Received: 30 November 2021 | Revised: 26 December 2022 | Accepted: 19 January 2023

DOI: 10.1002/rnc.6619

RESEARCH ARTICLE

WILEY

Sliding mode based fault diagnosis with deep reinforcement learning add-ons for intrinsically redundant manipulators

Nikolas Sacchi[1] | **Gian Paolo Incremona**[2] | **Antonella Ferrara**[1]

[1]Dipartimento di Ingegneria Industriale e dell'Informazione, University of Pavia, Pavia, Italy

[2]Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Milano, Italy

Correspondence
Gian Paolo Incremona, Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Piazza Leonardo da Vinci 32, 20133 Milano, Italy.
Email: gianpaolo.incremona@polimi.it

Abstract

This article presents a fault diagnosis control scheme for intrinsically redundant robot manipulators based on the combination of a deep reinforcement learning (DRL) approach and a battery of sliding mode observers. The DRL plays the role of detecting and isolating possible sensor faults, thus generating an alarm and pin-pointing the source. This in turn allows to compensate the sensor faults independently from the actuator ones. The latter are therefore detected and isolated by a set of sliding mode observers driven by input laws designed according to an optimal reaching algorithm. In order to design and apply such observers, a global feedback linearization is performed, which transforms the multi-input-multi-output (MIMO) nonlinear robot model into a chain of double integrators. The proposal is analyzed and assessed in realistic conditions using the PyBullet environment in which a 7 degrees-of-freedom (DOFs) Franka Emika Panda robot manipulator is reproduced.

KEYWORDS

deep reinforcement learning, fault diagnosis, robotics, sliding mode

1 | INTRODUCTION

Fault diagnosis (FD) in industrial contexts is nowadays an essential task to reduce the risk of damage, tear and wears in electro-mechanical systems. More precisely, FD consists in determining the causes of a deviation of the control status from the desired behavior, and in interpreting such status given the measurements from sensors, or on the basis of the process model.[1] Furthermore, since the continuity of the work processes must be guaranteed, it is significantly important to develop suitable schemes capable of compensating such faults and corruptions affecting the systems. However, such a compensation can be made if the fault is correctly detected, isolated and identified, even under critical conditions or significant measurement noises. Precisely, the *detection* procedure allows to understand when a fault occurs in the system, without any knowledge on the specific faulty component. If also the specific corrupted component is observed and the time evolution of the fault signals is reconstructed, then the *isolation* and *identification* tasks are accomplished.

Several FD methodologies have been proposed in the literature, basically distinguishing between two categories, that is *passive* techniques and *active* ones (see, e.g., Reference 2 for an overview). In the case of passive approaches, independently of any fault information, an input signal is fed into the actual process and also into its nominal

model, and then the corresponding output signals are analyzed (see, e.g., References 3 and 4). On the other hand, active approaches exploit the injection of specific signals in order to improve the detectability of the faults (see References 5-7, among many others). A recent alternative is the so-called *projection methods* which is based on the definition of a set of controllers from which the suitable control law associated to the specific detected fault is selected, thus giving rise to a switched mechanism. Other methods are instead based on adaptation and rely on robust control approaches, ranging from $\mathcal{H}_\infty$ methods,[8] to model following and multimodel approaches,[9] using linear quadratic controllers (LQR)[10] or model predictive controllers (MPC),[11] and relying on neural networks,[12] to cite a few.

Among these techniques, also sliding mode control (SMC) has resulted as a particularly promising method for FD applications.[13] SMC is indeed well known for its robustness in front of a wide class of uncertain terms, and in particular in front of matched uncertain terms acting directly on the input channel.[14] This property in certain case allows to avoid a controller reconfiguration, but requires some assumption on the knowledge of possible faults. Therefore, SMC based observers have been studied and introduced to design fault tolerant control schemes. In fact, SMC observers are capable of providing a robust estimate of the fault signals through a suitable scaling and filtering of the so-called *equivalent output error injection*. More precisely, the concept of the so-called equivalent control[15] is exploited, and the fault identification is operated without requiring any trade-off between state estimation and fault sensitivity. This technique has been exploited in several works (see, for instance,[16,17] among many others) and several fault scenarios.

In the context of FD a particular attention is devoted to robotics machines, since nowadays robots are rapidly expending to numerous applications. Robots are however susceptible to different types of faults affecting both sensors and actuators.[18-20] Moreover, such faults can occur simultaneously on both these components and more than one fault at a time can perturb the system operations. This scenario is made further complex by the nonlinear coupled nature of the robot model,[21,22] so that, if in most FD literature such coupling effects in the system are considered negligible,[19,20,23,24] this is generally false in robotics. A possible solution to decouple the effects of the fault signals acting on sensors and actuators can be the introduction of additional vision sensors, such as cameras externally located with respect to the robot systems. Instead, among the adopted methodologies to address the FD problem, the use of sliding mode based approaches to robot manipulators has been also investigated, both in the case of genuine controllers (see e.g., References 25-30), or in the case of observer design (see, for instance, References 31 and 32).

In Reference 31, a vision sensor is involved to enhance the fault diagnosis capability of the proposed scheme relying on the use of SMC based observers. An extension of this work is proposed in Reference 32, where a planar robotic manipulator has been considered and the MIMO nonlinear coupled model is transformed into a set of decoupled linear systems using the so-called inverse dynamics approach.[33] Moreover, detection, isolation and identification of sensor faults have been carried out using a low-cost IP camera, by processing the images in order to extrapolate the correct position of the end-effector in case of fault alarms. Suboptimal second order sliding mode (SSOSM)[34] based unknown input observers (UIOs) have been also adopted.

In this work, we put forward a novel fault diagnosis control scheme capable of detecting, isolating and identifying both sensor and actuator faults in the case of intrinsically redundant robot manipulators. It is in fact well known that kinematically redundant robots have more degrees of freedom than it is needed to execute a given task, thus implying possible infinite solutions to the inverse kinematics problem. This makes the use of the vision sensor not sufficient to correctly compute the joint positions given the end-effector coordinates in space. The novelty of the proposal lies in the application of a DRL model-free based reconstruction of signals in order to correct the corrupted measurements from sensors before they are used by the robot controller and by a battery of sliding mode observers aimed at the actuator fault diagnosis. The considered scenario is related to the so-called *recoverability problem* (see e.g., References 35 and 36 and the references therein), according to which a sufficient redundancy on the robot is needed to accommodate fault situations. Specifically, this article aims at providing an alternative method to the design of a FD control scheme enabling the following features:

1. a DRL approach relying on the twin delayed deep deterministic policy gradient (TD3) algorithm is adopted to generate the map capable of estimating multiple sensor faults independently of the actuator faults;
2. the sensor fault signals are compensated to provide the correct joint positions, which enable to close the loop and perform the actuator fault diagnosis via dedicated observers;
3. exploiting the diagnosis decoupling provided by the DRL mechanism, a battery of sliding mode observers is applied for multiple actuator fault detection, isolation and identification;

4. differently from previous works,[31,32] where a suboptimal second-order sliding mode (SSOSM) approach was used, a second-order sliding mode (SOSM) with optimal reaching[37] is here adopted. If on the one hand, differently from References 31 and 32, an estimate of the joint acceleration is needed, the proposal allows to improve the fault estimation thanks to the minimum time convergence property that such an approach can guarantee versus the worst realization of the uncertainties.

The proposal is finally assessed in a realistic simulation scenario created in the PyBullet environment, in which a 7 DOFs Franka Emika Panda robot manipulator is recreated using data provided by a real setup.

The article is organized as follows. In Section 2, the considered robot model and the faulty scenario are described, as well as the addressed problem is formulated. In Section 3 the adopted control scheme is described, while in Section 4 the proposed FD approach characterized by the DRL mechanism and the SOSM UIOs is discussed in detail and analyzed. Simulation results are illustrated and commented in Section 5, while some conclusions are gathered in Section 6.

2 | PROBLEM FORMULATION

In this section the considered robot model is presented and the faulty scenario is described.

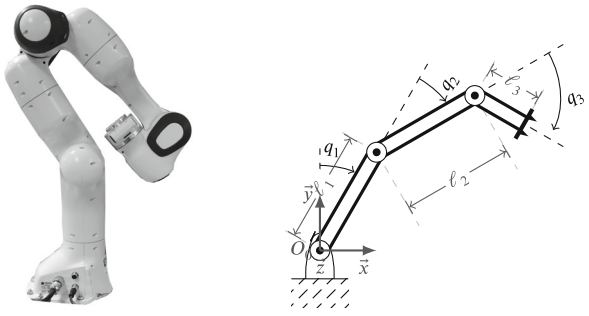
2.1 | Robot kinematic and dynamics

In this work, an intrinsically redundant robot manipulator is considered (see Figure 1), that is the dimension of the operational space is smaller than the dimension of the joint space[33(Ch. 2)]. The robotic system consists of n_q revolute joints and $n_q + 1$ links. The joint variables vector is denoted as $q \in \mathbb{R}^{n_q}$, where the orientation of the first link with respect to $\vec{y}$-axis is clockwise positive, while the displacement of the generic jth link, $j \neq 1$, is clockwise positive with respect to the $(j-1)$th one. Then, let $O_0, \{\vec{x}, \vec{y}, \vec{z}\}$, be the base-frame for the robotic manipulator, so that the center O_0 is placed in the center of the first joint of the robot, while $O_e, \{\vec{n}, \vec{s}, \vec{a}\}$ is the end-effector frame, with center O_e placed on the robot end-effector and associated to the axes $\{\vec{n}, \vec{s}, \vec{a}\}$. In the following, the proposed control scheme exploits both the kinematics and dynamics of the robot, defined as follows.

As for the kinematic model, the so-called *direct kinematics* to determine the pose of the end-effector with respect to the base frame can be computed through the homogeneous transformation matrix $T_0^e \in \mathbb{R}^{4\times4}$ given by

$$T_0^e(q) = \begin{bmatrix} R_0^e(q) & p_0^e(q) \\ 0_{1\times3} & 1 \end{bmatrix}, \tag{1}$$

where $p_0^e(q) \in \mathbb{R}^3$ is the position of the end-effector frame with respect to the base frame, $R_0^e(q) \in \mathbb{R}^{3\times3}$ is the rotation matrix between the two frames, while $0_{1\times3}$ is a row zero vector.



(a) (b)

FIGURE 1 Franka Emika Panda robot manipulator. Setup (a). Schematic view of a planar manipulator with three joints (b).

The dynamics of the n_q-joints robot manipulator is instead captured by

$$B(q)\ddot{q} + n(q, \dot{q}) = \tau, \tag{2a}$$

$$n(q, \dot{q}) = C(q, \dot{q})\dot{q} + F_v\dot{q} + F_s\text{sgn}(\dot{q}) + g(q), \tag{2b}$$

where $\dot{q} \in \mathbb{R}^{n_q}$ is the joint velocity vector, $\ddot{q} \in \mathbb{R}^{n_q}$ is the corresponding joint acceleration vector, $B(q) \in \mathbb{R}^{n_q \times n_q}$ is the manipulator inertia matrix, $C(q, \dot{q}) \in \mathbb{R}^{n_q \times n_q}$ is the matrix containing the Coriolis and centripetal effects, $F_v \in \mathbb{R}^{n_q \times n_q}$ is the viscous friction matrix, $F_s \in \mathbb{R}^{n_q \times n_q}$ is the static friction matrix, $g(q) \in \mathbb{R}^{n_q}$ is the vector which takes into account gravitational forces and torques, while $\tau \in \mathbb{R}^{n_q}$ is the vector of the motor torques.

2.2 | Faults modeling

In this article, faults on both sensors and actuators are considered. A fault on the ith sensor is modeled as an error $\Delta q_i \in \mathbb{R}$ which is added to the value of the ith element of the vector q. Analogously, a fault on the ith actuator is modeled as an additive term $\Delta\tau_i \in \mathbb{R}$ which sums up with the ith element of the torque vector τ. Moreover, letting $\Delta q \in \mathbb{R}^{n_q}$ and $\Delta\tau \in \mathbb{R}^{n_q}$ be the vectors containing sensor and actuator faults, the following assumption holds.

Assumption 1 (Faults boundedness). The faults vectors $\Delta q \in \mathbb{R}^{n_q}$ and $\Delta\tau \in \mathbb{R}^{n_q}$ are bounded, that is,

$$\Delta q \in \mathcal{Q} \subset \mathbb{R}^{n_q}, \tag{3a}$$

$$\Delta\tau \in \mathcal{T} \subset \mathbb{R}^{n_q}, \tag{3b}$$

with $\mathcal{Q}$ and $\mathcal{T}$ being compact sets containing the origin and limited by $\mathcal{Q}^{\text{sup}} = \sup_{\Delta q \in \mathcal{Q}}\{\|\Delta q\|\}$ and $\mathcal{T}^{\text{sup}} = \sup_{\Delta\tau \in \mathcal{T}}\{\|\Delta\tau\|\}$, respectively.

If single or multiple sensor faults (SFs) occur, the vector of the joint variables containing such anomalies is indicated as

$$\overline{q} \coloneqq q + \Delta q, \tag{4a}$$

while motor torques which are affected by actuator faults (AFs) are

$$\overline{\tau} \coloneqq \tau + \Delta\tau. \tag{4b}$$

2.3 | Problem statement

We are now in a position to introduce the considered scenario and formulate the problem to solve. Consider any possible task in the operational space described by $m < n_q$ variables, that is more DOFs than task variables are available. Assume that all the joint variables are measurable from the encoders fastened on the robot joints, and these are susceptible to faults, while an external sensor (e.g., a vision sensor) is capable to directly retrieve only the end-effector position p_0^e without being affected by any possible fault. Furthermore, assume that, during the task execution, the fault events (FEs) reported in Table 1 can occur.

TABLE 1 Possible FEs during the task execution.

FE #	Description
1	Single fault on an actuator
2	Single fault on a sensor
3	Multiple faults on actuators
4	Multiple faults on sensors
5	Multiple mixed faults

Therefore, the goal is to design a FD scheme capable of providing at each time instant t the estimates $\widehat{\Delta q} \in \mathbb{R}^{n_q}$ and $\widehat{\Delta \tau} \in \mathbb{R}^{n_q}$ of the sensor and actuator faults in order to minimize the following performance index

$$J := r_{\mathrm{s}} + r_{\mathrm{a}}, \tag{5a}$$

where the first term is related to the sensor fault estimation error, that is,

$$r_{\mathrm{s}} := \|\widehat{\Delta q} - \Delta q\|, \tag{5b}$$

while the second one corresponds to the actuator fault estimation error, that is

$$r_{\mathrm{a}} := \|\widehat{\Delta \tau} - \Delta \tau\|. \tag{5c}$$

3 | ADOPTED ROBUST CONTROL SCHEME

Before introducing the proposed FD approach, it is instrumental to clarify the role of the controller in the overall control scheme.

3.1 | Inverse dynamics

In order to achieve an equivalent controlled system behaving as the union of double integrators, the so-called inverse dynamics approach is applied.[33] At this stage, assume that $\Delta q = 0$ and $\Delta \tau = 0$ (i.e., no faults occur), and consider the control scheme in Figure 2. For the sake of simplicity, without losing any generality for the applicability of the proposal, the following assumption is considered.

Assumption 2 (Model knowledge). Both the inertia matrix $B(\cdot) \in \mathbb{R}^{n_q \times n_q}$ and the force vector $n(\cdot, \cdot) \in \mathbb{R}^{n_q}$ are known.

The inverse dynamics of a robot manipulator can be expressed as a nonlinear relationship between plant inputs and outputs. Given an auxiliary control vector $y \in \mathbb{R}^{n_q}$ dimensionally equal to an acceleration, the control torque is designed as

$$\tau = B(q)y + n(q, \dot{q}). \tag{6}$$

Hence, by applying (6) into (2), and by virtue of Assumption 2, one obtains

$$\ddot{q} = y, \tag{7}$$

that is a chain of n_q decoupled double integrators, one for each joint of the robot.

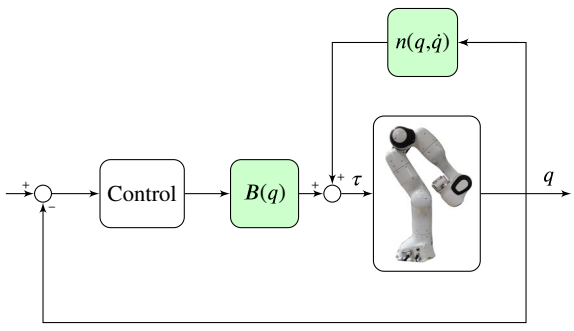


FIGURE 2 The adopted control scheme for the considered Franka Emika Panda robot manipulator.

3.2 | Controller design

We are now in a position to introduce the control algorithm. In fact, if Assumption 2 holds and the inverse dynamics law (6) is applied, the controller can be designed for each joint in order to stabilize an unperturbed double integrator. However, having in mind faulty scenarios, the inverse dynamics law affected by faults becomes

$$\tau = B(\overline{q})y + n(\overline{q}, \dot{\overline{q}}). \tag{8}$$

By adding the actuator faults $\Delta\tau$ to (8), and substituting it into (2), one has that the equivalent controlled system is

$$\ddot{q} = B^{-1}(q)B(\overline{q})y + B^{-1}(q)(n(\overline{q}, \dot{\overline{q}}) - n(q, \dot{q})) + B^{-1}(q)\Delta\tau = \eta(q, \dot{q}, \overline{q}, \dot{\overline{q}}) + \Delta y + G(q, \overline{q})y, \tag{9}$$

where $\eta(q, \dot{q}, \overline{q}, \dot{\overline{q}}) = B^{-1}(q)(n(\overline{q}, \dot{\overline{q}}) - n(q, \dot{q})) \in \mathbb{R}^{n_q}$ and $G(q, \overline{q}) = B^{-1}(q)B(\overline{q}) \in \mathbb{R}^{n_q \times n_q}$ are unknown vector fields, while the corresponding acceleration fault is given by

$$\Delta y = B^{-1}(q)\Delta\tau. \tag{10}$$

Therefore, the application of the inverse dynamics linearization law in presence of faults results in being a coupled uncertain MIMO system. The following assumption needs now to be introduced.

Assumption 3 (Boundedness of vector fields). The control effectiveness matrix $G(q, \overline{q}) \in \mathbb{R}^{n_q \times n_q}$ and the drift term $\eta(q, \dot{q}, \overline{q}, \dot{\overline{q}}) \in \mathbb{R}^{n_q}$ are bounded, that is,

$$\|G(q, \overline{q})\|_\infty \leq \Gamma, \tag{11a}$$

$$\eta(q, \dot{q}, \overline{q}, \dot{\overline{q}}) \in \mathcal{H}, \tag{11b}$$

with $\Gamma > 0$ and $\mathcal{H}$ being a compact set containing the origin, limited by $\mathcal{H}^{\sup} = \sup_{\eta \in \mathcal{H}}\{\|\eta\|\}$, respectively.

Moreover, from (8), the acceleration fault Δy is bounded by virtue of the boundedness of $\Delta\tau$.

Assumption 4 (Boundedness of the acceleration fault). The acceleration fault Δy is bounded, that is,

$$\Delta y \in \mathcal{Y}, \tag{12}$$

with $\mathcal{Y}$ being a compact set containing the origin, limited by $\mathcal{Y}^{\sup} = \|B^{-1}(q)\|_\infty \mathcal{T}^{\sup}$.

The previous assumptions are instrumental to design a suitable controller capable of guaranteeing the closed-loop stability in presence of model mismatches. A viable solution can be a SMC law for MIMO systems, see for example, References 38 and 39. Note that, the design of the controller is out of the scope of the present paper, which proposes a FD scheme aimed at detecting, isolating and identifying sensor and actuator faults, without compromising the closed-loop stability of the control system.

4 | PROPOSED FAULT DIAGNOSIS SCHEME

In order to solve the problem discussed in Section 2, in this work, we propose the FD control scheme illustrated in Figure 3.

Starting from the scheme in Figure 2, two additional key elements are included: the DRL based block for sensor faults diagnosis, and the battery of SOSM UIOs. In the following, some elements about the DRL approach are given before describing our proposal. Then, the SOSM UIOs are described and analyzed.

4.1 | Elements of reinforcement learning

The basic concept underlying any reinforcement learning (RL) algorithm is that a decision maker, called *agent*, interacts with an *environment* while performing a sequence of *actions* chosen according to a certain strategy, called *policy*. The

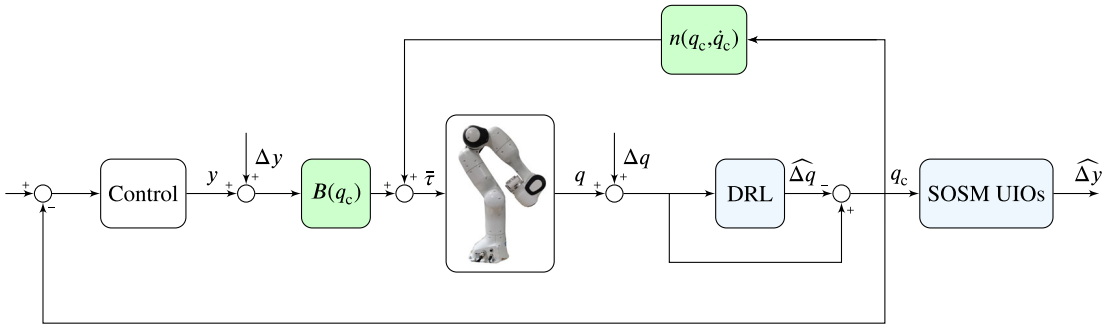


FIGURE 3 The proposed FD scheme for the considered Franka Emika Panda robot manipulator.

goodness of each action depends on the so-called *reward* function, which is assigned to the agent by the environment. The value of such a reward in turn depends on the task that the decision maker must perform. The main goal of the agent is to improve its policy in order to perform a sequence of actions that maximize not the single reward received after each action, but a long term reward which takes into account all the actions that have been performed during a certain time horizon. In order to do that, it is important that the agent *exploits* the actions that in the past lead to a higher instantaneous reward while *exploring* new actions that could lead to better results.

As for the environment, it is characterized by a certain set of *state* variables $s \in \mathcal{S}$, with $\mathcal{S}$ being the state space. The action that the agent performs is instead represented by a set of variables $a \in \mathcal{A}$, with $\mathcal{A}$ being the so-called action space. At each time instant t, the environment is described with a state $s_t \in \mathcal{S}$, and the agent, according to a policy $\pi(a_t|s_t)$, selects a certain action $a_t \in \mathcal{A}$ and, on the basis of a *transition function* $P(s_{t+1}|s_t, a_t)$, the environment changes its state from s_t to $s_{t+1} \in \mathcal{S}$. Depending on the landing state s_{t+1}, the environment gives a reward $r_{t+1} \in \mathcal{R}$, where $\mathcal{R}$ is the so-called *reward space*.

Therefore, given the current state s, the action a, and the next state indicated as s', the expected value of the reward, that is, $\mathbb{E}\{\cdot\}$, can be computed as

$$r(s, a, s') \coloneqq \mathbb{E}\{r_{t+1}|s_t, a_t, s_{t+1}\}. \tag{13}$$

The ultimate goal of the agent is then to maximize the so-called *cumulative reward* R_t, which takes into account all the rewards collected during the time horizon T_h. In particular, one has

$$R_t \coloneqq \sum_{k=0}^{T_\mathrm{h}} \gamma^k r_{t+k+1}, \tag{14}$$

where $0 < \gamma \leq 1$ is the *discount factor* which prioritizes earlier rewards over later ones. Given a policy $\pi(a|s)$, it is possible to compute the *value* of a state s, denoted as $V^\pi(s)$, and defined as the expected cumulative reward if the agent starts in state s and chooses actions according to policy π, that is,

$$V^\pi(s) \coloneqq \mathbb{E}_\pi\{R_t|s_t = s\}, \tag{15}$$

with R_t as in (14). Analogously, it is possible to define the value of an action a in a certain state s by computing the expected cumulative reward starting in s, taking action a and then following a policy π thereafter. Such value is called *action-value function* for policy π and it is defined as

$$Q^\pi(s, a) \coloneqq \mathbb{E}_\pi\{R_t|s_t = s, a_t = a\}. \tag{16}$$

If the state space $\mathcal{S}$ and the action space $\mathcal{A}$ have finite dimension, it is possible to approximate the action value function using the so-called *Q-Table*. Such a table has a number of rows equal to the number of states and a number of columns corresponding to the number of possible actions. The Q-Table can be then easily approximated using *Q-Learning* algorithms.[40] If instead the state space becomes too large or continuous, building the Q-Table becomes unfeasible, and a parametric approximator for the action-value function must be used. A way to build such an approximator is the use of a *deep neural network* (DNN), that is a parametric function which is able to model complex nonlinear relationships.

When dealing with DNN, it is possible to approximate the action-value function also using *deep Q network* (DQN) algorithms.[41] However, DQNs are no longer sufficient when dealing with continuous spaces. In these cases, several algorithms such as *deep deterministic policy gradient* (DDPG)[42] and *Twin Delayed Deep Deterministic Policy Gradient* (TD3)[43] can be used.

In this work, having in mind a robotic application with continuous state space, the TD3 algorithm is adopted (see Algorithm 1). It is aimed at concurrently learning the optimal action value function and the policy. It uses target networks and a replay buffer $\mathcal{B}$, and it presents some peculiar characteristics that allow to improve the performances with respect to other similar approaches such as DDPG. The first one is that it uses two Q-functions, approximated by two *critic* networks, Q_{θ_1} and Q_{θ_2}, parametrized by two sets of weights θ_1 and θ_2. Given a state s and an action $\bar{a}$, chosen according to a policy π_ϕ, the smaller of the two Q-values is used to represent the targets in the Bellman error loss functions (lines 13-16 in Algorithm 1). The second characteristic is that the update of the parameter ϕ of the *actor* network π_ϕ and the update of the target networks are not done every time step but with a certain frequency f_π (lines 17-23 in Algorithm 1). Finally, the third peculiarity is that, in order to reduce the risk that the policy exploits some error in the Q-functions, TD3 adds noise to the target action (see line 12 in Algorithm 1).

Algorithm 1. TD3 algorithm[43]

1: Initialize the "critic" networks Q_{θ_1}, Q_{θ_2}, and the "actor" network π_ϕ, with random parameters θ_1, θ_2 and ϕ
2: Initialize the target network $\theta'_1 \leftarrow \theta_1$, $\theta'_2 \leftarrow \theta_2$ and $\phi' \leftarrow \phi$
3: Initialize the replay buffer $\mathcal{B} \leftarrow 0$
4: Set the episode counter $\iota_{\text{ep}} = 1$
5: Set the iteration counter $\iota_{\text{it}} = 0$
6: **for** $\iota_{ep} \leq N_{\text{ep}}$ **do**
7: **for** $t \leq t_{\text{ep}}$ **do**
8: Select the action with added exploration noise $a \leftarrow \pi_\phi(s) + \epsilon$ with $\epsilon \sim \mathcal{N}(0, \sigma_{\text{e}})$
9: Observe the reward r and new state s'
10: Store the tuple $(s, a, r, s') \in \mathcal{B}$
11: Sample a mini-batch $\mathcal{M}$ of N elements (s, a, r, s') from $\mathcal{B}$
12: $\bar{a} \leftarrow \pi_\phi(s') + \epsilon$ with $\epsilon \sim \text{clip}(\mathcal{N}(0, \sigma_{\text{t}}), -c, c)$, $c > 0$
13: $y \leftarrow r + \gamma \left(\min\left(Q_{\theta'_1}(s', \bar{a}), Q_{\theta'_2}(s', \bar{a})\right)\right)$
14: Update the "critic" networks
15: $\theta_1 \leftarrow \arg\min_{\theta_1} \frac{1}{N} \sum_{(s,a,r,s') \in \mathcal{M}} (y - Q_{\theta_1}(s, a))^2$
16: $\theta_2 \leftarrow \arg\min_{\theta_2} \frac{1}{N} \sum_{(s,a,r,s') \in \mathcal{M}} (y - Q_{\theta_2}(s, a))^2$
17: **if** $\iota_{\text{it}} \bmod f_\pi$ **then**
18: Update ϕ using deterministic policy gradient $\nabla_\phi J(\phi) = \frac{1}{N} \sum_{s \in \mathcal{M}} \nabla_a Q_{\theta_1}(s, a)|_{a=\pi_\phi(s)} \nabla_\phi \pi_\phi(s)$
19: Update target networks
20: $\theta'_1 \leftarrow \tau\theta_1 + (1 - \tau)\theta'_1$
21: $\theta'_2 \leftarrow \tau\theta_2 + (1 - \tau)\theta'_2$
22: $\phi' \leftarrow \tau\phi + (1 - \tau)\phi'$
23: **end if**
24: $\iota_{\text{it}} = \iota_{\text{it}} + 1$
25: **end for**
26: **end for**

4.2 | Fault diagnosis for sensors: DRL algorithm

In order to perform detection, isolation and identification of faults concurrently occurring on multiple sensors, a model-free DRL agent is designed. Although the application of the proposal to an intrinsically redundant robot might increase the computational complexity for the DRL, its training phase is easily performed off-line, and it has the advantage to overcome possible issues due to the kinematic inversion which could lead to infinite solutions.

Remark 1. Note that, if the fault detection concerns only sensor faults, then, one could use a model based approach when the model is available. For example, one could apply the control signal τ both to the robot and to the model (2) and evaluate the differences in the output signals. In our case, the possible contemporary presence of actuator faults and sensor faults require to invent a method to decouple the effects of the two types of faults to allow for the fault diagnosis activity. ▽

Making reference to the previous section, the environment is represented by the state space $\mathcal{S}$ given by

$$\mathcal{S} \coloneqq \left\{ p_0^{\mathrm{e}}, \overline{q} \right\}, \tag{17}$$

depending on the end-effector position expressed in the base reference frame, and the joint variable vector affected by faults, as defined in Section 2.2. Note that, assuming to know the position of the end-effector as a state is quite reasonable in practice because it can be recovered either in simulation or by using, for instance, a stereo camera. The objective of the agent is therefore to provide at each time time step an estimate $\widehat{\Delta q} \in \mathbb{R}^{n_q}$ of the sensor faults. The bounded action space $\mathcal{A}$ for the DRL agent is instead defined by

$$\mathcal{A} \coloneqq \left\{ \widehat{\Delta q}_1, \widehat{\Delta q}_2, \ \dots \ , \widehat{\Delta q}_{n_q} \right\}, \tag{18}$$

with $\widehat{\Delta q}_i$ being the estimate of the fault on the ith sensor. Now, the following assumptions need to be introduced.

Assumption 5 (Boundedness of sensor fault estimation error). The sensor fault estimation error $\widehat{\Delta q} - \Delta q$ is bounded, that is there exists $E > 0$ such that

$$\| \widehat{\Delta q} - \Delta q \|_\infty \leq E. \tag{19}$$

Assumption 6 (Relationship between sensor fault estimation errors and end-effector position). Sensor fault estimation errors depend on the actual end-effector position through some unknown static function. This implies that when the end-effector is in a certain position then the associated sensor errors take always the same value.

Note that Assumption 6 is always true for constant sensor faults.

In order to train the agent, exploiting a realistic simulator of the robot, different sensor fault signals $\Delta q \in \mathbb{R}^{n_q}$ are injected into the system, and the reward function, used to assign a value to the action, is designed as

$$r_t = -r_{\mathrm{s}}, \tag{20}$$

where r_{s} is as in (5b), while the minus sign in the reward indicates that it is considered as a penalty.

During the training phase, the simulated robot is asked to follow random trajectories using the closed-loop scheme. At each time step, data from sensors are retrieved and constant faults $\Delta q_i, \ i = 1, \ \dots \ , n_q$ with random amplitude are added. The result is fed into the DNN together with the end-effector pose, which generates estimates for the fault, and, depending on the reward, updates its weights. Once the training phase is completed, the weights of the DNN are kept constant for the online test phase. Note that, since the auxiliary control y does not belong to $\mathcal{S}$, it does not affect the estimates of the sensor faults.

Once the DRL agent provides an estimate $\widehat{\Delta q}$, it is possible to perform both detection and isolation by building a vector $f_{\mathrm{s}} \in \mathbb{R}^{n_q}$, which contains a flag for each sensor component, that is

$$f_{\mathrm{s},i} = \begin{cases} 0, & \text{if } \underline{Q}_{\mathrm{s},i} \leq \widehat{\Delta q}_i \leq \overline{Q}_{\mathrm{s},i} \\ 1, & \text{otherwise} \end{cases}, \tag{21}$$

where $\underline{Q}_{\mathrm{s}_i}$ and $\overline{Q}_{\mathrm{s}_i}$ are constants indicating the ith lower and upper thresholds, respectively. Note that, in an ideal scenario, such thresholds are equal to zero, but noise on the sensors and on the vision system must be taken into account. Moreover, by analyzing the time evolution of the signals $\widehat{\Delta q}$, it is possible to perform a fault identification, thus compensating the corrupted signals in order to close the control loop. It is worth to highlight that this fault diagnosis on sensors is independent on the actuator faults. In other words, by virtue of the DRL mechanism the sensor fault can be

compensated, thus improving the fault diagnosis of the actuators failures. Specifically, let q_c be the joint position vector used to close the feedback, by virtue of the DRL based identification, one has

$$q_{c_i} = \begin{cases} \overline{q}_i - \widehat{\Delta q}_i, & \text{if } f_{s,i} = 1 \\ q_i, & \text{otherwise} \end{cases}, \quad \forall\, i = 1, \dots, n_q. \tag{22}$$

Remark 2. Note that, the previous definition of the corrected joint variables q_c means that, if no fault occurs, its ith component is equal to the actual position of the robot joint, otherwise, q_{c_i} is the corrected position after the compensation of the ith sensor fault. Relying on the control scheme described in Section 3, such a value is used to close the inner inverse dynamics loop and the outer feedback loop. If the DRL estimate $\widehat{\Delta q}_i$ differs from the actual fault Δq, since by Assumption 5, the estimation error is bounded, the controlled system becomes the one in (9), and, as mentioned in Section 3, it is assumed that a suitable robust controller is designed to guarantee the closed-loop stability even in this case. ▽

4.3 | Fault diagnosis for actuators: SOSM UIOs

In order to design the battery of observers aimed at the fault diagnosis on the actuators, a canonical form for the state model of the plant has to be introduced. This step indeed enables to describe the dynamics of the tracking error in a suitable form for SMC design.

Therefore, exploiting the nominal linearized model of the robot in (7) when no fault occurs, and using the corrected joint variables measures q_c such that $\ddot{q}_c = y + \Delta y$, one can design a battery of observers, one for each joint, in order to estimate possible actuator faults. Define the UIO model as

$$\ddot{\hat{q}}_c = y + u, \tag{23}$$

where $\hat{q}_c \in \mathbb{R}^{n_q}$ are the states of the observer, and $u \in \mathbb{R}^{n_q}$ is the observer input. Consider $e_{1,i} = q_{c,i} - \hat{q}_{c,i}$ and $e_{2,i} = \dot{e}_{1,i}$, for any $i = 1, \dots, n_q$. We are now interested in designing a sliding mode based observer capable of enabling an optimal reaching time of the estimation error which has to be steered to zero, and a minimum time convergence whenever the worst realization of the uncertain terms occurs. Making reference to Reference 37, a SOSM law is hereafter designed.

Define the sliding variable as the linear combination of the estimation errors as

$$\sigma_{1,i}(t) = \beta e_{1,i}(t) + e_{2,i}(t), \quad \forall\, i = 1, \dots, n_q, \tag{24}$$

where $\beta > 0$ is constant. By computing the time derivative of the sliding variable $\sigma_{1,i}$, relying on (23), the relative degree is equal to 1, so that a first order sliding mode law would apply. Since our goal is to design a SOSM law according to Reference 37, we introduce an auxiliary system with relative degree 2, given by

$$\begin{cases} \dot{\sigma}_{1,i}(t) = \sigma_{2,i}(t), \\ \dot{\sigma}_{2,i}(t) = f(e_{2,i}, q_{c,i}, \dot{y}_i) - v_i(t), \\ v_i(t) = \dot{u}_i(t), \end{cases} \tag{25}$$

where v_i is the new observer input, while $f(e_{2,i}, q_{c,i}, \dot{y}_i) = \beta_i \dot{e}_{2,i}(t) + \frac{d^{(3)} q_{c,i}(t)}{dt^3} - \dot{y}_i$. Due to the mechanical nature of the robotic system, the following assumption holds.

Assumption 7 (Boundedness of the auxiliary drift term). The drift term $f(e_{2,i}, q_{c,i}, \dot{y}_i) \in \mathbb{R}$ is bounded, that is,

$$|f(e_{2,i}, q_{c,i}, \dot{y}_i)| \leq F_i, \tag{26}$$

where $F_i > 0$ is the bound associated to the ith joint.

The auxiliary control input v_i is therefore designed as

$$v_i(t) = \alpha_i \text{sign}\left(\sigma_{1,i}(t) + \frac{\sigma_{2,i}(t)|\sigma_{2,i}(t)|}{2\alpha_{r,i}}\right), \tag{27}$$

where $\alpha_i > F_i$ are the control gains, while $\alpha_{r,i}$ are the so-called *reduced* control amplitudes defined as

$$\alpha_{r,i} := \alpha_i - F_i > 0, \tag{28}$$

that is the minimum control amplitudes to cope with the worst realization of the drift uncertain terms. The following result proves the minimum time convergence of the sliding variables to zero.

Proposition 1 (Convergence). *Consider system (2) with the inverse dynamics law in (6) based on measurements (22) provided by the DRL mechanism. If Assumption 7 holds and $f(e_{2,i}, q_{c,i}, \dot{y}_i) = F_i \text{sign}(v_i)$, then each of the components $\sigma_{1,i}(t)$ of the auxiliary system (25), controlled by (27), is steered to zero in minimum time.*

Proof. The proof of the minimum-time convergence to the origin of the space $\{\sigma_{1,i}, \sigma_{2,i}\}$ directly follows from Reference 44(Th. 2) and 45(Th. 2). Consider for the sake of simplicity that the initial condition is $(\sigma_{1,i}(0), \sigma_{2,i}(0))$ such that $\sigma_{1,i}(0) > -\frac{\sigma_{2,i}(0)|\sigma_{2,i}(0)|}{2\alpha_{r,i}}$ and $\sigma_{2,i}(0) > 0$ (all the other symmetric or specular cases are analogous). Since $\sigma_{1,i} = -\frac{\sigma_{2,i}|\sigma_{2,i}|}{2\alpha_{r,i}}$ corresponds to the minimum time curve in the nominal case, one has to prove that in the case of the worst realization of the uncertain terms, the auxiliary state trajectory under the control law (27) follows this curve, in an equivalent sense. In fact, computing the vector field, one has $[\sigma_{2,i}, F_i - \alpha_i] = [\sigma_{2,i}, -\alpha_{r,i}]$, that is the trajectory moves towards the curve, while pointing downward. When the curve $\sigma_{1,i} = -\frac{\sigma_{2,i}|\sigma_{2,i}|}{2\alpha_{r,i}}$ is reached, the control sign changes and the vector field becomes $[\sigma_{2,i}, -F_i + \alpha_i] = [\sigma_{2,i}, \alpha_{r,i}]$, with $\sigma_{2,i} < 0$. As a consequence this means that the trajectory is always tangent to the curve, and the state moves towards the origin in minimum time. ∎

The previous result guarantees the finite-time convergence to zero of the sliding variable and its derivatives, in turn implying that the error signals $e_{1,i}(t)$ and $e_{2,i}(t)$ exponentially decay to zero constrained to $\sigma_{1,i}(t) = 0,\ \forall t \geq \underline{t}$, with $\underline{t} \geq 0$ being the convergence time. The following proposition shows instead that, once in sliding mode $\sigma_{1,i} = \sigma_{2,i} \equiv 0$, the equivalent control, namely $\tilde{u}_i$, allows to identify the ith actuator fault.

Proposition 2 (Actuator faults identification). *Consider the auxiliary system (25), controlled by (27). If $\sigma_{1,i} = \sigma_{2,i} \equiv 0$, $\forall\ i = 1, \dots, n_q$, then it yields*

$$\lim_{t\to\infty} r_a(t) = 0, \tag{29}$$

with r_a defined in (5c).

Proof. In order to prove the proposition, we need to exploit the concept of equivalent control in Reference 15. More precisely, the ith element of the equivalent control signal $\tilde{u}_i$ can be computed by solving $\dot{\sigma}_{1,i} = 0$, that is,

$$\beta_i e_{2,i} + y_i + \Delta y_i - y_i - \tilde{u}_i = 0, \tag{30}$$

so that one has

$$\tilde{u}_i = \beta_i e_{2_i} + \Delta y_i. \tag{31}$$

Since by virtue of Proposition 1, enforcing the sliding mode $\sigma_{1,i} = \sigma_{2,i} \equiv 0$ implies the error $e_{2,i}$ exponentially going to zero, therefore

$$\tilde{u}_i^{\infty}(t) = \lim_{t\to\infty} \tilde{u}_i(t) = \Delta y_i(t), \tag{32}$$

meaning that, given $\Delta y = B^{-1}\Delta\tau$ so that $\tilde{u} = B^{-1}\widehat{\Delta\tau}$, as t goes to infinity then $\Delta\tau - \widehat{\Delta\tau} = 0$, in turn implying $r_a = 0$, which concludes the proof. ∎

Making reference to the previous proposition, a practical way to obtain the equivalent control is to apply a low-pass filter to the discontinuous control input (27). However, in our case, we do not need to add a filter because, despite the auxiliary control signal $v_i(t)$ is discontinuous, the observer input $u_i(t)$ is continuous and equal to

$$u_i(t) = \int_0^t v_i(z)\, \mathrm{d}z. \tag{33}$$

This means that the filtering operation is performed by the integrator in (33), and it is possible to conclude that the signal $u_i(t)$ can be used as an estimate of the actual actuator fault $\Delta y_i(t)$.

Analogously to the sensor fault diagnosis, it is possible to perform the fault detection and isolation by defining a vector $f_a \in \mathbb{R}^{n_q}$ such that

$$f_{a,i} = \begin{cases} 0, & \text{if } \underline{Y}_{a,i} \leq u_i \leq \overline{Y}_{a,i} \\ 1, & \text{otherwise} \end{cases} \tag{34}$$

with $\underline{Y}_{a_i}$ and $\overline{Y}_{a_i}$ being the ith lower and upper thresholds, respectively.

Remark 3. In order to tune the ith SOSM UIO, the knowledge of the bound of the drift term F_i, and of the reduced control amplitude $\alpha_{r,i}$ is required. The former can be estimated through data retrieved directly from experiments on the system. Then, the reduced control amplitude $\alpha_{r,i}$ can be computed according to (28), selecting $\alpha_i > F_i$. As for the residual threshold in (34), in order to avoid some intermittent detection when oscillating faults occur, possible practices could be the reduction of the thresholds (taking always into account the bounds on the modeling mismatches in order to avoid false detections), or the adoption of a waiting time between two subsequent detection moments (in this second case one has to accept possible misdetections). ▽

5 | CASE STUDY

In this section the performance of the proposed FD scheme are discussed relying on simulations carried out on PyBullet environment, which is a module used for physics simulation, robotics and DRL algorithms, developed on the Bullet Physics SDK. In particular, sensor and actuator faults have been injected in a virtualized model of the 7 DOFs Franka Emika Panda robot manipulator, which has to reach a specific point in the working space (see Figure 4).

5.1 | DRL training

As discussed in Section 4.2, the DRL algorithm to execute a sensor fault diagnosis is based on the TD3 agent, while the state space $\mathcal{S}$ and the action space $\mathcal{A}$ are defined as in (17) and (18). The training phase is divided into episodes, and each episode has a fixed duration of 5 s such that at each time-step the simulation advances of 4.2 ms, and a vector of random constant sensor faults Δq is injected into the system. Specifically, such faults belong to the set $\mathcal{Q} = \{\Delta q_i \in \mathbb{R} | -0.35 \leq \Delta q_i \leq 0.35, \ \forall\, i = 1, \dots, n_q\}$. Notice that the choice of considering constant sensor faults, even if the proposed approach can also deal with more general cases, is motivated by the fact that constant signals are commonly used as sensor faults to emulate a malfunction given by a bias in different applications, see, for example, Reference 46 for a classification of faults in robotics. The reward function is defined as in (20). Both the actor and critic are approximated by using multilayer perceptron (MLP) artificial neural networks, with an input layer composed of 10 neurons, 2 hidden layers composed of

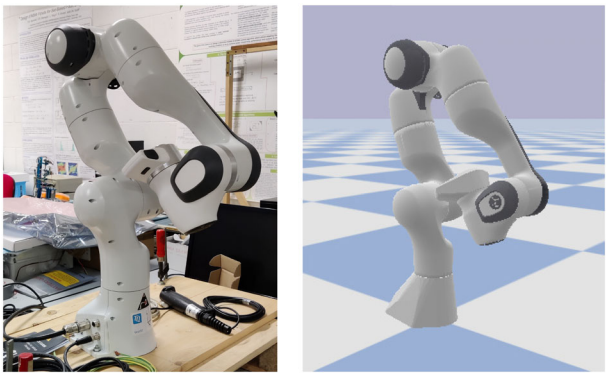


FIGURE 4 Laboratory setup (left) and virtual robot in PyBullet environment (right).

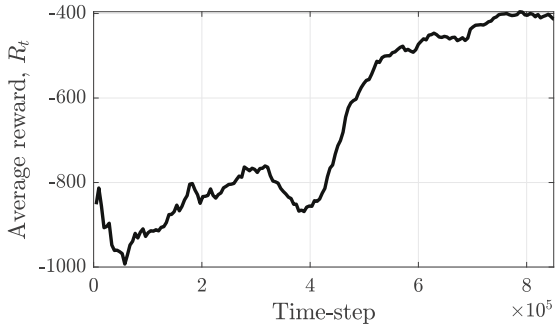


FIGURE 5 Average cumulative reward R_t during the training phase.

TABLE 2 RMS estimation errors for scenario FE3.

Joint #	RMS $(\widehat{\Delta y} - \Delta y)$ [rad s^{-2}]
1	0.1809
3	0.2153
4	0.1699
6	0.1863

64 neurons, and an output layer with 7 neurons. Figure 5 illustrates the average cumulative reward R_t during the training phase. One can notice that the reward is maximized, as expected, meaning that the agent learns how to perform the fault diagnosis whenever a sensor failure occurs.

5.2 | Simulation results

In order to assess the proposal, the behavior of the whole FD scheme in Figure 3 has been tested in case of scenarios FE3, FE4, and FE5 reported in Table 1. The initial values of the sensors estimates are set equal to the sensor readings, while they are equal to zero for the actuator fault estimates. The parameters of the SOSM UIOs are selected as $\alpha_i = 150$, that is, greater than $F_i = 50$, $i = 1, \dots, n_q$, and $\beta = 1$. The adopted control scheme consists of the inverse dynamics inner loop as discussed in Section 3, while the controller of the outer loop is the one selected by default in the simulator of the considered Franka Emika Panda robot manipulator, that is, a PD controller with control gains $K_\mathrm{P} = \mathrm{diag}(3, 3, 3, 3, 3, 3, 3)$ and $K_\mathrm{D} = \mathrm{diag}(2, 2, 2, 2, 2, 2, 2)$. The sampling time is instead selected as 1 ms.

5.2.1 | Faults on multiple actuators (scenario FE3)

In this scenario, the faulty actuators are those of joints 1, 3, 4, and 6, so that

$$\Delta y = \begin{bmatrix} 4\sin(6\pi t) & 0 & 7\sin(10\pi t) & 3\sin(2\pi t) & 0 & 8\sin(4\pi t) & 0 \end{bmatrix}^\top,$$

which can be interpreted, for instance, as a classical bearing failure (see e.g., Reference 46 for a brief overview on fault characteristics analysis). Faults occur at different time instants, in particular the fault on joint 1 occurs at $t = 1.5$ s, the one on joint 3 at $t = 2$ s, the one on joint 4 at $t = 1$ s, and the one in joint 6 at $t = 3.5$ s.

Figure 6 shows that the battery of SOSM UIOs is capable to identify in finite-time the actuator faults acting on the corrupted joints. The thresholds in (34) are selected as $\underline{Y}_{\mathrm{a},i} = -0.5$ and $\overline{Y}_{\mathrm{a},i} = 0.5$ in order to take into account possible measurement noises or disturbances. Then, the root mean square estimation error (RMS) has been computed for all the considered actuators, and the results are presented in Table 2.

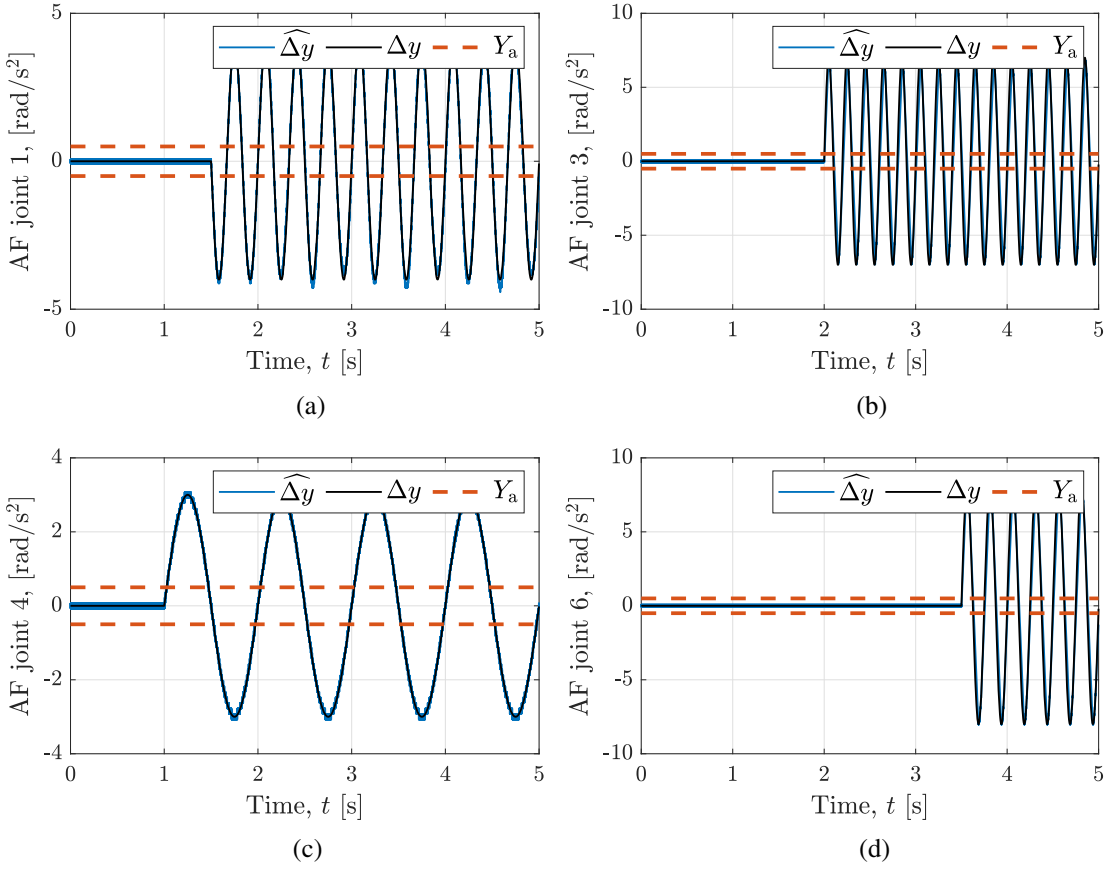


FIGURE 6 Time evolution of the actual and estimated actuator faults when joints 1, 3, 4, and 6 are subject to failures in the case of scenario FE3. (a) AF joint 1, (b) AF joint 3, (c) AF joint 4, (d) AF joint 6.

5.2.2 | Faults on multiple sensors (scenario FE4)

In this scenario, we assess the effectiveness of the proposed DRL algorithm. A constant fault equal to −0.0309 rad is first injected into the sensor of joint 2, while a constant fault equal to 0.1149 rad is added to the measurement of joint 6. The thresholds in (21) are selected as $\underline{Q}_{a,i} = -0.0262$ and $\overline{Q}_{a,i} = 0.0262$, to take into account possible measurement noises or disturbances.

The results are satisfactory and illustrated in Figure 7, while in Table 3 the RMS estimation errors are reported to confirm the effectiveness of the proposed approach.

Then, also joints 1, 4 and 5 are considered affected by sensor faults equal to 0.134, −0.202, and −0.08 rad, respectively. The results are illustrated in Figure 8, and the corresponding RMS estimation errors are indicated in Table 3.

It is worth to highlight that, although the estimated value slightly deviates from the actual one, the proposed approach has the merit to perform a completely model-free fault diagnosis, given only the measurements of the end-effector position and data from sensors, and without compromising the closed-loop stability of the controlled system.

5.2.3 | Faults on multiple sensors and actuators (scenario FE5)

In this scenario, both actuator and sensor faults can contemporarily occur on the same joint. Specifically, joint 1 is affected by a constant sensor fault equal to −0.3 rad and by a sinusoidal actuator fault equal to $3\sin(8\pi t)$ rad s^{-2}, which occur at $t = 1.5$ s and $t = 1$ s, respectively. Moreover, on joint 7, only a sinusoidal actuator fault equal to $5\sin(12\pi t)$ rad s^{-2} occurs at time $t = 3$ s.

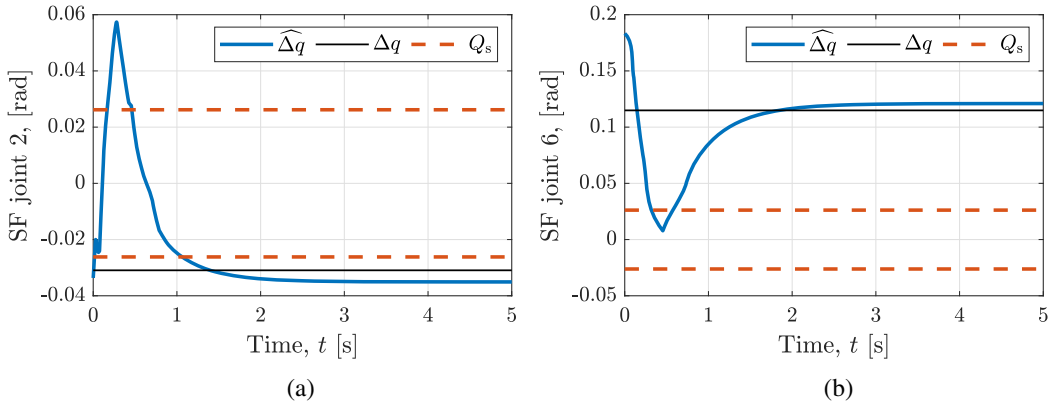


FIGURE 7 Time evolution of the actual and estimated sensor faults when joints 2 and 6 are subject to failures in the case of scenario FE4. (a) SF joint 2, (b) SF joint 6.

TABLE 3 RMS estimation errors for scenario FE4.

Joint #	RMS ($\widehat{\Delta q} - \Delta q$) [rad]
2	0.021
6	0.032
1	0.0231
4	0.0094
5	0.0395

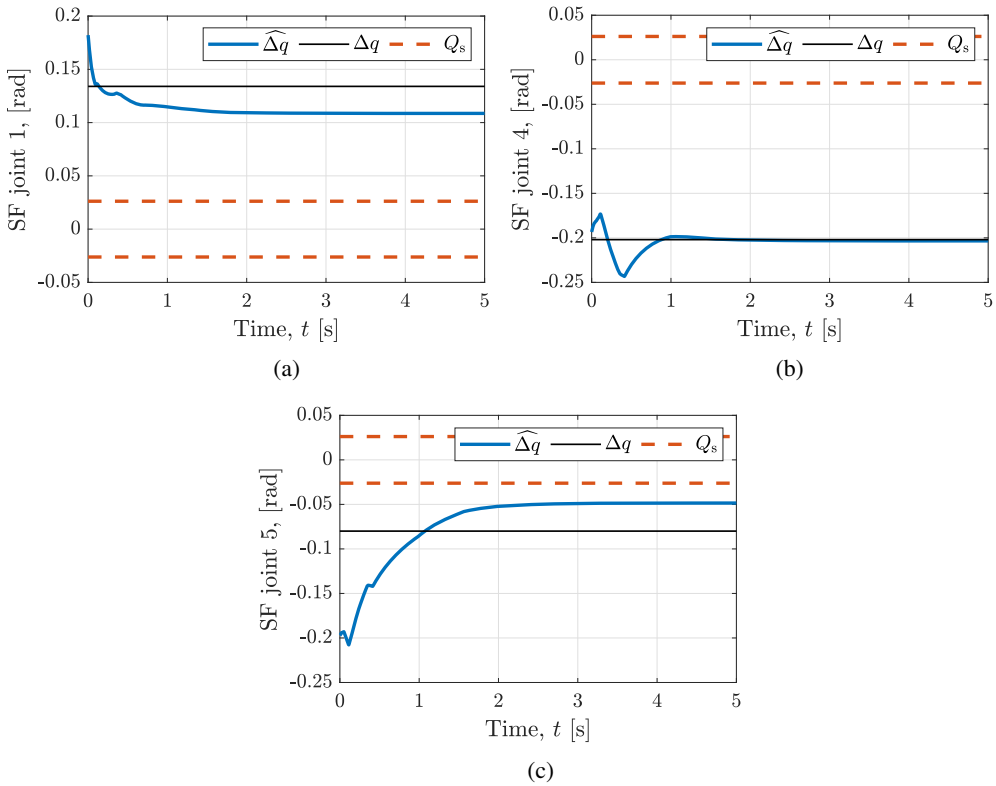


FIGURE 8 Time evolution of the actual and estimated sensor faults when joints 1, 4, and 5 are subject to failures in the case of scenario FE4. (a) SF joint 1, (b) SF joint 4, (c) SF joint 5.

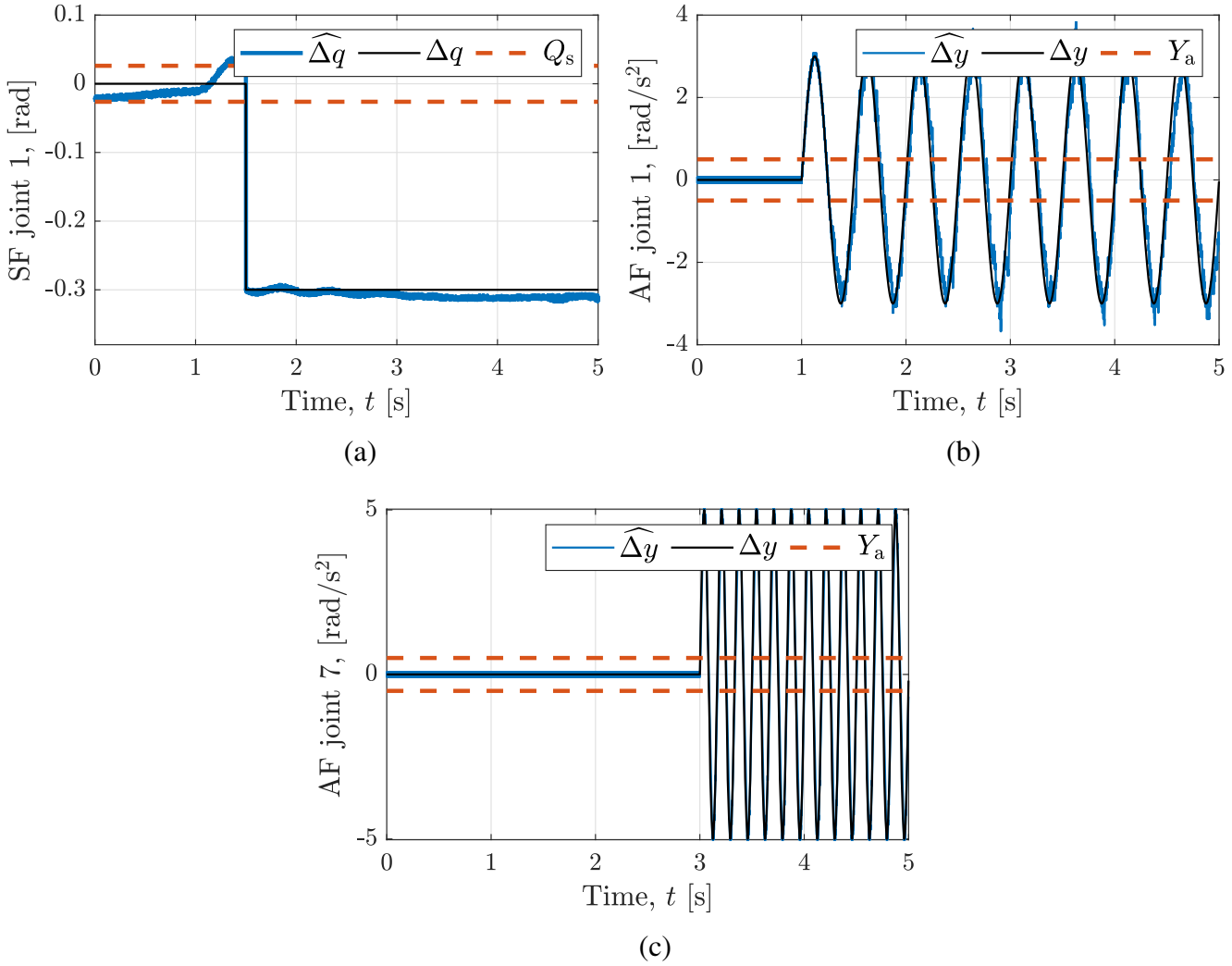


FIGURE 9 Time evolution of the actual and estimated faults on both sensors and actuators when joints 1 and 7 are subject to failures in the case of scenario FE5. (a) SF joint 1, (b) AF joint 1, (c) AF joint 7.

TABLE 4 RMS estimation errors for scenario FE5.

Joint #	RMS ($\widehat{\Delta q} - \Delta q$) [rad]	RMS ($\widehat{\Delta y} - \Delta y$) [rad s^{-2}]
1	0.013	0.0867
7	-	0.3682

The time evolution of the estimated faults are depicted in Figure 9. It is possible to observe that also in this more complex scenario the proposed control scheme is capable of distinguishing among the components and satisfactorily identify all the faults. These results are also confirmed by the RMS estimation errors reported in Table 4.

Finally, maintaining the same setting of parameters, we consider a scenario where no fault occurs on joints 1 and 7. The results, presented in Figure 10, highlight that, in absence of faults, no false alarm is generated by the proposed fault detection approach. Note that, all the results presented in this subsection have been obtained in presence of a random measurement noise, sampled uniformly between −0.1 deg and 0.1 deg, this in order to assess the performance of our proposal in a more realistic setting. As evident from Figures 9 and 10, even in the presence of measurement noise on the joint sensors, the fault detection approach results in being satisfactorily robust in detecting both sensor and actuator faults.

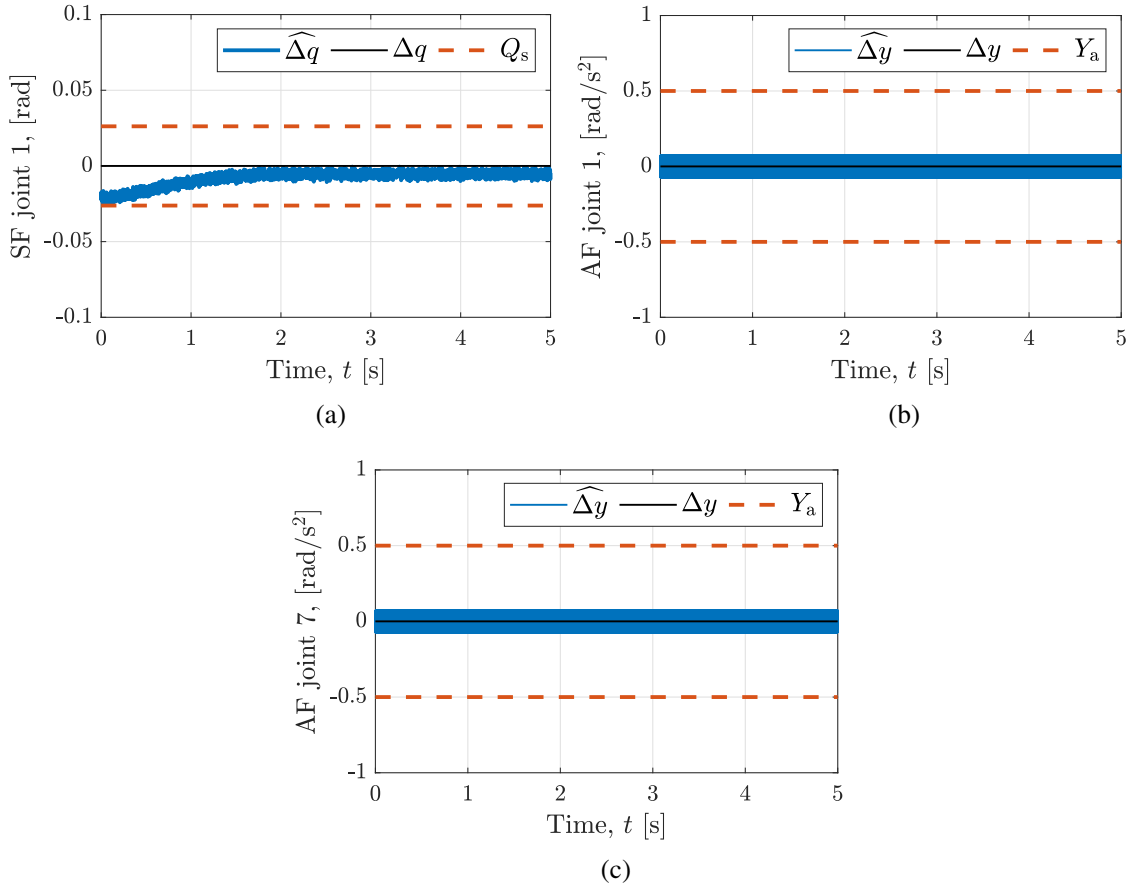


FIGURE 10 Time evolution of the sensors and actuators signals of joints 1 and 7 in the fault-free scenario. (a) SF joint 1, (b) AF joint 1, (c) AF joint 7.

6 | CONCLUSIONS

A fault diagnosis control scheme for intrinsically redundant robot manipulators has been presented in this paper. The proposal relies on the joint application of a global feedback linearization, and a fault diagnosis mechanism for multiple sensor and actuator faults. The use of a model-free DRL mechanism, which is trained off-line, allows one to attain a complete sensor fault diagnosis. Moreover, it provides estimates of the faults which can be used to compensate the actual sensor failures so as to close the control feedback, making the whole control scheme sensor-fault tolerant. This in turn enables the decoupling between sensor and actuator faults, thus making the diagnosis of possible actuator faults feasible. This is performed by means of n_q UIOs based on a SOSM with optimal reaching. These observers perform the actuator fault detection and isolation in a robust way. Realistic simulation tests, considering multiple faults occurring on different robot sensors and actuators have been carried out with satisfactory results, relying on a 7 DOFs Franka Emika Panda robot manipulator reproduced in the PyBullet environment.

DATA AVAILABILITY STATEMENT

Research data are not shared.

ACKNOWLEDGMENT

Open Access Funding provided by Politecnico di Milano within the CRUI-CARE Agreement.

CONFLICT OF INTEREST

All authors declare no conflict of interest.

ORCID

Gian Paolo Incremona https://orcid.org/0000-0003-1974-5646
Antonella Ferrara https://orcid.org/0000-0002-1977-8248

REFERENCES

1. Isermann R. *Fault-Diagnosis Systems. 1*. Springer; 2006.
2. Punčochář I, Škach J. A survey of active fault diagnosis methods. *IFAC-PapersOnLine*. 2018;51(24):1091-1098.
3. Chiang LH, Russell EL, Braatz RD. *Fault Detection and Diagnosis in Industrial Systems*. Springer; 2000.
4. Isermann R. *Fault-Diagnosis Applications: Model-Based Condition Monitoring: Actuators, Drives, Machinery, Plants, Sensors, and Fault-Tolerant Systems*. Springer; 2011.
5. Andjelkovic I, Sweetingham K, Campbell SL. Active fault detection in nonlinear systems using auxiliary signals. *American Control Conference*. IEEE; 2008:2142-2147.
6. Šimandl M, Punčochář I. Active fault detection and control: Unified formulation and optimal design. *Automatica*. 2009;45(9):2052-2059.
7. Scott JK, Findeisen R, Braatz RD, Raimondo DM. Input design for guaranteed fault diagnosis using zonotopes. *Automatica*. 2014;50(6):1580-1589.
8. Henry D, Cieslak J, Zolghadri A, Efimov D. A non-conservative $\mathscr{H}_/\mathscr{H}_\infty$ solution for early and robust fault diagnosis in aircraft control surface servo-loops. *Control Eng Pract*. 2014;31:183-199.
9. Wolfram A, Fussel D, Brune T, Isermann R. Component-based multi-model approach for fault detection and diagnosis of a centrifugal pump. *American Control Conference*. IEEE; 2001:4443-4448.
10. Zhan Y, Jiang J. An interacting multiple-model based fault detection, diagnosis and fault-tolerant control approach. *38th IEEE Conference on Decision and Control*. IEEE; 1999:3593-3598.
11. Jarrou A, Sauter D, Alami K. Fault diagnosis and fault tolerant control based on model predictive control for nearly zero energy buildings. *4th Conference on Control and Fault Tolerant Systems*. IEEE; 2019:219-225.
12. Baimukashev D, Rakhim B, Rubagotti M, Varol HA. End-to-End Deep Fault Tolerant Control. *IEEE/ASME Trans Mechatron*. 2021;27(4):2224-2234.
13. Alwi H, Edwards C, Tan CP. *Fault Detection and Fault-Tolerant Control Using Sliding Modes*. Springer; 2011.
14. Ferrara A, Incremona GP, Cucuzzella M. *Advanced and Optimization Based Sliding Mode Control: Theory and Applications*. Society for Industrial and Applied Mathematics; 2019.
15. Utkin VI. *Sliding Modes in Optimization and Control Problems*. Springer; 1992.
16. Davila J, Fridman L, Poznyak A. Observation and identification of mechanical systems via second order sliding modes. *Int J Control*. 2006;79(10):1251-1262.
17. Spurgeon SK. Sliding mode observers: A survey. *Int J Syst Sci*. 2008;39(8):751-764.
18. Halder B, Sarkar N. Robust fault detection of a robotic manipulator. *Int J Robotics Res*. 2007;26(3):273-285.
19. De Luca A, Mattone R. An adapt-and-detect actuator FDI scheme for robot manipulators. *IEEE International Conference on Robotics and Automation*. IEEE; 2004:4975-4980.
20. Capisani LM, Ferrara A, De Loza FA, Fridman L. Manipulator fault diagnosis via higher order sliding-mode observers. *IEEE Trans Ind Electron*. 2012;59(10):3979-3986.
21. Halder B. *Robust Nonlinear Fault Detection and Isolation of Robotic System: A Novel Nonlinear Analytic Redundancy Method*. Springer; 2009.
22. Rigatos GG. *Fault Diagnosis in Robotic and Industrial Systems*. iConcept Press Ltd.; 2012.
23. De Luca A, Mattone R. An identification scheme for robot actuator faults. *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE; 2005:1127-1131.
24. Papadimitropoulos A, Rovithakis GA, Parisini T. Fault Detection in Mechanical Systems With Friction Phenomena: An Online Neural Approximation Approach. *IEEE Trans Neural Networks Learn Syst*. 2007;18(4):1067-1082.
25. Bartolini G, Caputo W, Cecchi M, Ferrara A, Fridman L. Vibration damping in elastic robotic structures via sliding modes. *J Robot Syst*. 1997;14(9):675-696.
26. Ferrara A, Rubagotti M. A dynamic obstacle avoidance strategy for a mobile robot based on sliding mode control. *IEEE Control Application Intelligent Control Conference*. IEEE; 2009:1535-1540.
27. Capisani LM, Ferrara A, Magnani L. Design and experimental validation of a second-order sliding-mode motion controller for robot manipulators. *Int J Control*. 2009;82(2):365-377.
28. Incremona GP, Saccon A, Ferrara A, Nijmeijer H. Trajectory tracking of mechanical systems with unilateral constraints: Experimental results of a recently introduced hybrid PD feedback controller. *54th IEEE Conference on Decision and Control*. IEEE; 2015:920-925.
29. Ferrara A, Incremona GP. Design of an integral suboptimal second-order sliding mode controller for the robust motion control of robot manipulators. *IEEE Trans Control Syst Technol*. 2015;23(6):2316-2325.
30. Incremona GP, Ferrara A, Magni L. MPC for robot manipulators with integral sliding modes generation. *IEEE/ASME Trans Mechatron*. 2017;22(3):1299-1307.
31. Capisani LM, Ferrara A, Pisu P. Sliding mode observers for vision-based fault detection, isolation and identification in robot manipulators. *American Control Conference*. IEEE; 2010:4540-4545.
32. Incremona GP, Ferrara A. Fault diagnosis for robot manipulators via vision servoing based suboptimal second order sliding mode. *European Control Conference*. IEEE; 2019:3090-3095.

33. Siciliano B, Sciavicco L, Villani L, Oriolo G. *Robotics: Modelling, Planning and Control*. Springer; 2009.
34. Bartolini G, Ferrara A, Usai E. Chattering avoidance by second-order sliding mode control. *IEEE Trans Autom Control*. 1998;43(2):241-246.
35. Wu N, Zhou K, Salomon G. Control reconfigurability of linear time-invariant systems. *Automatica*. 2000;36(11):1767-1771.
36. Yang H, Jiang B, Staroswiecki M, Zhang Y. Fault recoverability and fault tolerant control for a class of interconnected nonlinear systems. *Automatica*. 2015;54:49-55.
37. Dinuzzo F, Ferrara A. Higher order sliding mode controllers with optimal reaching. *IEEE Trans Autom Control*. 2009;54(9):2126-2136.
38. DeCarlo R, Zak S, Matthews G. Variable structure control of nonlinear multivariable systems: a tutorial. *Proc IEEE*. 1988;76(3):212-232.
39. Bartolini G, Ferrara A, Usai E, Utkin V. On multi-input chattering-free second-order sliding mode control. *IEEE Trans Autom Control*. 2000;45(9):1711-1717.
40. Watkins CJ, Dayan P. Q-learning. *Mach Learn*. 1992;8(3-4):279-292.
41. Mnih V, Kavukcuoglu K, Silver D, et al. Playing Atari with deep reinforcement learning. arXiv preprint, arXiv:1312.5602, 2013.
42. Lillicrap TP, Hunt JJ, Pritzel A, et al. Continuous control with deep reinforcement learning. arXiv preprint, arXiv:1509.02971, 2015.
43. Dankwa S, Zheng W. Twin-delayed DDPG: A deep reinforcement learning technique to model a continuous movement of an intelligent robot agent. *3rd International Conference on Vision, Image and Signal Processing*. ACM; 2019:1-5.
44. Dinuzzo F, Ferrara A. Finite-time output stabilization with second order sliding modes. *Automatica*. 2009;45(9):2169-2171.
45. Incremona GP, Rubagotti M, Ferrara A. Sliding mode control of constrained nonlinear systems. *IEEE Trans Autom Control*. 2017;62(6):2965-2972.
46. Jiao J, Zheng X. Fault Characteristics Analysis of Industrial Robot Based on Fault Tree. *2nd International Conference on Mechatronics Engineering and Information Technology*. Atlantis Press; 2017:179-182.

How to cite this article: Sacchi N, Incremona GP, Ferrara A. Sliding mode based fault diagnosis with deep reinforcement learning add-ons for intrinsically redundant manipulators. *Int J Robust Nonlinear Control*. 2023;1-19. doi: 10.1002/rnc.6619