2022 30th Mediterranean Conference on Control and Automation (MED)
June 28 - July 1, 2022. Athens, Greece

Learning-Based Diagnostics for Fault Detection and Isolation in Linear Stochastic Systems

Erfaun Noorani, Christoforos Somarakis, Raman Goyal, Alexander Feldman, and Shantanu Rane

Abstract— **AI-enabled mechanisms are deployed to guard controlled systems against sensor anomalies. We explore a two-level architecture design in which a low-level feedback controller of a linear system uses measurements from one or more potentially unreliable sensors. These observations are prone both to sensor noise but unknown additive faults. Our proposed, high-level, guard mechanism consists of a Reinforcement Learning (RL) agent that monitors available vitals of the system. In the event of a fault on the sensor components, the RL agent automatically detects, estimates the fault, localizes and takes action to cancel the fault. In addition, we develop design methodologies for efficient training of the RL agent that take advantage of system dynamics and sensor fusion schemes. We show that the associated training cost functions can be designed so that their optimal policy achieves efficient of arbitrary constant or piece-wise constant sensor faults. To illustrate our theoretical results, we consider a linearized version of a chemical process with multiple sensors, controlled by a Linear Quadratic Gaussian (LQG) Servo-Controller with Integral Action. Our simulations show that the RL-agent is successful in localizing the faulty sensors and mitigating the effects of faults in an online fashion.**

I. Introduction

The smart revolution with developments in smart grid, smart building, smart transportation systems, etc, in conjunction with so called "network society" and proliferation of Internet of Things (IoT) devices, have made the resilience and security of Cyber Physical Systems a paramount priority [1], [2]. Such complex systems are prone to multiple fault scenarios, e.g. system/process fault, actuator fault, controller fault, cyber attacks, sensor faults, etc. We focus our attention on abnormalities in the sensing aperture of the system. The complex and interconnected nature of the today's Cyber Physical Systems allow for escalation and propagation of small faults in subsystems. Thus, the ability to quickly detect, isolate (localize), mitigate (accommodate), and characterize faults in a system have become synonymous to trustworthiness, reliability, and resilience.

Accounting for all possible failure scenarios at the design stage is a complex and in times impractical, if not impossible, task, and typically leads to conservative designs, e.g. H_2 and H_∞. Thus, designing Fault Tolerant (FT) systems that allow for mission continuation in the case of unexpected failures are of importance for designing resilient and trustworthy systems. The complexity of the problem has led to a taxonomy of the types of faults and attacks [3] and development of approaches restricted to particular type of systems and abnormalities with low generalizability over the CPS systems or abnormalities.

E. Noorani, C. Somarakis, R. Goyal, A. Feldman, and S. Rane are with Palo Alto Research Center - A Xerox Company, Palo Alto, CA, USA. E. Noorani is with the Department of Electrical and Computer Engineering and the Institute for System Research (ISR) at the University of Maryland College Park, College Park, MD, USA. E. Noorani is a Clark doctoral Fellow. {enoorani}@umd.edu, {enoorani,somarakis,rgoyal}@parc.com, {afeldman,srane}@parc.com

Model-based diagnostics is a well-established research [1], [4], [5]. Fault Tolerant Control (FTC) [6], [7] concerns the use of model-based approaches in the design of control systems. The traditional FTC methods rely on prior knowledge of the abnormalities and their effects on the system, i.e., known system dynamics and faults. The performance of such model-based approaches are limited by model inaccuracies and are sensitive to model misspecifications. The traditional design is typically composed of a fault detection module which switches to a low-level controller appropriate for that specific fault from a set of predefined fault controllers. The sequential nature of such design introduces a time-delay into the system which might de-stabilize the system.

To mitigate the delay introduced by the traditional FTC design, Blended Control (BC) proposes a hierarchical design with a weighted combination of low-level controllers. A deep-learning BC design [8] uses an RL algorithm to set the blending weights, to provide a data-driven approach to BC, which in turn eliminates the need for prior knowledge of the system dynamics. In such architecture, the high-level control effectively implements the low-level controller and does not directly interact with the controlled system. However, BC designs are intrusive in the sense that they aim to synthesize a fault-tolerant controller. BC designs are not capable of fault localization.

This work is an outgrowth of [9], from which we draw results freely to keep this work self-contained. Our emphasis in [9] was to design RL agents for simple fault detection and mitigation strategies. In [9], we proposed an augmented feedback-loop to a controlled system that change the input to the low-level controller based on the input-output of the low level system in order to help the controlled system to continue its mission in the presence of sensor faults.

This paper focuses on the problem of mitigation *and* localization in an over-observed system with unreliable sensors. Our working hypothesis is that one or more sensors may undergo a fault event and subsequently affect the plant dynamics. It is desirable for the agent not only to be able to mitigate the fault but also to detect which sensor(s) are the ones that infuse the fault in the system.

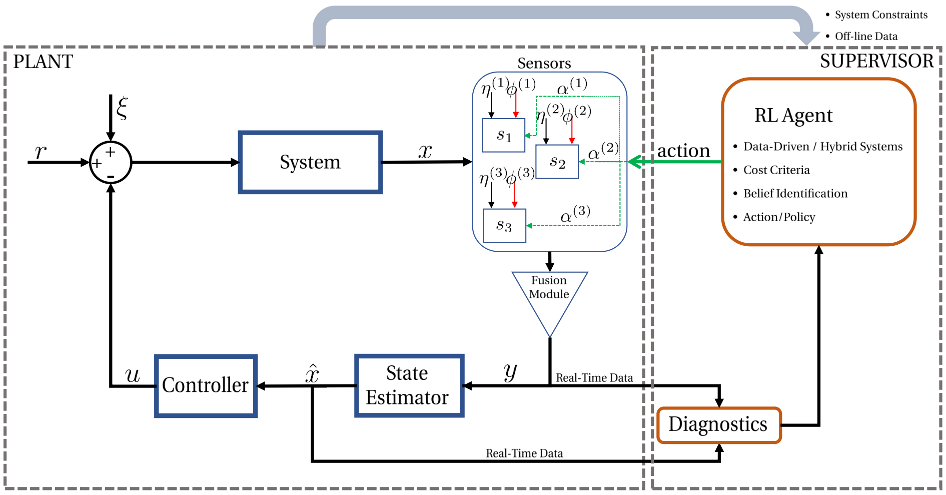

Fig. 1: The bi-level architecture. The agent takes action to defend the closed-loop plant against sensor faults. The input information arrives from the sensor fusion out of a collection of heterogeneous sensors into some output signal y that is fed into the low-level controller. The supervisory agent is exposed to the output of the sensor and acts on each sensor separately.

II. Control Systems Framework

We begin with laying the groundwork for our analysis. The plant components to be considered are: a discrete time linear control system in presence of stochastic process noise, assumed Gaussian - with known statistics; an ensemble of sensors that measure system state, and a Kalman state-estimator. Furthermore, a low-level controller regulates the system around a reference vector.

A. State Equation

In particular, for $t \in \mathbb{N}_0$ we assume the linear time invariant update law

$$x_{t+1} = A\,x_t + B\,u_t + \Gamma\,\xi_t, \tag{1}$$

where $x \in \mathbb{R}^{n_x}$ is the state vector, $A \in \mathbb{R}^{n_x \times n_x}$ is the state matrix, $u \in \mathbb{R}^{n_u}$ is the control signal, $B \in \mathbb{R}^{n_x \times n_u}$ is the input matrix, $\xi_t \in \mathbb{R}^{n_\xi}$ is a noise vector of Gaussian white noise, and $\Gamma \in \mathbb{R}^{n_x \times n_\xi}$ is the diffusion matrix associated with the process noise.

B. Sensing Component

To address the isolation problem, our assumption is that state x in (1) is observed at every instant from more than one sensors, independently. There are $s{>}1$ sensors each of which with their own inaccuracy modelled via additive dynamic signals, of Gaussian type with given statistics. We consider $s{>}1$ sensors, so that k^{th} sensor delivers

$$y_t^{(k)} = C_k\,x_t + N_k\,\eta_t^{(k)}, \qquad k = 1, \ldots, s.$$

Here $C_k \in \mathbb{R}^{n_y \times n_x}$ is the output sensor matrix and $\eta_t^{(k)} \in \mathbb{R}^{n_\eta}$ the white noise vector with $N_k \in \mathbb{R}^{n_y \times n_\eta}$ the associated diffusion matrix.

1) Fault Model: Our working hypothesis is that sensors are vulnerable to faults. They modeled as time-dependent additive bias. Sensor biases can occur due to incorrect calibration, physical changes in the sensor system or it can be the result of types of jamming attacks [10], [11]. With this in mind, the k^{th} potentially biased sensor's output is:

$$y_t^{(k)} = C_k\,x_t + N_k\,\eta_t^{(k)} + \phi_t^{(k)}, \qquad k = 1, \ldots, s. \tag{2}$$

Here, vector $\phi_t^k \in \mathbb{R}^{n_y}$ is the disturbance that models bias.

2) Sensor Fusion: Over-observed systems are an example of multi-sensing configurations addressed in e.g. robotic [12], or industrial applications [13], where redundancy of estimators is necessary for ensuring acceptable accuracy of measurement that will prevent system instabilities or cascading failures . It is also typical that a fusion scheme smooths out raw sensor measurements before further use. The scheme to be considered in this work is the direct average of s sensors' outputs, i.e.:

$$\begin{aligned} y_t &= \frac{1}{s}\sum_{k=1}^{s}\left[C_k\,x_t + N_k\,n_t^{(k)} + \phi_t^{(k)}\right], \\ &= C x_t + \frac{1}{s}\sum_{k=1}^{s}\left[N_k\,n_t^{(k)} + \phi_t^{(k)}\right], \end{aligned} \tag{3}$$

where $C \in \mathbb{R}^{n_y \times n_x}$ is the cumulative output matrix of x_t. Provided C preserves necessary observability, the rationale behind selecting (3) as the input signal is that through this observation redundancy, the effect of sensor noise is averaged out [12]. It is remarked that multiple sensors and the fusion scheme we consider are, from a system theory point of view, an extension of single sensor measurements.

C. State Estimator

We use the standard Kalman estimator, with dynamics

$$x_{t+1|t} = A\,x_{t|t-1} + B\,u_t + L\,(y_t - C\,x_{t|t-1}), \tag{4}$$

where $x_{t+1|t} =: \hat{x}_{t+1} \in \mathbb{R}^{n_x}$ is the prediction at time $t+1$ and $L \in \mathbb{R}^{n_x \times n_x}$ is the Kalman gain. The filter output is

$$x_{t|t} = (I_{n_x} - MC)x_{t|t-1} + My_t, \tag{5}$$

where I_{n_x} is the $n_x \times n_x$ identity matrix, and M is the innovation gain.

D. Controller

Assuming controllability and observability of (1), and that we have information about the process and sensor noise statistics, we can design a linear Quadratic Gaussian Servo Controller with Integral Action [14]. The result is that output y_t asymptotically tracks reference signal r_t also in the sense of mean value. The integrator dynamics are:

$$x_{t+1}^{(i)} = x_t^{(i)} + (r_t - y_t) \tag{6}$$

where $r_t \in \mathbb{R}^{n_y}$ is the reference signal. The control law reads

$$u_t = -\begin{bmatrix} K_{\hat{x}} & K_{x^{(i)}} \end{bmatrix} \begin{bmatrix} x_{t|t} \\ x_t^{(i)} \end{bmatrix} \tag{7}$$

for matrices $K_{\hat{x}}$ and $K_{x^{(i)}}$ that minimize some quadratic cost functional J. In the event of a sensor anomaly, controller's effect on plant will be to keep measured output on track, at the expense of the actual state. The latter gets diverged to an uncontrollable, perhaps dangerous range of values. This is illustrated in Figure 2 of §V.

III. High-Level Diagnostic Module

In this section, we elaborate on how the RL-agent perceives the situation (observation-space), acts on the controlled system (action-space) and the task that it aims to accomplish. We design a cost function to achieve this task.

At the beginning of each learning episode, the agent starts in an initial state, that is a function of $\mathbb{X}$, the controlled-system's augmented state[1]. At each time-step, the RL-agent receives an observation (e.g. from the diagnostic module) and executes an action (e.g. by adding a term to the sensor measurement) according to its policy. Upon the execution of the action, the system transitions to a successor state, and the agent receives an instantaneous cost c_t. The cost defines the task at hand. The RL agent aims to find a policy to minimize the expectation of the system's cumulative cost over a given horizon, denoted by T, that is,

$$\mathcal{J}(\pi) := \mathbb{E}\Big[\mathcal{C}(\tau)\Big], \tag{8}$$

where $\mathcal{C}{=}\sum_{t=0}^{T-1} c_t$ is the cumulative cost over an episode, and the expectation is taken over the system's trajectory. Once the agent is trained, it can be deployed to run simultaneously in parallel with the controlled system.

Feature Extraction: In our case-study, we use three types of input signals from the low-level system at each time-step as the set of observations to the RL algorithm. The observation space of the RL algorithm consists of the low-level control signal u, the state estimate $\hat{x}$, and sensor residuals, i.e., $y^{(k)} - C_k\,\hat{x}$.

[1]Therefore RL state should not be confused with the state x of the system.

Action Space: The way the RL agent applies its actions on the low-level controller is through additive intervention on the sensing module as shown in Figure 1. The formal expression on RL's integration into system dynamics is the following modification of (2):

$$y_t = C_k\,x_t + \eta_t^{(k)} + \phi_t^{(k)} + \alpha_t^{(k)}, \qquad k = 1, \dots, s, \tag{9}$$

with the restriction of $\alpha_t^{(k)} \in \mathcal{A} \subset \mathbb{R}$, $\forall k, t$. Here $\mathcal{A}$ is a compact subset of $\mathbb{R}$ modeling the range of feasible faults that can appear on the sensors. In other words, the RL agent, by design, intervenes in the sensor module, with a set of actions spanning the set of sensors. Following the minimal intervention constraint to the closed-loop plant dynamics, the agent has no information about the location and magnitude of the fault, nor the low-level controller action on output and state. The RL agent aims to apply an action at each time step that blocks the injection of faults into the system, i.e. $\alpha_t \equiv -\phi_t$ where $\alpha = [\alpha^{(1)}, \dots, \alpha^{(s)}]^T$ and $\phi = [\phi^{(1)}, \dots, \phi^{(s)}]^T$ are the stacked vectors of actions and faults, respectively.

RL algorithm: We chose to use a Deep Deterministic Policy Gradient (DDPG) RL algorithm [15]. Other RL algorithms that can be applied to continuous action-spaces, e.g. our case-study, could potentially be used as an alternative to DDPG here. Our results show that the proposed architecture can be applied with satisfactory results, and without the need to extensively search in the space of available algorithms or hyper-parameters.

IV. Reward Function Design Principles

Reward function for the training phase of agents are crucial for their performance. The design principles of cost (8) within the context of control systems, and the bi-level architecture proposed, are the main contribution of this work. We propose two different types of cost functions and explains the diagnostic objectives they achieve.

A. Conditions for Fault Mitigation

By taking the tracking error at each time-step as the instantaneous cost, the RL agent will try to learn a mapping from the observations to actions, a policy, that minimizes (8) for the choice of $c_t{=}\|Cx_t - r_t\|^2$. Non-trivial RL action signals that monitor and safeguard plant against faults, but do not intervene in the interior system dynamics may turn out to be a challenging task due to the action of low-level controller. The latter module always imposes tracking of r_t from the output y_t. One way for RL to mitigate this effect is to learn a policy that will minimize:

$$\mathcal{C}_M(\tau) = \sum_{t=0}^{T-1} \|C\,x_t - r_t\|^2,$$

in the sense of (8). We define $c_t{:=}\|C\,x_t - r_t\|^2$ as *mitigation* cost function term because the policy that RL agent achieves cancellation of fault without necessarily identifying, where the error comes from, i.e., which sensor is at fault. The following result outlines this property by examining the policy that minimizes long-term cost, i.e., $\lim_t \mathbb{E}[c_t]$.

Theorem 1: Consider the closed-loop system dynamics (1) - (7) with RL agent acting on output observables according to (9). Assume that reference signal is $r_t \equiv r \in \mathbb{R}^{n_y}$ i.e. constant, and that faults $\phi_t^{(k)} \equiv \phi^{(k)} \in \mathbb{R}^{n_y}$ are also constant, yet arbitrary. Then on the space of time-invariant actions implemented by the RL agent, it holds that

$$\lim_{t\to\infty} \mathbb{E}\big[c_t\big] = \theta_\infty + (\phi+\alpha)^T \bigg(\frac{1}{s^2}\mathbb{J}_s \otimes I_{n_y}\bigg)(\phi+\alpha),$$

where $\mathbb{J}_s$ is the $s\times s$ matrix of ones and $\phi, \alpha \in \mathbb{R}^{sn_y}$ are the group fault and action vectors, respectively. Moreover, the steady-state minimization vector of $\lim_t \mathbb{E}[c_t]$ lies in

$$\mathcal{M} = \bigg\{\alpha \in \mathbb{R}^{sn_y} \;:\; \sum_k \alpha^{(k)} = -\sum_k \phi^{(k)}\bigg\}.$$

Further discussion of mitigation strategies is held in [9].

B. Conditions for Mitigation & Localization

As Theorem 1 explains successful minimization of mitigation cost function will result in cancelling the sensor faults in the system, however one can say very little about where these faults come from. The problem of *localization*, that is for the RL agent to label the faults onto sensors would require a modified cost function. Our idea is that the agent can be trained to minimize

$$\mathcal{C}_{ML}(\tau) = \sum_{t=0}^{T-1} \big(\|C\,x_t - r_t\|^2 + \sum_{k=1}^{s} \|F_k\,y_t^{(k)} - G_k\,y_t\|^2\big)$$

in the sense of (8). In this case, the instantaneous cost is

$$c_t = \|C\,x_t - r_t\|^2 + \sum_{k=1}^{s} \|F_k\,y_t^{(k)} - G_k\,y_t\|^2. \tag{10}$$

Matrices $\{F_k, G_k\}_{k=1}^s$ have dimensions $n_s \times n_y$ and will be designed for a particular purpose: to decouple the fault that comes from the closed-loop system and the fault that enters through sensors. The fault occurring from the system's state x_t is the result of faulty sensors that are already in the loop. These dynamics are handled with the properties of the mitigation cost term (this is the first term, $\mathcal{R}_M$, within $\mathcal{C}_{ML}$). The second term is summed over all the s sensors with their companion matrices F_k and G_k chosen to satisfy

$$F_k C_k = G_k C, \quad k = 1, \dots, s. \tag{11}$$

This condition tells us that the localization term effectively cancels the signal coming from the system and compares the rest of the sources, that is potential faults that are newly injected into the system. The second result of this work explains under what condition the steady-state constant policy minimized the $\mathcal{C}_{ML}(\tau)$ in the sense of (8). For the exposition of Theorem 2 below, we introduce the matrix $D = [D_{ij}]_{n_y\times n_y}$:

$$D_{ii} = F_i^T F_i - \frac{1}{s}(G_i^T F_i + F_i^T G_i) + \frac{1}{s^2}\sum_{k=1}^{s} G_k^T G_k,$$

$$D_{ij} = -\frac{1}{s}(F_i^T G_i + F_j^T G_j) + \frac{1}{s^2}\sum_{k=1}^{s} G_k^T G_k,$$

and the matrix

$$X_k = \bigg[-\frac{1}{s}G_1 N_1 - \dots + (F_k N_k - \frac{1}{s}G_k N_k) - \dots - \frac{1}{s}G_s N_s\bigg].$$

Theorem 2: Consider the closed-loop system dynamics (1) - (7) with the same set of assumptions as Theorem 1. Let the matrices F_k and G_k satisfy the condition in Eq. (11) for the $\mathcal{C}_{\mathrm{ML}}$ cost. Then

$$\lim_{t\to\infty} \mathbb{E}\big[c_t\big] = \sigma_\infty + (\phi+\alpha)^T \bigg(\frac{1}{s^2}\mathbb{J}_s \otimes I_{n_y} + D\bigg)(\phi+\alpha)$$

where $\sigma_\infty = \theta_\infty + \sum_{k=1}^{s} \mathrm{Tr}\big[X_k^T X_k\big]$. Moreover, if

$$\frac{1}{s^2}\mathbb{J}_s \otimes I_{n_y} + D > 0,$$

the steady-state minimization vector of $\lim_t \mathbb{E}[c_t]$ lies in

$$\mathcal{ML} = \bigg\{\alpha \in \mathbb{R}^{sn_y} \;:\; \alpha^{(k)} = -\phi^{(k)}\bigg\}.$$

Note that matrices $\{F_k, G_k\}_k$ are agnostic to system dynamics and relate only to the output matrices C_k of the sensor module. The next corollary describes an elegant application of condition (11).

Corollary 1: Assume that sensor module consists of s identical sensors (i.e. $C_k = C_{k'} \;\; \forall\, k, k'$). Then (11) holds for $F_k = G_k = I_{n_y}$ and

$$\lim_{t\to\infty} \mathbb{E}\big[c_t\big] = \sigma_\infty + (\phi+\alpha)^T \bigg(\big(\frac{1}{s^2}\mathbb{J}_s + \mathcal{L}\big) \otimes I_{n_y}\bigg)(\phi+\alpha)$$

where $\mathcal{L}$ is the laplacian matrix of a fully connected graph of s nodes with edge weights $1/s$. Furthermore, $\big(\frac{1}{s^2}\mathbb{J}_s + \mathcal{L}\big)$ is positive definite and the steady-state cost function has a unique global minimum in $\mathcal{ML}$.

The design principles discussed in this section seem restricted to constant faults and actions. This hypothesis facilitated tractable analysis and closed-form expressions in contrast to involved Dynamic Programming techniques that RL typically relies on. However the method can be applicable to more general faults and actions in plants where a well-designed plant controller will bring the linear system to steady-state behavior, faster than the duration or impulsiveness of a fault.

V. Simulation Results

As an application to our approach we consider a level and temperature control problem, addressed in industrial systems [16]. Three tanks with liquid of different temperature are connected with pipes. The objectives are to maintain a constant liquid level and a constant temperature in the reactor tank 2 and, thus, producing a constant product outflow. To achieve this, hot and cold liquids are supplied by Tanks 1 and 3, respectively, while an extra heater device is installed in Tank 2. The system states of interest are the level of water in tanks 2 and 3, and the temperature of water in tank 2, i.e. $n_x = 3$. The control inputs is two flow pumps, one valve and one heater (i.e. $n_u = 4$). To fit with our current framework, we consider a linearized version of the system dynamics along the lines of [16], [17], we refer for more

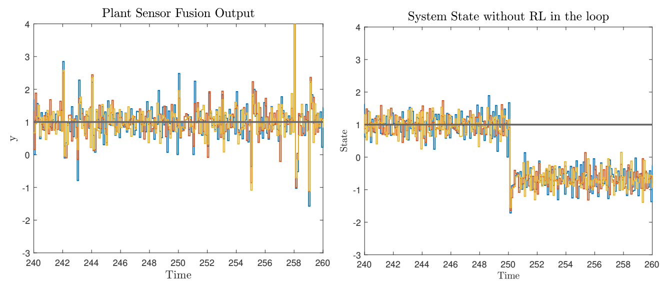

Fig. 2: The effect of LQG-i controller in the event of sensor fault. Black horizontal line illustrates reference target. The low-level controller is designed to regulate output around reference. In the event of fault, like $\phi_t^{(3)} = 5$ for $t \geq 250$, system output y will remain unchanged (left). On the contrary the system state will get perturbed (right).

details. The actuator noise is assumed zero-mean Gaussian with covariance $\Gamma^T\Gamma{=}0.05I_3$. The plant controller objective is to regulate state vector around a reference value that for simplicity was taken equal to $r = [1, 1, 1]^T$. The linearized system matrices are drawn from [17]. The three-tank system is LQG controlled with integral action to regulate output while optimizing some given cost functional J. Figure 2 illustrates plant response in the event of a sensor fault. At time $t = 250$ a fault on sensor 3 is induced, with constant magnitude. The fused sensor output remains unchanged, at the expense of system state, In other words, controllers of such type can tolerate sensor faults at the expense of state of the system. We trained an RL agent with reward functions for mitigation, $\mathcal{C}_M$ and mitigation and isolation $\mathcal{C}_{ML}$, respectively. Training conducted by inducing constant yet arbitrary faults with magnitude between -20 and 20, and identified with the action range, i.e. $\mathcal{A} = [-20, 20]$. It is important to remark that all results presented in the section are with agent took the constant fault training only.

A. Fault Mitigation

The first round of result is agent trained for mitigation, and briefly discussed and mainly for illustrative purposes and comparison with mitigation and localization results. We used reward function $\mathcal{C}_M$ to train agent for 1000 episodes. We present a scenario where a constant fault is induced at time $t = 250$ through sensor 3. Figure 3 demonstrates the performance of supervisory agent trained to detect and mitigate a fault. The implemented action is in line with the desired policy according to Theorem 1. Exploration of mitigation diagnostics is the purpose of [9].

B. Fault Mitigation & Isolation

A supervisory agent was trained to explore policies that minimize $\mathcal{C}_{ML}$. Our objective is to illustrate the effectiveness of the reward function under essentially Corollary 1. Here, we show a simple illustration of the effectiveness of training. Our agent is able to apply what it has learned in a constant fault applied at $t = 250$ on sensor 3. The simulation results are illustrated in Figure 4. It is interesting to compare these results with Figure 3. Unlike mitigation, actions $\alpha^{(2)}$ and

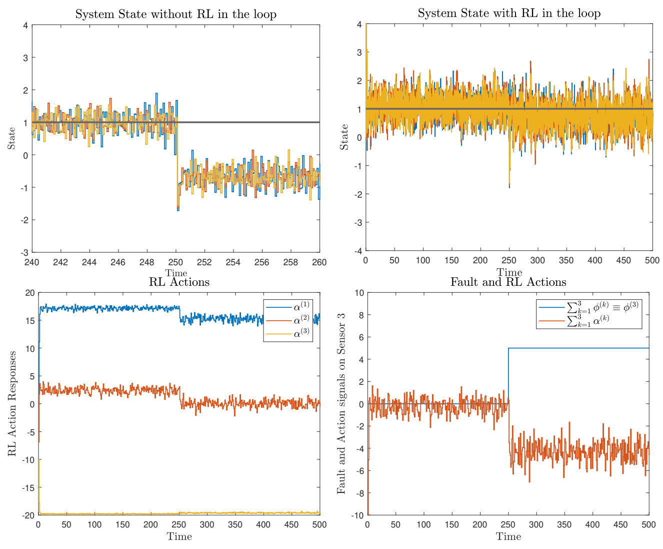

Fig. 3: Mitigation. RL agent responses collectively to a fault through sensor fault. The results illustrate Theorem 1, under which agent's policy is able to detect and correct the fault but not identify its entry point.

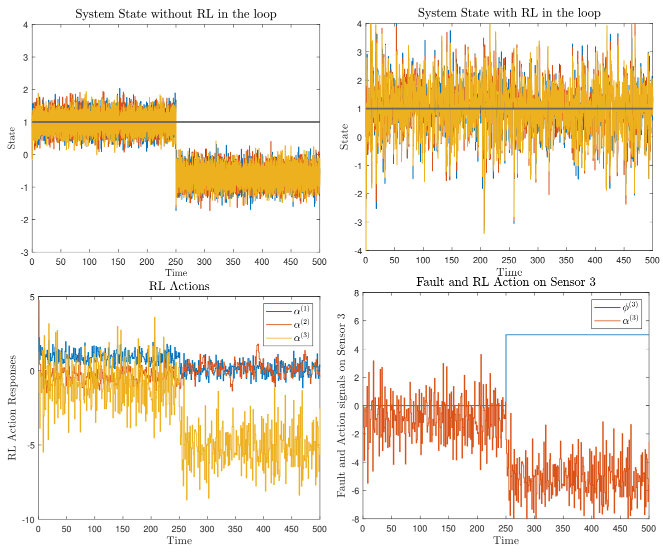

Fig. 4: Upper left: Timed behavior of state of plant without RL module, when a fault of magnitude 5 enters the system at time $t = 250$. Upper right: Behavior of the RL augmented plant, in the event of same fault scenario. Lower left: RL agent's policies over time and its reaction to fault event. Lower right: Fault/action plot.

$\alpha^{(3)}$, remain roughly indifferent. The last two simulation results regard piece-wise constant and slowly time varying fonts. We explore the effectiveness of our supervisory agent that was trained with constant faults only and see that despite its simple training, the agent conducts successfully mitigation and localizing piece-wise constant faults, as shown in Figure 5 but also exhibits remarkable success in dealing with sinusoidal signal fault applied on sensor 3, as shown in Figure 6. Despite the fact that general time-varying fault signals are beyond Theorem 2 and Corollary 1, $\mathcal{C}_{ML}$ delivers satisfactory results, by tracking the injected fault.

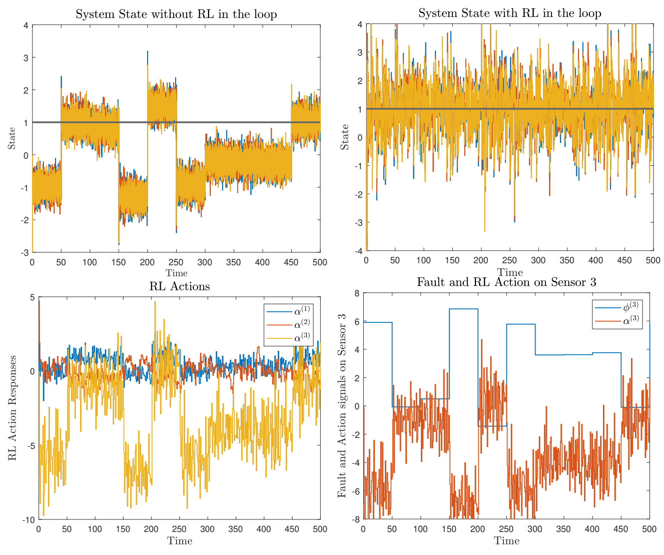

Fig. 5: Upper left: Timed behavior the controlled system without RL module when a piece-wise fault with random magnitudes, and jumps at every 50 time steps enters the system. Upper right: The corrective effect of RL under same effect. Lower left: The behavior of the RL. Lower right: Fault/action plot.

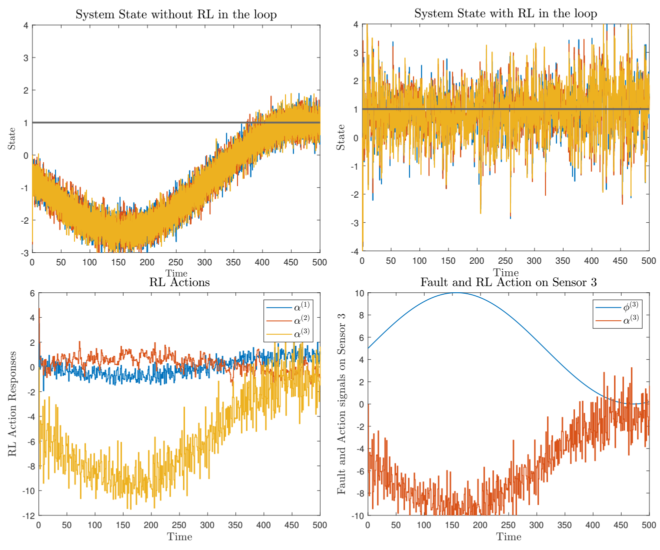

Fig. 6: Upper left: Timed behavior the controlled system without RL module when $\phi_t^{(3)} = 5 + \sin(0.01 \cdot t)$ enters the system. Upper right: The corrective effect of RL under same effect. Lower left: The behavior of the RL. Lower right: Fault/action plot.

VI. Concluding Remarks

We presented a method to design and implement AI-based techniques for non-intrusive online sensor anomaly detection, mitigation and isolation for class of linear control systems. Our method suggest that RL monitoring can deliver interesting results with minimal intervention, explicit information on plant dynamics and anything but involved training recipes. In our future work, we will extend our framework to non-linear systems and general adversarial attacks on the sensor modules.

References

[1] R. Patton, P. Frank, and R. Clark, *Issues of Fault Diagnosis for Dynamic Systems*. Springer London, 2000.

[2] V. Venkatasubramanian, R. Rengaswamy, K. Yin, and S. N. Kavuri, “A review of process fault detection and diagnosis: Part i: Quantitative model-based methods,” *Computers & Chemical Engineering*, vol. 27, no. 3, pp. 293–311, 2003.

[3] S. M. Dibaji, M. Pirani, D. B. Flamholz, A. M. Annaswamy, K. H. Johansson, and A. Chakrabortty, “A Systems and Control Perspective of CPS Security,” *Annual Reviews in Control*, vol. 47, pp. 394–411, 2019.

[4] V. Venkatasubramanian, R. Rengaswamy, and S. N. Kavuri, “A review of process fault detection and diagnosis: Part ii: Qualitative models and search strategies,” *Computers & Chemical Engineering*, vol. 27, no. 3, pp. 313–326, 2003.

[5] G. J. Vachtsevanos and G. J. Vachtsevanos, *Intelligent fault diagnosis and prognosis for engineering systems*. Wiley Online Library, 2006, vol. 456.

[6] J. Selkäinaho and A. Halme, “An intelligent fault tolerant control system,” *IFAC Proceedings Volumes*, vol. 20, no. 5, Part 3, pp. 69–73, 1987, 10th Triennial IFAC Congress on Automatic Control - 1987 Volume III, Munich, Germany, 27-31 July.

[7] Y. Diao and K. M. Passino, “Intelligent fault-tolerant control using adaptive and learning methods,” *Control engineering practice*, vol. 10, no. 8, pp. 801–817, 2002.

[8] Y. Sohège, G. Provan, M. Quinones-Grueiro, and G. Biswas, “Deep reinforcement learning and randomized blending for control under novel disturbances,” *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 8175–8180, 2020.

[9] E. Noorani, C. Somarakis, R. Goyal, A. Feldman, and R. Shantanu, “Model-based inverse reinforcement learning for online diagnostics in stochastic controlled systems,” *The 17th IEEE International Conference on Control Automation.*, June 2022.

[10] E. Balaban, A. Saxena, P. Bansal, K. F. Goebel, and S. Curran, “Modeling, detection, and disambiguation of sensor faults for aerospace applications,” *IEEE Sensors Journal*, vol. 9, no. 12, pp. 1907–1917, 2009.

[11] D. Adamy, *EW 102: A Second Course in Electronic Warfare*, ser. Artech House radar library. Artech House, 2004.

[12] Y. Guo, X. Fang, Z. Dong, and H. Mi, “Research on multi-sensor information fusion and intelligent optimization algorithm and related topics of mobile robots.” *EURASIP J. Adv. Signal Process*, vol. 11, pp. 1–17, 2021.

[13] Y. Shen, Z. Wang, H. Dong, and H. Liu, “Multi-sensor multi-rate fusion estimation for networked systems: Advances and perspectives,” *Information Fusion*, vol. 82, pp. 19–27, 2022.

[14] P. C. Young and J. Willems, “An approach to the linear multivariable servomechanism problem,” *International journal of control*, vol. 15, no. 5, pp. 961–979, 1972.

[15] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. M. O. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, “Continuous control with deep reinforcement learning,” *CoRR*, vol. abs/1509.02971, 2016.

[16] M. Blanke, M. Kinnaert, J. Lunze, and M. Staroswiecki, *Diagnosis and Fault-Tolerant Control*. Springer Berlin Heidelberg, 2015.

[17] J. Milošević, H. Sandberg, and K. H. Johansson, “Estimating the impact of cyber-attack strategies for stochastic networked control systems,” *IEEE Transactions on Control of Network Systems*, vol. 7, no. 2, pp. 747–757, 2019.