electronics

MDPI

Article

Facilitating Autonomous Systems with AI-Based Fault Tolerance and Computational Resource Economy

Kyriakos M. Deliparaschos [1,2,*,†], Konstantinos Michail [1,3,†] and Argyrios C. Zolotas [2,†]

[1] Electrical and Computer Engineering and Informatics, Cyprus University of Technology, 3036 Limassol, Cyprus; kon_michael@ieee.org

[2] Centre for Autonomous and Cyber-Physical Systems, SATM, Cranfield University, Bedford MK43 0AL, UK; a.zolotas@cranfield.ac.uk

[3] SignalGeneriX, 23C Gregory Afxentiou str., P.O. Box 59627, 4011 Limassol, Cyprus

* Correspondence: k.deliparaschos@cranfield.ac.uk or k.deliparaschos@cut.ac.cy

† The authors contributed equally to this work.

Received: 14 April 2020; Accepted: 1 May 2020; Published: 11 May 2020

Abstract: Proposed is the facilitation of fault-tolerant capability in autonomous systems with particular consideration of low computational complexity and system interface devices (sensor/actuator) performance. Traditionally model-based fault-tolerant/detection units for multiple sensor faults in automation require a bank of estimators, normally Kalman-based ones. An AI-based control framework enabling low computational power fault tolerance is presented. Contrary to the bank-of-estimators approach, the proposed framework exhibits a single unit for multiple actuator/sensor fault detection. The efficacy of the proposed scheme is shown via rigorous analysis for several sensor fault scenarios for an electro-magnetic suspension testbed.

Keywords: fault tolerance; reconfigurable control; Maglev; neural networks; artificial intelligence

1. Introduction

Modern control systems require careful, reliable and economic design with maximum performance, normally imposing several design trade-offs (economic design, reliability, performance). In particular, reliability in control systems is vital especially in safety-critical systems (i.e., where faults must be accommodated before the impaired system becomes unstable). In non-safety-critical system cases, like production lines, reliability supports a normal operation regime avoiding production delays and/or unnecessary maintenance. In areas more aligned to autonomy, such as in Unmanned Area Vehicles (UAVs), the problem of considering control methods for reliability adds to the computational power of the already limited resources [1–6].

Autonomous systems must be trustworthy, and trustworthiness has been a popular topic of discussion in the current autonomous systems literature [7,8]. Reliability facilitates trustworthiness, and in this context fault accommodation can be achieved with a priori design of a controller that has the ability to take remedial actions so that the stability of the control system is maintained even with degraded performance. The stability and performance of a control system depends upon the healthy operation of its interfaces (actuators and sensors) and various approaches to design such capability appear in the literature (both model-based and model-free methods) [9–17]. Fault-tolerant control (FTC) supports reliability [18], the approach normally classified as either Passive (PFTC) or Active (AFTC) [19]. Passive FTC type requires a prior knowledge of the faults, while the Active type (used in this work) does not necessitate such knowledge of the fault rather a Fault Detection and Isolation (FDI) mechanism with reconfigurable control. Reconfigurable FTC control has gained significant attention

over recent years given the demand on reliable system design [20–22] and in the area of cyber-physical systems [23].

Referring to sensor fault tolerance, especially after sensors failure, a few methods exist that use the information from the remaining healthy sensors, in order to reconstruct the lost signal of the faulty ones [24]. The latter methods include, use of a bank of Neural Networks (NNs) or use of Kalman Estimators (KE) [1,25]. Both approaches are worth considering when aiming in avoiding sensor redundancy. In contrast to the KEs approach, NNs have increased False Alarm Rates (FAR), mainly because they leave a very small residual after fault estimation [24]. Despite that, they are widely used since they can be designed without having precise knowledge of the model of the system under test [26–29]. In the above approaches, where many actuators/sensors exist, an (in-parallel) bank of estimators for multiple faults detection is employed [30]. For example, if there is one actuator with n_y sensors, then the number of sensor fault combinations that could happen is $2^{n_y} - 1$ (where n_y is the total number of sensors, assuming that not all sensors can fail). Hence, to be able to detect those faults it requires the same number of estimators. However, this increases the complexity of the control design and requires additional computational resources, since the estimators must work in parallel.

NNs have been used to a great extend in many engineering fields including control systems [31,32], as well as for Fault Detection (FD) methods in FTC systems [33,34] and more specifically in Sensor FDI [24,35,36]. We proposed an AI-based FD mechanism, referred to as *i*FD, based on the use of Neural Network approach that performs a similar task to the conventional bank-of-estimators FD albeit offers substantially reduced computational complexity. Visually this is depicted in Figure 1, the bank of the estimators running in parallel (dotted lines). The authors, in their brief paper [19], presented the original concept framework from an automatic fault-tolerant control viewpoint. This paper considerably extends the original results (with a particular emphasis on their interpretation) and present how reliable system autonomy can be facilitated.

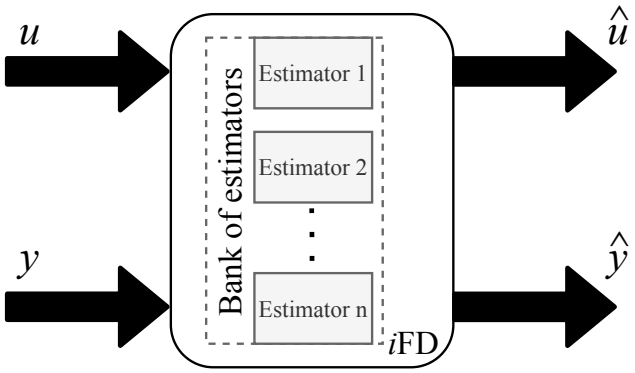


Figure 1. Bank of estimators (dotted lines) vs. proposed *i*FD (bold lines) for actuators/sensors estimates.

We discuss the framework solution with the help of a practical example, i.e., an Electro-Magnetic Suspension (EMS) testbed (typically forms the suspension platform of Maglev train) to support the vehicle compartment and maintain acceptable passenger ride quality. The rationale behind this choice being that EMS (Maglev) is an inherently unstable, non-linear, safety-critical system, subject to non-trivial control performance and reliability requirements (hence offering a challenging application for the validation of the work). Sensor faults are first modelled and then the simulation results using various fault scenarios showcase the proposed method.

The rest of the paper is organized as follows: Section 2 describes the proposed *i*FD approach including a short description of the NN training, and Section 3.1 shows the efficacy of the proposed method based on the analysis of the results (with the help of the practical example of the EMS—the test case details been discussed in Appendix A). Conclusions are discussed in Section 4.

2. Proposed Fault Detection Scheme

We employ the FD unit to detect actuator/sensor faults and activate controller reconfiguration. The proposed FD scheme is NN-based and the concept is depicted in Figure 2. Industrial systems typically exhibit Multiple-Input, Multiple Outputs. Control inputs relate to actuation (i.e., industrial drives, motors, pumps, electro-magnets etc.), in a control setting indicated by variable $\mathcal{U}$. Outputs measure several useful parameters (both for control purposes and monitoring purposes), typically indicated by variable $\mathcal{Y}$. Control design satisfies a desired performance relevant to the application.

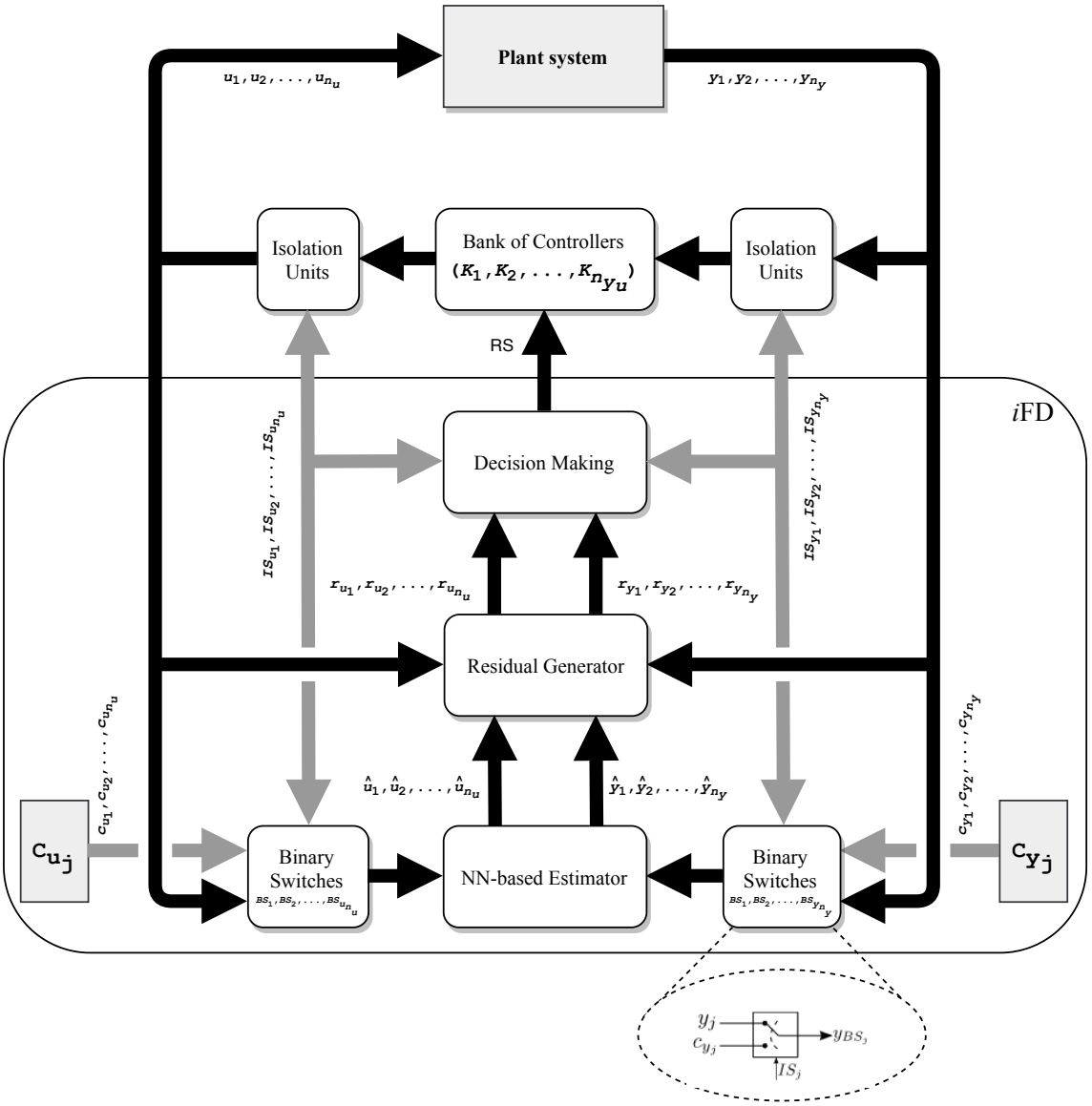


Figure 2. FDI general diagram with proposed *i*FD and including the binary switch operator.

When one or more actuators and/or sensors are impaired, those signals are distorted leading to performance degradation or possibly instability of the closed loop. The sets of actuators and sensors are defined as $\mathcal{U} = [u_1, u_2, \ldots u_{n_u}]$ and $\mathcal{Y} = [y_1, y_2, \ldots y_{n_y}]$, where u_j is the jth actuator, y_j is the jth sensor, n_u is the total number of actuators and n_y is the total number of sensors.

The control loop features a bank of controllers $[K_1, K_2, \ldots K_{n_{yu}}]$ and two isolation units for isolating faulty actuator and sensor signals when these happen. The *i*FD mechanism is employed to detect the faults and is comprised of a NN-based estimator, a Residual Generator (RG) and a Decision Mechanism (DM). The NN estimator is trained in such a way that the actuators and sensors signals are estimated

and fed into the RG which in its turn compares the real signals with the estimated ones. Immediately after the RG has completed the comparison, it advances the residuals to the DM, leading to a decision whether a component is faulty or not.

The key point in the proposed *i*FD method is the estimator training approach to observe $\mathcal{U}$ and $\mathcal{Y}$ (the training process discussed in Section 2.1).

The inputs to the estimator are obtained from the Binary Switches (BS) depicted in Figure 2. The BS have three inputs; one represents the real measured values of the $\mathcal{U}$ and $\mathcal{Y}$ and the other comes from the functions $\mathcal{C}_{u_j}$ and $\mathcal{C}_{y_j}$, defined as $\mathcal{C}_{u_j} = [c_{u_1}, c_{u_2} \dots c_{u_{n_u}}]$ and $\mathcal{C}_{y_j} = [c_{y_1}, c_{y_2} \dots c_{y_{n_y}}]$. $\mathcal{C}_{u_j}$ and $\mathcal{C}_{y_j}$ represent two arrays that contains predefined functions, used during the training and operation of the *i*FD. Typically, calculating these values is benefited by designer experience as they tend to be application dependent. The third input (IS_{y_j}) is a binary input which controls the switching operation between the inputs e.g., from y_1 to c_{y_1}. The output y_{BS_j} of the BS is given by (1)

$$y_{BS_j} = \begin{cases} y_j, & if \quad IS_{y_j}=1 \\ c_{y_j}, & if \quad IS_{y_j}=0 \end{cases} \tag{1}$$

The residual generation is a vital task although is out of scope in this paper. The moving average filter defined in (2), manages to accommodate the noise coming from the sensors to reduce the FAR [37].

$$r_{y_j} = \sum_{j-(N-1)}^{j} \frac{(y_j - \hat{y}_j)^2}{N} \tag{2}$$

where r_{y_j} is the residual, y_j and $\hat{y}_j$ are the jth real and estimated signals (for the actuators the y is replaced by u), and N is the total number of past samples.

The DM decides whether one or more actuating and/or sensing components are impaired or not by comparing the relative residual r_j with a predefined threshold (engineer must define the threshold). Threshold selection is a non-trivial task to perform, since it impacts both fault detection sensitivity of the DM and the FAR. A very sensitive DM to faults (thresholds too low) causes increase of FAR, while a less sensitive DM (thresholds too high) may cause instability due to delayed reconfiguration. Threshold selection is normally done as trial-and-error process with designer experience in the application beneficial. Within the AI remit of fault estimation, threshold selection has received attention [38] but the details of this aspect are beyond the scope of this paper. In the proposed *i*FD, the Reconfiguration Signal (RS) at the output of DM enables controller reconfiguration. The proposed *i*FD setup works as follows:

- Given normal operation, the actuator(s)/sensors signals are estimated and then fed into the residual generator. The generator calculates the residual (which would correspond to a very small value under normal operation) for each actuator/sensor and feeds it to the DM. The DM outputs the IS_j and RS signals.
- With one or more actuator(s)/sensors failing the corresponding residuals r_j increase and the DM will detect the change by comparison with the relevant threshold levels. Next, the ISs will switch in order to activate the corresponding BS to modify its output to c_j; while the RS will feed the appropriate data value to enable controller reconfiguration. The BSs will also isolate the signals of the faulty devices from the estimator so that the latter 'sees' some 'known' data based on its training/knowledge (this is described in the next section).
- Finally, the isolation units will remove faulty devices from the loop and the controller reconfigures to maintain performance and stability.

2.1. Offline Training of the iFD Unit: Obtaining the Learning Set

As in all AI-based solutions, training of the algorithm is the key point. The NN *i*FD unit is trained based on data accumulated from an extensive set of scenaria on subsets of the main sensor set, $\mathcal{Y}$.

The collected data are then packed in a structure shown on Table 1. The first column presents the sensor set number from 1 to n_{yn}, defined as,

$$n_{yu} = 2^{(n_u+n_y)} - 2^{n_u} - 2^{n_y} + 1 \tag{3}$$

The second column shows the status of the sensor set, i.e., all possible sensor/actuator fault scenarios are covered, and the next two columns show the measured sensors and actuators signals. The last two columns reflect the estimated sensors and actuators signals. The data set $\mathcal{D}$ with dimensions $d_r \times d_c$ is given by,

$$d_r = n_{yu} \times k \tag{4}$$

$$d_c = n_y + n_u + n_{\hat{y}} + n_{\hat{u}} \tag{5}$$

where $n_{\hat{u}}$ and $n_{\hat{y}}$ are the number of estimated signals for actuators and sensors, respectively. $\mathcal{D}$ is constructed with data from numerical simulations with each sensor/actuator fault scenario. Where the sensor(s) and/or actuator(s) is (are) assumed to be faulty, a known function c_{u_j}, c_{y_j} replaces the k data points. In an automatic setup the design engineer is required to select a set of functions $\mathcal{C}_{u_j}$ and $\mathcal{C}_{y_j}$ to replace the non-predicted outputs from the faulty actuators and sensors respectively. In an autonomous setting, experience from automation and use of reinforced learning design will enable the aforementioned choice (this part is studied as future research and is not the main aspect of the work presented here).

When an actuator and/or sensor fault occurs, the corresponding function c_{u_j} and/or c_{y_j} is/are connected to the *i*FD. This is a result of the *i*FD learning capability to respond to sensor/actuator faults in a way that the *i*FD unit itself continually checks for faults on the full sensor set and its subsets.

An electro-magnetic suspension (EMS) system testbed (details in Appendix A) is used as the practical platform to illustrate the proposed solution.

2.2. Visualizing the iFD Applied on the EMS Example

The EMS system is excited by a single control input (i.e., $n_u = 1$), $\mathcal{U} = \{u_c\}$, and provides four output measurements (i.e., $n_y = 4$), namely $\mathcal{Y} = \{i, (z_t - z), \dot{z}, \ddot{z}\}$. The outputs are employed for controller design. The $\mathcal{H}_\infty$ Loop-Shaping Design Procedure (LSDP) robust control approach was used to enable the necessary closed-loop performance. Please note that the airgap $(z_t - z)$ is required by default as a standard input [39]. With $(z_t - z)$ a default measurement, the total number of actuator/sensor sets are $n_{yn} = 8$. Hence, eight LSDP controllers (i.e., $K_{(z_t-z)}$, $K_{i,(z_t-z)}$, $K_{(z_t-z),\dot{z}}$, $K_{(z_t-z),\ddot{z}}$, ..., etc.) are used to cover the spectrum of (seven) possible sensor combinations that could occur as indicated in Figure 3. Details on the design of LSDP controllers is discussed in [40]. The *i*FD concept presented here can be considered for the detection and isolation of sensor faults in a typical FTC system arrangement for other engineering test platforms. In the case of one or more sensors failures, the relevant fault is detected and isolated (note that the isolation switches and the binary switches are merged together as shown in the figure) and the system switches to the alternative controller $K_\bullet$ (for closed-loop purposes).

Table 1. Structure of the data used for the *i*FD training (basis from [19]).

Actuator/ Sensor Set Number	Actuator/ Sensor Status	Measured										Estimated									
		y_1	y_2	y_3	…	y_{n_y}	u_1	u_2	u_3	…	u_{n_u}	$\hat{y}_1$	$\hat{y}_2$	$\hat{y}_3$	…	$\hat{y}_{n_{\hat{y}}}$	$\hat{u}_1$	$\hat{u}_2$	$\hat{u}_3$	…	$\hat{u}_{n_{\hat{u}}}$
1	healthy set	$D^1_{y^1_1}$	$D^1_{y^1_2}$	$D^1_{y^1_3}$	…	$D^1_{y^1_{ny}}$	$D^1_{u^1_1}$	$D^1_{u^1_2}$	$D^1_{u^1_3}$	…	$D^1_{u^1_{nu}}$	$D^1_{\hat{y}^1_1}$	$D^1_{\hat{y}^1_2}$	$D^1_{\hat{y}^1_3}$	…	$D^1_{\hat{y}^1_{n\hat{y}}}$	$D^1_{\hat{u}^1_1}$	$D^1_{\hat{u}^1_2}$	$D^1_{\hat{u}^1_3}$	…	$D^1_{\hat{u}^1_{n\hat{u}}}$
	$\mathcal{Y} = y_1, y_2 \ldots y_{n_y}$	⋮	⋮	⋮		⋮	⋮	⋮	⋮		⋮	⋮	⋮	⋮		⋮	⋮	⋮	⋮		⋮
	$\mathcal{U} = u_1, u_2, \ldots u_{n_u}$	$D^1_{y^k_1}$	$D^1_{y^k_2}$	$D^1_{y^k_3}$	…	$D^1_{y^k_{ny}}$	$D^1_{u^k_1}$	$D^1_{u^k_2}$	$D^1_{u^k_3}$	…	$D^1_{u^k_{nu}}$	$D^1_{\hat{y}^k_1}$	$D^1_{\hat{y}^k_2}$	$D^1_{\hat{y}^k_3}$	…	$D^1_{\hat{y}^k_{n\hat{y}}}$	$D^1_{\hat{u}^k_1}$	$D^1_{\hat{u}^k_2}$	$D^1_{\hat{u}^k_3}$	…	$D^1_{\hat{u}^k_{n\hat{u}}}$
2	Faulty	$c^2_{y^1_1}$	$D^2_{y^1_2}$	$D^2_{y^1_3}$	…	$D^2_{y^1_{ny}}$	$D^2_{u^1_1}$	$D^2_{u^1_2}$	$D^2_{u^1_3}$	…	$D^2_{u^1_{nu}}$	$c^2_{\hat{y}^1_1}$	$D^2_{\hat{y}^1_2}$	$D^2_{\hat{y}^1_3}$	…	$D^2_{\hat{y}^1_{n\hat{y}}}$	$D^2_{\hat{u}^1_1}$	$D^2_{\hat{u}^1_2}$	$D^2_{\hat{u}^1_3}$	…	$D^2_{\hat{u}^1_{n\hat{u}}}$
	y_1	⋮	⋮	⋮		⋮	⋮	⋮	⋮		⋮	⋮	⋮	⋮		⋮	⋮	⋮	⋮		⋮
		$c^2_{y^k_1}$	$D^2_{y^k_2}$	$D^2_{y^k_3}$	…	$D^2_{y^k_{ny}}$	$D^2_{u^k_1}$	$D^2_{u^k_2}$	$D^2_{u^k_3}$	…	$D^2_{u^k_{nu}}$	$c^2_{\hat{y}^k_1}$	$D^2_{\hat{y}^k_2}$	$D^2_{\hat{y}^k_3}$	…	$D^2_{\hat{y}^k_{n\hat{y}}}$	$D^2_{\hat{u}^k_1}$	$D^2_{\hat{u}^k_2}$	$D^2_{\hat{u}^k_3}$	…	$D^2_{\hat{u}^k_{n\hat{u}}}$
3	Faulty	$c^3_{y^1_1}$	$c^3_{y^1_2}$	$D^3_{y^1_3}$	…	$D^3_{y^1_{ny}}$	$D^3_{u^1_1}$	$D^3_{u^1_2}$	$D^3_{u^1_3}$	…	$D^3_{u^1_{nu}}$	$c^3_{\hat{y}^1_1}$	$c^3_{\hat{y}^1_2}$	$D^3_{\hat{y}^1_3}$	…	$D^3_{\hat{y}^1_{n\hat{y}}}$	$D^3_{\hat{u}^1_1}$	$D^3_{\hat{u}^1_2}$	$D^3_{\hat{u}^1_3}$	…	$D^3_{\hat{u}^1_{n\hat{u}}}$
	y_1, y_2	⋮	⋮	⋮		⋮	⋮	⋮	⋮		⋮	⋮	⋮	⋮		⋮	⋮	⋮	⋮		⋮
		$c^3_{y^k_1}$	$c^3_{y^k_2}$	$D^3_{y^k_3}$	…	$D^3_{y^k_{ny}}$	$D^3_{u^k_1}$	$D^3_{u^k_2}$	$D^3_{u^k_3}$	…	$D^3_{u^k_{nu}}$	$c^3_{\hat{y}^k_1}$	$c^3_{\hat{y}^k_2}$	$D^3_{\hat{y}^k_3}$	…	$D^3_{\hat{y}^k_{n\hat{y}}}$	$D^3_{\hat{u}^k_1}$	$D^3_{\hat{u}^k_2}$	$D^3_{\hat{u}^k_3}$	…	$D^3_{\hat{u}^k_{n\hat{u}}}$
4	Faulty	$c^4_{y^1_1}$	$c^4_{y^1_2}$	$D^4_{y^1_3}$	…	$D^4_{y^1_{ny}}$	$c^4_{u^1_1}$	$D^4_{u^1_2}$	$D^4_{u^1_3}$	…	$D^4_{u^1_{nu}}$	$c^4_{\hat{y}^1_1}$	$c^4_{\hat{y}^1_2}$	$D^4_{\hat{y}^1_3}$	…	$D^4_{\hat{y}^1_{n\hat{y}}}$	$c^4_{\hat{u}^1_1}$	$D^4_{\hat{u}^1_2}$	$D^4_{\hat{u}^1_3}$	…	$D^4_{\hat{u}^1_{n\hat{u}}}$
	y_1, y_2, u_1	⋮	⋮	⋮		⋮	⋮	⋮	⋮		⋮	⋮	⋮	⋮		⋮	⋮	⋮	⋮		⋮
		$c^4_{y^k_1}$	$c^4_{y^k_2}$	$D^4_{y^k_3}$	…	$D^4_{y^k_{ny}}$	$c^4_{u^k_1}$	$D^4_{u^k_2}$	$D^4_{u^k_3}$	…	$D^4_{u^k_{nu}}$	$c^4_{\hat{y}^k_1}$	$c^4_{\hat{y}^k_2}$	$D^4_{\hat{y}^k_3}$	…	$D^4_{\hat{y}^k_{n\hat{y}}}$	$c^4_{\hat{u}^k_1}$	$D^4_{\hat{u}^k_2}$	$D^4_{\hat{u}^k_3}$	…	$D^4_{\hat{u}^k_{n\hat{u}}}$
5	Faulty	$c^5_{y^1_1}$	$D^5_{y^1_2}$	$D^5_{y^1_3}$	…	$D^5_{y^1_{ny}}$	$c^5_{u^1_1}$	$c^5_{u^1_2}$	$D^5_{u^1_3}$	…	$D^5_{u^1_{nu}}$	$c^5_{\hat{y}^1_1}$	$D^5_{\hat{y}^1_2}$	$D^5_{\hat{y}^1_3}$	…	$D^5_{\hat{y}^1_{n\hat{y}}}$	$c^5_{\hat{u}^1_1}$	$c^5_{\hat{u}^1_2}$	$D^5_{\hat{u}^1_3}$	…	$D^5_{\hat{u}^1_{n\hat{u}}}$
	y_1, u_1, u_2	⋮	⋮	⋮		⋮	⋮	⋮	⋮		⋮	⋮	⋮	⋮		⋮	⋮	⋮	⋮		⋮
		$c^5_{y^k_1}$	$D^5_{y^k_2}$	$D^5_{y^k_3}$	…	$D^5_{y^k_{ny}}$	$c^5_{u^k_1}$	$c^5_{u^k_2}$	$D^5_{u^k_3}$	…	$D^5_{u^k_{nu}}$	$c^5_{\hat{y}^k_1}$	$D^5_{\hat{y}^k_2}$	$D^5_{\hat{y}^k_3}$	…	$D^5_{\hat{y}^k_{n\hat{y}}}$	$c^5_{\hat{u}^k_1}$	$c^5_{\hat{u}^k_2}$	$D^5_{\hat{u}^k_3}$	…	$D^5_{\hat{u}^k_{n\hat{u}}}$
⋮	⋮	⋮	⋮	⋮		⋮	⋮	⋮	⋮		⋮	⋮	⋮	⋮		⋮	⋮	⋮	⋮		⋮
n_{yu}	Faulty	$c^{n_{yu}}_{y^1_1}$	$c^{n_{yu}}_{y^1_2}$	…	$c^{n_{yu}}_{y^1_{ny-1}}$	$D^{n_{yu}}_{y^1_{ny}}$	$c^{n_{yu}}_{u^1_1}$	$c^{n_{yu}}_{u^1_2}$	…	$c^{n_{yu}}_{u^1_{nu-1}}$	$D^{n_{yu}}_{u^1_{nu}}$	$c^{n_{yu}}_{\hat{y}^1_1}$	$c^{n_{yu}}_{\hat{y}^1_2}$	…	$c^{n_{yu}}_{\hat{y}^1_{ny-1}}$	$D^{n_{yu}}_{\hat{y}^1_{n\hat{y}}}$	$c^{n_{yu}}_{\hat{u}^1_1}$	$c^{n_{yu}}_{\hat{u}^1_2}$	…	$c^{n_{yu}}_{\hat{u}^1_{nu-1}}$	$D^{n_{yu}}_{\hat{u}^1_{n\hat{u}}}$
	$y_1 \ldots y_{n_y-1}$	⋮	⋮		⋮	⋮	⋮	⋮		⋮	⋮	⋮	⋮		⋮	⋮	⋮	⋮		⋮	⋮
	$u_1, \ldots u_{n_u-1}$	$c^{n_{yu}}_{y^k_1}$	$c^{n_{yu}}_{y^k_2}$	…	$c^{n_{yu}}_{y^k_{ny-1}}$	$D^{n_{yu}}_{y^k_{ny}}$	$c^{n_{yu}}_{u^k_1}$	$c^{n_{yu}}_{u^k_2}$	…	$c^{n_{yu}}_{u^k_{nu-1}}$	$D^{n_{yu}}_{y^k_{nu}}$	$c^{n_{yu}}_{\hat{y}^k_1}$	$c^{n_{yu}}_{\hat{y}^k_2}$	…	$c^{n_{yu}}_{\hat{y}^k_{ny-1}}$	$D^{n_{yu}}_{\hat{y}^k_{n\hat{y}}}$	$c^{n_{yu}}_{\hat{u}^k_1}$	$c^{n_{yu}}_{\hat{u}^k_2}$	…	$c^{n_{yu}}_{\hat{u}^k_{nu-1}}$	$D^{n_{yu}}_{\hat{y}^k_{n\hat{u}}}$

k is the total number of samples at each actuator/sensor set.

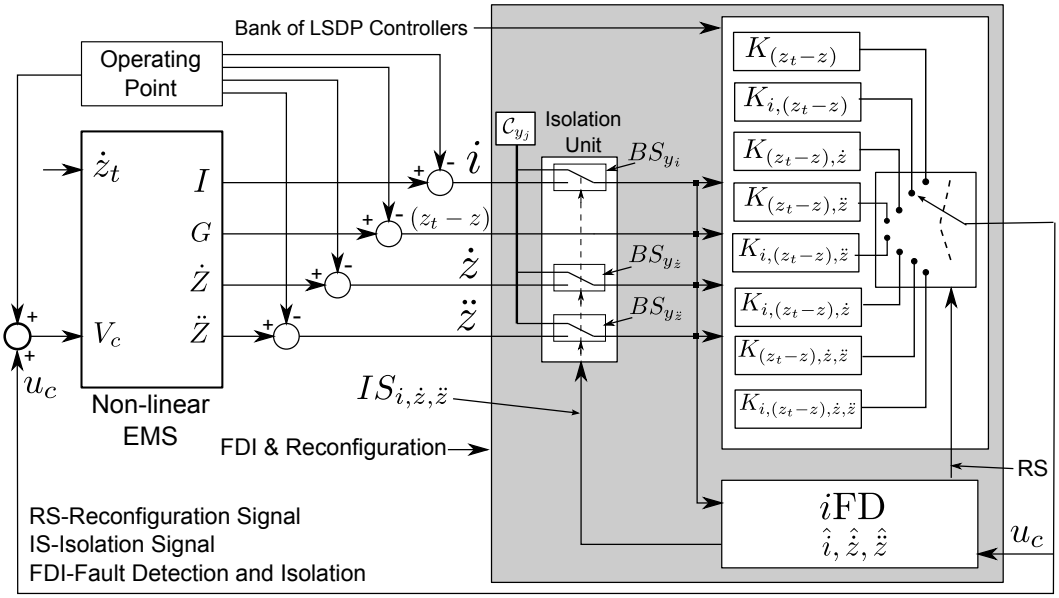
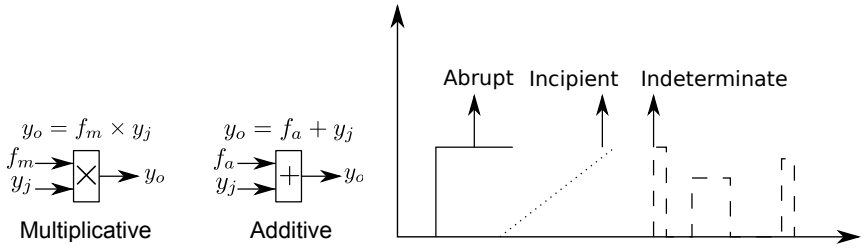


Figure 3. The proposed *i*FD solution using the EMS application example (basis from [19]).

Sensor faults can be categorized into additive and multiplicative categories. As clearly shown in Figure 4, there exist three types of faults in each category, namely: (i) abrupt or step-type fault, (ii) incipient or soft fault and (iii) indeterminate fault. The faults accounted for in this work fall into the additive and multiplicative categories and are both abrupt and incipient types of faults. In the first case, the output of the sensor is added or multiplied with a function f_a and f_m respectively. Indeterminate faults could also belong to any combinations of the aforementioned faults, but they are assumed to occur in random width time windows and amplitudes. These faults are treated like permanent by the *i*FD, and they are 'captured' once they appear.



(**a**) Fault categories. (**b**) Fault types.

Figure 4. Sensor fault categories (**a**) and types (**b**) (basis from [19]).

2.3. Neural Network Algorithm

A plethora of NN algorithm types exist in the research literature; however in this paper a dynamic non-linear input-output Neural Network model with tapped delay lines at the input is employed for time-series prediction. The NN's task is to perform similar to that of the—more conventional—bank of Kalman estimators (KE) in the feedback loop, as well as to predict future values based on past values of one or multiple time-series. In particular, to predict $\phi(t)$ series based on n_c past values of $\theta(t)$ series so that $\phi(t) = \lambda(\theta(t-1), \ldots, \theta(t-n_c))$.

This type of NN algorithm is adapted to fit the inputs and targets of the suspension for the *i*FD realization. It has a total of five inputs (u_c and $i, (z_t - z), \dot{z}, \ddot{z}$) and three estimated outputs ($\hat{i}, \hat{\dot{z}}, \hat{\ddot{z}}$). Its internal architecture is shown in Figure 5, and is realized as a hidden layer (with one delay and

20 hidden neurons) and an output layer with sigmoid and linear functions, respectively. Each neuron's output is generally described by (6),

$$o_n = \sum_{j=1}^{n_n} \Delta_j w_j + \psi \tag{6}$$

where o_n is the neuron's output, n_n is the number of neuron's inputs, w are the weights (is a vector equal to the size of the neuron's inputs), Δ are the delay lines and ψ is the bias point (considered to be one for all neurons).

A fast convergence method for training moderate sized feed-forward neural networks is the Levenberg-Marquardt backpropagation algorithm (for details see [41,42] (Chapters 11–12)).

The training data that were (later used to train the NN), were successively collected in equal time windows T for all failing set combinations with a sampling time τ_s, on an online working state model with the $\mathcal{H}_\infty$ LSDP controllers in the feedback loop. The stopping criteria was set to a Mean Square Error (MSE $\leq 10^{-5}$) or a maximum of 1000 epochs.

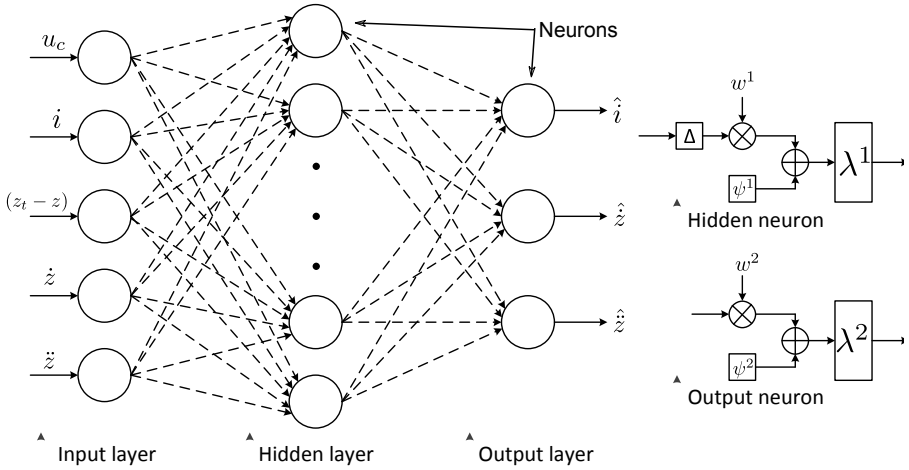


Figure 5. The Neural network architecture for the *i*FD as applied on the EMS.

3. Results Discussion

3.1. Efficacy and Assessment of the Proposed iFD

A rigorous analysis follows next, to show the effectiveness of the proposed *i*FD unit with results from realistic simulations (MATLAB/Simulink) on the EMS system.

An offline training with a typical NN-based estimator, described in Section 2.3, was performed prior to the FTC design of the suspension. The total training process (1000 epochs) required c.7 min on a mobile workstation laptop (equipped with Intel® Core™ i7-9750H @ 2.6 GHz, 32 GB RAM), with the parameters described in the previous section. The total number of sensor and actuators, their estimated signals and the number of actuator/sensor sets are, $n_y = 4$, $n_u = 1$, $n_{\hat{y}} = 3$, $n_{\hat{u}} = 1$ and $n_{yu} = 8$ respectively.

The training data from each sensor set comprising $\mathcal{D}$, were collected using sample rate $\tau_s = 1$ kHz under total simulation time $T = 6.6$ s. The data set $\mathcal{D}$ consists of data subsets $\mathcal{D}_d$ and $\mathcal{D}_s$, drawn from the deterministic and stochastic suspension responses (Subscripts d and s indicate the deterministic and stochastic cases respectively). The data set used for training is as follows,

$$\mathcal{D}^{d_r \times d_c} = \begin{bmatrix} \mathcal{D}_d^{d_{r_d} \times d_{c_d}} \\ \mathcal{D}_s^{d_{r_s} \times d_{c_s}} \end{bmatrix} \tag{7}$$

where $d_r = d_{r_d} + d_{r_s}$ and $d_c = d_{c_d} = d_{c_s}$ are both found from (4). The total number of columns is calculated as $d_c = 9$, while the total samples per set is $k = 6.6 \times 1000$, hence $d_{r_d} = d_{r_s} = 52{,}800$ and

$d_r = 105{,}600$. The functions used for the training of the NN are $\mathcal{C}_{y_j} = \{c_i = 0, c_{\dot{z}} = 0, c_{\ddot{z}} = 0\}$ with dimensions $k \times 3$. These functions are both used in the stochastic and deterministic responses of the suspension where a fault is assumed, as explained in Section 2.1.

Overall, there are 4 sensors in the full sensor set ($\mathcal{Y}$), while it is taken for granted that the actuator, u_c, and the airgap sensor $(z_t - z)$ cannot fail (in case where fault tolerance is required for the airgap then redundant components can be used under a voting scheme). Three sensor fault possibilities are considered: the current i, the vertical velocity $\dot{z}$ and the acceleration $\ddot{z}$. Sensor faults are generally classified into abrupt and incipient fault types. We also use additive and multiplicative faults with bias for each one of the sensors $i, \dot{z}$ and $\ddot{z}$. Figure 6 illustrates current sensor measurement, i, fault profile (similar pattern fault profiles are used for other sensors in this work). For all cases, faults start developing at time $t_f = 1$ s (this is marked at point A in all relevant figures).

Figure 6a for the impaired sensor, illustrates the normal current value superimposed with a low frequency band-limited random signal, $\nu(t)$, at frequencies of 10 rad/s and zero mean, white noise characteristics and power spectral density of $S_i = 5$ limited to $\omega_i = 1.6$ Hz. Consequently, the additive/abrupt fault profile for the current sensor (f_{aa_i}) is given by,

$$f_{aa_i}(t) = \begin{cases} \nu(t) \; if & t_f \leq t < \infty \\ 0 \quad if & t < t_f \end{cases} \tag{8}$$

Sensor output $\dot{z}$ and $\ddot{z}$ follow a similar pattern (same bandwidth) and $S_{\dot{z}} = 0.03$ and $S_{\ddot{z}} = 2$ respectively. Next, Figure 6b depicts the multiplicative/abrupt case (f_{ma_i}) where the current sensor is suddenly damaged at $t = 1$ s and as a result its output becomes five times larger than normal (9).

$$f_{ma_i}(t) = \begin{cases} 5 \quad if & t_f \leq t < \infty \\ 1 \quad if & 0 \leq t < t_f \end{cases} \tag{9}$$

The same fault profile is used for the other two sensors, $\dot{z}$ and $\ddot{z}$. For the current measurement, a bias (abrupt) type of failure is shown in Figure 6e. Clearly the sensor output abruptly increases (in a step manner) to its maximum value, i.e., in this case max $y_{o_i} = 10$ A.

The incipient types of faults are illustrated in Figure 6c and Figure 6d respectively. In the former figure the additive/incipient fault on the current measurement (f_{ai_i}) are described by (10). The latter is a ramp type signal with σ_i slope superimposed with a low frequency random signal with band-limited white noise characteristics as previously explained. The aforementioned figure, depicts the multiplicative/incipient (f_{mi_i}) fault described by (11), where the fault starts developing at $t_f = 1$ s and then falls to zero (due to multiplication by zero).

$$f_{ai_i}(t) = \begin{cases} \sigma_i(t - t_f) + \nu(t) & if \quad t_f \leq t < \infty \\ 0 & if \quad 0 < t < t_f \end{cases} \tag{10}$$

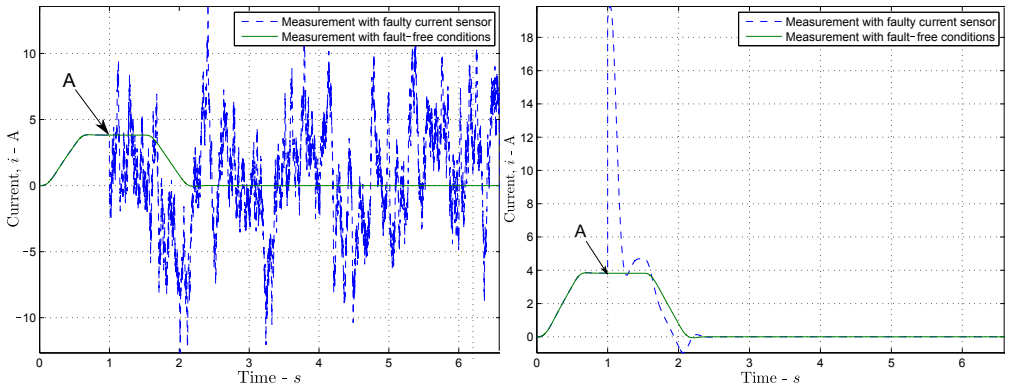
$$f_{mi_i}(t) = \begin{cases} \sigma_i(t - t_f) + 1 + \nu(t) & if \quad t_f \leq t < \infty \\ 1 & if \quad 0 < t < t_f \end{cases} \tag{11}$$

The fault profiles for $\dot{z}$ and $\ddot{z}$ follow the same behavior as above with effective slopes, $\sigma_i = 20$, $\sigma_{\dot{z}} = 6$ and $\sigma_{\ddot{z}} = 20$ and for the PSD, $S_i = 0.5$, $S_{\dot{z}} = 0.03$ and $S_{\ddot{z}} = 0.5$ respectively. The following scenario supports explaining the *i*FD working principle:

- Three sensors are subsequently impaired with a time difference as follows: accelerometer at 0.5 s, velocity at 1.5 s and current at 2 s),
- the deterministic disturbance to the suspension is used and,
- a multiplicative/abrupt fault profile is injected for each sensor at each time instant mentioned above (e.g., for the current sensor see Figure 6b).

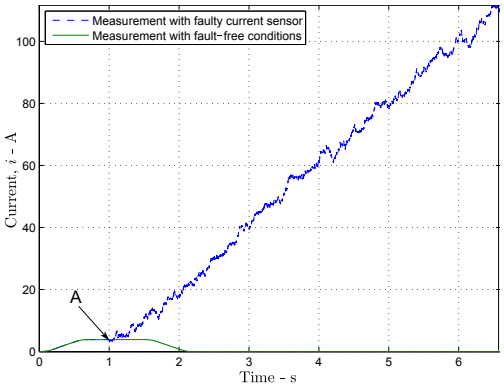
The airgap sensor output with fault-free case and with the aforementioned fault scenario is depicted in Figure 7. The figure illustrates the airgap with a fault-free case (i.e., healthy sensor set, $\mathcal{Y}$, with $K_{i,(z_t - z),\dot{z},\ddot{z}}$) and under the fault scenario mentioned. The acceleration sensor is impaired at 0.5 s (point A) and immediately after a controller reconfiguration follows (a new controller, $K_{i,(z_t - z),\dot{z}}$, is introduced

in the loop) in order to maintain the stability and performance of the EMS. The EMS response with both fault-free and fault scenario comply with the control performance requirements described in Appendix A. Following the acceleration fault, the velocity one fails at $t = 1.5$ s (designated at point B) and the current sensor follows at $t = 2$ s (marked at point C). The subsequent faults are successfully detected and accommodated via appropriate switching on $K_{i,(z_t - z)}$ and $K_{(z_t - z)}$ respectively.

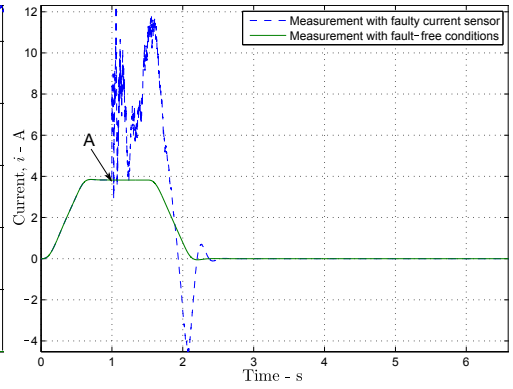


(**a**) Additive/abrupt fault profile, f_{aa_i}.

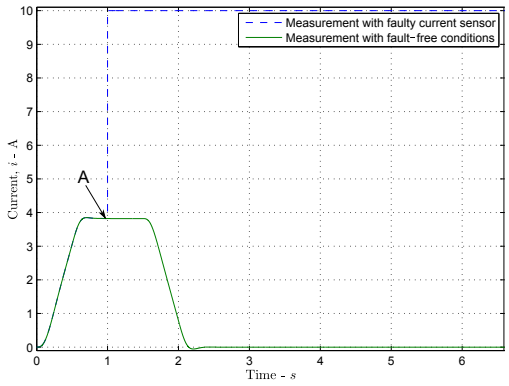
(**b**) Multiplicative/abrupt fault profile, f_{ma_i}.



(**c**) Additive/incipient profile, f_{ai_i}.



(**d**) Multiplicative/incipient fault profile, f_{mi_i}.



(**e**) Bias/abrupt fault profile.

Figure 6. Coil's current sensor, i, fault profiles.

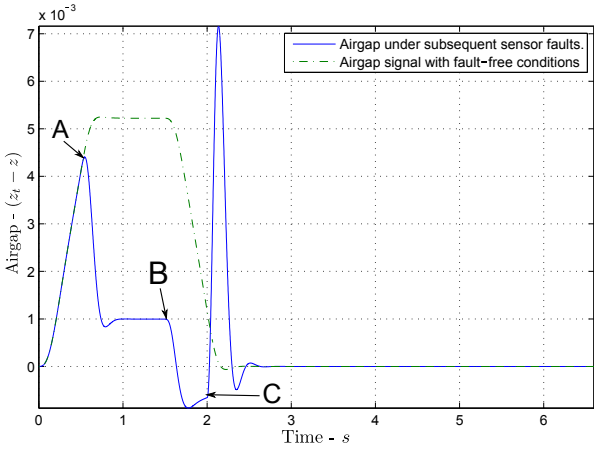


Figure 7. Airgap sensor signal with fault-free and with the fault scenario, *id*: 8 of Table 2.

The sensor fault accommodation for each sensor failure is integrated in three steps. To assist in explaining the steps of the procedure, the current sensor fault will be interpreted: (i) *Sensor FD*: when the fault occurs at $t = 1$ s, the residual of the current measurement, r_{y_i} starts increasing and as soon as it passes the threshold (see Figure 8) the fault is detected. (ii) *Fault Isolation*: at this stage the faulty sensor is removed from the loop using a BS, while a 'known' function $c_{y_i} = 0$ is connected at the input of the *i*FD. Figure 9 clearly shows the signal at the input and output of the BS, as well as the signal at the output of the *i*FD. (iii) *Controller reconfiguration*: after the faulty sensor isolation, a reconfiguration signal is generated and the new controller, $K_{(z_t - z)}$, is introduced in the loop.

Careful investigation of Figure 9 (after the fault occurs at point C) shows that the unit detect the fault after a few time steps and that one time step is required for the BS_{y_i} portion to permanently change its output to $c_{y_i} = 0$. Hence, the residual remains large which justifies the reason the BS output will never return to its previous stage when/if the fault vanishes. The same figure also shows the input to BS_{y_i} with the two previous sensor faults, i.e., acceleration and velocity (at point A and B respectively).

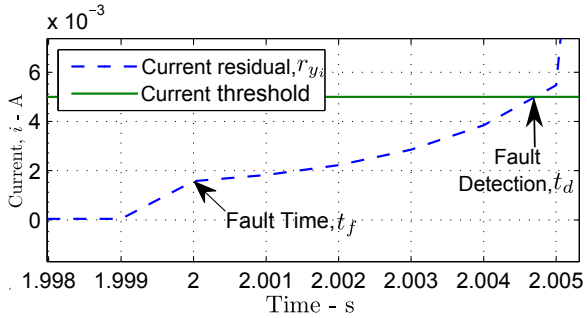


Figure 8. Current sensor residual, r_{y_i}, when fault occurs at 2 s.

Table 2 indicates the resulted performance of the suspension and the false alarm (FA) after 70 tests, analytically 35 for each deterministic and stochastic responses of the suspension. The first column of the table describes the sensor fault scenarios used to test the proposed *i*FD. Typically, rows 2–4 show the results for single sensor faults that occur at $t = 1$ s, while the rest rows show the results with subsequence faults starting from 0.5 s with a time difference of 1 s. The first six columns present the performance with abrupt faults while the rest four, show the performance with incipient faults. The track inputs exciting the EMS were discussed in Appendix A. Multiplicative (Mult.) and Additive (Add.) faults are used, as well as a bias (Bis.) fault that occurs abruptly. Per the scenario case, entry ✓ indicates that EMS performance is successfully maintained, while entry "x" indicates the opposite. In addition, if a FA arises is marked with a red color ▽.

Close investigation of the aforementioned table of results indicates that the *i*FD successfully detects and reconfigures the controller under all scenaria (maintaining the appropriate performance levels). In some cases, i.e., in *id*: 7–8, an FA appears meaning that a sensor is 'shown' impaired although truly is healthy. This is an important finding towards facilitating reliable system autonomy. The threshold setting for the residual plays a substantial role and this needs to be addressed in a reliable autonomous system (as an FA could hinder perception and hence impact system stability).

Given that the residual threshold of such sensor cases is increased to avoid the FA, and the sensors are impaired themselves, then the FD could delay long enough to cause instability. Two particular issues that have been noted and looked further as future research are: (i) in the NN-trained FD unit, in general a small residual remains after a fault occurs in some scenaria, (ii) the coupled nature of the closed-loop (FD unit, reconfiguration mechanisms, decision making). A deep learning approach is currently investigated to address that uncertainty envelope.

Table 2. Performance with various sensor fault scenarios for the EMS.

id	Faulty Sensor(s)	Abrupt Fault						Incipient Fault			
		Mult./FA		Add./FA		Bis./FA		Mult./FA		Add./FA	
		Sth.	Dtm.	Sth.	Dtm.	Sth.	Dtm.	Sth.	Dtm.	Sth.	Dtm.
1	Fault-free	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x
2	i	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x
3	$\dot{z}$	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x
4	$\ddot{z}$	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x
5	$i \to \dot{z}$	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x
6	$i \to \ddot{z}$	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x	✓/x
7	$\dot{z} \to \ddot{z}$	✓/x	✓/▽	✓/x	✓/▽	✓/x	✓/▽	✓/x	✓/▽	✓/▽	✓/▽
8	$\ddot{z} \to \dot{z} \to i$	✓/x	✓/▽	✓/x	✓/▽	✓/x	✓/▽	✓/x	✓/▽	✓/x	✓/▽

Mult.—Multiplicative, Add.—Additive, FA—False Alarm, Sth.—Stochastic, Dtm.—Deterministic, Bis.—Bias.

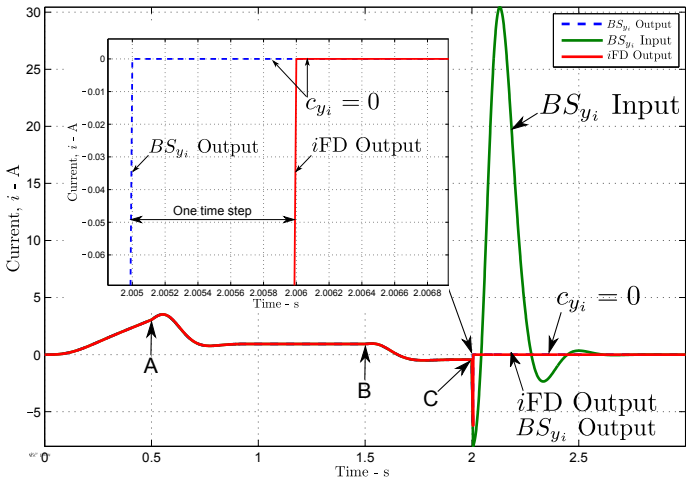


Figure 9. Input-output of the BS_{y_i} of the current sensor, i, and the output of the *i*FD.

Sensor FD time investigation is seen on Table 3. Column-wise: column 1 maps the identifier number for the chosen scenario, column 2 lists the sensors (remark: underlined channels indicate faulty ones with incident occurring at t_f which is shown column 3). The rest of the columns refer to FD time, t_d, in particular the ones marked using boldface font indicated fault occurrence is detected (while boldface entries with superscript "*" are the false alarms).

Careful investigation of the results shows that with abrupt single sensor failures (*id*: 1–3) the fault detection succeeds at the instance the sensor fails. In the incipient fault cases, a short delay within 0.030–0.399 s is noted. However, this delay does not hinder performance due to the robust controller. Although delays are observed in FD throughout these three and the next two scenaria, none of these cause FAs. In the last two scenarios (i.e., in scenario 6: two subsequent faults on velocity and acceleration outputs are considered and in scenario 7: acceleration, velocity and current sensors are impaired sequentially) FAs appear mainly in the current sensor case. Please note that other observed fault delays in *i*FD are successfully accommodated via the control reconfiguration.

Table 3. Sensor (-Snr.) fault detection time, t_d, for the various fault scenarios.

id	Snr.	t_f (s)	Fault Detection Time, $t_d(s)$									
			Abrupt Fault						Incipient Fault			
			Mult.		Add.		Bis.		Mult.		Add.	
			Sth.	Dtm.	Sth.	Dtm.	Sth.	Dtm.	Sth.	Dtm.	Sth.	Dtm.
1	$\underline{i}$	1	1.000	1.000	1.000	1.000	1.000	1.000	**1.052**	1.000	1.000	1.000
	$\underline{\dot{z}}$	-	-	-	-	-	-	-	-	-	-	-
	$\underline{\ddot{z}}$	-	-	-	-	-	-	-	-	-	-	-
2	$\underline{i}$	-	-	-	-	-	-	-	-	-	-	-
	$\underline{\dot{z}}$	1	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	**2.399**	1.000
	$\underline{\ddot{z}}$	-	-	-	-	-	-	-	-	-	-	-
3	$\underline{i}$	-	-	-	-	-	-	-	-	-	-	-
	$\underline{\dot{z}}$	-	-	-	-	-	-	-	-	-	-	-
	$\underline{\ddot{z}}$	1	1.000	1.000	1.000	1.000	1.000	1.000	**1.030**	1.000	**2.075**	1.000
4	$\underline{i}$	0.5	0.500	0.500	0.500	0.500	0.500	0.500	**0.527**	0.500	0.500	0.500
	$\underline{\dot{z}}$	1.5	1.500	**1.502**	1.500	1.500	1.500	1.500	**1.517**	1.500	**2.335**	**1.511**
	$\underline{\ddot{z}}$	-	-	-	-	-	-	-	-	-	-	-
5	$\underline{i}$	0.5	0.500	0.500	0.500	0.500	0.500	0.500	**0.527**	0.500	0.500	0.500
	$\underline{\dot{z}}$	-	-	-	-	-	-	-	-	-	-	-
	$\underline{\ddot{z}}$	1.5	1.500	1.500	1.500	1.500	1.500	1.500	1.500	**1.510**	**1.514**	1.500
6	$\underline{i}$	-	-	**1.700** *	-	**1.700** *	-	**1.700** *	-	**1.700** *	**2.400** *	**1.700** *
	$\underline{\dot{z}}$	0.5	0.500	0.500	**2.070**	0.500	0.500	0.500	0.500	0.500	**2.423**	**0.534**
	$\underline{\ddot{z}}$	1.5	1.500	1.500	1.500	1.500	1.500	**1.700**	1.500	1.500	**2.403**	1.500
7	$\underline{i}$	2.0	2.000	**0.654** *	2.000	**0.654** *	2.000	**0.650** *	**2.614**	**0.654** *	2.000	**0.654** *
	$\underline{\dot{z}}$	1.5	**1.517**	**1.502**	1.500	1.500	1.500	1.500	**1.517**	**1.502**	**2.336**	**1.511**
	$\underline{\ddot{z}}$	0.5	**0.506**	0.500	0.500	0.500	0.500	0.500	**0.508**	0.500	0.500	0.500

3.2. Comparison of the Execution Time

Using the full sensor set, $\mathcal{Y}$ is possible to compare the time taken for a simulation to complete i.e., the execution time (t_e) using the *i*FD with that for a bank of eight in-parallel KEs as shown in Figure 10a. The execution time is measured at a high-level simulation in Simulink platform iteratively ($\times$100) with each estimator. The execution time for each simulation is illustrated in Figure 10b. The mean simulation time for the *i*FD is 0.5 s, while that of the bank-of-estimators setup 7.9 s. Clearly the *i*FD is c. $\times$16 faster, in terms of mean exec. time, compared to the bank of KEs (in addition the proposed scheme offers c. $\times$13 lesser standard deviation i.e., 0.15 s). The comparison showcases a two-fold aspect: (i) the efficacy of the proposed fault detection approach in terms of fast actuators/sensors fault detection, (ii) a level of re-assurance as the proposed approach performs within the same envelope of performance of the conventional bank-of-estimators approach.

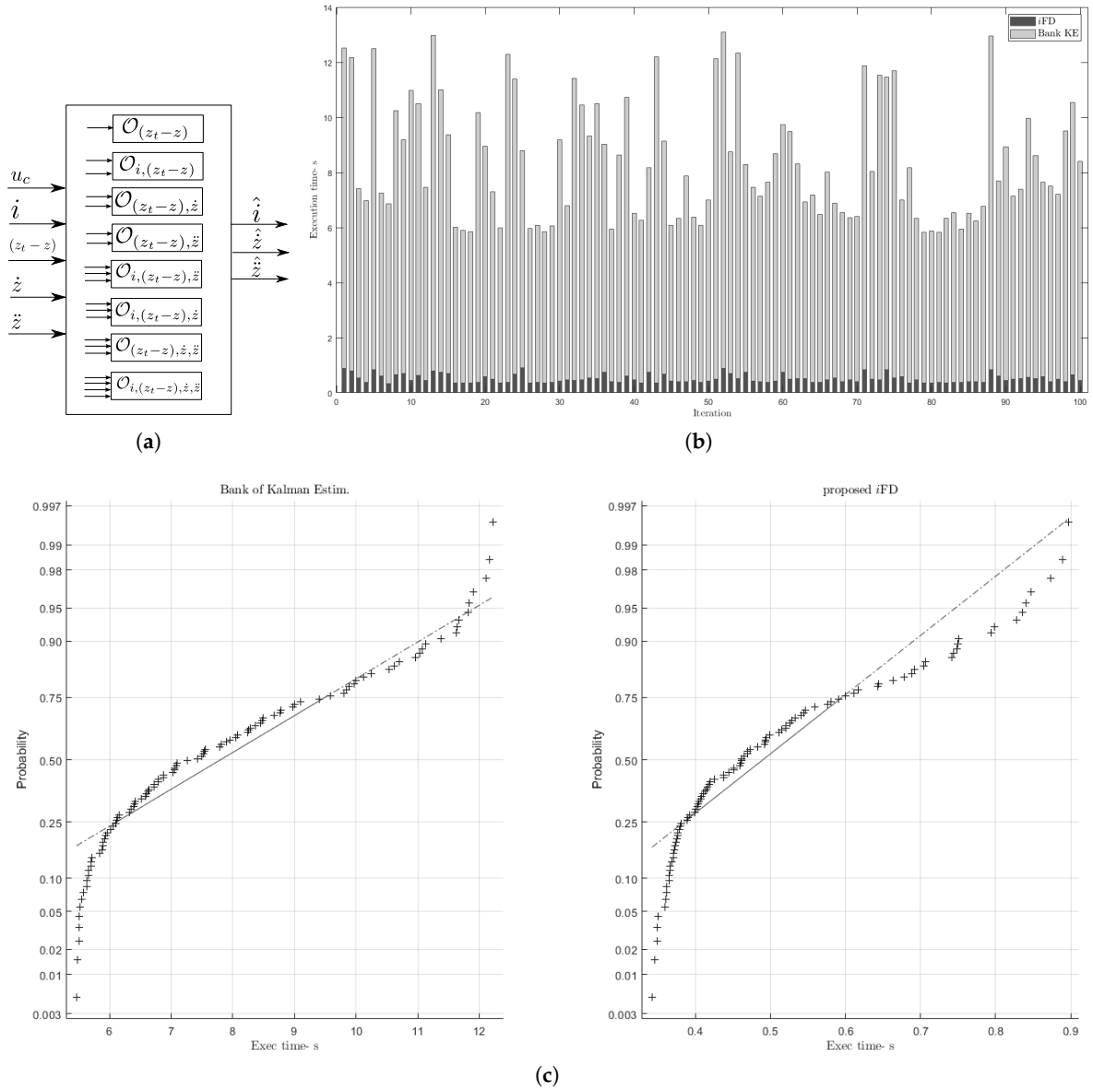


Figure 10. Computational complexity comparison bank of KE vs. *i*FD (AI-based) for multiple sensors failure detection on the EMS system: (**a**) bank of KE (inner box) vs. *i*FD architecture (outer box), (**b**) execution time performance, (**c**) normal probability plot of execution time data.

Figure 10c shows the normal probability plots for the conventional bank of estimators and proposed AI-based schemes. Clearly with very different execution time average, it is seen that the former setup is closer to a normal distribution, the latter NN-based solution favors faster execution time (while providing comparable performance). The importance of this result is two-fold, which we study further, (i) maintaining similar level of performance/reliability to an acceptable conventional solution enables certification [43] and (ii) correlating the process with descriptive statistics supports faster training for the autonomous system solution. To summarize, the following specific comments are highlighted for the proposed scheme:

1. a single AI-based estimator unit can be used in the FDI instead of the conventional bank of estimators benefiting substantial computational resource reduction;
2. the AI-based unit maintaining similar level of performance/reliability to an acceptable conventional solution enables certification [43];

3. faults of multiple characteristics can be handled appropriately by the single AI-based unit as in the more conventional cases;
4. correlating the process with descriptive statistics will support faster autonomous system training.

4. Conclusions

We presented an AI-based, using neural networks, method referred to as *i*FD, towards more reliable system autonomy via detecting actuator/sensor faults. The methods were supported by a detailed analysis on multiple (70) fault scenarios using an electro-magnetic suspension system testbed. Various sensor fault types were studied i.e., abrupt/multiplicative, abrupt/additive, abrupt/bias, incipient/multiplicative and incipient/additive characteristics. The results clearly show that: a single AI-based estimator can be used in the FDI instead of the conventional bank of estimators, which benefits computational resource reduction. The paper indicates important point towards reliable AI-based system autonomy from the FD and threshold discussion. Currently the scheme follows a conservative approach that considers indeterminate faults as permanent ones which are removed as soon as they are detected (to enforce a level of reliability in performance). It is worth mentioning that training (and availability of data) plays an important role in the development.

Author Contributions: The authors contributed to the article as follows: Conceptualization, K.M.D., A.C.Z.; Methodology, K.M.D. and K.M.; Software, K.M.D.; Validation, K.M.D., K.M., A.C.Z.; Resources, K.M.D., A.C.Z.; Data curation, K.M.D., A.C.Z.; Writing–original draft preparation, K.M., K.M.D.; Writing–review and editing, K.M.D., A.C.Z.; Visualization, K.M.D., K.M., A.C.Z.; Supervision, A.C.Z., K.M.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: The authors would like to thank Roger Goodall (now at University of Huddersfield, UK) for his advice during the early stages of this work on the electro-magnetic suspension model considerations and testbed configuration setup. The authors would also like to acknowledge Spyros Tzafestas' (Deceased 13 April 2019) advice and contributions on AI and Neural Networks development in the earlier developments of this work; this paper is dedicated to his memory.

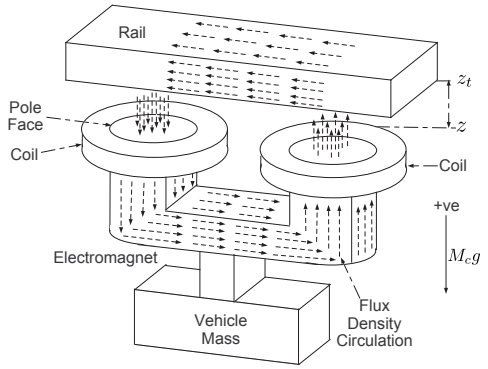
Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. The EMS System—A Test Case

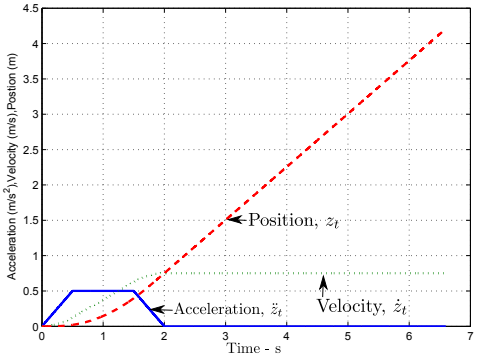
We use a single-stage EMS system platform representation (typically representing a quarter of a Maglev vehicle [44]. The model is actually extensively discussed in [45]. It is the nature of the EMS system that makes it appealing as a test system [46], i.e., being inherently unstable, safety-critical system, under non-trivial control requirements. Figure A1a presents the EMS schematic diagram, the input excitation profile and Figure A1c lists the performance constraints.

The characteristics of the system are discussed in [45]. In particular, the vehicle mass (M_c) is supported on the electromagnet with the distance between them being the so-called airgap $(z_t - z)$. Normally the airgap does not exceed 15 mm. The rest of the variables of interest are: flux B, force F, current I, airgap $(z_t - z)$, input voltage to the EM V_c, vehicle position Z. Please note that the designed operating condition for the EMS system is at airgap of 15 mm, with nominal force 9810 N, nominal flux 1 T, nominal current $I_o = 10$ A, and operating voltage 100 V. For the linearized time invariant state space model of the EMS and the electromagnet parameter calculations the reader is referred to [19,45].

In addition, for simulation purposes the operational scenario uses deterministic rail track information due to the intended changes of the rail's inclination. This transition, vertical direction only, onto the rail's gradient is simulated by the signal shown in Figure A1b. We also consider stochastic track irregularities as random variations of the rail's vertical layout typically caused by the installation process i.e., accumulated inaccuracies and unevenness. Such track elements normally are difficult to measure (some information can be provided from a monitored track database, or a specialist track condition measurement vehicle), while attempts to estimate irregularities from vehicle-based sensor exist in the current literature [47]. Considering the vertical direction, the velocity variations can be approximated by a double-sided power spectrum density [19].



(**a**) The single-stage EMS system.



(**b**) EMS' deterministic input profile.

EMS limitations	Value
Stochastic rail profile	
RMS of acceleration, $\ddot{z}_{rms}$	$\leq 1ms^{-2}$
RMS of airgap variation, $(z_t - z)_{rms}$	$\leq 5mm$
RMS of control effort, $u_{c_{rms}}$	$\leq 300V$
Deterministic rail profile	
Maximum of airgap deviation, $(z_t - z)_p$	$\leq 7.5mm$
Maximum of control effort, u_{c_p}	$\leq 300V$
Airgap settling time, t_s	$\leq 3s$
Airgap steady state error, $(z_t - z)_{e_{ss}}$	$= 0$

(**c**) EMS control constraints.

Figure A1. EMS systems and its constraints.

Moreover, the control design requirements of an EMS system are dependent on the train type and its speed [48]. The EMS system should follow the gradient onto the rail (deterministic) and remain insensitive to track irregularities. Figure A1c summarizes the control performance requirements.

References

1. Napolitano, M.; Windon, D.; Casanova, J.; Innocenti, M.; Silvestri, G. Kalman filters and neural-network schemes for sensor validation in flight control systems. *IEEE Trans. Control Syst. Technol.* **1998**, *6*, 596–611. [CrossRef]
2. Rago, C.; Prasanth, R.; Mehra, R.K.; Fortenbaugh, R. Failure detection and identification and fault tolerant control using the IMM-KF with applications to the Eagle-Eye UAV. In Proceedings of the 37th IEEE Conference on Decision and Control, Tampa, FL, USA, 18 December 1998; Volume 4, pp. 4208–4213.
3. Ranjbaran, M.; Khorasani, K. Fault recovery of an under-actuated quadrotor Aerial Vehicle. In Proceedings of the 49th IEEE Conference on Decision and Control, Atlanta, GA, USA, 15–17 December 2010; pp. 4385–4392.
4. Petritoli, E.; Leccese, F.; Ciani, L. Reliability and Maintenance Analysis of Unmanned Aerial Vehicles. *Sensors* **2018**, *18*, 3171. [CrossRef] [PubMed]
5. Nguyen, N.P.; Hong, S.K. Fault Diagnosis and Fault-Tolerant Control Scheme for Quadcopter UAVs with a Total Loss of Actuator. *Energies* **2019**, *12*, 1139. [CrossRef]
6. Chen, X.M.; Wu, C.X.; Wu, Y.; Xiong, N.X.; Han, R.; Ju, B.B.; Zhang, S. Design and Analysis for Early Warning of Rotor UAV Based on Data-Driven DBN. *Electronics* **2019**, *8*, 1350. [CrossRef]
7. Lussier, B.; Chatila, R.; Guiochet, J.; Ingrand, F.; Lampe, A.; Olivier Killijian, M.; Powell, D. Fault Tolerance in Autonomous Systems: How and How Much? In Proceedings of the 4th IARP/IEEE-RAS/EURON Joint Workshop on Technical Challenge for Dependable Robots in Human Environments, Nagoya, Japan, 16–18 June 2005; 10p.
8. Kate Devitt, S. Trustworthiness of Autonomous Systems. In *Foundations of Trusted Autonomy*; Abbass, H.A., Scholz, J., Reid, D.J., Eds.; Studies in Systems, Decision and Control; Springer International Publishing: Cham, Switzerland, 2018; pp. 161–184. [CrossRef]

9. Napolitano, M.; An, Y.; Seanor, B. A fault tolerant flight control system for sensor and actuator failures using neural networks. *Aircr. Des.* **2000**, *3*, 103–128. [CrossRef]
10. Campa, G.; Fravolini, M.L.; Seanor, B.; Napolitano, M.R.; Gobbo, D.D.; Yu, G.; Gururajan, S. On-line learning neural networks for sensor validation for the flight control system of a B777 research scale model. *Int. J. Robust Nonlinear Control* **2002**, *12*, 987–1007. [CrossRef]
11. Lunze, J.; Schroder, J. Sensor and actuator fault diagnosis of systems with discrete inputs and outputs. *IEEE Trans. Syst. Man Cybern. Part B* **2004**, *34*, 1096–1107. [CrossRef]
12. Edwards, C.; Tan, C. Sensor fault tolerant control using sliding mode observers. *Control Eng. Pract.* **2006**, *14*, 897–908. [CrossRef]
13. Talebi, H.; Khorasani, K. An intelligent sensor and actuator fault detection and isolation scheme for nonlinear systems. In Proceedings of the 46th IEEE Conference on Decision and Control, New Orleans, LA, USA, 12–14 December 2007; pp. 2620–2625.
14. Talebi, H.A.; Khorasani, K. A neural network-based actuator gain fault detection and isolation strategy for nonlinear systems. In Proceedings of the 46th IEEE Conference on Decision and Control, New Orleans, LA, USA, 12–14 December 2007; pp. 2614–2619.
15. Heredia, G.; Ollero, A.; Bejar, M.; Mahtani, R. Sensor and actuator fault detection in small autonomous helicopters. *Mechatronics* **2008**, *18*, 90–99. [CrossRef]
16. Yetendje, A.; Seron, M.M.; Doná, J.A.D.; Martínez, J.J. Sensor fault-tolerant control of a magnetic levitation system. *Int. J. Robust Nonlinear Control* **2010**, *20*, 2108–2121. [CrossRef]
17. Realpe, M.; Vintimilla, B.X.; Vlacic, L. A Fault Tolerant Perception system for autonomous vehicles. In Proceedings of the 2016 35th Chinese Control Conference (CCC), Chengdu, China, 27–29 July 2016; pp. 6531–6536.
18. Ding, S.X. *Data-Driven Design of Fault Diagnosis and Fault-Tolerant Control Systems*; Springer: London, UK, 2014.
19. 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.* **2016**, *24*, 293–301. [CrossRef]
20. Hwang, I.; Kim, S.; Kim, Y.; Seah, C.E. A survey of fault detection, isolation, and reconfiguration methods. *IEEE Trans. Control Syst. Technol.* **2010**, *18*, 636–653. [CrossRef]
21. Zhang, Y.; Jiang, J. Bibliographical review on reconfigurable fault-tolerant control systems. *Annu. Rev. Control* **2008**, *32*, 229–252. [CrossRef]
22. Chibani, A.; Chadli, M.; Ding, S.X.; Braiek, N.B. Design of robust fuzzy fault detection filter for polynomial fuzzy systems with new finite frequency specifications. *Automatica* **2018**, *93*, 42–54. [CrossRef]
23. Ruiz-Arenas, S.; Rusák, Z.; Horváth, I.; Mejí-Gutierrez, R. Systematic exploration of signal-based indicators for failure diagnosis in the context of cyber-physical systems. *Front. Inf. Technol. Electron. Eng.* **2019**, *20*, 152–175. [CrossRef]
24. Samy, I.; Postlethwaite, I.; Gu, D.W. Survey and application of sensor fault detection and isolation schemes. *Control Eng. Pract.* **2011**, *19*, 658–674. [CrossRef]
25. Samy, I.; Postlethwaite, I.; Gu, D.W. A comparative study of NN- and EKF-based SFDA schemes with application to a nonlinear UAV model. *Int. J. Control* **2010**, *83*, 1025–1043. [CrossRef]
26. Tzafestas, S. System Fault Diagnosis Using the Knowledge-Based Methodology. In *Fault Diagmosis in Dynamic Systems*; Patton, R., Frank, P., Clark, R., Eds.; Prentice Hall: Upper Saddle River, NJ, USA, 1989; pp. 502–552.
27. Mok, H.; Chan, C. Online fault detection and isolation of nonlinear systems based on neurofuzzy networks. *Eng. Appl. Artif. Intell.* **2008**, *21*, 171–181. [CrossRef]
28. Skoundrianos, E.N.; Tzafestas, S.G. Fault diagnosis via local neural networks. *Math. Comput. Simul.* **2002**, *60*, 169–180. [CrossRef]
29. Skoundrianos, E.N.; Tzafestas, S.G. Modelling and FDI of Dynamic Discrete Time Systems Using a MLP with a New Sigmoidal Activation Function. *J. Intell. Robot. Syst.* **2004**, *41*, 19–36. [CrossRef]
30. Isermann, R. Supervision, fault-detection and fault-diagnosis methods–an introduction. *Control Eng. Pract.* **1997**, *5*, 639–652. [CrossRef]
31. Albu, A.; Precup, R.E.; Teban, T.A. Results and challenges of artificial neural networks used for decision-making and control in medical applications. *Facta Univ. Ser. Mech. Eng.* **2019**, *17*, 285. [CrossRef]

32. Hunt, K.; Sbarbaro, D.; Żbikowski, R.; Gawthrop, P. Neural networks for control systems-a survey. *Automatica* **1992**, *28*, 1083–1112. [CrossRef]
33. Polycarpou, M.; Helmicki, A. Automated fault detection and accommodation: A learning systems approach. *IEEE Trans. Syst. Man Cybern.* **1995**, *25*, 1447–1458. [CrossRef]
34. Polycarpou, M.; Vemuri, A. Learning methodology for failure detection and accommodation. *IEEE Control Syst. Mag.* **1995**, *15*, 16–24.
35. Frank, P.M.; Köppen-Seliger, B. New developments using AI in fault diagnosis. *Eng. Appl. Artif. Intell.* **1997**, *10*, 3–14. [CrossRef]
36. Reppa, V.; Polycarpou, M.M.; Panayiotou, C.G. Adaptive Approximation for Multiple Sensor Fault Detection and Isolation of Nonlinear Uncertain Systems. *IEEE Trans. Neural Netw. Learn. Syst.* **2013**, *25*, 137–153. [CrossRef]
37. Heredia, G.; Ollero, A.; Mahtani, R.; Béjar, M.; Remuss, V.; Musial, M. Detection of sensor faults in autonomous helicopters. In Proceedings of the IEEE International Conference on Robotics and Automation, Barcelona, Spain, 18–22 April 2005; pp. 2229–2234.
38. Wang, Z.; Lu, C. An Adaptive Threshold Based on RBF Neural Network for Fault Detection of a Nonlinear System. In *Advances in Computer, Communication, Control and Automation*; Wu, Y., Ed.; Lecture Notes in Electrical Engineering; Springer: Berlin/Heidelberg, Germany, 2012; Volume 121, pp. 495–502.
39. McFarlane, D.C.; Glover, K. A loop-shaping design procedure using $\mathcal{H}_\infty$ synthesis. *IEEE Trans. Autom. Control* **1992**, *37*, 759–769. [CrossRef]
40. Michail, K.; Zolotas, A.C.; Goodall, R.M.; Halikias, G. Optimal selection for sensor fault tolerant control of an EMS system via loop-shaping robust control. In Proceedings of the 19th Mediterranean Conference on Control and Automation, Corfu, Greece, 20–23 June 2011; pp. 1112–1117.
41. Hagan, M.; Menhaj, M. Training feedforward networks with the Marquardt algorithm. *IEEE Trans. Neural Netw.* **1994**, *5*, 989–993. [CrossRef]
42. Hagan, M.; Demuth, H.; Beale, M. *Neural Network Design*; PWS Pub. Co.: Boston, MA, USA, 1996.
43. Aravena, J.; Zhou, K.; Li, X.R.; Chowdhury, F. Fault tolerant safe flight controller bank 1. *IFAC Proc. Vol.* **2006**, *39*, 807–812. doi:10.3182/20060829-4-CN-2909.00134. [CrossRef]
44. Goodall, R.M. Generalised Design Models for EMS Maglev. In Proceedings of the Maglev 2008—The 20th International Conference on Magnetically Levitated Systems and Linear Drives, San Diego, CA, USA, 15–18 December 2008.
45. Michail, K. Optimised Configuration of Sensing Elements for Control and Fault Tolerance Applied to an Electro-Magnetic Suspension System. Ph.D. Thesis, Loughborough University, Loughborough, UK, 2009.
46. Bojan-Dragos, C.A.; Radac, M.B.; Precup, R.E.; Hedrea, E.L.; Tanasoiu, O.M. Gain-scheduling control solutions for magnetic levitation systems. *Acta Polytech. Hung.* **2018**, *15*, 89–108. [CrossRef]
47. Rosa, A.D.; Alfi, S.; Bruni, S. Estimation of lateral and cross alignment in a railway track based on vehicle dynamics measurements. *Mech. Syst. Signal Process.* **2019**, *116*, 606–623. [CrossRef]
48. Goodall, R.M. Dynamics and control requirements for EMS Maglev suspensions. In Proceedings of the International Conference on Maglev, Shanghai, China, 26–28 October 2004; pp. 926–934.