum size of the disturbances vector that can be decoupled can not exceed the number of independent measurements which are done. In other words, sensor faults isolation encourages redundancy either in the measurement or in the number of the used sensors. The second condition is a milder condition related to observability. If

$(C, A - HCA)$ is already observable, a pole placement method can be directly used to determine the value of K_1 . If it is not observable, an observable canonical decomposition procedure must be performed to do a matrix transformation before using pole placement to find the value of K_1 . The details of this procedure are explained in [31].

2.2. UIO for Sensor Faults Isolation

Sensor faults isolation refers to a process for determining in which sensor the fault has occurred. One way to accomplish this is by using structured residual signals. These structured residual signals are made in such a way that each of them is sensitive (or insensitive) to a certain group of faults while insensitive (or sensitive) to the others. This property becomes the basis to solve the isolation problem. As mentioned earlier, because UIO can decouple some unknown input signals so that it is insensitive to them, it has the potential to produce the needed structured residual signals for sensor faults isolation. One remark is that this scheme can only isolate a single or a certain group of faults. It can not isolate simultaneous faults. This is called weak sensor faults isolation.

Consider a linear system with sensor faults and disturbance as follows:

$$\begin{aligned}\dot{x}(t) &= Ax(t) + Bu(t) + Ed(t) \\ y(t) &= Cx(t) + f_s(t)\end{aligned}\quad (5)$$

To apply the UIO for weak sensor faults isolation, we delete one by one of the rows in the C matrix in Equation (5) related with each sensor fault as shown in Equation (6) below:

$$y^i(t) = C^i x(t) + f_s^i(t) \quad (6)$$

where $C^i \in \mathbb{R}^{n \times (m-1)}$ is the C matrix without the i th row, $y^i(t) \in \mathbb{R}^{m-1}$ is y vector without the i th entry, and $f_s^i(t) \in \mathbb{R}^{m-1}$ is $f_s(t)$ vector without the i th entry.

Then, a bank of m UIOs for the system can be derived with the following dynamics:

$$\begin{aligned}\dot{z}^i(t) &= F^i z^i(t) + T^i Bu(t) + K^i y^i(t) \\ r^i(t) &= (I - C^i H^i) y^i(t) - C^i z^i(t)\end{aligned}\quad (7)$$

with $i = 1, 2, 3, \dots, m$; $z^i(t) \in \mathbb{R}^n$ is the i th sensor fault UIO's state vector; $r^i(t) \in \mathbb{R}^{m-1}$ is the i th sensor fault UIO's residual signal; and the F^i , T^i , K^i , and H^i matrices are chosen in such a way to satisfy the following conditions:

$$\begin{aligned}H^i C^i E &= E \\ T^i &= I - H^i C^i \\ F^i &= T^i A - K_1^i C^i \\ K_2^i &= F^i H^i \\ K^i &= K_1^i + K_2^i\end{aligned}\quad (8)$$

where K_1^i is chosen to stabilize F^i .

Remark that Equation (7) shows $z^i(t)$ is affected by all inputs except the i th entry related to a specific sensor fault ($y^i(t)$). If a fault occurs, all the residual signals will be triggered except for the i th one. Hence, this bank of UIOs can be used as residual generators for weak sensor faults isolation.

3. Sensor Faults Isolation in Networks of Control Systems

3.1. Sensor Faults in Networked Control Systems

In networked control systems, besides the local control inputs, the states of each subsystem in that network are also affected by the states of its neighbouring subsystems. This is shown in Equation (9) as follows:

$$\begin{aligned}\dot{\mathbf{x}}_j &= A_j \mathbf{x}_j + B_j \mathbf{u}_j + I_j \mathbf{i}_j \\ \mathbf{y}_j &= C_j \mathbf{x}_j\end{aligned}\quad (9)$$

where subscript $j = 1, 2, 3, \dots, N$ represents j th subsystem, $\mathbf{x}_j \in \mathbb{R}^n$ is the state vector, $\mathbf{y}_j \in \mathbb{R}^m$ is the measured output vector, $\mathbf{u}_j \in \mathbb{R}^r$ is the control input vector, and $\mathbf{i}_j \in \mathbb{R}^p$ is the interconnection input vector which represents the states from the neighbouring subsystems. I_j is an interconnection input matrix with appropriate dimension.

By assuming static linear interconnections, the interconnection inputs \mathbf{i}_j can be generally written as:

$$\mathbf{i}(t) = L\mathbf{y}(t)$$

$$\begin{bmatrix} i_1(t) \\ i_2(t) \\ i_3(t) \\ \vdots \\ \vdots \\ \vdots \\ i_N(t) \end{bmatrix} = \begin{bmatrix} 0 & L_{12} & L_{13} & \dots & \dots & L_{1N} \\ L_{21} & 0 & L_{23} & \dots & \dots & L_{2N} \\ L_{31} & L_{32} & 0 & \dots & \dots & L_{3N} \\ \vdots & \vdots & \vdots & \dots & \dots & \vdots \\ \vdots & \vdots & \vdots & \dots & \dots & \vdots \\ \vdots & \vdots & \vdots & \dots & \dots & \vdots \\ L_{N1} & L_{N2} & L_{N3} & \dots & \dots & 0 \end{bmatrix} \begin{bmatrix} y_1(t) \\ y_2(t) \\ y_3(t) \\ \vdots \\ \vdots \\ \vdots \\ y_N(t) \end{bmatrix}\quad (10)$$

where L is called the interconnection matrix. This L matrix contains the matrices L_{jk} which describes the relations between output measurements from k th subsystem being connected as interconnection inputs to j th subsystem.

For N connected Linear Time Invariant (LTI) subsystems subject to sensor faults and measurement noises, each of them has dynamics described by state-space equations as follows:

$$\begin{aligned}\dot{\mathbf{x}}_j &= A_j \mathbf{x}_j + B_j \mathbf{u}_j + I_j(\mathbf{i}_j + \delta \mathbf{i}_j) \\ \mathbf{y}_j &= C_j \mathbf{x}_j + \mathbf{f}_{sj} + \mathbf{w}_j\end{aligned}\quad (11)$$

where $\mathbf{f}_{sj} \in \mathbb{R}^m$ is the fault sensor vector, $\mathbf{w}_j \in \mathbb{R}^m$ is the measurement noises vector, and $\delta \mathbf{i}_j \in \mathbb{R}^p$ is the disturbance term that describes the sensor faults propagation through network such that:

$$\delta \mathbf{i}_j = \sum_{k=1}^N L_{jk}(\mathbf{f}_{sk} + \mathbf{w}_k)\quad (12)$$

Now, consider a general linear control algorithm where the control inputs \mathbf{u}_j depend on the state variable which is obtained from the output measurements \mathbf{y}_j so that:

$$\mathbf{u}_j = K_j \mathbf{y}_j + \mathbf{u}_{cj}\quad (13)$$

where K_j is the controller gain matrix and \mathbf{u}_{cj} is a feed-forward term.

Hence, by substituting Equations (12) and (13) into Equation (11) and rearranging it, it can be seen that the sensor faults are now affecting the dynamics of the subsystem as follows:

$$\begin{aligned}\dot{\mathbf{x}}_j &= [A_j + B_j K_j C_j] \mathbf{x}_j + [B_j \quad I_j] \begin{bmatrix} \mathbf{u}_{cj} \\ \mathbf{i}_j \end{bmatrix} + [I_j \delta \mathbf{i}_j + B_j K_j \mathbf{f}_{sj} + B_j K_j \mathbf{w}_j] \\ \mathbf{y}_j &= C_j \mathbf{x}_j + \mathbf{f}_{sj} + \mathbf{w}_j\end{aligned}\quad (14)$$

3.2. UIO for Sensor Faults Isolation in Networked Control Systems

In this section, we show that the UIO-based weak sensor faults isolation scheme presented in the earlier section can be applied to develop proper sensor faults isolation despite fault propagation through the network. The resulting observer can be implemented in a distributed way which means that it uses local subsystem's measurements and the

measurements of the neighbouring subsystems. The model in Equation (14) can be written in a more compact way as follows:

$$\begin{aligned}\dot{x}_j &= \hat{A}_j x_j + \hat{B}_j \hat{u}_j + \hat{E}_j \hat{d}_j \\ y_j &= C_j x_j + f_{sj} + w_j\end{aligned}\quad (15)$$

where

$$\begin{aligned}\hat{A}_j &= A_j + B_j K C_j \\ \hat{B}_j &= [B_j \quad I_j] \\ \hat{E}_j &= [I_j \quad B_j K_j] \\ \hat{d}_j &= \begin{bmatrix} \delta i_j \\ f_{sj} + w_j \end{bmatrix} \\ \hat{u}_j &= \begin{bmatrix} u_{cj} \\ i_j \end{bmatrix}\end{aligned}\quad (16)$$

Remark that Equation (15) has the same form as Equation (5) such that the weak sensor faults isolation scheme can be applied directly. In addition, it is also seen that the input \hat{u}_j depends only on the control inputs of the j th subsystem and the measured outputs of the neighbouring subsystems.

3.3. Threshold Computation for Fault Isolation

As previously explained, to perform weak sensor faults isolation, the UIOs are treated as residual generators in which each residual is insensitive to a specific fault while sensitive to the others. Consequently, because we determine the faulty sensor by observing noisy measurement signals, a suitable threshold need to be specified. By substituting Equation (6) into Equation (7), we get:

$$\begin{aligned}\dot{z}^i(t) &= F^i z^i(t) + T^i B u(t) + K^i (C^i x(t) + f_s^i(t) + w^i(t)) \\ r^i(t) &= (I - C^i H^i) (C^i x(t) + f_s^i(t) + w^i(t)) - C^i z^i(t)\end{aligned}\quad (17)$$

Thus, the sole effect of the measurement noises $z_w^i(t)$ and $r_w^i(t)$ on the residuals are defined by the following equation:

$$\begin{aligned}z_w^i(t) &= F^i z^i(t) + K^i w^i(t) \\ r_w^i(t) &= (I - C^i H^i) w^i(t) - C^i z^i(t)\end{aligned}\quad (18)$$

To minimize the occurrence of false alarm situation, the highest peak gain of the system's frequency response is chosen as threshold values (th^i) as follows:

$$th^i = \|G_{rw}^i\|_\infty \|w^i\|_\infty\quad (19)$$

where $\|G_{rw}^i\|_\infty$ is the infinity norm of the system in Equation (18).

Hence, the weak sensor faults isolation is achieved by formulating the decision signal σ_i in which the fault happened at the i th sensor using the relation:

$$\sigma_i = \begin{cases} 1 & \text{if } \|r^i(t)\| \leq th^i \text{ and } \|r^i(t)\| > th^i, \forall \hat{i}(t) \neq i \\ 0 & \text{otherwise.} \end{cases}\quad (20)$$

4. Modelling of Controlled Robots Platoon

4.1. Fault Free Model of Controlled Robots Platoon

In this paper, the subsystems in networked control systems are represented by robots in a controlled robots platoon. The modelling of this controlled robots platoon is inspired

by Adaptive Cruise Control (ACC) which is used for vehicle following system [32]. Several robots are assumed to move along x axis and each of them has a double integrator dynamic as follows:

$$m_j \ddot{x}_j = u_j \quad (21)$$

where $j = 2, 3, 4, \dots, n$ represents the j th follower robot with n being the number of robots ($j = 1$ is allocated to represent the leader robot), $\dot{x}_j = \dot{v}_j = a_j$ with x_j is position, v_j is velocity, a_j is acceleration, m_j is mass, and u_j is the control input of the j th robot.

According to [30], two conditions that must be achieved in this leader-following control system are individual stability and string stability.

Individual stability means that the spacing error of the concerned robot should converge to zero when the preceding robot has a constant speed. The spacing error describing the distance with the preceding robot can be expressed as:

$$\begin{aligned} \epsilon_j &= x_j - x_{j-1} + l_{j-1} \\ e_j &= x_j - x_{j-1} + L_j \end{aligned} \quad (22)$$

where ϵ_j is the measured inter-robot spacing, e_j is the spacing error of the j th robot, and L_j is some desired value of inter-robot spacing and includes the length of the preceding robot l_{j-1} . Figure 1 shows the block diagram of the concerned robots platoon.

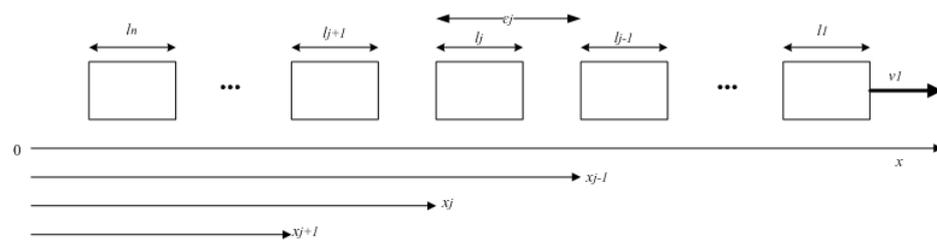


Figure 1. Robots platoon block diagram.

Meanwhile, string stability means that the spacing errors on each robot should not amplify towards the end of the string ($\|e_j\|_\infty \leq \|e_{j-1}\|_\infty$). To satisfy both individual stability and string stability, a constant time-gap policy must be used to formulate the control input for each robot. In this policy, the desired inter-robot spacing L_j is not made constant. Instead, it varies in proportional with velocity as well as the spacing error e_j as follows:

$$\begin{aligned} L_j &= l_{j-1} + h\dot{x}_j \\ e_j &= \epsilon_j + h\dot{x}_j \end{aligned} \quad (23)$$

where h is a prescribed positive constant parameter called time-gap and ϵ_j is as given in the Equation (22).

A previous research [30] found that a suitable control input u_j for this constant time-gap policy is in the form of:

$$\begin{aligned} u_j &= \ddot{x}_j = -\frac{1}{h}(\dot{\epsilon}_j + \lambda e_j) \\ &= -\frac{1}{h}(\dot{x}_j - \dot{x}_{j-1} + \dot{l}_{j-1} + \lambda x_j - \lambda x_{j-1} + \lambda l_{j-1} + \lambda h \dot{x}_j) \\ &= -\frac{\lambda}{h}x_j + \left(-\frac{1}{h} - \lambda\right)\dot{x}_j + \frac{\lambda}{h}x_{j-1} + \frac{1}{h}\dot{x}_{j-1} + \left(-\frac{\lambda}{h}\right)l_{j-1} \end{aligned} \quad (24)$$

where λ is a constant chosen such that $\lambda > 0$.

For the sake of generality, Equation (24) is rewritten as:

$$u_j = k_{1j}x_j + k_{2j}\dot{x}_j + k_{3j}x_{j-1} + k_{4j}\dot{x}_{j-1} + k_{5j}l_{j-1} \quad (25)$$

where $k_{1j} = -\frac{\lambda}{h}$, $k_{2j} = -\frac{1}{h} - \lambda$, $k_{3j} = \frac{\lambda}{h}$, $k_{4j} = \frac{1}{h}$, and $k_{5j} = -\frac{\lambda}{h}$.

Hence, the related state-space model for the j th robot is:

$$\begin{bmatrix} \dot{x}_{1j} \\ \dot{x}_{2j} \\ \dot{x}_{3j} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ k_{1j}/m_j & k_{2j}/m_j & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_{1j} \\ x_{2j} \\ x_{3j} \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ k_{5j}/m_j & k_{3j}/m_j & k_{4j}/m_j \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} l_{j-1} \\ x_{j-1} \\ \dot{x}_{j-1} \end{bmatrix} \quad (26)$$

where $x_{1j} = x_j$, $x_{2j} = \dot{x}_j$, and $x_{3j} = x_j - x_{j-1}$.

4.2. Model of Controlled Robots Platoon with Sensor Faults

To implement the robots platoon's control system, it is assumed that each robot is equipped with conventional sensors which are a GPS-based sensor, a velocity sensor, and a radar sensor. Thus, the considered measurement outputs are:

1. y_{1j} is a position measurement to obtain the x_{1j} state using a GPS-based sensor (S1).
2. y_{2j} is a velocity measurement obtained from the same GPS-based sensor. For GPS-based velocity measurement, please refer to [33].
3. y_{3j} is a velocity measurement by using the wheel-mounted velocity sensor (S2) on the robot to obtain the x_{2j} state.
4. y_{4j} is an inter-vehicle distance measurement to obtain the x_{3j} state using a radar-based sensor (S3).

In addition, it is also assumed that each robot receives the position information from the preceding robot through wireless inter-vehicle network communication (x_{j-1}). The velocity from the preceding robot can also be calculated by differentiating this information ($v_{j-1} = \dot{x}_{j-1}$).

Note that the common use of local sensors and inter-vehicle communication introduces measurements redundancy. Velocity can either be obtained from the GPS-based sensor or measured by the wheel-mounted velocity sensor. In the same manner, the inter-vehicle distance also can either be measured directly or obtained by the difference between GPS-based measurement and the position of the preceding vehicle which is received through the wireless communication network.

Besides that, the sensors are affected by additive faults and measurement noises. The measurement noises are considered high frequency and low amplitude signals. Meanwhile, sensor faults are due to malfunctioning of sensors, the measurements signal from them is expected to have a sudden or incremental change of amplitude at the time of the incident. Thus, the faults are assumed to be high magnitude signals (e.g., sensor biases) as compared to measurements noise that has a considerable influence on the dynamic behaviour of the controlled robots platoon. Hence, the output measurements can be modelled as:

$$\begin{aligned} y_j &= C_j x_j + f_{sj} + w_j \\ C_j &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{aligned} \quad (27)$$

where $y_j = [y_{1j} \ y_{2j} \ y_{3j} \ y_{4j}]^T$ is the measurement vector, $x_j = [x_{1j} \ x_{2j} \ x_{3j}]^T$ is the state vector, $f_{sj} = [f_{s1j} \ \dot{f}_{s1j} \ f_{s2j} \ f_{s3j}]^T$ is the sensor faults vector which contains position sensor fault (f_{s1j}), derivative of the position sensor fault (\dot{f}_{s1j}), velocity sensor fault (f_{s2j}), distance sensor fault (f_{s3j}), and $w_j = [w_{1j} \ \dot{w}_{1j} \ w_{2j} \ w_{3j}]^T$ is the measurement noises vector.

The received information from the network is also considered to be affected by network disturbances as follows:

$$i_{j-1} = x_{j-1} + \delta i_{j-1} \quad (28)$$

where i_{j-1} is the received position information from the preceding robot, x_{j-1} is the position information from the preceding robot, and δi_{j-1} represents the disturbance on the signal

received through the network due to the possible position sensor fault of the preceding robot.

The control input for each robot in this controlled robots platoon can be calculated from the enumerated sensor measurements which are affected both by sensor faults and network disturbance. Because of the redundancy in measurements, there are several ways to compute the control signal u_j . One possible way to compute is:

$$\begin{aligned} u_j &= k_{1j}y_{1j} + k_{2j}y_{3j} + k_{3j}i_j + k_{4j}\dot{i}_j + k_{5j}l_{j-1} \\ u_j &= k_{1j}(x_{1j} + f_{s1j} + w_{1j}) + k_{2j}(x_{2j} + f_{s2j} + w_{2j}) + k_{3j}(x_{j-1} + \delta i_{j-1}) \\ &\quad + k_{4j}(\dot{x}_{j-1} + \delta \dot{i}_{j-1}) + k_{5j}l_{j-1} \\ u_j &= k_{1j}x_j + k_{2j}\dot{x}_j + k_{3j}x_{j-1} + k_{4j}\dot{x}_{j-1} + k_{5j}l_{j-1} + D_j \end{aligned} \quad (29)$$

where D_j represents an unknown disturbance term which sums up the additive faults and the noises terms that arrive through the networks or from the local faulty sensors as follows:

$$D_j = k_{1j}f_{s1j} + k_{1j}w_{1j} + k_{2j}f_{s2j} + k_{2j}w_{2j} + k_{3j}\delta i_{j-1} + k_{4j}\delta \dot{i}_{j-1} \quad (30)$$

If we implement the control input using measurements from other sensors, u_j will have the same form as in Equation (29) and only D_j will change.

Thus, the state-space equation with unknown disturbance term for the j th robot in a controlled robots platoon is:

$$\begin{aligned} \begin{bmatrix} \dot{x}_{1j} \\ \dot{x}_{2j} \\ \dot{x}_{3j} \end{bmatrix} &= \begin{bmatrix} 0 & 1 & 0 \\ k_{1j}/m_j & k_{2j}/m_j & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_{1j} \\ x_{2j} \\ x_{3j} \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ k_{5j}/m_j & k_{3j}/m_j & k_{4j}/m_j \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} l_{j-1} \\ x_{j-1} \\ \dot{x}_{j-1} \end{bmatrix} \\ &\quad + \begin{bmatrix} 0 & 0 & 0 & 0 \\ k_{3j}/m_j & k_{4j}/m_j & k_{1j}/m_j & k_{2j}/m_j \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \delta i_{j-1} \\ \delta \dot{i}_{j-1} \\ f_{s1j} + w_{1j} \\ f_{s2j} + w_{2j} \end{bmatrix} \\ \begin{bmatrix} y_{1j} \\ y_{2j} \\ y_{3j} \\ y_{4j} \end{bmatrix} &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{1j} \\ x_{2j} \\ x_{3j} \end{bmatrix} + \begin{bmatrix} f_{s1j} \\ f_{s1j} \\ f_{s2j} \\ f_{s3j} \end{bmatrix} + \begin{bmatrix} w_{1j} \\ \dot{w}_{1j} \\ w_{2j} \\ w_{3j} \end{bmatrix} \end{aligned} \quad (31)$$

4.3. Special Case: The Leader Robot

It is considered that the leader (first) robot moves with a prescribed controlled speed and it does not receive information from the other robots. In a similar manner, this leader is assumed to move along x axis and has a double integrator dynamic as follows:

$$m_1\ddot{x}_1 = u_1 \quad (32)$$

where $\ddot{x}_1 = \dot{v}_1 = a_1$ with x_1 is leader's position, v_1 is leader's velocity, a_1 is leader's acceleration, m_1 is leader's mass, and u_1 is the control input for this leader robot.

The needed control input for leader is calculated only from its sensor measurements without being affected by the states of the other robots. A proportional integrator (PI) velocity controller is assumed as it is shown in the Equation (33):

$$\begin{aligned} u_1 &= k_i \int_0^t (v_{ref} - v_1) d\tau + k_p(v_{ref} - v_1) \\ u_1 &= -k_i x_1 - k_p v_1 + u_c \\ u_c &= k_i \int_0^t v_{ref} d\tau + k_p v_{ref} \end{aligned} \quad (33)$$

where k_i is the integral control gain, k_p is the proportional control gain, u_c is the feed-forward control term, and v_{ref} is a predefined speed set-point for the leader robot.

The related state-space model of the leader is as seen in the Equation (34).

$$\begin{bmatrix} \dot{x}_{11} \\ \dot{x}_{21} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -k_i/m_1 & -k_p/m_1 \end{bmatrix} \begin{bmatrix} x_{11} \\ x_{21} \end{bmatrix} + \begin{bmatrix} 0 \\ 1/m_1 \end{bmatrix} u_c \quad (34)$$

where $x_{11} = x_1$ and $x_{21} = \dot{x}_1 = v_1$.

4.4. Model of the Leader Robot with Sensor Faults

The assumed measurement outputs for this leader robot are:

1. y_{11} is a GPS-based position measurement to obtain the x_{11} state.
2. y_{21} is a velocity measurement based on the same GPS sensor.
3. y_{31} is a velocity measurement by using the wheel-mounted velocity sensor on the robot to obtain the x_{21} state.

The control input of the leader only depends on its own sensor measurements. The unknown inputs, which act as disturbance for the faulty model of the leader robot, come from this leader's sensor fault. With this additive sensor fault, the leader's measurements become as shown in the Equation (35):

$$\begin{aligned} \mathbf{y}_1 &= C_1 \mathbf{x}_1 + \mathbf{f}_{s1} + \mathbf{w}_1 \\ C_1 &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \end{aligned} \quad (35)$$

where $\mathbf{y}_1 = [y_{11} \ y_{21} \ y_{31}]^T$ is the leader's measurement vector, $\mathbf{x}_1 = [x_{11} \ x_{21}]^T$ is the leader's state vector, $\mathbf{f}_{s1} = [f_{s11} \ \dot{f}_{s11} \ f_{s21}]^T$ is the leader's sensor faults vector which contains position sensor fault (f_{s11}), derivative of the position sensor fault (\dot{f}_{s11}), velocity sensor fault (f_{s21}), and $\mathbf{w}_1 = [w_{11} \ \dot{w}_{11} \ w_{21}]^T$ is the leader's measurement noises vector.

By taking into consideration the sensor faults, one possible way to compute the control input of the leader is as shown in the Equation (36):

$$\begin{aligned} u_1 &= -k_i y_{11} - k_p y_{31} + u_c \\ u_1 &= -k_i(x_{11} + f_{s11} + w_{11}) - k_p(x_{21} + f_{s21} + w_{21}) + u_c \\ u_1 &= -k_i x_{11} - k_p x_{21} + u_c + D_1 \end{aligned} \quad (36)$$

where D_1 contains the additive fault-induced disturbance and noise terms as follows:

$$D_1 = -k_i f_{s11} - k_i \dot{w}_{11} - k_p f_{s21} - k_p w_{21} \quad (37)$$

Thus, the state-space equation with sensor faults for the leader robot in a controlled robots platoon is as seen in Equation (38):

$$\begin{aligned} \begin{bmatrix} \dot{x}_{11} \\ \dot{x}_{21} \end{bmatrix} &= \begin{bmatrix} 0 & 1 \\ -k_i/m_1 & -k_p/m_1 \end{bmatrix} \begin{bmatrix} x_{11} \\ x_{21} \end{bmatrix} + \begin{bmatrix} 0 \\ 1/m_1 \end{bmatrix} u_c + \begin{bmatrix} 0 & 0 \\ -k_i/m_1 & -k_p/m_1 \end{bmatrix} \begin{bmatrix} f_{s11} + w_{11} \\ f_{s21} + w_{21} \end{bmatrix} \\ \begin{bmatrix} y_{11} \\ y_{21} \\ y_{31} \end{bmatrix} &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_{11} \\ x_{21} \end{bmatrix} + \begin{bmatrix} f_{s11} \\ \dot{f}_{s11} \\ f_{s21} \end{bmatrix} + \begin{bmatrix} w_{11} \\ \dot{w}_{11} \\ w_{21} \end{bmatrix} \end{aligned} \quad (38)$$

5. Sensor Faults Isolation in Controlled Robots Platoon

5.1. Sensor Faults Isolation in the Follower Robots

In the case of controlled robots platoon, the network disturbance yields from the position sensor fault measurement of the preceding robot which is transmitted to the robot behind it. For the j th robot, we can treat both the network disturbance from the preceding

$(j - 1)$ th robot and the effect of its own sensor faults on control as unknown input part \hat{d}_j . By comparing Equation (15) with Equation (31), we can write:

$$\begin{aligned}
 \hat{A}_j &= \begin{bmatrix} 0 & 1 & 0 \\ k_{1j}/m_j & k_{2j}/m_j & 0 \\ 0 & 1 & 0 \end{bmatrix} \\
 \hat{B}_j &= \begin{bmatrix} 0 & 0 & 0 \\ k_{5j}/m_j & k_{3j}/m_j & k_{4j}/m_j \\ 0 & 0 & -1 \end{bmatrix} \\
 \hat{E}_j &= \begin{bmatrix} 0 & 0 & 0 & 0 \\ k_{3j}/m_j & k_{4j}/m_j & k_{1j}/m_j & k_{2j}/m_j \\ 0 & 0 & 0 & 0 \end{bmatrix} \\
 x_j &= \begin{bmatrix} x_{1j} \\ x_{2j} \\ x_{3j} \end{bmatrix} \\
 \hat{u}_j &= \begin{bmatrix} l_{j-1} \\ x_{j-1} \\ \dot{x}_{j-1} \end{bmatrix} \\
 \hat{d}_j &= \begin{bmatrix} \delta i_{j-1} \\ \dot{\delta} i_{j-1} \\ f_{s1j} + w_{1j} \\ f_{s2j} + w_{2j} \end{bmatrix}
 \end{aligned} \tag{39}$$

The output matrix C_j is given by Equation (27). The C_j^1 , C_j^2 , and C_j^3 matrices, which correspond to the design of UIO for weak sensor faults isolation, are obtained in the same manner as in Equation (6) so that we get:

$$\begin{aligned}
 C_j^1 &= \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \\
 C_j^2 &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \\
 C_j^3 &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}
 \end{aligned} \tag{40}$$

One remark for UIO which is designed to be insensitive to a fault in S1 is that both the 1st and 2nd row of the C matrix is deleted to form C_j^1 because y_{2j} represents velocity which is obtained from the same sensor to obtain y_{1j} . Thus, GPS-based velocity measurement is not treated as an extra sensor that needs faults isolation.

By using each of those three new C_j matrices, three UIOs for each sensor faults isolation can be designed for each robot according to Equation (7) to produce residual signals r_{1j} , r_{2j} , and r_{3j} which are insensitive to a fault in S1, S2, and S3, respectively. Afterwards, these residual signals are compared to each threshold value th^i computed from Equation (19). When a fault occurs, the decision signal σ_i will rise based on that comparison results as shown in Equation (20). For example, if the residual signals from S1 (r_{1j}) and S3 (r_{3j}) are higher than the threshold values while a residual signal from S2 (r_{2j}) is lower, then decision signal for S2 (σ_2) will rise from zero to one indicating that a fault is happening in S2. The block diagram of this bank of UIOs along with its isolation logic is shown in Figure 2.

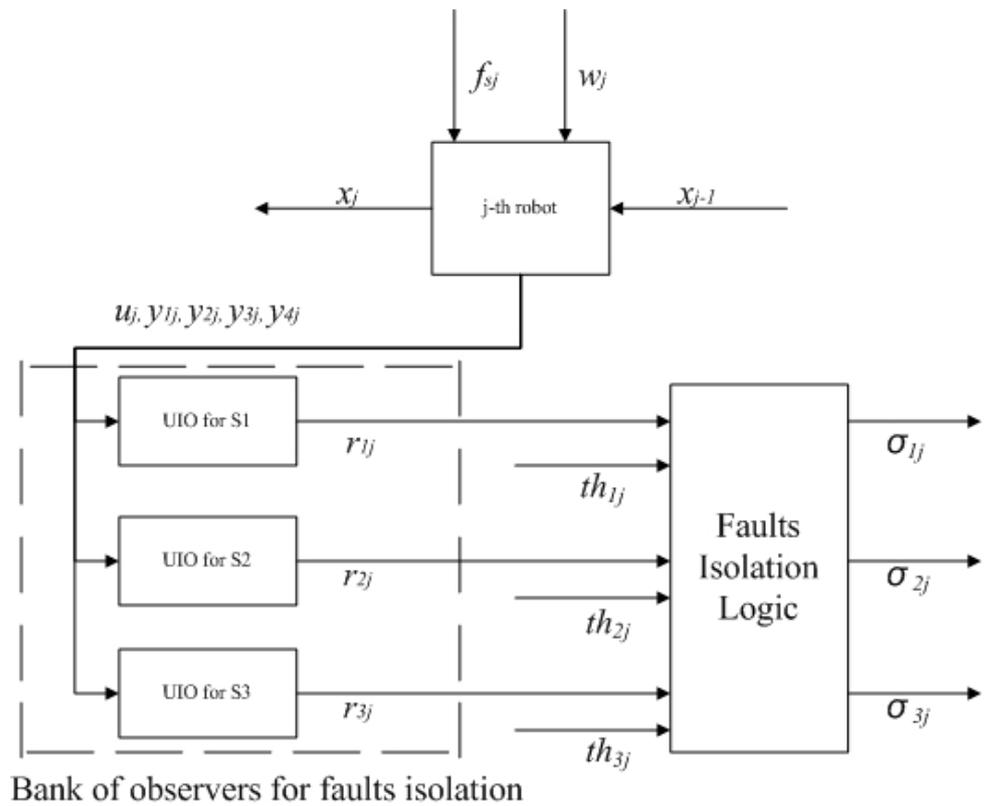


Figure 2. Local UIOs for sensor faults isolation on each robot.

5.2. Sensor Faults Isolation in the Leader Robot

Sensor faults isolation for the leader robot has a rather different scheme because its states are not affected by the states of the other robots. One more thing to be noted is that it only has redundancy in velocity measurement which comes from both the GPS-based sensor measurement (S1) and the wheel-mounted velocity sensor measurement (S2).

Because the leader robot has redundancy in the velocity measurement, we can directly use the method as in the case of the follower robots to design the UIO which is insensitive to fault in S2. By comparing Equation (15) with Equation (38), we use these matrices in constructing the UIO which produces r_{21} as follows:

$$\begin{aligned}
 \hat{A}_1^2 &= \begin{bmatrix} 0 & 1 \\ -k_i/m_1 & -k_p/m_1 \end{bmatrix} \\
 \hat{B}_1^2 &= \begin{bmatrix} 0 \\ 1/m_1 \end{bmatrix} \\
 \hat{C}_1^2 &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \\
 \hat{E}_1^2 &= \begin{bmatrix} 0 & 0 \\ -k_i/m_1 & -k_p/m_1 \end{bmatrix} \\
 \hat{x}_1^2 &= \begin{bmatrix} x_{11} \\ x_{21} \end{bmatrix} \\
 \hat{u}_1^2 &= u_c = k_i \int_0^t v_{ref} d\tau + k_p v_{ref} \\
 \hat{d}_1^2 &= \begin{bmatrix} f_{s11} + w_{11} \\ f_{s21} + w_{21} \end{bmatrix}
 \end{aligned} \tag{41}$$

As the position measurement of the leader does not have any redundancy, we refer to Equation (36) to design a residual generator for fault isolation in S1 by neglecting the

disturbance parts E_1^1 and D_1 so that the produced residual signal will respond to any sensor bias fault in S2. We can do that because we perform weak sensor faults isolation that assumes no simultaneous fault. Thus, we get:

$$\dot{x}_{21} = -\frac{k_p}{m_1}x_{21} - \frac{k_i}{m_1}x_{11} + \frac{1}{m_1}u_c \quad (42)$$

After that, by comparing the equation above with Equation (15), the matrices to design the UIO to produce residual signal r_{11} are:

$$\begin{aligned} \hat{A}_1^1 &= -k_p/m_1 \\ \hat{B}_1^1 &= [-k_i/m_1 \quad 1/m_1] \\ C_1^1 &= 1 \\ \hat{E}_1^1 &= 0 \\ x_1^1 &= x_{21} \\ \hat{u}_1^1 &= \begin{bmatrix} x_{11} \\ u_c \end{bmatrix} \\ u_c &= k_i \int_0^t v_{ref} d\tau + k_p v_{ref} \end{aligned} \quad (43)$$

6. Results and Discussion

6.1. Simulation Method

To verify the behaviour of the proposed model and the performance of the proposed weak sensor faults isolation method, we performed simulations using the MATLAB programming environment. In these simulations, the controlled robots platoon consists of five robots where one robot acts as the leader while the remaining four robots act as the followers. Each of them transmits its GPS-based position measurement to the robot behind it.

Firstly, the M-file feature in MATLAB is used to initialize the state-space parameters of both the leader and followers robots, to check the fulfilment of the UIO's existence requirement, and, if it exists, to compute the required parameters of the UIO's matrices for weak sensor faults isolation purpose. In this simulation, the parameters for the controlled leader robot in both Equations (41) and (43) are: $k_i = 0.05$, $k_p = 2$, and $m_1 = 1$ where the values of k_i and k_p are the PI controller gain values for leader's velocity which is determined to get fast settling time with small overshoot.

Meanwhile, the parameters for the remaining four of the follower robots in Equation (39) are obtained from Equation (25) in which $m_j = 1$, $l_{j-1} = 3$, $h = 2$, and $\lambda = 20$ refers to the adaptive cruise control examples in [30]. It yields that: $k_{1j} = -10$, $k_{2j} = -20.5$, $k_{3j} = 10$, $k_{4j} = 0.5$, and $k_{5j} = -10$.

Because each robot sends its GPS-based position measurement (S1) to the robot behind it, the interconnection matrix L in the Equation (10) is:

$$L = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

By using the new C matrices as in Equations (40), (41) and (43), the rank and detectability conditions necessary to UIO existence in Equation (4) are checked and it turns out that they are fulfilled.

Subsequently, the UIOs H_j^i , T_j^i , F_j^i , and K_j^i matrices are computed to design residual generators where i represents i th sensor and j represents j th robot. Hence, the UIO-based residual generators for each sensor in each robot as shown in Figure 2 are implemented

based on those matrices. The observer gains were determined using pole placement to satisfy all the equations in Equation (8) with poles: at -15 for the leader's sensor S1; at -15 and -10 for the leader's sensor S2; at -15 and -10 for both the follower's sensor S1 and sensor S3; and poles at -15 , -10 , and -5 for the follower's sensor S2.

After that, Equations (18) and (19) are used to calculate the residuals threshold by considering measurement noises such that $\|w^i\|_\infty = 0.35$ in the simulation. The infinity norm gain values of the residual generators are 1 for the followers and 2.6 for the leader. By observing the produced residual signals from the UIOs and comparing them with the threshold values, weak sensor faults isolation are done using the detection/isolation logic shown in Equation (20). Based on that relation, assuming a single sensor fault happens in a robot, the decision signal related to the malfunctioning sensor will arise.

Lastly, the Simulink feature in MATLAB is used to build and simulate the controlled robots platoon model with sensor faults and fault diagnostics. The computation results of UIOs are passed to the Simulink model which represents the residual generators.

6.2. Simulation Results

In this simulation, the leader's desired velocity was set to 10 m/s. For the non-faulty case, the system's response for every robot (position, velocity, and distance) and their residual signals (residual signal for S1, residual signal for S2, and residual signal for S3) are shown in Figures 3 and 4 respectively. It can be seen that all of them achieve the desired velocity of 10 m/s and have a distance of 22.4 m between each robot with a slower settling time consecutively. Furthermore, all residual signals show very small amplitudes below the threshold indicating that no false alarms are produced.

Meanwhile, for the faulty case, we simulate four different scenarios. In these scenarios, all of the sensor faults are simulated as sensor bias errors using a step function where the magnitude is increased from 0 to 10 at the 10th second.

In the first scenario, the fault is happening at sensor S1 in the 1st robot (leader) by which the system's response and UIO's residual signals are shown in Figures 5 and 6 respectively.

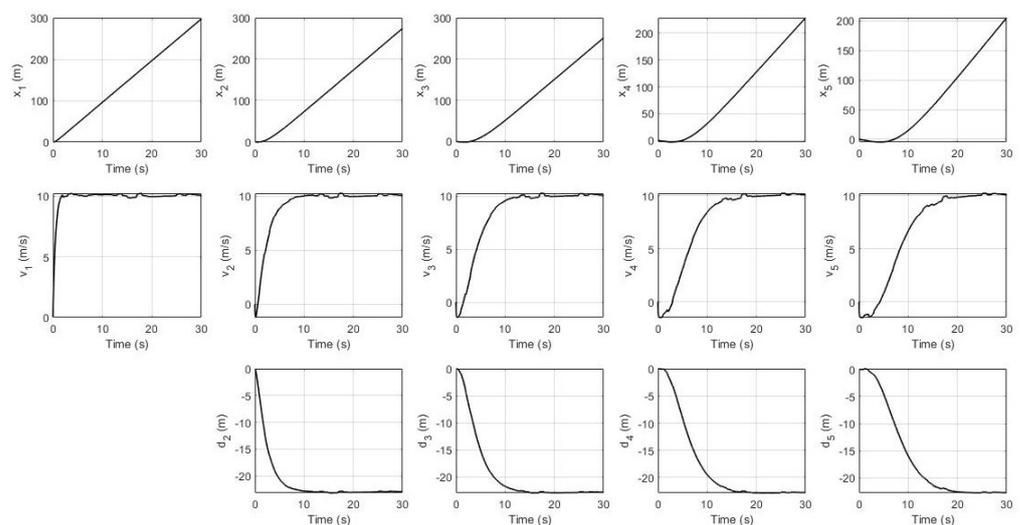


Figure 3. System's response without fault.

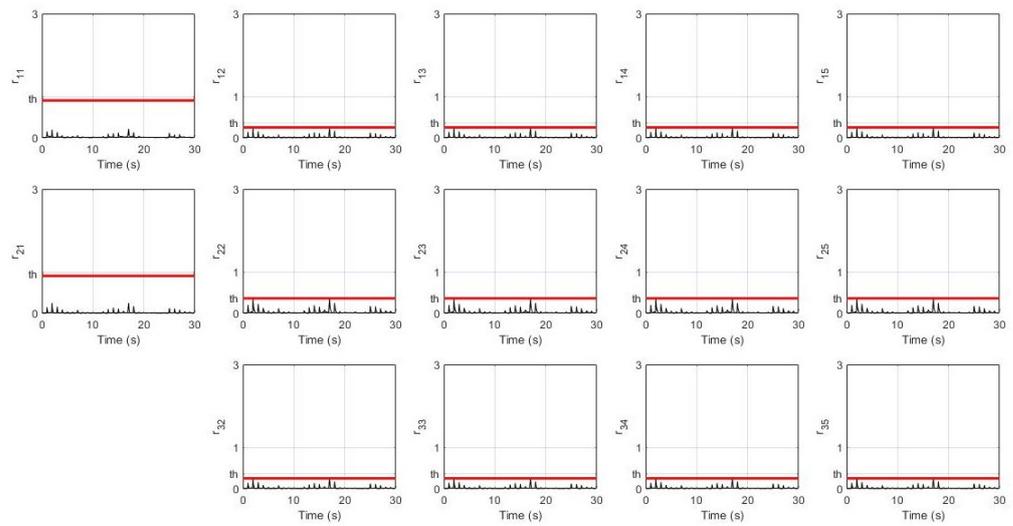


Figure 4. Residual signals without fault.

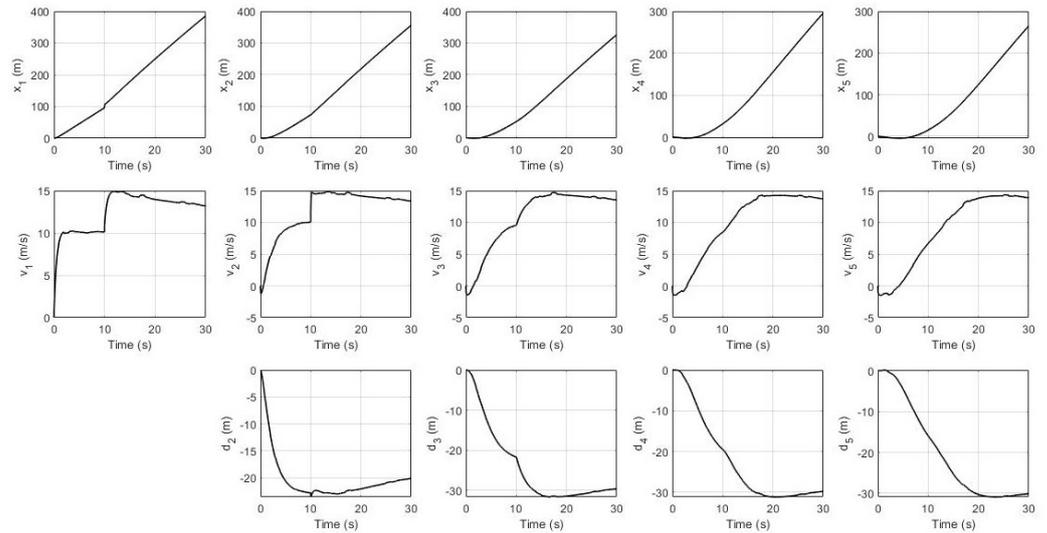


Figure 5. System's response with sensor S1 fault in the 1st robot.

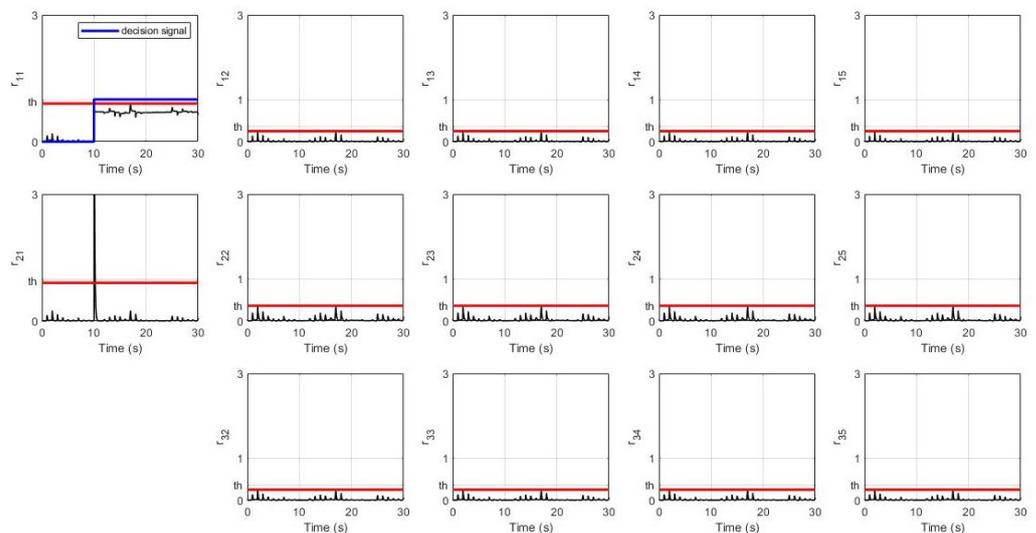


Figure 6. Residual signals with sensor S1 fault in the 1st robot.

In the second scenario, the fault is happening at sensor S1 in the 2nd robot by which the system's response and UIO's residual signals are shown in Figures 7 and 8 respectively.

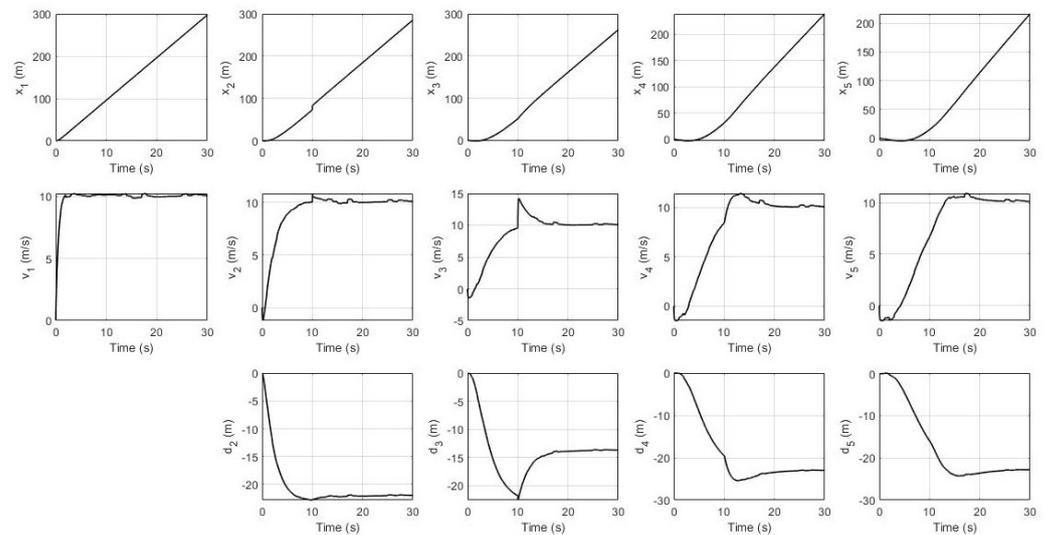


Figure 7. System's response with sensor S1 fault in the 2nd robot.

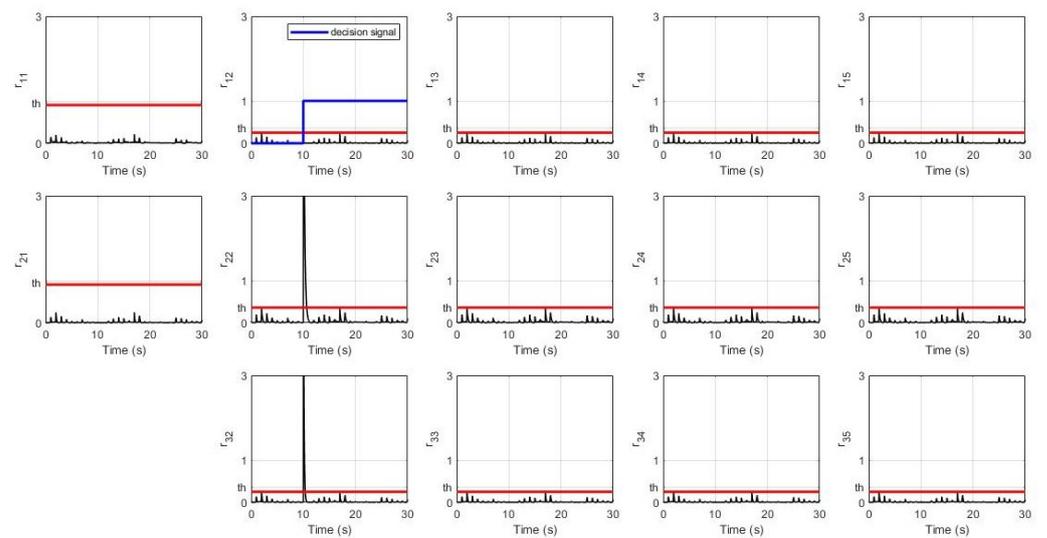


Figure 8. Residual signals with sensor S1 fault in the 2nd robot.

In the third scenario, the fault is happening at sensor S2 in the 3rd robot by which the system's response and UIO's residual signals are shown in Figures 9 and 10 respectively.

Finally, in the fourth scenario, the fault is happening at sensor S3 in the 5th robot by which the system's response and UIO's residual signals are shown in Figures 11 and 12 respectively.

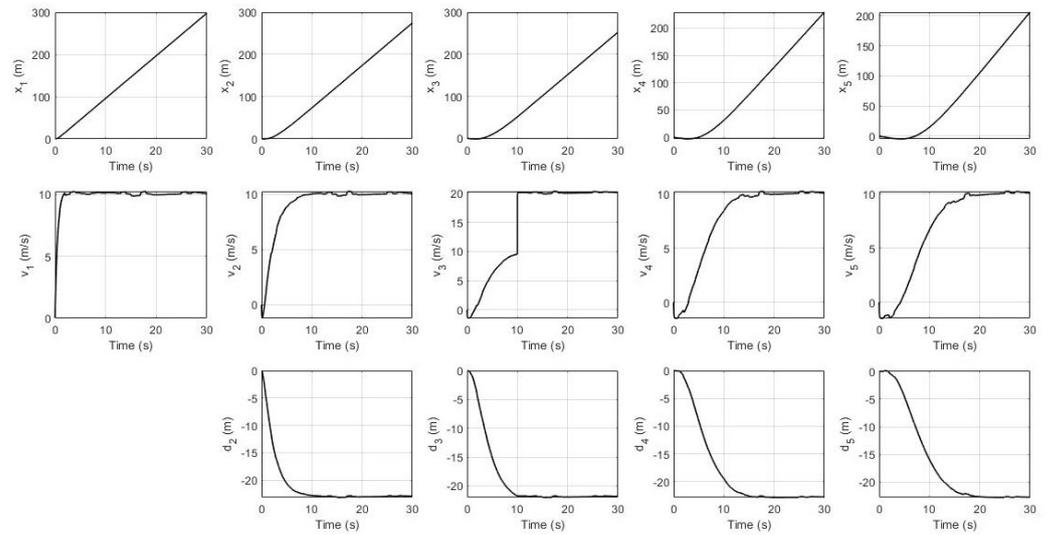


Figure 9. System's response with sensor S2 fault in the 3rd robot.

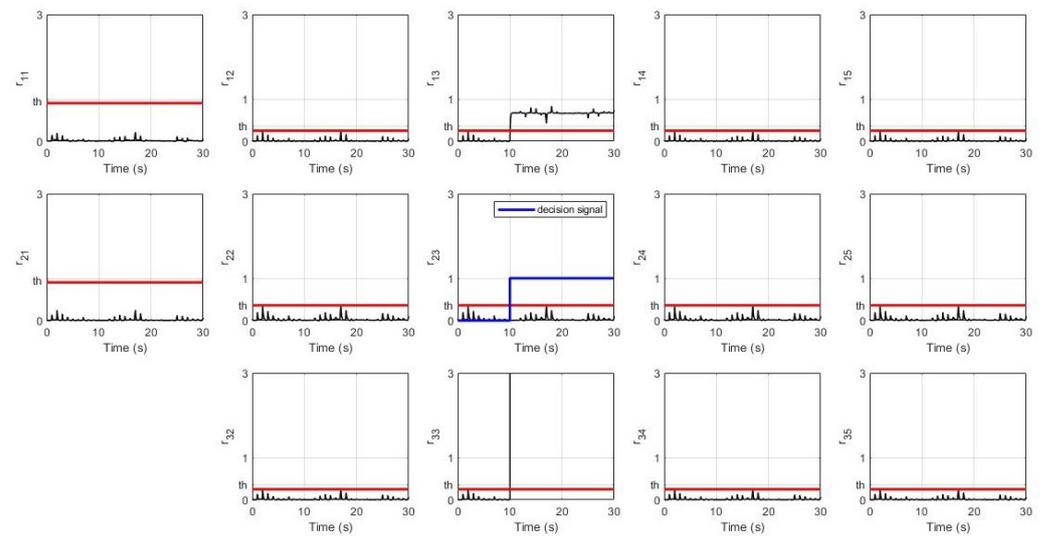


Figure 10. Residual signals with sensor S2 fault in the 3rd robot.

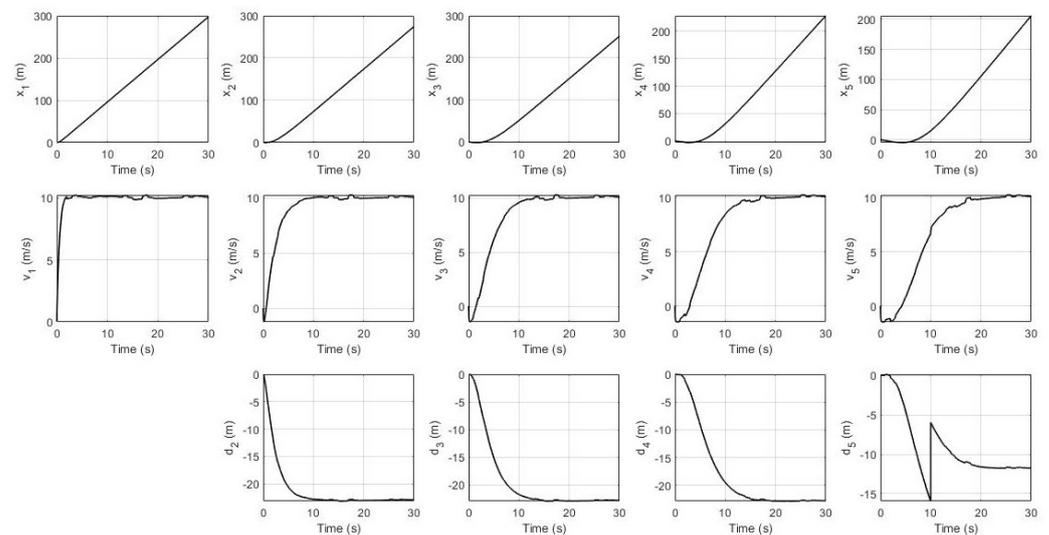


Figure 11. System's response with sensor S3 fault in the 5th robot.

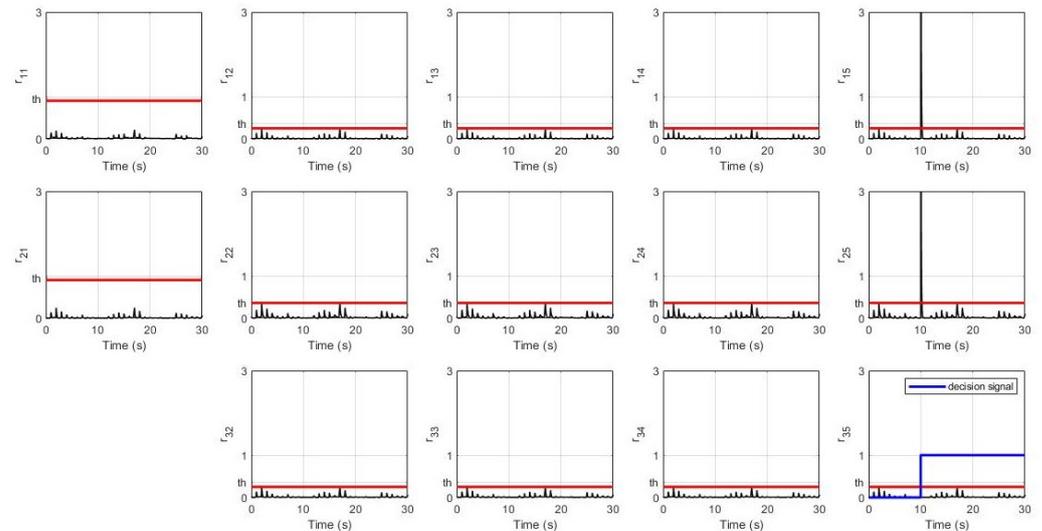


Figure 12. Residual signals with sensor S3 fault in the 5th robot.

For the first scenario, Figure 5 shows that a fault in S1 in the 1st robot is affecting all the states (position, velocity, and distance) of the 2nd, 3rd, 4th, and 5th robots. This is happening because, as mentioned earlier, the transmitted information in these networked control systems comes from this GPS-based position measurement (S1). Besides that, Figure 6 shows a significant change of amplitude in the residual signals produced by the UIO for sensor S2 while the other residual for sensor S1 is below the threshold value. Thus, based on the previous relation shown in the Equation (20), a decision signal for sensor S1 arises indicating that a fault occurred in this sensor at the 10th second.

For the second scenario, Figure 7 shows a similar situation as in the first scenario where a fault in S1 in the 2nd robot is affecting all the states of the 3rd, 4th, and 5th robots. In terms of sensor faults isolation, Figure 8 shows a significant change of amplitude in the residual signals for both sensor S2 and sensor S3 exceeding the threshold value while residual for sensor S1 is not. Hence, the decision signal for S1 arises revealing an occurrence of a fault in this sensor of the 2nd robot at the 10th second.

For the third scenario, Figure 9 shows that fault in S2 in the 3rd robot does not directly affect the states of the 4th and 5th robots because the measurement of this sensor (S2) is not the information that is transmitted to the robot behind it. In addition, Figure 10 shows a change of amplitude exceeding the threshold in both the residual signals from this robot's UIO for sensor S1 and sensor S3 triggering a decision signal to indicate that sensor S2 is faulty at the 10th second.

Lastly, for the fourth scenario, Figure 11 shows that fault in S3 in the 5th robot does not directly affect any other robots at all because this 5th robot is the last in these networked control systems. Furthermore, Figure 12 shows a significant change of amplitude in this last robot's UIO residual signal for both sensor S1 and sensor S2 triggering a decision signal to indicate that sensor S3 is faulty at the 10th second.

6.3. Discussion

The simulation results presented in the previous section show that the proposed faults isolation method ensures fast response time, the interval between the rise of the fault signal and the rise of the decision signal is less than 5 ms in each case. Moreover, those decision signals are always correct and accurate in determining where the fault is happening. In the case of the robot platoons, the UIO existence conditions are fulfilled.

Comparing with other similar works, this UIO-based approach shows its popularity as a favourite choice in the field of fault detection and isolation in the multi-agents system. Liu et al. implemented multiple UIOs in each agent for fault detection and isolation in the multi-agents system that is robust to disturbance [27]. While their method prioritizes the

detection of which agents are failed, our approach is more refined in terms of it can detect a specific sensor fault in each agent. Other works by Shames et al. also used UIO to detect a faulty agent in a power network [28]. Yet, they also did not aim to detect a specific sensor fault in an agent.

The research object of Zhang et al. is the most similar to ours compared with the others. They derived a global augmented system model included with actuator and sensor faults and then used it to design a global UIO able to undertake faults estimation processes [29]. Their proposed faults isolation logic was a centralized one. The used case study was a simulation of five aircraft in a leader-follower network scheme. Even though we are lacking in actuator faults feature compared to them, our approach of local distributed UIOs is more scalable.

7. Conclusions

The extension of model-based sensor fault isolation schemes to networked control systems is essential to deal with fault effect propagation through the interconnections. In this research, we proposed a networked control system model that can be used to describe the expected behaviour of the robotics platoon with sensor faults and their propagation through the network. Distributed Unknown Input Observers (UIO) for weak sensor fault isolation in each robot are also designed using this model as the basis. In designing these UIOs, the fault propagation is treated as the unknown input part. We have also introduced a threshold computation method for UIO-based sensor fault isolation in the presence of measurement noises.

The verification and validation are done in MATLAB using a case study where five subsystems (robots) are connected with a communication-based connection. In these five robots, one robot acts as the leader which has two sensors while the remaining four robots act as the followers by which each of them has three sensors. The simulation results show that the weak sensor faults isolation in the robotics platoon is successfully achieved by observing the produced residual signals from the UIOs in each robot.

Some limitations of the UIO-based faults isolation approach in networked control systems are related to the assumptions, see Equation (4), that the subsystem models have to be satisfied to decouple the propagation of the faults through the network. Besides that, the detection and isolation logic depends on a threshold value determined by the maximum value of measurements noise. Thus, if the measurements noise has excessive values, the threshold value will also be high so that low magnitude fault can not be detected.

In many communication networks, the delay can not be neglected. According to the previous works, if the delay is small enough (under 250 ms), then the model of the system with small delay can be approximated with a model without delay [34]. Hence, if the delay in the communication network between the subsystems is small, we can implement our fault isolation approach based on the approximated model.

As the proposed model for faults isolation has low complexity, this approach has a great potential to be implemented on embedded systems. The bank of UIOs that perform the faults isolation can be discretized using standard methods. Furthermore, the approach utilizes only conventional sensors which are available on most mobile robots.

Possible future works include exploring this approach to perform actuator fault isolation in networked control systems. Another possible extension is to take into consideration the communication time delay in the form of distributed delay among the subsystems.

Author Contributions: Conceptualization and methodology, W.K. and L.M.; software and validation, W.K.; formal analysis, W.K. and L.M.; investigation, W.K. and L.M.; writing—original draft preparation, W.K.; writing—review and editing, W.K. and L.M.; supervision, L.M. All authors have read and agreed to the published version of the manuscript.

Funding: This study was supported by the National Research, Development and Innovation Fund of Hungary, funded under the K_19 funding scheme, Project No. 131501.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors thank Katalin Hangos from University of Pannonia, Veszprem, Hungary, for her useful comments and remarks that have improved the quality of this research.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Mohamed, N.; Al-Jaroodi, J.; Jawhar, I. A review of middleware for networked robots. *Int. J. Comput. Sci. Netw. Secur.* **2009**, *9*, 139–148.
2. Roveda, L.; Spahiu, B.; Terkaj, W. On the Proposal of a Unified Safety Framework for Industry 4.0 Multi-Robot Scenario. In Proceedings of the SEBD, Grosseto, Italy, 16–19 June 2019.
3. Gonzalez-Ruiz, A.; Ghaffarkhah, A.; Mostofi, Y. A comprehensive overview and characterization of wireless channels for networked robotic and control systems. *J. Robot.* **2011**, *2011*, 340372. [[CrossRef](#)]
4. Fehér, Á.; Márton, L. Transient Behaviour Analysis and Design for Platoons with Communication Delay. *IFAC-PapersOnLine* **2019**, *52*, 13–18. [[CrossRef](#)]
5. Blanke, M.; Kinnaert, M.; Lunze, J.; Staroswiecki, M.; Schröder, J. *Diagnosis and Fault-Tolerant Control*; Springer: Berlin/Heidelberg, Germany, 2006; Volume 2.
6. Varga, A. Solving fault diagnosis problems. *Studies in Systems, Decision and Control*, 1st ed.; Springer: Berlin/Heidelberg, Germany, 2017.
7. Pózna, A.I.; Fodor, A.; Hangos, K.M. Model-based fault detection and isolation of non-technical losses in electrical networks. *Math. Comput. Model. Dyn. Syst.* **2019**, *25*, 397–428. [[CrossRef](#)]
8. Gertler, J. Fault detection and isolation using parity relations. *Control Eng. Pract.* **1997**, *5*, 653–661. [[CrossRef](#)]
9. Song, T.; Collins, E.G., Jr. Robust H estimation with application to robust fault detection. *J. Guid. Control Dyn.* **2000**, *23*, 1067–1071. [[CrossRef](#)]
10. Niemann, H.; Saberi, A.; Stoorvogel, A.A.; Sannuti, P. Optimal fault estimation. *IFAC Proc. Vol.* **2000**, *33*, 267–272. [[CrossRef](#)]
11. Menke, T.E.; Maybeck, P.S. Sensor/actuator failure detection in the Vista F-16 by multiple model adaptive estimation. *IEEE Trans. Aerosp. Electron. Syst.* **1995**, *31*, 1218–1229. [[CrossRef](#)]
12. White, N.A.; Maybeck, P.S.; DeVilbiss, S.L. MMAE detection of interference/jamming and spoofing in a DGPS-aided inertial system. In Proceedings of the 11th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GPS 1998), Nashville, TN, USA, 15–18 September 1998; pp. 905–914.
13. Simani, S.; Fantuzzi, C.; Patton, R.J. Model-based fault diagnosis techniques. In *Model-based Fault Diagnosis in Dynamic Systems Using Identification Techniques*; Springer: Berlin/Heidelberg, Germany, 2003; pp. 19–60.
14. Michail, K.; Deliparaschos, K.M.; Tzafestas, S.G.; Zolotas, A.C. AI-based actuator/sensor fault detection with low computational cost for industrial applications. *IEEE Trans. Control Syst. Technol.* **2015**, *24*, 293–301. [[CrossRef](#)]
15. Nasiri, S.; Khosravani, M.R.; Weinberg, K. Fracture mechanics and mechanical fault detection by artificial intelligence methods: A review. *Eng. Fail. Anal.* **2017**, *81*, 270–293. [[CrossRef](#)]
16. Xiao, S.; Yao, J.; Chen, Y.; Li, D.; Zhang, F.; Wu, Y. Fault Detection and Isolation Methods in Subsea Observation Networks. *Sensors* **2020**, *20*, 5273. [[CrossRef](#)] [[PubMed](#)]
17. Li, D.; Wang, Y.; Wang, J.; Wang, C.; Duan, Y. Recent advances in sensor fault diagnosis: A review. *Sensors Actuators A Phys.* **2020**, *309*, 111990. [[CrossRef](#)]
18. Chen, J.; Patton, R.J. *Robust Model-Based Fault Diagnosis for Dynamic Systems*; Springer: Berlin/Heidelberg, Germany, 2012; Volume 3.
19. Chakrabarty, A.; Corless, M.J.; Buzzard, G.T.; Žak, S.H.; Rundell, A.E. State and Unknown Input Observers for nonlinear systems with bounded exogenous inputs. *IEEE Trans. Autom. Control* **2017**, *62*, 5497–5510. [[CrossRef](#)]
20. Chakrabarty, A.; Fridman, E.; Žak, S.H.; Buzzard, G.T. State and Unknown Input Observers for nonlinear systems with delayed measurements. *Automatica* **2018**, *95*, 246–253. [[CrossRef](#)]
21. Xu, F.; Tan, J.; Wang, X.; Puig, V.; Liang, B.; Yuan, B. A novel design of Unknown Input Observers using set-theoretic methods for robust fault detection. In Proceedings of the 2016 American Control Conference (ACC), Boston, MA, USA, 6–8 July 2016; pp. 5957–5961.
22. Zuo, L.; Yao, L.; Kang, Y. UIO Based Sensor Fault Diagnosis and Compensation for Quadrotor UAV. In Proceedings of the 2020 Chinese Control And Decision Conference (CCDC), Hefei, China, 21–23 May 2020; pp. 4052–4057.
23. Taha, A.F.; Elmahdi, A.; Panchal, J.H.; Sun, D. Unknown input observer design and analysis for networked control systems. *Int. J. Control* **2015**, *88*, 920–934. [[CrossRef](#)]
24. Shames, I.; Teixeira, A.M.; Sandberg, H.; Johansson, K.H. Distributed fault detection for interconnected second-order systems. *Automatica* **2011**, *47*, 2757–2764. [[CrossRef](#)]
25. Chen, W.; Saif, M. Unknown input observer design for a class of nonlinear systems: An LMI approach. In Proceedings of the 2006 American Control Conference, Minneapolis, MN, USA, 14–16 June 2006; pp. 834–838.

26. Chakrabarty, A.; Sundaram, S.; Corless, M.J.; Buzzard, G.T.; Žak, S.H.; Rundell, A.E. Distributed Unknown Input Observers for interconnected nonlinear systems. In Proceedings of the 2016 American Control Conference (ACC), Boston, MA, USA, 6–8 July 2016; pp. 101–106.
27. Liu, X.; Gao, X.; Han, J. Robust Unknown Input Observer based fault detection for high-order multi-agent systems with disturbances. *ISA Trans.* **2016**, *61*, 15–28. [[CrossRef](#)] [[PubMed](#)]
28. Shames, I.; Teixeira, A.M.; Sandberg, H.; Johansson, K.H. Distributed fault detection and isolation with imprecise network models. In Proceedings of the 2012 American Control Conference (ACC), Montreal, QC, Canada, 27–29 June 2012; pp. 5906–5911.
29. Zhang, K.; Liu, G.; Jiang, B. Robust Unknown Input Observer-based fault estimation of leader–follower linear multi-agent systems. *Circuits Syst. Signal Process.* **2017**, *36*, 525–542. [[CrossRef](#)]
30. Rajamani, R. *Vehicle Dynamics and Control*; Springer: Berlin/Heidelberg, Germany, 2011.
31. Chen, C. Linear System Theory and Design. Available online: http://ebadi.profcms.um.ac.ir/imagesm/474/stories/modern_control/chen%20c%20-t%20linear%20system%20theory%20and%20design.pdf (accessed on 18 August 2021).
32. Xiao, L.; Gao, F. A comprehensive review of the development of adaptive cruise control systems. *Veh. Syst. Dyn.* **2010**, *48*, 1167–1192. [[CrossRef](#)]
33. Serrano, L.; Kim, D.; Langley, R.B.; Itani, K.; Ueno, M. A GPS velocity sensor: How accurate can it be? . . . A first look. In Proceedings of the 2004 National Technical Meeting of The Institute of Navigation, Long Beach, CA, USA, 16–19 May 2004; pp. 875–885.
34. Fehér, Á.; Márton, L.; Pituk, M. Approximation of a linear autonomous differential equation with small delay. *Symmetry* **2019**, *11*, 1299. [[CrossRef](#)]