IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 29, 2021, accepted February 6, 2021, date of publication February 16, 2021, date of current version March 1, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3059560

Dynamic Data Integration for Resilience to Sensor Attacks in Multi-Agent Systems

LUIS BURBANO[1,2], (Member, IEEE), LUIS FRANCISCO CÓMBITA[1,3], (Member, IEEE), NICANOR QUIJANO[1], (Senior Member, IEEE), AND SANDRA RUEDA[4], (Member, IEEE)

[1]Departamento de Ingeniería Eléctrica y Electrónica, Universidad de Los Andes, Bogotá 111711, Colombia
[2]Department of Computer Science and Engineering, University of California at Santa Cruz, Santa Cruz, CA 95064, USA
[3]Facultad de Ingeniería, Universidad Distrital Francisco José de Caldas, Bogotá 110231, Colombia
[4]Departamento de Ingeniería de Sistemas y Computación, Universidad de Los Andes, Bogotá 111711, Colombia

Corresponding author: Luis Francisco Cómbita (ca.luis10@uniandes.edu.co)

This work was supported in part by the Air Force Office of Scientific Research under Award FA9550-19-1-0014, in part by the Comisión de Estudios No. 015 de 2014 by Universidad Distrital Francisco José de Caldas, and in part by the Convocatoria 727 Doctorados Nacionales 2015 by Colciencias.

ABSTRACT In recent years the number of security incidents affecting control systems has increased. These incidents have shown the need to develop strategies to improve system resilience to cyber-attacks. This paper presents a practical implementation of a strategy to detect cyber-attacks and mitigate their effects on sensors of a multi-agent system. The proposed approach computes, in real-time, a convex combination of measurements from main and redundant sensors, producing a trust value of the measurement and feeding it to the controller. We implemented this approach on a testbed of ground robots in formation. Experimental results to various kinds of attacks and a key performance index show that the proposed strategy reduces the effects of attacks not only on the affected agent, but also prevents the propagation of the attack over the remaining agents.

INDEX TERMS Cyber-physical systems, multi-agent systems, resilience to sensor attacks, sensor attacks.

NOMENCLATURE

$\alpha^{(i,j)}$	Confidence value for sensor i of measured variable j.
$\mathbb{R}$	Set of real numbers.
$\mathbb{R}_+$	Set of the strictly positive quadrant of $\mathbb{R}$.
$\bar{\mathbf{z}}_j^{(i)}$	Measurement reported by the i^{th} sensor (possibly compromised) for the j^{th} agent.
$\check{\mathbf{z}}$	Output vector for a differential drive robot used to implement the feedback linearization control.
$\hat{\mathbf{x}}$	Estimated state.
$\hat{\mathbf{x}}^-$	Prediction of the state.
$\mathbf{a}_j^{(i)}$	Attack on sensor i of the agent j.
$\mathbf{d}_{ij}$	Relative distance between agents i and j.
$\mathbf{e}$	Difference between the predicted $\hat{\mathbf{z}}$ and read output $\bar{\mathbf{z}}$.
$\mathbf{F}$	Jacobian of f_d evaluated at the estimated state.
$\mathbf{H}$	Jacobian of h_d evaluated at the estimated state.
$\mathbf{K}$	Kalman Gain.
$\mathbf{P}$	Sensor error covariance matrix.
$\mathbf{Q}$	Process noise covariance matrix.
$\mathbf{R}$	Measurement noise covariance matrix.
$\mathbf{r}$	Residue (difference between the measurement and its estimation).
$\mathbf{u}$	Control action of a differential drive robot.
$\mathbf{u}_{FL_1}$	Control law for the leader robot.
$\mathbf{u}_{FL_i}$	Control law for the i^{th} follower robot.
$\mathbf{u}_{FL}$	Tangential velocities in the direction of the first and second component of $\check{\mathbf{z}}$.
$\mathbf{v}$	Measurement noise.
$\mathbf{w}$	Multivariate process noise.
$\mathbf{x}$	Discrete-time nonlinear state.
$\mathbf{z}$	Discrete-time nonlinear output.
$\mathcal{E}$	Edges set of a communication graph.
$\mathcal{G}$	Model of a communication graph.
$\mathcal{H}$	Hybrid system.

The associate editor coordinating the review of this manuscript and approving it for publication was Aijun Yang.

$\mathcal{K}_j^{(i)}$	Attack duration on the i^{th} sensor for the j^{th} agent.
$\mathcal{N}_i$	Neighbors of the node i.
$\mathcal{S}_j$	List of n_s redundant sensors for the j agent.
$\mathcal{V}$	Vertices set of a communication graph.
ω_l	Angular velocity of the left wheel.
ω_r	Angular velocity of the right wheel.
θ	Orientation angle of the agent with respect to the x axis (unicycle).
$\tilde{z}^{(j)}$	Trust value of the j^{th} measured variable.
$C_{\mathcal{H}}$	Flow set of a hybrid system.
$D_{\mathcal{H}}$	Jump set of a hybrid system.
d_{FL}	Look-ahead distance used to compute the feedback linearization.
$f_{\mathcal{H}}$	Flow map of a hybrid system.
$g_{\mathcal{H}}$	Jump map of a hybrid system.
L	Distance between wheels.
m	Number of outputs for the nonlinear discrete-time system.
N	Cardinality of the set $\mathcal{V}$, and the number of agents in the formation.
n_u	Number of inputs for the nonlinear discrete-time system.
n_x	Number of state variables for the nonlinear discrete-time system.
r	Wheel radius.
$S^{(i,j)}$	CUSUM for the j^{th} measurement of the i^{th} sensor.
T	Total number of iterations in the simulation and experiments.
T_s	Sampling time.
v	Tangential velocity of a robot.
w	Angular velocity of a robot.
x	Horizontal position of the agent (unicycle) in a two-dimensional plane.
y	Vertical position of the agent (unicycle) in a two-dimensional plane.
$\nu^{(i,j)}$	CUSUM discount parameter for the j^{th} measurement of the i^{th} sensor.
$\tau^{(i,j)}$	CUSUM threshold for the j^{th} measurement of the i^{th} sensor.
$z_j^{(i)}$	Actual reading of the i^{th} sensor for the j^{th} agent.

I. INTRODUCTION

Recently, attackers have deployed different malicious actions with targets that range from industrial systems, such as the Stuxnet attack deployed in an Iranian uranium enrichment plant in 2010 [1], to robotic systems where attacks can affect civil global position systems (GPS) [2]. These malicious actions have shown the necessity of developing security strategies to increase system resilience against attacks. When an attack has been deployed, those strategies should not only ensure system and users safety, but also present a performance similar to a scenario without attacks.

The number of works that make automated decisions to mitigate attack impacts have increased in the last years [3]. Regarding attacks on sensors and actuators of ground robots, most of the works focus on detection. These works implement a monitor to verify that the system is properly working. For that purpose, some works use analytical redundancy to verify that the behavior of a system is similar to the behavior generated with a mathematical model. For instance, a linear monitor that accumulates the quadratic error in a time window is proposed in [4].

As vehicles present nonlinear dynamics, a monitor with a nonlinear model is used in [5]. A hardware-based redundancy strategy that uses multiple sensors to identify an attack is presented in [6]. This strategy is based on a modified principal component analysis (PCA) to find the envelopes where the system is working without attacks. Then, an attack can be identified if a sensor is outside such an envelope.

Although the mentioned works detect various attacks, they do not propose any automated mitigation strategy. Regarding strategies to mitigate attack effects on ground robotic systems, we found works that either focus on single robot systems or multi-robot systems. In the first group, a linear model is used to estimate the system output and detect an attack in [7]. Once an attack is revealed, the controller is fed with the estimated output. Additionally, a modification of the linear Kalman filter such that the influence of the attacked sensor is decreased is proposed in [8]. Regarding the works in multi-agent systems, authors in [9] propose a moving target defense (MTD) strategy to decrease the effects of attacks on the communication network. The work shows that randomly changing the communication graph can reduce the deviation produced by the attack. This strategy is finally extended to attacks on system's sensors and controllers. A formalization of the strategy to face attacks is developed in [10]. The strategy presented in [11] mitigates attacks by isolating communication links and nodes in unmanned aerial vehicles (UAV) formations. To detect such attacks, each agent implements an unknown input observer (UIO) bank. Once the attacked agent has been identified, it is removed from the UAV formation [12].

To sum up, attacks on multi-agent systems sensors are a real concern for the system safety. In such a scenario, agents make wrong decisions based on false data, decreasing the system performance and producing other adverse effects such as collisions in multi-vehicle applications. Those collisions might damage the vehicles, and, therefore, a strategy that maintains the integrity of the agents and the system performance is required. Several strategies presented above show a method to detect attacks acting on the system. However, this is insufficient since it does not necessarily prevent collisions or unsafe scenarios. Additionally, even if a mitigation strategy is presented, the techniques presented above do not maintain the performance (e.g., the formation shape is modified). Consequently, a strategy to mitigate sensor attacks in multi-agent systems, preserving formation geometry and the system performance, is needed.

The main contribution of this paper is to design such a strategy to increase system resiliency to attacks. Our approach dynamically detects and mitigates attacks in real time by

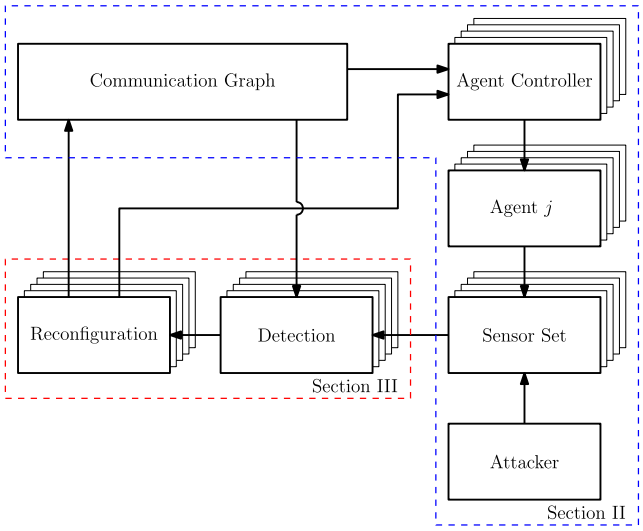

FIGURE 1. Complete strategy proposed to mitigate attack effects on agent sensors.

computing control actions that compensate deviations from expected behavior. It uses a convex combination among redundant sensors to compute trust values for all physical variables that are being measured in the system. These trust values are used to compute a control action to mitigate the impact on the attacked agent and on the entire system. The strategy is not only simulated, but it is also implemented in a system of multiple differential drive robots. Our results show that the proposed strategy reduces the effects on the attacked agent and, additionally, it prevents the attack propagation to the whole system through the communication network. Thus, the strategy preserves the geometry of the formation when the system is facing an attack on its sensors.

The remaining of the paper is organized as follows. Section II describes the differential drive robot model, the formation model, and the controller for every agent. It also shows attacks definitions and capabilities and limitations of attackers (see the blue dotted box in Figure 1). Section III presents detection and mitigation strategies (see the red dotted box in Figure 1). Section IV presents the simulations and experimental results. Those results are discussed in Section V. Finally, Section VI presents conclusions and future work.

II. SYSTEM DESCRIPTION

In robotic formations, one of the main objectives is to maintain relative distances between the involved robots. One approach to achieve a particular formation is to use a leader-follower structure. While the leader robot moves in the space tracking a reference without depending on the neighbors position, the follower robots must modify their positions to maintain formation. Therefore, to accomplish the objective, each agent must implement a controller to achieve a position and each robot must share its position with a set of local robots (neighbors in a graph). Such formation and controller can be affected by attacks on the agent This section presents the controller and the threat model.

A. GRAPH THEORY

We model communications between pairs of robots with a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. This graph has a finite set of vertices $\mathcal{V} = \{1, 2, \ldots, N\}$, and a set of edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. In a directed graph an edge (v_i, v_j), denoted as (i, j), means that node j can obtain information from node i (it represents communication in only one direction). The neighbors of node i will be denoted as $\mathcal{N}_i = \{j \in \mathcal{V} : (i, j) \in \mathcal{E}\}$.

To implement the leader-follower structure, the leader does not need information from other agents. Therefore, communications are directed; from the leader to the followers.

B. AGENT MODEL

Each agent in this work is a ground robot with differential locomotion. A differential drive robot has two parallel wheels attached to independent actuators. The left and right actuator move the wheels at different angular velocities ω_r, ω_l. The robot position in a two-dimensional space x, y, and the angle θ with respect to the x axis may be modeled with the kinematics of an unicycle, which describes the robot as a particle with tangential velocity v and angular velocity ω, i.e.,

$$\begin{aligned} x &= v\cos(\theta) \\ y &= v\sin(\theta) \\ \theta &= \omega. \end{aligned} \tag{1}$$

The model shown in (1) is useful to design the system controller. Although an agent's tangential and angular velocities are not the actual inputs of the system, these values are linearly related with wheel velocity:

$$v = \frac{r}{2}(\omega_r + \omega_l), \quad \omega = \frac{r}{L}(\omega_r - \omega_l), \tag{2}$$

where r is wheel radius, and L is distance between wheels.

C. CONTROLLER

As shown in (1), a differential ground robot presents nonlinear dynamics. Various works have proposed different controllers for these robotic systems (see [13] and the references therein). This work implements a feedback linearization controller. This technique linearizes the relationship between the system input and output. Thus, a linear controller can be used to achieve a formation shape. For the differential drive robot, let us define the output as

$$\check{\mathbf{z}} = \begin{bmatrix} x + d_{FL}\cos(\theta) \\ y + d_{FL}\sin(\theta) \end{bmatrix}, \tag{3}$$

with $d_{FL} > 0$. The feedback linearization controller is given by,

$$\mathbf{u} = \begin{bmatrix} v \\ \omega \end{bmatrix} = \frac{1}{d_{FL}} \begin{bmatrix} d_{FL}\cos\theta & d_{FL}\sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \mathbf{u}_{FL},$$

where $\mathbf{u}_{FL} \in \mathbb{R}^2$ is the vector of the tangential velocities in the direction of the first and second component of $\check{\mathbf{z}}$, respectively. The zero dynamics are stable (see [14] and [15] for details).

We assume that the robot 1, denoted with the lower index 1, is the leader, and also that the follower robots denoted with the

lower index i, share information via an unweighted directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set that represents the N robots, and $\mathcal{E}$ is the set that represents the communication links between robots. The leader robot uses a proportional controller to follow a desired position $\check{\mathbf{z}}_{1,d} \in \mathbb{R}^2$, hence

$$\mathbf{u}_{FL_1} = k_p(\check{\mathbf{z}}_1 - \check{\mathbf{z}}_{1,d}).$$

The i^{th} follower robot has the control law

$$\mathbf{u}_{FL_i} = \frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} \left[\check{\mathbf{z}}_j - \check{\mathbf{z}}_i - \mathbf{d}_{ij}\right], \tag{4}$$

where $\mathbf{d}_{ij} \in \mathbb{R}^2$ is the relative distance between agents i and j and depends on the robot formation geometry. Under conditions without attack, a system using this controller converges if there is a directed path (a set of edges that connect a sequence of different vertices) between the leader and all the agents. Note that we can measure all the system variables, and the output defined in (3) is only used to achieve system formation.

Note also that the formation is achieved with the output (3), instead of the position x and y. Then, the robot position will converge to a neighborhood of the output (3). Consequently, we have chosen the parameter $d_{FL} = 0.035\,m$. Additionally, the leader controller is determined to be $k_p = 1$. These parameters have been selected independently of the reconfiguration and mitigation strategy parameters, which will be presented in Section III. The objective is that the strategy increases the resiliency to attacks on sensors without changing the controller parameters and, then, sacrificing the controller performance.

D. THREAT MODEL

We assume that for every measured variable in the system, there are redundant information sources (i.e., different sensors). Let $\mathcal{S}_j = \left\{s_j^{(1)}, \ldots, s_j^{(n_s)}\right\}$ be the list of n_s redundant sensors for the j agent, whose measurements (possibly compromised) are $\left[\bar{\mathbf{z}}_j^{(1)}, \ldots, \bar{\mathbf{z}}_j^{(n_s)}\right], \bar{\mathbf{z}}_j^{(i)} \in \mathbb{R}^m$. The purpose of the attacker is to modify sensors data to prevent the controllers to compute the required control action. We assume that the attacker has the actual systems' output and can inject any arbitrary value $\mathbf{a}_j^{(i)} \in \mathbb{R}^m$ at any time to sensor i of agent j. However, we assume that there is at least one sensor on each agent that is not under attack, but the strategy does have information about which sensors are not attacked.

Let $\mathcal{K}_j^{(i)} = \{k_{j,f}^{(i)}, \ldots, k_{j,l}^{(i)}\}$ represent the attack duration on the i^{th} sensor of the j^{th} agent, between the first time step $k_{j,f}^{(i)}$ and the last time step $k_{j,l}^{(i)}$. The model of the attack is [16]:

$$\bar{\mathbf{z}}_j^{(i)}[k] = \begin{cases} \mathbf{z}_j^{(i)}[k] & \text{for } k \notin \mathcal{K}_j^{(i)} \\ \mathbf{a}_j^{(i)}[k] & \text{for } k \in \mathcal{K}_j^{(i)} \end{cases}, \tag{5}$$

where $\mathbf{z}_j^{(i)}[k]$ is the actual reading of the i^{th} sensor of j^{th} agent, $\bar{\mathbf{z}}_j^{(i)}[k]$ is the measurement reported by the i^{th} sensor of j^{th} agent, and $\mathbf{a}_j^{(i)}$ is the attack signal affecting the i^{th} sensor of j^{th} agent. The attacker can also attack the pre-processing stage implemented for the sensors (e.g., the position estimation using robot encoders).

The threat model has some additional assumptions. First, as we are focusing on attacks to the sensors, we assume that an attacker only deploys attacks to those devices measurements. By a similar reason, we assume that the integrity of the information sent through the communication links is preserved and each agent trusts the information received from its neighbors. As a consequence, if there is an attack to an agent's sensor, this agent will send false data to its neighbors, potentially affecting the whole system. Finally, we also assume an attacker knows the system model, controller, agent neighbors state, detection strategy parameters, and that attacker can deploy a stealthy attack with that information. We argue that if the strategy can mitigate such a strong attacker, it can also mitigate less intelligent attacks.

III. ATTACK DETECTION AND MITIGATION

This work proposes a model-based strategy to detect and mitigate attacks on sensors of cyber-physical systems with multiple agents. The strategy uses redundant sensors to measure the system output and every reading is monitored using analytical redundancy. Such redundancy compares the sensor measurements with the expected ones given by the system model. Using this information, the redundant sensors information is fused to estimate the system output without attack. The overall strategy and the integration of both stages is inside the red box in Figure 1, and a detailed procedure of the strategy is presented in Algorithm 1. In the algorithm, $\mathbf{0}_m$ is a vector of zeros with m rows. In this section, the agent subindex is avoided for notation simplicity. However, all the variables refer to only one agent.

The detection stage is performed with an extended Kalman filter (EKF). The filter estimates the system states integrating the model with the input and sensor measurements, while handling system uncertainties such as noise. As our strategy is mainly based on the EKF, the error convergence and stability of the extended Kalman filter is a crucial concern for the practical implementation. For these reason, the design of the filter must guarantee its error convergence and stability. Several works have been studying conditions to ensure those properties, see [17] and [18] for instance. Indeed, we follow [17], and we verify that the initialization of $\mathbf{P}[0]$ is positive definite and that the matrix that weights the noise in the state equation is full rank, which in this case is the identity and therefore satisfies the condition. Let a discrete-time nonlinear system be given by,

$$\begin{aligned} \mathbf{x}[k] &= f_d(\mathbf{x}[k-1], \mathbf{u}[k]) + \mathbf{w}[k] \\ \mathbf{z}[k] &= h_d(\mathbf{x}[k]) + \mathbf{v}[k], \end{aligned}$$

where $f_d : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$ and $h_d : \mathbb{R}^{n_x} \to \mathbb{R}^m$, $\mathbf{w}[k] \in \mathbb{R}^{n_x}$ is a multivariate zero-mean Gaussian process noise with covariance matrix $\mathbf{Q} \in \mathbb{R}^{n_x \times n_x}$, $\mathbf{Q} \succeq 0$, and measurement noise $\mathbf{v}[k] \in \mathbb{R}^m$ with covariance matrix $\mathbf{R} \in \mathbb{R}^{m \times m}$, $\mathbf{R} \succeq 0$.

Algorithm 1 Attack detection and mitigation strategy

Data: Sensor list, $\mathcal{S} = \{s^{(1)}, \ldots, s^{(n_s)}\}$. Sensor measurement, $\bar{\mathbf{z}}^{(i)}[k]$. Sensor error covariance matrix, $\mathbf{P}^{(i)}[k-1]$. Process noise covariance Matrix, $\mathbf{Q}^{(i)}$. Measurement noise covariance matrix, $\mathbf{R}^{(i)}$. Sensor CUSUM, $S^{(i)}[k-1]$. CUSUM thresholds, $\boldsymbol{\tau}^{(i)}$. CUSUM discount parameter, $\boldsymbol{\nu}^{(i)}$. Sensor state estimation, $\hat{\mathbf{x}}^{(i)}[k-1]$. The input calculated for every sensor, $\mathbf{u}^{(i)}[k]$.
Result: Trust value of output $\mathbf{z}[k]$, $\tilde{\mathbf{z}}[k]$
$\Sigma_\alpha \leftarrow \mathbf{0}_m$;
foreach *sensor i in* $\mathcal{S}$ **do**
  $\hat{\mathbf{x}}^{(i)}[k] \leftarrow$
  $EKF(\hat{\mathbf{x}}^{(i)}[k-1], \mathbf{u}^{(i)}[k], \mathbf{P}^{(i)}[k-1], \mathbf{Q}^{(i)}, \mathbf{R}^{(i)})$;
  $\mathbf{r}^{(i)}[k] \leftarrow |\bar{\mathbf{z}}^{(i)}[k] - h(\hat{\mathbf{x}}^{(i)}[k])|$;
  foreach *measured variable j* **do**
    if $S^{(i,j)}[k-1] > \tau^{(i,j)}$ **then**
      $S^{(i,j)}[k] \leftarrow 0$;
      Raise an alarm on sensor i;
    else
      $S^{(i,j)}[k] \leftarrow$
      $\max(0; S^{(i,j)}[k-1] + r^{(i,j)}[k] - \nu^{(i,j)})$;
    if *an alarm has not been raised in sensor i* **then**
      $\alpha^{(i,j)}[k] \leftarrow 1 - S^{(i,j)}[k]/\tau^{(i,j)}$;
    else
      $\alpha^{(i,j)}[k] \leftarrow 0$;
    $\Sigma_\alpha^j \leftarrow \Sigma_\alpha^j + \alpha^{(i,j)}$;
$\tilde{\mathbf{z}}[k] \leftarrow \mathbf{0}_m$;
foreach *sensor i in* $\mathcal{S}$ **do**
  foreach *measured variable j* **do**
    if $\Sigma_\alpha^j > 0$ **then**
      $b^{(i,j)} \leftarrow \alpha^{(i,j)}/(\Sigma_\alpha^j)$;
    else
      $b^{(i,j)} \leftarrow 1/n_s$;
    $\tilde{z}^{(j)}[k] \leftarrow \tilde{z}^{(j)}[k] + b^{(i,j)}\bar{z}^{(i,j)}[k]$;

The EKF is iteratively defined as follows. First, the predicted state, $\hat{\mathbf{x}}^-[k] \in \mathbb{R}^{n_x}$, is calculated evaluating the model on the previous prediction, $\hat{\mathbf{x}}[k-1]$, and the current input, $\mathbf{u}[k]$, i.e.,

$$\hat{\mathbf{x}}^-[k] = f_d(\hat{\mathbf{x}}[k-1], \mathbf{u}[k]).$$

Then, the system dynamics are approximated using the Jacobian of f_d evaluated at the previous prediction, $\hat{\mathbf{x}}[k-1]$, and the current input, $\mathbf{u}[k]$, and the Jacobian of h_d evaluated at the predicted state $\hat{\mathbf{x}}^-[k]$, i.e.,

$$\mathbf{F}[k] = \left.\frac{\partial f_d}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}[k-1],\mathbf{u}[k]}, \quad \mathbf{H}[k] = \left.\frac{\partial h_d}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}^-[k]}.$$

Using these matrices, the error covariance matrix $\mathbf{P} \in \mathbb{R}^{n_x \times n_x}$ is predicted by,

$$\mathbf{P}^-[k] = \mathbf{F}[k]\mathbf{P}[k-1]\mathbf{F}^\top[k] + \mathbf{Q},$$

where $\hat{\mathbf{x}}[0]$ and $\mathbf{P}[0]$ are the filter initial conditions.

Afterwards, the correction is performed by adding the prediction with the error (i.e., difference between the predicted $\hat{\mathbf{z}}$ and read output $\bar{\mathbf{z}}$) multiplied by the Kalman gain $\mathbf{K}$. This gain depends on the predicted error covariance $\mathbf{P}^-$, measure noise covariance matrix $\mathbf{R}$, and the Jacobian of the observation function, i.e.,

$$\begin{aligned}\mathbf{e}[k] &= \bar{\mathbf{z}}[k] - h_d(\hat{\mathbf{x}}^-[k])\\ \mathbf{K}[k] &= \mathbf{P}^-[k]\mathbf{H}^\top[k](\mathbf{H}[k]\mathbf{P}^-[k]\mathbf{H}^\top[k] + \mathbf{R})^{-1}\\ \hat{\mathbf{x}}[k] &= \hat{\mathbf{x}}^-[k] + \mathbf{K}[k]\mathbf{e}[k].\end{aligned}$$

Finally, the error covariance matrix is updated using

$$\mathbf{P}[k] = (\mathbf{I} - \mathbf{K}[k]\mathbf{H}[k])\,\mathbf{P}^-[k].$$

On a scenario without attacks, changes on the input and output modify the estimation residues $\mathbf{r}[k] = |\bar{\mathbf{z}}[k] - \hat{\mathbf{z}}[k]|$, where $|\cdot|$ is the element wise absolute value. However, when a sensor information is maliciously modified, the magnitude of the estimation residues increases. To determine such modification on the residues behavior produced by an attack, the non-parametric cumulative sum (CUSUM) is used. Test raises an alarm when the residues magnitude is big or it presents a persistent deviation. CUSUM has shown to detect more attacks and outperform other tests [19]. To mitigate attacks on a system's agent, multiple sensors information is fused. For every sensor, a different EKF and CUSUM are used to monitor the sensor measurement. The CUSUM for the j^{th} measurement of the i^{th} sensor is iteratively defined by,

$$\begin{aligned}S^{(i,j)}[k] &= \max\{0, S^{(i,j)}[k-1] + r^{(i,j)}[k] - \nu^{(i,j)}\},\\ S^{(i,j)}[0] &= 0,\end{aligned}$$

where $\nu^{(i,j)} \in \mathbb{R}_+$ is a CUSUM parameter that prevents the statistic to increase under normal operation, and $r^{(i,j)}$ is the estimation residue for the j^{th} measurement of i^{th} sensor.[1] Every sensor has a discount parameter $\boldsymbol{\nu}^{(i)} = [\nu^{(i,1)}, \ldots, \nu^{(i,m)}]$ and it is selected as the minimum value such that, in a scenario without attack,

$$\mathbb{E}[r^{(i,j)}[k] - \nu^{(i,j)}] < 0, \tag{6}$$

where $\mathbb{E}[\cdot]$ is the expected value. This condition ensures that CUSUM is bounded when there are no attacks in sensors, and therefore false alarms due CUSUM unboundedness are avoided. The CUSUM raises an alarm on sensor i when the statistic of the measured variable j exceeds a threshold $S^{(i,j)}[k] > \tau^{(i,j)}$, $\tau^{(i,j)} \in \mathbb{R}_+$. As the parameter $\boldsymbol{\nu}^{(i)}$, every sensor has a threshold $\boldsymbol{\tau}^{(i)} = [\tau^{(i,1)}, \ldots, \tau^{(i,m)}]^\top$. These parameters are determined based on a false alarm rate: the bigger the threshold is, the lower the false rate is but detection

[1] $\mathbb{R}_+$ represents the strictly positive quadrant of $\mathbb{R}$

time increases. In this work, we tune the CUSUM parameters as presented in [16].

It is important to clarify that the distribution of the residues is affected by some parameters of the controller (e.g., sampling time). Therefore, the CUSUM parameters should be selected once the control strategy has been completely designed. In other words, the controller parameters can be picked to accomplish some performance specifications, and then the CUSUM parameters are tuned. Consequently, the performance of the controller is not affected in order to increase the system security.

The previous paragraphs present a strategy to detect modifications of sensor data. However, we also need an automated response to maintain the system in a safe state and meet the control system objective. The remaining of this subsection shows the proposed mitigation strategy integrated with the detection process.

In order to mitigate the attack effects on the robotic system, a different EKF monitors every reading of each sensor to obtain the residues. Using that information, redundant sensors information is fused. A confidence value is computed in each iteration for every sensor i and every measured variable j, which is defined by,

$$\alpha^{(i,j)}[k] = 1 - \frac{S^{(i,j)}[k]}{\tau^{(i,j)}},$$

where $\alpha^{(i,j)} = 0$, if an alarm has been raised in the j^{th} measured variable of i^{th} sensor. Note that confidence value is near to zero when CUSUM is near to the threshold, i.e., when there is a difference between the expected and actual behavior. In contrast, the confidence is near to one when the CUSUM does not show a difference between the actual and estimated measurements.

Finally, the sensors information is fused based on a convex combination, i.e.,

$$\tilde{z}^{(j)}[k] = \frac{1}{\sum_{i\in\mathcal{S}} \alpha^{(i,j)}[k]} \sum_{i\in\mathcal{S}} \alpha^{(i,j)}[k]\bar{z}^{(i,j)}[k], \tag{7}$$

where $\mathcal{S}$ is the sensor list and $\bar{z}^{(i,j)}$ is the reading from the i^{th} with the j^{th} variable, and $\tilde{z}^{(j)}$ is the trust value of the j^{th} measured variable. The calculated trust value $\tilde{\mathbf{z}}$ is then used to compute the control action and it is also sent to the communication network to prevent attack propagation.

IV. RESULTS

This section presents the implementation of some attacks to show the strategy usefulness. To apply the strategy, the discount parameter is $\boldsymbol{\nu}^{(i)} \in \mathbb{R}^3_+$ and threshold $\boldsymbol{\tau}^{(i)} \in \mathbb{R}^3_+$. We will denote the measurement of x, y, θ from sensor i by $\bar{z}^{(i,1)}$, $\bar{z}^{(i,2)}$, $\bar{z}^{(i,3)}$, respectively. The discount parameter of the i^{th} sensor will also be denoted by $\nu^{(i,1)}$, $\nu^{(i,2)}$, $\nu^{(i,3)}$ for the measurement of x, y, θ, respectively. Additionally, as we can measure all the system variables, the output function $h_d(\mathbf{x})$ for the EKF is,

$$\mathbf{z}[k] = h_d(\mathbf{x}[k]) = \mathbf{C}\begin{bmatrix} x[k] & y[k] & \theta[k] \end{bmatrix}^\top,$$

where $\mathbf{C}$ is the 3×3 identity. We also use a discrete-time version of differential-drive robot model (1), i.e.,

$$\begin{aligned} x[k+1] &= x[k] + v[k]\cos\left(\theta[k] + \frac{\omega[k]}{2}T_s\right)T_s \\ y[k+1] &= y[k] + v[k]\sin\left(\theta[k] + \frac{\omega[k]}{2}T_s\right)T_s \\ \theta[k+1] &= \theta[k] + \omega[k]T_s, \end{aligned}$$

where T_s is the sample time.

Note that the integration of the model presented in (1) with the controller and the mitigation strategy can be modeled as a hybrid system. A hybrid system $\mathcal{H}$ can be described with four elements: the flow set $C_{\mathcal{H}}$, flow map $f_{\mathcal{H}}$, jump set $D_{\mathcal{H}}$ and jump map $g_{\mathcal{H}}$. The jump and flow set are the points where the system flows or jumps, respectively. The behavior of the system when it flows or jumps is described by the flow map and the jump map. Then, to model the mitigation strategy as a hybrid system, we add an additional state β which is a timer. This timer is initialized at 0 and constantly grows at a rate of one, i.e., $\dot{\beta} = 1$, when $\beta \in [0, T_s]$. When the timer is less than the sampling time $\beta \leq T_s$, the system is in the flow set and the continuous states change. That is, the states of the robot are updated according to equation (1). Once the timer increases to $\beta \geq T_s$ the system is in the jump set and the discrete states change. That is, the states of the robot remain constant while the controller, states estimations, CUSUM, confidence values and trust values are updated. Additionally, the timer is reset to zero $\beta = 0$ and the system arrives again to the flow set [20], [21].

A. EXPERIMENTAL SETUP

The strategy has been implemented in a system with 6 e-puck version 2 robots [22]. The robot has a maximum velocity of 15.4 cm/s, radius $r = 2.05$ cm, and distance between wheels $L = 5$ cm. Additionally, it has two stepper motors, a Wi-Fi and Bluetooth module, and multiple sensors as the inertial measurement unit.

To measure the robot states, odometry with the motor step counting has been implemented. Every wheel revolution represents 1000 steps and the robot sends its information every 0.05 s. Additionally, to perform the redundancy strategy, a camera is used to measure the robot angle and position. The camera can take up to 30 frames per second at 1080p. However, with such resolution the image recognition is expensive. Therefore, the image size has been changed to 600×450 pixels and 25 frames per second. This results in a resolution of 4 pixels per centimeter. Therefore, to ensure that the sensors are updated at each step, the discretization time has been set to 0.1 s.

A central processor runs image recognition, odometry, robot communication, and attack detection. This central processor is a laptop with 16 GB of RAM, with an Intel Core i7-8750H (which is a six-cores), 2.2 GHz processor, running a program developed in Python 3. Every process (e.g., image processing) runs in parallel in a different core of the

processor, and the communication with each robot is made in a different thread. Additionally, each robot has a program provided by the robots' developer that is used to implement the communication between the robot and the central processor.

B. CUSUM PARAMETERS

A proper parameter selection ensures a fast attack detection and low false alarm rate. In this work, we use the procedure presented in [16] to tune the CUSUM parameters. To guarantee the CUSUM boundness condition of (6), 10000 simulations for fifty seconds without attack have been performed to approximate the expected value. Every 1000 simulations, the maximum reference change rate is incremented 1 cm/s until a maximum velocity of 10 cm/s. For every simulation, the residues are calculated for the measured variable (i.e., x, y, and θ) and the mean is calculated to estimate the expected value of the residues. Therefore, the parameters found for both sensors for measurement in x is $\nu^{(i,1)} = 0.0023$, for measurement in y is $\nu^{(i,2)} = 0.0024$, and for measurement of θ is $\nu^{(i,3)} = 0.1103$ for sensors $i = \{1, 2\}$.

Finally, the CUSUM is calculated for every simulation with different threshold values to find the number of false alarms. High thresholds for x, y and θ have been selected to avoid a large false alarm rate, i.e., $\tau^{(i,1)} = 0.8$, $\tau^{(i,2)} = 0.8$, $\tau^{(i,3)} = 4$, for sensors $i = \{1, 2\}$.

C. NUMERICAL RESULTS

A six-robots system has been used to test the developed strategy. To implement the formation controller and show that the proposed strategy can work with different graphs, two communication topologies are used. The first communication graph, graph 1, used in this paper is defined by $\mathcal{G}_1 = \{\mathcal{V}_1, \mathcal{E}_1\}$, where $\mathcal{V}_1 = \{1, \ldots, 6\}$ and $\mathcal{E}_1 = \{(1, 2), (1, 6), (2, 3), (3, 2), (3, 4), (4, 3), (4, 5), (5, 4), (5, 6), (6, 5)\}$. The second graph, graph 2, is a directed graph that connects the leader and the follower robots with a directed path. This graph is defined by $\mathcal{G}_2 = \{\mathcal{V}_2, \mathcal{E}_2\}$, where $\mathcal{V}_2 = \{1, \ldots, 6\}$ and $\mathcal{E}_2 = \{(1, 2), (2, 3), (3, 4), (4, 5), (5, 6)\}$.

Figure 2 shows a photo of the system, graph 1 (represented with red arrows) and graph 2 (represented with blue arrows). Note that both graphs have a directed path between the leader and the followers to ensure that robots can achieve the formation when there are no attacks. Additionally, Figure 3 presents behavior for simulation and implementation of the system with a triangular formation and communications defined by graph 1. The simulations has been run in Matlab/Simulink 2018A.

The first attack is deployed on the leader robot odometry. Specifically, between 3 and 12 sec., the right motor step counting is stopped. The second attack is deployed on the camera sensor, adding a sigmoidal signal to the measurement of position x of the leader robot:

$$\bar{z}_1^{(2,1)}[k] = z_1^{(2,1)}[k] + \frac{0.2}{1 + e^{-3(0.1k-3)}}, \quad \mathcal{K}_1^{(2)} = \{0, 1, \ldots\}.$$

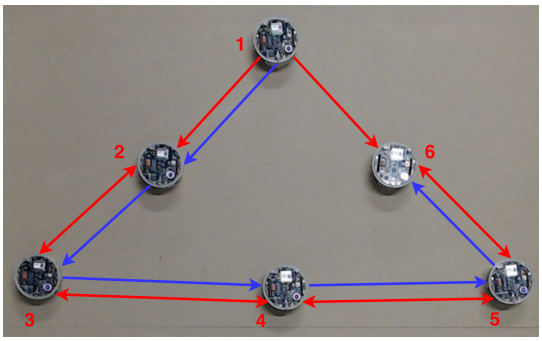

FIGURE 2. Aerial photography of e-puck version 2 robots formation. The red lines are a graphical representation of the communications graph. The red arrows indicate the flow direction of the information exchange in the graph $\mathcal{G}_1$, and the blue arrows represent the information exchange in the graph $\mathcal{G}_2$.

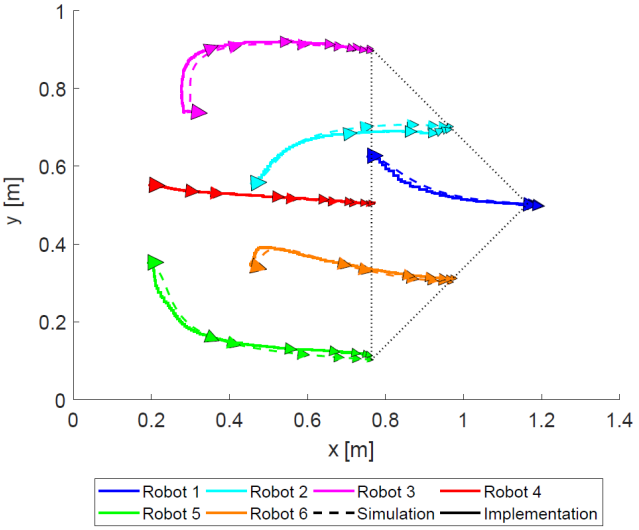

FIGURE 3. Trajectories without attack on the experimental setup (continuous lines) and simulation (dotted lines), using the graph 1.

It is important to clarify that we have also run two simulations deploying the every attack in robots 1 (leader), 2, 4, and 6 simultaneously. In contrast, only the second attack has been deployed in multiple robots in the physical system due to safety reasons. Additionally, a stealthy attack has been deployed on the camera sensor. For a detector that uses the CUSUM statistic, an optimal attack is given by [19],

$$\bar{z}_1^{(2,1)}[k] = \hat{z}_1^{(2,1)}[k] + (\tau^{(2,1)} + \nu^{(2,1)} - S_1^{(2,1)}[k]),$$
$$\mathcal{K}_1^{(2)} = \{20, 21, \ldots\}.$$

This attack implies that the attacker knows not only the current state, but also the detection strategy and model parameters, i.e., the functions f_d, h_d, robot parameters r, L, CUSUM parameters $\boldsymbol{\tau}$, $\boldsymbol{\nu}$ and EKF covariance matrices $\mathbf{Q}$, $\mathbf{R}$, and $\mathbf{P}$. Although this powerful attacker could be unrealistic, we deploy this attack to evaluate strategy results.[2]

[2]Some videos of the presented attacks can be found at: youtu.be/B0Vy33MbH4Q

TABLE 1. Normalized key performance index for the different experiments.

Graph 1				
	Implementation		Simulation	
Experiment	No reconfiguration	Reconfiguration	No reconfiguration	Reconfiguration
AO* on leader robot	1.46	1.02	1.42	1.06
AC* on leader robot	1.10	1.02	1.13	1.01
AO* on multiple robots	-	1.03	3.24	1.06
AC* on multiple robots	1.70	1.00	1.66	1.05
SA* on leader robot	1.39	1.08	1.30	1.00
Graph 2				
	Implementation		Simulation	
Experiment	No reconfiguration	Reconfiguration	No reconfiguration	Reconfiguration
AO* on leader robot	1.69	1.03	1.59	1.04
AC* on leader robot	1.12	1.00	1.21	1.07
AO* on multiple robots	-	1.00	4.30	1.07
AC* on multiple robots	1.76	1.00	1.80	1.01
SA* on leader robot	1.72	1.00	1.58	1.00

*AC: Attack on camera. AO: Attack on odometry. SA: stealthy attack.

To quantify the attack mitigation strategy efficiency, a key performance index (KPI) is defined. Let $\check{\mathbf{z}}_i[k]$ be the i^{th} robot output at instant k, and $\check{\mathbf{z}}_{i,d}[k]$ be the i^{th} robot desired output when the formation is achieved. We accumulate the Euclidean norm between the robot output at instant k, and the position that the robot should arrive once the formation is achieved in a scenario without attacks. Thus, the KPI is defined as:

$$KPI = \sum_{i=1}^{N} \sum_{k=0}^{T} \left\| \check{\mathbf{z}}_i[k] - \check{\mathbf{z}}_{i,d} \right\|_2^2 T_s, \tag{8}$$

where N is the number of robots, and T is the last instant of the experiment. To compare the different scenarios (i.e., without an attack, and attacked with and without reconfiguration), the KPI is normalized by the result obtained in the non-attack scenario. Table 1 shows KPIs of the aforementioned experiments and simulations.

Some KPIs constantly increase due to the attack characteristics. The KPI in the table was taken at 30 s (i.e., $K = 300$) when all scenarios have reached their steady-state. At this instant, the KPIs of scenarios with attack and reconfiguration have converged on all performed experiments.

Finally, we have run 1000 simulations, changing initial conditions, time of attacks, attacked agents and communication topology (i.e., selecting either $\mathcal{G}_1$ or $\mathcal{G}_2$). We have selected experiment conditions such that one sensor of every agent may have been attacked at different times during the simulation. We have also selected an intelligent attacker that deploys a stealthy attack. For these simulations we present the normalized KPI in function of time and not only the final value. Figure 4 shows the average of the normalized KPI together with the maximum and minimum KPI obtained in the whole simulation set, for the attack scenarios with and without the presented mitigation strategy. Note that, in order to obtain the normalized KPI, we require the simulation of the scenario without an attack.

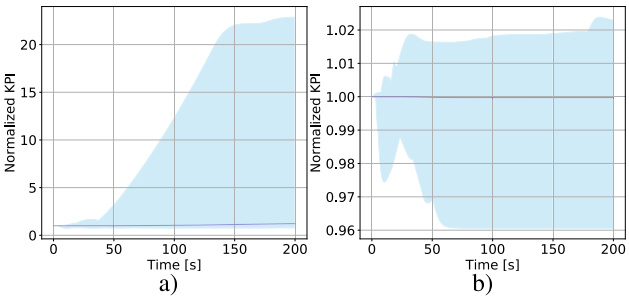

FIGURE 4. Average (blue line), maximum and minimum (blue area plot) normalized key performance index when a stealthy attack is deployed, randomly changing initial conditions, communication graph, attack instants and attacked agents. The KPI is presented when a) mitigation strategy is not implemented, and b) implementing the strategy.

V. DISCUSSION

The attack on the leader robot odometry has an important effect on the system behavior. At the second 30, the KPI of the experimental setup is greater than the scenario without attack when the graph 1 is used. The KPI shows that performance of the system is similar before and after deploying the mitigation strategy. When an attack starts, it disturbs the system and the KPI grows.

When this attack is deployed on multiple agents, the effects are amplified but the reconfiguration strategy can detect the attack on agents and reduce its effects. Similarly, in our simulations the strategy mitigates attack effects for both scenarios (i.e., attack on leader and simultaneous robots).

The bias attack on the camera does not allow the system to reach the desired formation. Although this attack presents smaller effects than the attack on odometry, the KPI shows

that the attack on the leader robot negatively affects system performance. When the attack is deployed on multiple robots, the system performance without the reconfiguration strategy is even worse. In contrast, the system with the reconfiguration strategy can achieve the expected formation, although the normalized KPI is greater than one. In other words, the system achieves the formation but the attack disturbs the system.

Note that the KPI of scenarios with attacks on multiple agents remain near one, i.e., the system has a similar performance to the scenario without attack. This shows that the reconfiguration strategy can handle attacks on various agents, without affecting the overall system performance. The strategy prevents every agent from misbehaving due to false information from one of its sensors or due to false information sent by a neighbor.

When the graph 2 is used, the attacks effects increase compared to the experiments using the graph 1. In the graph 1, each agent communicates with two agents. Then, when an agent is attacked, its neighbor makes decisions using the information of an attacked and a not attacked robot. In contrast, with the graph 2, all agents receive information from only one agent, which possibly is under attack. Then, with the graph 2, agents could make decisions using only false information even if just one robot is under attack. Nevertheless, with the reconfiguration strategy, the system can achieve the formation when attacks have been deployed on the system. Thus, we conclude that graph topology does not affect the reconfiguration strategy capacity to mitigate attack effects.

As expected, a stealthy attack cannot be detected by the implemented monitor and the CUSUM statistic remains below the threshold. The simulation of multiple scenarios shows that the stealthy attack can produce an important deviation compared to the non-attack scenario. However, when the reconfiguration strategy is implemented, the average, together with the maximum and minimum KPI is near one (i.e., near to the performance of the non-attack scenario). A similar result is obtained when the stealthy attack is deployed in the physical system. This shows that the strategy can mitigate a powerful attacker that modifies an attack to remain undetected, both in simulations and empirical experiments.

Remember that the parameter $\boldsymbol{\nu}$ is selected greater than the residues mean to prevent false alarms produced by the CUSUM unboundedness. However, if the difference between the residues mean and the parameter $\boldsymbol{\nu}$ is too large, the CUSUM would slowly increase and the time spent to detect an attack would also increase. Moreover, small changes in the residues distribution produced by some attacks would not change the CUSUM statistic, which could remain near zero. Therefore, changes in the parameter $\boldsymbol{\nu}$ change the classifier sensitivity. When the parameter $\boldsymbol{\nu}$ is incremented, the detector becomes less sensible to attacks. As a consequence, the system disturbance could become more notorious or the reconfiguration could not be properly made because the controller would be computed with the average between the not attacked and attacked sensors. Thus, the estimated output will be close to the actual output if the number of not attacked sensors is larger than the attacked ones. Similarly, if the parameter $\boldsymbol{\tau}$ is increased, some attacks could become undetectable or the time to detect could increase. However, when the threshold decreases, the number of false alarms could increase and, as a consequence, sensors could remain unused even if there are no attacks.

The simulation and implementation results show that the strategy can reduce attack effects on the system. Those attacks can have similar characteristics to a fault. Additionally, the strategy can identify the attacked sensor. However, it cannot differentiate between attacks on the sensors or actuators. Therefore, as the strategy only takes actions on the sensors, an attack (or fault) on agent actuators cannot be mitigated. Nevertheless, an alarm would be raised for all sensors and the attack would be detected. In that scenario, as there is not enough information to determine the attack, the trust value is calculated as the average among sensors. The same action would be taken if all sensors are attacked.

Although the simulation and implementation results are similar, there are some differences. Such differences may be explained by three main reasons. First, in the experimental setup, the attacks could be deployed in different time instants by some fraction of seconds. Second, in simulation, both sensors are assumed to be equal in the absence of attacks. However, in the implementation there is an error between measurements obtained from the camera and odometry. Third, the actuator dynamics have not been considered in the simulation. This results in a simpler strategy that requires less physical parameters (e.g., motor inductances and resistances), and less mathematical operations to monitor the sensors because there are fewer state variables to estimate. Nonetheless, the proposed strategy has shown to decrease the effects of attacks applied to real robots using a simplified model.

Finally, we have implemented the strategy in a six-robot system, but it is important to clarify that the strategy can be implemented in systems with more robots. Note that the strategy only uses local information to detect attacks. That is, each robot only uses the information of its sensors to determine the trust value, and the information from its neighbors to compute the control action. Therefore, the addition of more agents to the formation will not translate into a higher computational overhead to every agent.

VI. CONCLUSION

A model-based detection and reconfiguration strategy has been implemented to mitigate attack effects on a multiple robot system. The detection has been performed with an extended Kalman filter and the non-parametric cumulative sum. Measurements of the physical variables from redundant sensors in each robot are dynamically integrated using a convex combination to increase system resilience. The strategy has been simulated and implemented in a formation of six robots with one leader, using two different communication graphs. Attacks with different characteristics on the leader

robot and multiple robots have been deployed: one attack similar to a fault, a softer bias attack, and a stealthy attack. We have shown that the strategy can mitigate even intelligent attacks, with different initial conditions and communication topologies.

One of the redundant sensors, the camera, is not scalable and it is not available in many real scenarios. Therefore, the strategy could be tested with a different sensor as a global positioning system (GPS). On multi-agent systems, the communication links between the agents could be subject of attacks. In that scenario, the information shared by each agent can be modified and system safety would be compromised. In the future, we would like to address such problem. A strategy to mitigate those attacks may be integrated with the strategy presented in this work.

REFERENCES

[1] P. Shakarian, J. Shakarian, and A. Ruef, "Attacking Iranian nuclear facilities: Stuxnet," *Introduction to Cyber-Warfare: A Multidisciplinary Approach*. Boston, MA, USA: Syngress, 2013, pp. 223–239.

[2] S. Parkinson, P. Ward, K. Wilson, and J. Miller, "Cyber threats facing autonomous and connected vehicles: Future challenges," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 11, pp. 2898–2915, Nov. 2017.

[3] Y. Z. Lun, A. D'Innocenzo, F. Smarra, I. Malavolta, and M. D. D. Benedetto, "State of the art of cyber-physical systems security: An automatic control perspective," *J. Syst. Softw.*, vol. 149, pp. 174–216, Mar. 2019.

[4] H. Choi, W.-C. Lee, Y. Aafer, F. Fei, Z. Tu, X. Zhang, D. Xu, and X. Deng, "Detecting attacks against robotic vehicles: A control invariant approach," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2018, pp. 801–816.

[5] P. Guo, H. Kim, L. Guan, M. Zhu, and P. Liu, "VCIDS: Collaborative intrusion detection of sensor and actuator attacks on connected vehicles," in *Proc. Int. Conf. Secur. Privacy Commun. Syst.*, 2017, pp. 377–396.

[6] A. Tiwari, B. Dutertre, D. Jovanović, T. D. Candia, P. D. Lincoln, J. Rushby, D. Sadigh, and S. Seshia, "Safety envelope for security," in *Proc. 3rd Int. Conf. High Confidence Netw. Syst.*, Apr. 2014, pp. 85–94.

[7] G. Sabaliauskaite, G. S. Ng, J. Ruths, and A. Mathur, "Experimental evaluation of stealthy attack detection in a robot," in *Proc. IEEE 21st Pacific Rim Int. Symp. Dependable Comput. (PRDC)*, Nov. 2015, pp. 70–79.

[8] N. Bezzo, J. Weimer, M. Pajic, O. Sokolsky, G. J. Pappas, and I. Lee, "Attack resilient state estimation for autonomous robotic systems," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2014, pp. 3692–3698.

[9] J. Giraldo and A. A. Cardenas, "Moving target defense for attack mitigation in multi-vehicle systems," in *Proactive and Dynamic Network Defense*. Cham, Switzerland: Springer, 2019, pp. 163–190.

[10] J. Giraldo, A. Cardenas, and R. G. Sanfelice, "A moving target defense to detect stealthy attacks in cyber-physical systems," in *Proc. Amer. Control Conf. (ACC)*, Jul. 2019, pp. 391–396.

[11] L. Negash, S.-H. Kim, and H.-L. Choi, "Distributed observes for cyberattack detection and isolation in formation-flying unmanned aerial vehicles," *J. Aerosp. Inf. Syst.*, vol. 14, no. 10, pp. 551–565, Oct. 2017.

[12] S.-H. Kim, L. Negash, and H.-L. Choi, "Cubature Kalman filter based fault detection and isolation for formation control of multi-UAVs," *IFAC-PapersOnLine*, vol. 49, no. 15, pp. 63–68, 2016.

[13] W. E. Dixon, D. M. Dawson, E. Zergeroglu, and A. Behal, *Nonlinear Control of Wheeled Mobile Robots*, vol. 175. London, U.K.: Springer-Verlag, 2001.

[14] J.-B. Pomet, B. Thuilot, G. Bastin, and G. Campion, "A hybrid strategy for the feedback stabilization of nonholonomic mobile robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 1992, pp. 129–134.

[15] J. R. T. Lawton, R. W. Beard, and B. J. Young, "A decentralized approach to formation maneuvers," *IEEE Trans. Robot. Autom.*, vol. 19, no. 6, pp. 933–941, Dec. 2003.

[16] A. A. Cárdenas, S. Amin, Z.-S. Lin, Y.-L. Huang, C.-Y. Huang, and S. Sastry, "Attacks against process control systems: Risk assessment, detection, and response," in *Proc. 6th ACM Symp. Inf., Comput. Commun. Secur. (ASIACCS)*, 2011, pp. 355–366.

[17] S. Elizabeth and R. Jothilakshmi, "Convergence analysis of extended Kalman filter in a noisy environment through difference equations," *Int. J. Differ. Equ. Appl.*, vol. 14, no. 2, pp. 1–8, 2015.

[18] K. Reif, S. Gunther, E. Yaz, and R. Unbehauen, "Stochastic stability of the discrete-time extended Kalman filter," *IEEE Trans. Autom. Control*, vol. 44, no. 4, pp. 714–728, Apr. 1999.

[19] D. I. Urbina, J. A. Giraldo, A. A. Cardenas, N. O. Tippenhauer, J. Valente, M. Faisal, J. Ruths, R. Candell, and H. Sandberg, "Limiting the impact of stealthy attacks on industrial control systems," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2016, pp. 1092–1105.

[20] R. Goebel, R. G. Sanfelice, and A. Teel, "Hybrid dynamical systems," *IEEE Control Syst. Mag.*, vol. 29, no. 2, pp. 28–93, Apr. 2009.

[21] R. Goebel, R. G. Sanfelice, and A. Teel, *Hybrid Dynamical Systems: Modeling Stability, and Robustness*. Princeton, NJ, USA: Princeton Univ. Press, 2012.

[22] F. Mondada, M. Bonani, X. Raemy, J. Pugh, C. Cianci, A. Klaptocz, S. Magnenat, J.-C. Zufferey, D. Floreano, and A. Martinoli, "The e-puck, a robot designed for education in engineering," in *Proc. 9th Conf. Auto. Robot Syst. Competitions*, 2009, pp. 59–65.

LUIS BURBANO (Member, IEEE) received the B.S. degree in electronic engineering and the M.S. degree in computer and electronic engineering from the Universidad de los Andes, Bogotá, Colombia, in 2017 and 2019, respectively. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, University of California at Santa Cruz. His current research interest includes the development of security strategies to increase resiliency of cyber-physical systems.

LUIS FRANCISCO CÓMBITA (Member, IEEE) received the B.S. degree in electronics engineering from Universidad Distrital, Bogotá, Colombia, in 1992, and the M.S. degree in electrical engineering from the Universidad de los Andes, Bogotá, in 2002, where he is currently pursuing the Ph.D. degree.

He joined the Engineering Faculty, Universidad Distrital, as an Auxiliar Professor, in 1997, where he is currently an Assistant Professor. His current research interests include cyber-physical systems security, modeling and simulation of dynamical systems, and industrial control systems.

NICANOR QUIJANO (Senior Member, IEEE) received the B.S. degree in electronics engineering from Pontificia Universidad Javeriana, Bogotá, Colombia, in 1999, and the M.S. and Ph.D. degrees in electrical and computer engineering from The Ohio State University, Columbus, OH, USA, in 2002 and 2006, respectively.

He joined the Department of Electrical and Electronics Engineering, Universidad de los Andes (UAndes), Bogotá, as an Assistant Professor, in 2007, where he is currently a Full Professor and the Director of the Research Group in Control and Automation Systems. His current research interests include hierarchical and distributed optimization methods using bio-inspired and game-theoretical techniques for dynamic resource allocation problems, especially those in energy, water, and transportation.

SANDRA RUEDA (Member, IEEE) received the B.S. and M.S. degrees in systems and computing engineering from the Universidad de los Andes, Colombia, and the Ph.D. degree in computer science and engineering from The Pennsylvania State University, PA, USA. She is currently an Associate Professor with the Universidad de los Andes. Her research interests include security of software systems and security of the IoT, and mobile devices.