Distributed Reconfiguration for Resilient Synchronization of Multi-agent Systems

GILBERTO DÍAZ-GARCÍA1, (Student Member, IEEE), DANIEL GUATIBONZA2, and LUIS FELIPE GIRALDO2, (Member, IEEE)

1 Department of Electrical and Computer Engineering, University of California, Santa Barbara, United States.
2 Department of Electrical and Electronic Engineering, Universidad de los Andes, Bogotá, Colombia.

Corresponding author: Luis Felipe Giraldo (e-mail: lf.giraldo404@uniandes.edu.co).

This work was supported in part by the Vice Precidency for Research & Creation publication fund at Universidad de los Andes.

ABSTRACT The performance in output synchronization in multi-agent systems (MAS) can degrade in the presence of misbehaving agents that are affected by external attacker actions that try to desynchronize the system. In this paper, we introduce a heuristic algorithm based on best-response games to define an online distributed reconfiguration strategy to dynamically mitigate these actions, to make the system reach a synchronization state. This algorithm provides each unattacked agent with a decision-making based on local information to reconfigure the interaction patterns with its neighbors such that the system eventually synchronizes. We present some simulations of the proposed strategy for several systems in realistic scenarios to show ability of the of the proposed algorithm to mitigate different types of attacks.

INDEX TERMS Resilience, multi-agent systems, output synchronization, reconfiguration, best response games.

I. INTRODUCTION

The study of distributed control systems has become a relevant topic in control engineering. Its significance lies on the advantages that this type of controllers provide in large scale systems that include scalability and the capability to work in scenarios without full information [1], [2], [3]. A family of distributed control protocols that stand out are the synchronization algorithms. Their many applications in different fields involve formation control and flocking [4], synchronization of coupled oscillators [5], [6], data fusion in sensor networks [7], reinforcement learning [8], [9], and dynamics of human social systems [10], [11], [12], [13], [14]. Typically, these algorithms assume that all agents in the network are trustful, implying that they will follow a distributed pre-defined synchronization protocol. However, the misbehavior of at least one agent, deliberate or not, could deviate the system from a synchronization state. For this reason, the design of distributed control protocols that can be resilient to the action of misbehaving agents has gained importance over the past years [15], [16], [17], [18]. Resilient synchronization refers to the coordinated action of agents to reach a synchronization state in the presence of such misbehaving agents.

Several protocols that provide solutions to overcome the vulnerability of synchronization systems to malicious agents have been proposed, typically at the expense of increasing the complexity of them [19], [20], [21]. For example, in [22], agents that are considered to have the most dissimilar behavior are discarded, allowing unattacked agents to synchronize. This method, known as weighted-mean subsequence-reduced algorithm, can only work in a specific type of graphs, defined as robust networks, and with agents that are characterized by a 1-dimensional state variable. Similarly, the authors in [23] proposed to randomly change the connections over the graph to produce a resilient network. This solution, known as moving target defense, dramatically reduces the effect of misbehaving agents on the network but, due to randomness, malicious agents still affect the whole performance of the system. The study in [24] presented a mitigation strategy in linear systems that is based on the comparison of error sequences using the Kullback-Leibler divergence. With the information provided by the divergence, the nodes in the network can successfully neglect the effect of malicious agents. All the aforementioned algorithms share an important characteristic: they mitigate the effect of misbehaving agents over a network by changing the connections of the topology of the communication graph, preserving the pre-defined control algorithm. This supposes an advantage for the...
A. GRAPH THEORY CONCEPTS

Communication between agents is described by a weighted directed graph (digraph) $G = (V, E, w)$, where $V = \{1, ..., N\}$ is the set of agents, $E \subseteq V \times V$ is the edge set that describes the interaction between agents, and $w : V \times V \rightarrow \mathbb{R}_{\geq 0}$ is the weighting function of the edge set. The existence of edge $(i, j)$ states that agent $i$ receives information from agent $j$. The adjacency matrix of a graph $A = [a_{ij}]$ is a $N \times N$ matrix where $a_{ij} = w(i,j)$ if $(i,j) \in E$ and 0 otherwise. Let us define the neighbor set of node $i$ as $\mathcal{N}_i := \{j : (i, j) \in E\}$. The degree of a node is defined as the sum of all the weights of its neighbors, i.e. $d_i = \sum_j a_{ij}$.

A directed path between two nodes, $x$ and $y$, is a sequence of edges $((v_0, v_1), (v_1, v_2), \ldots, (v_{n-2}, v_{n-1}), (v_{n-1}, v_n))$ where $v_0 = x$, $v_n = y$ and $(v_i, v_{i+1}) \in E$ for all possible $i$. A graph is strongly connected if there exists a directed path between every pair of nodes in $V$. Also, the $\kappa-$connectivity of a graph is defined as the maximum number of nodes that can be removed while keeping the graph strongly connected.

Katz centrality of a matrix, denoted as $M = [m_{ij}]$, measures the relative importance of node $j$ for node $i$ in a directed graph taking into account all the possible walks between them. This metric is defined as [25]

$$M = \sum_{k=0}^{\infty} \eta^k A^k = (I - \eta A)^{-1} - I,$$

where $I$ is the identity matrix, $\eta \in [0, 1/d]$ is an attenuation factor that ensures that the summation converges and $d := \max_i d_i$. In Equation (1), $[A^k]_{ij}$ represents the weight of a walk from node $i$ to node $j$ in $k$ steps. Then, Katz centrality measures all the possible ways to connect two different nodes giving more importance to shorter walks. This metric has been widely used to characterize the importance in citation networks and web pages networks. Also, it characterizes the influence of every node in social networks based on their connections [26], [27], [28]. We propose to use it to show the indirect influence of malicious agents in cooperative control algorithms.

B. GAME THEORY CONCEPTS

A game is defined as a tuple $(V, \{S_i\}_{i \in V}, \{L_i\}_{i \in V})$, where $S_i$ is the set of available strategies for agent $i$, and $S_{-i} := \Pi_{j \in V \setminus \{i\}} S_j$. Let $L_i : S_i \times S_{-i} \rightarrow \mathbb{R}$ be a loss function associated with every agent, which reflects how profitable is to make an action when other agents already chose their actions. Therefore, each agent can compute its own optimal action using the best response for its neighbors’ actions $s_{-i} \in S_{-i}$, that is,

$$s_i^* = \arg\min_{s_i \in S_i} L_i(s_i, s_{-i}).$$

An important trait of finite $N$-player non-zero sum games is that they admits a Nash equilibrium in mixed strategies [29]. Therefore, it is ensured that agents will use an optimal strategy in steady-state conditions, i.e. every agent will be able to have the minimum possible loss in a situation in which all agents try to minimize of their own.

II. PRELIMINARIES
III. OUTPUT SYNCHRONIZATION IN MAS

Consider a network of agents described by a graph $\mathcal{G}$ as mentioned before, governed by the discrete dynamics

$$x_i(k+1) = f_i(x_i(k), u_i(k))$$
$$y_i(k) = h_i(x_i(k)),$$

where $x_i(k) \in \mathbb{R}^n$, $u_i(k) \in \mathbb{R}^m$ and $y_i(k) \in \mathbb{R}^p$ are the agent’s $i$ state, input and output, respectively, for the discrete time $k$. The objective in output synchronization in MAS is to drive all agents to an agreement space, that is,

$$\lim_{k \to \infty} ||y_i(k) - y_j(k)||^2 = 0 \quad \forall i, j \in \mathcal{V}.$$

The aforementioned task describes the consensus or output synchronization problem. This task has been addressed and solved for a wide variety of systems under different assumptions. Some of the best known systems that can achieve consensus are modeled by simple and double integrators, homogeneous linear systems, passive systems, among others, even in presence of communication delays or time-varying communication graphs [30], [31], [32], [33]. An important feature of these synchronization algorithms is that they can be described by the control action

$$u_i(k) = \sum_{j \in \mathcal{N}_i} a_{ij} \chi(y_j(k) - y_i(k)), \quad (2)$$

where $\chi : \mathbb{R}^p \to \mathbb{R}^m$ is an odd, continuous and locally Lipschitz function called coupling function. This control input makes all agents minimize

$$J_{ij}(k) := ||y_i(k) - y_j(k)||^2 \quad \forall (i, j) \in \mathcal{E}, \quad (3)$$

driving them to the agreement space [34, Theorem 6]. We refer to $J_{ij}$ as the performance function of the relationship between $i$ and $j$ from agent $i$’s perspective. To ensure the convergence of these algorithms, the communication graph $\mathcal{G}$ must be strongly connected [35]. The strongly connectivity condition states that every node is reachable from every other node in the graph, ensuring that all agents will synchronize using only local information. Although the parameters of this family of control algorithms affect how the agents synchronize, in future sections we assume that the control strategy is properly designed, i.e., the chosen $\chi(\cdot)$ guarantees synchronization for all agents in the absence of an attack.

IV. THREAT MODEL

We describe the attack with a subset of agents $\mathcal{V}_A \subseteq \mathcal{V}$, called attacked nodes, that do not follow the protocol described by the control algorithm. These attacked nodes are able to modify either their input or directly their output such that the function $J_{ij}$ is not minimized for all $\nu_i \in \mathcal{V}_A$. To this end, the attacker has access to all information from the attacked nodes including the control input calculated by the controller $u_i$, its state $x_i$, and output $y_i$. Therefore, when the unattacked nodes try to synchronize with the attacked nodes, they desynchronize, making the whole system diverge.

To design our attack mitigation strategy in a consensus system, we study a scenario where we can guarantee that an agreement can be reached. Therefore, we assume that there are some limitations on the capabilities of the attacker given in the following assumption:

**Assumption 1.** The subgraph that only includes the unattacked nodes $\mathcal{G}_U = (\mathcal{V}_U, \mathcal{E}_U, \mathcal{w}_U)$, with $\mathcal{V}_U = \mathcal{V} \setminus \mathcal{V}_A$, is strongly connected.

Assumption 1 guarantees that, if a mitigation strategy can isolate the attacked nodes, ignoring them does not affect the convergence of the synchronization algorithm. In other words, making this assumption means that the synchronization state can be reached once the attacked nodes have been ignored. Without this assumption, the reconfiguration strategy still isolates the attacked nodes, but no consensus algorithm will synchronize all the unattacked nodes in network.

Some sufficient conditions to ensure Assumption 1 are:

- the number of attacked nodes is less than the $\kappa$-connectivity of the graph; or
- the number of neighbors for every unattacked node is greater than the number of attacked nodes.

Even though the first condition can be less conservative, the second one is useful in scenarios where the whole communication graph is unknown, as in distributed control scenarios or large scale systems where the $\kappa$-connectivity can be difficult to compute.

V. GAME THEORY-BASED RECONFIGURATION STRATEGY

In schemes where the attacker manipulates a subset of nodes to perturb the network, a straight-forward strategy that mitigates the impact of the attack is to reconfigure the network such that unattacked nodes attenuate the attacked nodes’ actions to be able to synchronize, as illustrated in Fig. 1. Therefore, we propose a decision-making process based on game theory as a mechanism to reconfigure the weights over the network during the synchronization process to mitigate an attack. Each agent $i$ uses the information available from the synchronization strategy to define a game with its neighbors through a loss function $L_i(s_j, s_{-i})$ as defined in Section II. If the loss function is determined by acting in a way that function $J_{ij}$ in Equation (3) is minimized, then agent $i$’s decision-
making will be such that it will prefer to interact with those neighbors that will allow it to reach a synchronization state. This interaction is quantified by the weighting element $a_{ij}$ in the control action presented in Equation (2). Now, since agents only have partial information of the complete multiagent system, they cannot anticipate their neighbor’s performance $J_{ij}$ without losing the distributed property of the synchronization protocol. Hence, we introduce a prediction strategy that allows every agent in the network to estimate the behavior of its neighbors without having full information of the system. Fig. 2 shows a block diagram of the proposed reconfiguration strategy. At each time step, an individual anticipates the behavior of its neighbors and modifies its interaction patterns by adjusting the weights in the control action $a_{ij}$. Following, we discuss both the prediction strategy and the loss function.

### A. TRAJECTORY ANTICIPATION

Let $I_{ij}^{(k)}$ be the set that contains historical information between nodes $i$ and $j$ at time $k$ defined as

$$I_{ij}^{(k)} = (J_{ij}(0), J_{ij}(1), \ldots, J_{ij}(k-1), J_{ij}(k)).$$

This set contains all the previous information that agent $i$ has from agent $j$ characterized by its performance $J_{ij}$. Agent $i$ assumes that its neighbor $j$ behaves in a way that its performance can be described by some function $g(\cdot)$ as:

$$\hat{J}_{ij}(k+1) = g_{ij}\left(J_{ij}(k)\right) = g\left(J_{ij}(k) \mid \theta_{ij}^{(k)}\right),$$

where $\theta_{ij}^{(k)}$ are the parameters of the function $g$ at time $k$. This function allows agent $i$ to predict the behavior of its neighbor $j$ at the next step. Parameters $\theta_{ij}^{(k)}$ are calculated using the information contained in $I_{ij}^{(k)}$, and depends on the chosen prediction function $g(\cdot)$. These parameters can be estimated at each time step $k$ by any online estimation method, ranging from a linear regression to a neural network that uses transfer learning concepts. Given the parameters $\theta_{ij}^{(k)}$, at time step $k$, every agent has access to an estimation of the behavior of all its neighbors for a prediction horizon of $m$ steps using the recursive relation of the form

$$\hat{J}_{ij}(k + m) = g_{ij}^{(m)}(J_{ij}(k))$$

where

$$g_{ij}^{(m)}(J_{ij}(k)) = g_{ij}^{(m-1)}(J_{ij}(k))$$

(4)

Parameter $m$ depends on the dynamics of the agents in which the reconfiguration algorithm is being applied and can be tuned.

### B. NETWORK RECONFIGURATION

The reconfiguration strategy allows each unattacked agent to change its neighbor’s weights to mitigate the effect of an attacker’s actions. With the information provided by the estimation in (4), that represents in some way the cost to synchronize with each neighbor, the agent can effectively choose the network’s weights used to synchronize. In a game-theoretic framework, we can describe the available pure strategies for every agent as its neighbors, that is, $\mathcal{S}_i = \mathcal{N}_i$. Therefore, every agent calculates the loss function $L_i(s_j, s_{-i})$ according to their neighbors behavior to determine which neighbor is preferable to interact with. The performance of agent $i$ when it only chooses agent $j$, i.e. uses strategy $s_j \in \mathcal{S}_i$, can be computed as

$$L_i^{(k)}(s_j, s_{-i}) = \sum_{m=1}^{H} \xi^{H - m} g_{ij}^{(m)}(J_{ij}(k)),$$

where $s_{-i}$ are the strategies used by its neighbors, $H$ is the prediction horizon of the system performance, and $\xi \in [0, 1]$ is a discount factor that gives more relevance to future outcomes. Given that mixed strategies $\alpha_i \in \mathcal{P}(\mathcal{S}_i)$ are allowed, the expected loss is defined as

$$L_i^{(k)}(\alpha_i, \alpha_{-i}) = \sum_{j \in \mathcal{N}_i} \alpha_{ij} L_i^{(k)}(s_j, \alpha_{-i})$$

$$= \sum_{j \in \mathcal{N}_i} \alpha_{ij} \sum_{m=1}^{H} \xi^{H - m} g_{ij}^{(m)}(J_{ij}(k)),$$

(5)
where $\alpha_i = [\alpha_{ij}]$ for all $j \in N_i$ is the mixed strategy used by the agent $i$, and $\alpha_{ij}$ is the probability of agent $i$ to choose strategy $s_j$. Hence, $\sum_{j \in N_i} \alpha_{ij} = 1$.

The proposed loss function in Equation (5) drive agents to only choose their neighbors with minimum cost $L_i^k(s_j, s_{-i})$, completely discarding the other ones. This situation could lead agents to take premature decisions, leading to unnecessary disconnections between agents and, consequently, a deterioration in the system’s performance. Thus, we propose to include a randomization mechanism in the loss function through the Shannon entropy. With this incorporation, the agents not only are forced to minimize the synchronization performance loss, but also they try to maintain the most possible connections. The loss function can be redefined as

$$
\hat{L}_i^k(\alpha_i, \alpha_{-i}) = L_i^k(\alpha_i, \alpha_{-i}) + \lambda(k) \sum_{j \in N_i} \alpha_{ij} \log \alpha_{ij},
$$

being $\lambda(k)$ a regularization term that can change at every iteration $k$. This regularization term ensures that the performance loss and the entropy loss are comparable. We use a decreasing regularization term such that the performance loss gains importance over time. With a proper definition of the loss function, agents can choose the most profitable connections, i.e. the less expensive strategy, using the best response

$$
\alpha_i^*(k) = \arg \min_{\alpha_i \in \mathcal{P}(S_i)} \hat{L}_i^k(\alpha_i, \alpha_{-i}).
$$

and define the weights over network as $a_{ij}^*(k) = \alpha_{ij}^*(k)d_i$, where $d_i$ is the degree of node $i$. Note that these modified weights preserve the node degrees $d_i$ through time. The protocol defined in Equation (7) can be seen as a perturbed best response dynamics [36]. In them, the players of a game try to choose their best mixed strategy while perceiving a noisy version of their pay-off. In our case, by introducing noise to the perceived pay-off $L_i^k(\alpha_i, \alpha_{-i})$ with the Shannon entropy we force them to maintain more connections. Thus, each agent compute its input as

$$
u_i^*(k) = \sum_{j \in N_i} a_{ij}^*(k) \chi(y_j(k) - y_i(k)) = \sum_{j \in N_i} \alpha_{ij}^*(k)d_i \chi(y_j(k) - y_i(k)).
$$

The following theorem shows that, for each agent, we can find a closed form solution to this problem at each step $k$.

**Theorem 1.** The solution to the optimization problem in Equation (7) is

$$
\alpha_{ij}^*(k) = \frac{\exp(-L_i^k(s_j, \alpha_{-i})/\lambda(k))}{\sum_{l \in N_i} \exp(-L_i^k(s_l, \alpha_{-i})/\lambda(k))},
$$

Proof: The Lagrangian of the constrained problem in (7) is given by

$$
F_i(\alpha_i, \mu_i) = \sum_{j \in N_i} c_{ij} \alpha_{ij} + \lambda(k) \sum_{j \in N_i} \alpha_{ij} \log \alpha_{ij} + \mu_i \left( \sum_{j \in N_i} \alpha_{ij} - 1 \right),
$$

with $c_{ij} = L_i^k(s_j, \alpha_{-i})$. The gradient of $F_i$ is calculated as

$$
\frac{\partial F_i(\alpha_i^*, \mu_i^*)}{\partial \alpha_{ij}} = c_{ij} + \lambda(k)(1 + \log \alpha_{ij}) + \mu_i.
$$

At the optimum point $(\alpha_i^*, \mu_i^*)$, the gradient of $F_i$ is equal to 0. Then,

$$
\frac{\partial F_i(\alpha_i^*, \mu_i^*)}{\partial \alpha_{ij}} = 0 \Rightarrow \alpha_{ij}^* = \exp \left( -c_{ij} - \mu_i - \lambda(k) \right) / \lambda(k).
$$

Also, at the optimum point $(\alpha_i^*, \mu_i^*)$, the constraints of the optimization problem must be satisfied, leading to

$$
\sum_{l \in N_i} \alpha_{il} = 1 \Rightarrow \mu_i^* = \lambda(k) \left( \log \sum_{l \in N_i} \exp \left( -c_{il} / \lambda(k) \right) - 1 \right).
$$

Finally, substituting Equation (11) into Equation (10) we obtain Equation (9).

**Corollary 1.1.** When $\lambda(k) = 0$ for a given $k$, the problem is formulated as

$$
\alpha_i^*(k) = \arg \min_{\alpha_i \in \mathcal{P}(S_i)} L_i^k(s_j, \alpha_{-i}),
$$

whose solution corresponds to choosing any convex combination of neighbors that have minimum individual cost $L_i^k(s_j, \alpha_{-i})$.

It means that if one neighbor has the lowest $L_i^k$, then this neighbor is selected. When two or more neighbors have the lowest $L_i^k$, then the result of the optimization problem can be any convex combination of those neighbors. The whole reconfiguration strategy to compute the weights in the control action for synchronization is summarized in the Algorithm 1. Interestingly, the result in Theorem 1 has a similar form to the update rule computed by an agent in resilient distributed hypothesis testing, where each agent estimates a local belief hypothesis testing to map arbitrary values into probability distributions, providing a smooth version of argmin operator [39]. The main parameter of the soft-min function, in our
case \( \lambda(k) \), can be seen as a threshold defining when extreme numbers are discarded. Therefore, if the difference between a number and the real minimum is comparable to \( \lambda(k) \), then they are treated as similar numbers. On the other hand, if the difference is larger than \( \lambda(k) \), then those options are discarded. For our mitigation strategy, this point of view restates the importance of the regularization term. If \( \lambda(k) = 0 \), then the agents will discard many of the neighbors, preserving only the connections with agents with minimum loss, as noted in Corollary 1.1, making the reconfiguration strategy inconsistent. In addition, analyzing our strategy as a soft-min function provides us some information on how to choose \( \lambda(k) \). First, note that \( L_{ij} \) and \( J_{ij} \) satisfy

\[
J_{ij}(k + 1) \geq J_{ij}(k) \Rightarrow L^{(k+1)}_{i}(s_j, \alpha_{-i}) \geq L^{(k)}_{ij}(s_j, \alpha_{-i}),
\]

\[
J_{ij}(k + 1) < J_{ij}(k) \Rightarrow L^{(k+1)}_{i}(s_j, \alpha_{-i}) < L^{(k)}_{ij}(s_j, \alpha_{-i}).
\]

Knowing this, the choice of \( \lambda(k) \) must ensure that none of the nodes should be discarded prematurely because attacked nodes do not necessarily diverge at the initial time steps. Therefore, the initial condition of the parameter is chosen to be large enough to make all the neighbors comparable, i.e,

\[
\lambda(0) \geq \sum_{m=1}^{H} \xi^{H-m} \hat{J}^{+}_{ij}(0),
\]

with \( \hat{J}^{+}_{ij}(0) \) being the maximum of possible value that \( \hat{J}_{ij}(0) \) can take given some initial conditions. Additionally, \( \lambda(k) \) must be decreasing with respect to \( k \), in a way that the proposed algorithm is able to discard attacked nodes. However, if it decreases too fast, some unattacked nodes could be discarded. To avoid this situation, we can use the fact that the performance cost of every unattacked node will decrease. Then, there exists a constant \( \beta \in (0, 1) \) such that

\[
L^{(k+1)}_{i}(s_j, \alpha_{-i}) \leq \beta L^{(k)}_{i}(s_j, \alpha_{-i}), \quad \forall j \in \mathcal{N}_{i}.
\]

This constant denotes the slowest decreasing rate for all unattacked nodes. Thus, the evolution of \( \lambda \) can be set as

\[
\beta \lambda(k) \leq \lambda(k + 1) \leq \lambda(k).
\]

This condition leads to state that, with a proper choice of \( \lambda(k) \), the mitigation strategy described in Equation (7) will discard the attacked nodes. At some time step \( k^* \), the loss \( L_{ij} \) of every attacked node will, at least, remain the same while the loss of the unattacked nodes will decrease. This affirmation not only guarantees that the proposed reconfiguration algorithm will remove the attacked nodes. Also, it states that the output synchronization algorithm still works without the presence of an attack because the loss of every node will decrease and their connections will be preserved.

It is important to mention that this is a heuristic algorithm, and that convergence to synchronization is not guaranteed, unless all unattacked nodes are disconnected. Through simulation study cases we show that this algorithm works in a wide variety of synchronization environments in which agents are under attack.

### VI. STUDY CASES

We implemented the strategy shown in Equation (7) in three different systems to show the performance of the proposed mitigation strategy: a discrete consensus, consensus filters for sensor networks, and formation control in differential-drive robots in a realistic scenario. In these study cases, we conducted a weighted linear regression at each time step \( k \) to find the parameters of the prediction function \( g(\cdot) \),

\[
J_{ij}(k + 1) = \theta_0 + \theta_1 J_{ij}(k),
\]

with

\[
\theta_{ij}(k) = \arg \min_{(\theta_0, \theta_1)} \sum_{m=0}^{k-1} \gamma^{k-1-m} [\theta_0 + \theta_1 J_{ij}(m) - J_{ij}(m + 1)]^2,
\]

where \( \gamma \) is a discount factor that gives priority to latest data. We chose linear regression because of its simplicity of implementation and it has an explicit solution through the least square minimization problem, facilitating an online estimation. Additional parameters for the following simulations are \( \xi = 0.75 \), \( H = 5 \) and \( \gamma = 0.75 \). For each multi-agent system, we conducted multiple simulations to observe the performance of the proposed algorithm for different communication graphs, number of attacked nodes, and initial conditions. For comparison purposes, we showed the system’s performance in four different scenarios for each study case. The first two scenarios are in the absence of an attack. The first one does not have the reconfiguration strategy to show the performance of the system in normal operation conditions. The second one implements the mitigation strategy to observe how it affects the aforementioned expected performance. The last two scenarios are in the presence of an attack. First, when the mitigation strategy is not implemented, and then when it is implemented.
A discrete consensus

In the first set of simulations, we implemented the proposed reconfiguration strategy for discrete consensus, which is a well-studied case of output synchronization. The dynamics of this system is described as

\[ x_i(k + 1) = x_i(k) + \delta \sum_{j \in N_i} a_{ij} [y_j(k) - y_i(k)] \]

where \( \delta \) is the desynchronization bias. For the initial communication graph, we used a 7-node graph generated as a binomial random graph with probability of connections \( p = 0.5 \) and weights \( a_{ij}(0) \in [0.8, 1.2] \). The regularization parameter is set to follow the rule \( \lambda(k) = 7(0.95)^k \) following the recommendations in Section V-C. To measure the performance of the system, we computed the mean of the average difference between every pair of unattacked agents. This can be described as

\[ \mathcal{D} := \{(i, j) \mid (i, j) \in \mathcal{V}_U \land (i > j)\} \]

\[ P(k) = \frac{1}{|\mathcal{D}|} \sum_{(i,j) \in \mathcal{D}} ||y_i(k) - y_j(k)||^2, \]

where \( \mathcal{D} \) is the set of all possible pairs of unattacked nodes, and \( P(k) \in \mathbb{R} \) is the performance index at the time step \( k \). To make the performance comparable for all simulations disregarding initial conditions, we used a normalized performance index

\[ \hat{P}(k) = \frac{P(k)}{P(0)}. \]

We run a set of 1000 simulations to verify the algorithm performance under different communication graphs and initial conditions. Fig. 3 shows the behavior of \( y_i \) and \( \hat{P} \) under four different scenarios. Figs. 3(a) and 3(e) show that the performance decreases to 0, meaning that agents achieve consensus in absence of an attack and with no reconfiguration. Similarly, in Figs. 3(b) and 3(f), \( \hat{P}(k) \) decreases as time increases, verifying that reconfiguration algorithm does not affect the synchronization algorithm. On the other hand, Figs. 3(c) and 3(g) show how the desynchronization bias does not allow the agents to synchronize. Conversely, they keep separated when they try to get closer to the attacked agent. Finally, Figs. 3(d) and 3(h) show that the inter-agent distance decreases even in presence of an attacked node, showing the proper functioning of the proposed algorithm for attack mitigation. This can be verified in Fig. 3(h) where unattacked agents ignore the malicious agent and synchronize.

In addition to the previous set of simulations, we compare the performance of the proposed algorithm with other strategies for resilient reconfiguration in this simple yet illustrative study case. We implemented the moving target defense (MTD) strategy presented in [23] and the weighted-mean subsequence-reduced (W-MSR) algorithm presented in [22]. In MTD, at each time step each agent randomly...
connects and disconnects communication with its neighbors according with a probability $p_d$. Here, we set $p_d = 0.5$. In W-MSR, at each time step an agent disconnects those neighbors who have a state variable that are strictly larger or lower than its own, as long as there are at most $F$ neighbors that satisfy this. Here, we set $F = 1$. We run an additional set of 1000 simulations under different communication graphs and initial conditions and tested our proposed algorithm and the competing algorithms. Fig. 4 shows the performance $\hat{P}$ in Equation (15) for the different reconfiguration strategies, and Fig. 5 shows the trajectories of the agents for a single simulation. Note that our algorithm outperforms the other algorithms in this scenario. While random disconnections in MTD lead to a reduced effect of the attacker in the network, the attack still affects the synchronization between unattacked nodes. Likewise, the W-MSR algorithm guarantees convergence when it is applied to a special family of network called robust networks [22]. However, since the
proposed networks are not necessarily considered as robust, there exist cases when this algorithm disconnects the network, perturbing the synchronization process even in absence of an attack. On the other hand, our algorithm based on best-response games conducts a decision-making based on the prediction of a general performance criterion and does not try to not reduce arbitrarily the connectivity of the network, allowing for a resilient synchronization in a wide variety of scenarios.

B. CONSENSUS FILTERS FOR SENSOR NETWORKS

After showing the performance in a standard synchronization multi-agent system, we tested our reconfiguration strategy in noisy environments. For this purpose, we implemented our strategy in the sensor fusion network scheme developed in [7]. All the sensor agents compare their measurements with their neighbors driving them to the consensus agreement space. This makes that all agents’ states preserve the same mean of the measurements, but reducing the variance of them. The dynamics of these sensor agents are described as

\[ x_i(k+1) = x_i(k) + \delta \left( \sum_{j \in \mathcal{N}_i} a_{ij}[y_j(k) - y_i(k)] + \sum_{j \in \mathcal{J}_i} a_{ij}[u_j(k) - y_i(k)] \right) \]

(16)

where \( \mathcal{J}_i := \mathcal{N}_i \cup \{i\} \), \( a_{ii} = 1 \) for all \( i \), and \( u_i \) is the measurement of the \( i \) sensor. With the dynamics described in Equation (16), the agents will merge their measurements reducing the variance along the network. In this simulation, all agents are sampling a standard normal random variable.

\[ y_i(k) = u_i(k) + 10\sigma(0.025(k - 150)) \]

\[ \sigma(x) = \frac{1}{1 + \exp(-x)}. \]

This attack introduces not only a bias but also noisier measurements. In this case, the attack disperses the agents, making all states differ from the real measure. This misinformation can lead the system to make wrong decisions because the observed measurement is not correct and, in most cases, different for all nodes.

To check the performance in large-scale systems, we implemented the system described in Equation (16) with a 100-node Erdos-Renyi unweighted random graph as an initial communication graph and, to test it in the worst possible scenario, we attacked the \( \kappa - 1 \) most important nodes according to Katz centrality, as defined in Equation (1), to preserve Assumption 1. The regularization parameter is defined by the rule \( \lambda(k) = 7(0.9)^k + 2 \). The performance of the system is evaluated by analyzing the average of the unattacked agents, i.e.,

\[ P(k) = \frac{1}{|\mathcal{V}_{\text{dr}}|} \sum_{i \in \mathcal{V}_{\text{dr}}} y_i(k), \]

(17)

For a set of 150 simulations, Fig. 6 shows the behavior of \( y_i \), and \( P \) from Equation (17). According to [7], unattacked
nodes will preserve the mean of the original measurements as shown in Figs. 6(a) and 6(e). The reconfiguration strategy does not perturb the behavior of the network in absence of an attack, as presented in Figs. 6(b) and 6(f), which is a desirable property in this type of cyber-security strategies. When the described attack occurs, the average of the agents deviates from the expected mean, leading to misinformation in the sensor network, as shown in Figs. 6(c) and 6(g). When the mitigation strategy is implemented, the nodes along the network can ignore the attacked nodes, preserving again the mitigation strategy is implemented, the nodes along the network can ignore the attacked nodes, preserving again the mean of the sampled variable \( u_i \) as presented in Figs. 6(d) and 6(h).

C. FORMATION CONTROL IN DIFFERENTIAL DRIVE ROBOTS

Finally, we tested our reconfiguration strategy in a realistic environment involving the control of a set of differential-drive robots. We implemented five pioneer-like differential-drive robots in an environment created in CoppeliaSim from Coppelia Robotics on the Robot Operating System (ROS). This environment recreates the physics of the robots and allows for the application of a local-level control policy to drive the speed of each wheel and a high-level control to define each robot’s decision-making. Fig. 7 shows a snapshot of the differential drive robots in the simulation environment.

![FIGURE 7. Pioneer-like differential drive robots on CoppeliaSim used to test the proposed reconfiguration strategy.](image)

The main goal of the set of robots is to reach a formation of a regular pentagon and to avoid those whose control policy has been attacked. The decision-making of each robot is defined as shown in Fig. 8 under a sampling time of 0.25 seconds. Robot \( i \) has a position at time step \( k \) denoted by \( y_i(k) \in \mathbb{R}^3 \) that is defined by the 2-dimensional coordinates of the middle point between its wheels in the global reference frame along with the rotation angle \( \theta \) within this frame and the robot’s. Robot \( i \) is able to read its neighbors’ position \( y_j(k) \), for all \( j \in N_i \), according to a communication topology. In this set of simulations, we assumed that they communicate following fully-connected network. Given its own position and its neighbors’, robot \( i \) estimates the reference position \( y_i^{ref} \) that it should follow at the next time step. This reference is computed using a protocol based on a consensus scheme and the proposed reconfiguration strategy in order to reach the desired formation and to avoid attacked robots. The formation control can be achieved using the Laplacian-based control:

\[
y_i^{ref}(k+1) = y_i^{ref}(k) + 0.02 \sum_{j \in N_i} a_{ij}(y_j(k) - y_i(k) + \Delta_{ij}),
\]

where \( \Delta_{ij} \) is the expected distance between agents that defines the pentagonal formation and \( a_{ij} \) is defined by Algorithm 1.

Given the reference position \( y_i^{ref} \), robot \( i \) computes the required change in \( y_i \), defined as \( \delta_i \), to reach the reference position. This is achieved by a data-driven controller, known as MFAC [40], which conducts an data-based online estimation of the nonlinear dynamics of the robot in the environment. To associate this change in the components of the robot in the global frame with the angular velocities of each of robot’s wheels \( \phi_{i,1} \) and \( \phi_{i,2} \), a set of kinematic equations that consider a rotation map and the rolling and non-sliding control are applied. A proportional-integral-derivative (PID) controller defines the policy that drives each wheel given the angular velocities \( \phi_{i,1} \) and \( \phi_{i,2} \). Details of the kinematic equation of the robot and the MFAC control are given in the appendix.

To disturb this system, one of the agents drives the whole formation to a new final position. This agent defines its reference position as

\[
y_i^{ref}(k) = r_a
\]

where \( r_a \in \mathbb{R}^3 \) is the position of the attacked robot, moving all the formation to a new synchronization point as shown in Fig. 9(c). Here, by perturbing only one node, all agents move to an unexpected region making this type of attacks a real hazard in scenarios that involve autonomous vehicles [23]. With this in mind, we measured the performance of the system by comparing how much every agent deviates from the ideal case. This case is when there is neither an attack nor a reconfiguration strategy, as for example in Fig. 9(a). This can be described mathematically as,

\[
P(k) = \frac{1}{|\mathcal{V}_d|} \sum_{i \in \mathcal{V}_d} \| y_i(k) - y_i^* \|^2,
\]

where \( y_i^* \) is the final position of agent \( i \) in the ideal case. As in discrete consensus simulations, we used a normalized version of the performance index to make all simulations comparable, that is, \( \hat{P}(k) \) as described in Equation (15). We implemented the mitigation strategy with \( \xi = 0.75 \), and
FIGURE 9. For a set of 10 simulations (mean: solid line, and standard deviation: light-colored shade), behavior of a formation control system (a)(e) without attack and without reconfiguration, (b)(f) without attack and with reconfiguration, (c)(g) with attack and without reconfiguration, and (d)(h) with attack and reconfiguration. Figs. (a)-(d) present the trajectory of the attacked node (dashed line), and Figs. (e)-(h) present the performance index in Equation (18).

\[ \lambda(k) = 3000(0.99)^k \] as regularization parameter. Furthermore, the prediction horizon is set to \( H = 5 \) to capture the nonlinear behavior of the robotic system. We tested our algorithm using 10 simulations to verify its performance under different initial conditions in the position of the robots.

Figs. 9(a) and 9(e) show how the indicator of the system decreases to 0 as expected by the performance definition in Equation (18) since there are no attacks. Figs. 9(b) and 9(f) show the case when our strategy is implemented in absence of a malicious attack. The performance index decreases indicating that the reconfiguration strategy slightly perturbs the agents’ final positions when is implemented. However, it can be observed that the performance decreases at a slower rate than the ideal case due to changes in the network’s weights. On the other hand, the attacker’s deviation makes all agents move away from the expected position \( y_i^r \) as presented in Figs. 9(c) and 9(g). However, the proposed reconfiguration strategy clearly reduces the effect of the attacker on the network, bounding the dissimilarity between the original formation and the one after reconfiguration, as shown in Figs. 9(d) and 9(h).

VII. CONCLUSIONS

We introduced a heuristic algorithm based on a best-response game for attack mitigation in which every unattacked agent in the network dynamically changes its weights to achieve a synchronization state. The mechanism used to choose the weights is based on the game theory concept of Nash equilibrium, where all agents try to minimize a cost related to the performance of their neighbors. We presented some simulations of the proposed mitigation strategy implemented on several systems that we considered provided a wide variety of scenarios. In these simulations, we showed the performance of the reconfiguration algorithm in different scenarios, verifying the capabilities of the algorithm to mitigate different kinds of attacks.

As future work, we propose to implement and evaluate the suggested strategy in a real physical systems. Since the purpose of these secure strategies is to prevent and mitigate malfunctions in cyber-physical systems, it is important to analyze the performance of our strategy taking into consideration implementation aspects that also involve the presence of delays, faults, and communication noise [41], [42], [43]. Also, other types of threat models, regression models based on online estimations, and synchronization challenges such as containment control in distributed systems can be studied [44], [45].

APPENDIX A DETAILS OF THE IMPLEMENTATION OF THE ROBOTIC SYSTEM

A. KINEMATICS OF THE ROBOT

A differential robot is represented by \( w = [x, z, \theta]^T \), which is the 2-dimensional coordinates of the middle point between its wheels in the global reference frame \((x, z)\) along with the rotation angle \( \theta \) within this frame and the robot’s. The rotation matrix \( R(\theta) \), which is given by

\[
R(\theta) = \begin{bmatrix}
\cos(\theta) & \sin(\theta) & 0 \\
-\sin(\theta) & \cos(\theta) & 0 \\
0 & 0 & 1
\end{bmatrix},
\]

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/
maps the point $w$ to the robot’s frame as $w_R = R(\theta)w = [x_R, z_R, \theta_R]^T$. The subindex $R$ refers to the robot’s reference frame.

To associate the velocity components of the robot in the global frame to the velocities of each of its wheels, the rolling and non-sliding equations associated with the physical structure of the robot are used. The rolling equation that describes the motion of the robot for each of the wheels based on the physical structure of the wheels in the robot chassis are defined as:

$$\dot{x}_R \sin(\alpha + \beta) - z_R \cos(\alpha + \beta) - \theta_R l \sin(\beta) = q \dot{\varphi},$$

where $2l$ represents the distance between wheels, $q$ is the radius of the particular wheel, and $\dot{\varphi}$ is its angular speed. Constants $\alpha$ and $\beta$ are constants to each wheel: $\alpha = -\pi/2$ and $\beta = \pi$ for the right wheel, and $\alpha = \pi/2 \ y \ \beta = 0$ for the left one.

The non-sliding equation limits, under the assumption of unidirectional motion of the wheels, the direction in which the robot does not move, and is defined as

$$\dot{x}_R \cos(\alpha + \beta) + z_R \sin(\alpha + \beta) + \theta_R l \sin(\beta) = 0.$$

By posing the associated system of equations and clearing the angular velocities, the following matrix system is obtained:

$$\begin{bmatrix} \dot{\varphi}_1 q \\ \dot{\varphi}_2 q \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & l \\ 1 & 0 & -l \\ 0 & 1 & 0 \end{bmatrix} R(\theta) \begin{bmatrix} \dot{x}_R \\ \dot{z}_R \\ \theta_R \end{bmatrix}.$$

### B. MFAC TYPE CONTROL

Model free adaptive control (MFAC) is a data-based type of control used to determine the rate of change of the position of robot $i$ that is required to reach the reference point $y_{ref}^i$. This strategy uses an iterative estimation of the partial pseudo-derivative matrix $\phi_i$ [40], which estimates the relationship between the output and the input of a nonlinear system. In our case, $\phi_i$ estimates the relationship between robot $i$ position $y_i$ and the velocity $\dot{y}_i$ that is required by the robot to reach the reference position $y_{ref}^i$. The update rule for $\phi_i$ is given by

$$\dot{\phi}_i(k) = \phi_i(k - 1) + \frac{\eta \Delta y_i(k) - \phi_i(k - 1) \Delta \dot{y}_i(k - 1)}{\mu + \|\phi_i(k - 1)\|^2} \Delta \dot{y}_i^2(k - 1),$$

where $\phi_i(1) = I, \eta = 1$ and $\mu = 1$. This rule updates matrix $\phi_i$ based on the changes in the input and the output $\Delta y_i(k) = y_i(k) - y_i(k - 1)$ and $\Delta \dot{y}_i(k) = \dot{y}_i(k) - \dot{y}_i(k - 1)$. Given a reference signal $y_{ref}^i$, based on the calculation of the pseudo-partial derivative $\phi_i$, the control signal $\dot{w}_i$ is given by the expression

$$\dot{w}_i(k) = \rho \rho_i^T(k) \|y_{ref}^i - y_i(k)\| \nu + \|\phi_i(k)\|^2,$$

where $\nu = 1$ and $\rho = 2.7$. 

### REFERENCES

[1] V. Gazi and K. M. Passino, “Stability analysis of swarms,” IEEE Transactions on Automatic Control, vol. 48, no. 4, pp. 692–697, 2003.

[2] N. Chopra and M. W. Spong, “On exponential synchronization of kuromoto oscillators,” IEEE Transactions on Automatic Control, vol. 54, no. 3, pp. 353–357, 2009.

[3] N. Quijano and K. M. Passino, “Honey bee social foraging algorithms for resource allocation: Theory and application,” Engineering Applications of Artificial Intelligence, vol. 23, no. 6, pp. 845–861, 2010.

[4] G. Díaz-García, L. Burbano, N. Quijano, and L. F. Giraldo, “Distributed MPC and potential game controller for consensus in multiple differential-drive robots,” in 2019 IEEE 4th Colombian Conference on Automatic Control (CCAC). IEEE, 2019, pp. 1–6.

[5] J. Giraldo, E. Mojica-Nava, and N. Quijano, “Synchronization of isolated microgrids with a communication infrastructure using energy storage systems,” International Journal of Electrical Power & Energy Systems, vol. 63, pp. 71–82, 2014.

[6] B. Wei, F. Xiao, and Y. Shi, “Synchronization in kuramoto oscillator networks with sampled-data updating law,” IEEE Transactions on Cybernetics, vol. 50, no. 6, pp. 2378–2388, 2019.

[7] R. Olfati-Saber and J. S. Shamma, “Consensus filters for sensor networks and distributed sensor fusion,” in Proceedings of the 44th IEEE Conference on Decision and Control. IEEE, 2005, pp. 6698–6703.

[8] Y. Yang, K. G. Vanmoudakis, and H. Modares, “Safe reinforcement learning for dynamical games,” International Journal of Robust and Nonlinear Control, vol. 30, no. 9, pp. 3706–3726, 2020.

[9] Y. Yang, D.-W. Ding, H. Xiong, Y. Yin, and D. C. Wunsch, “Online barrier-actor-critic learning for h infinity control with full-state constraints and input saturation,” Journal of the Franklin Institute, vol. 357, no. 6, pp. 3316–3344, 2020.

[10] L. F. Giraldo and K. M. Passino, “Dynamic task performance, cohesion, and communications in human groups,” IEEE Transactions on Cybernetics, vol. 46, no. 10, pp. 2207–2219, 2016.

[11] L. F. Giraldo, K. M. Passino, J. D. Clapp, and D. Ruderman, “Dynamics of metabolism and decision making during alcohol consumption: Modeling and analysis,” IEEE Transactions on Cybernetics, vol. 47, no. 11, pp. 3955–3966, 2017.

[12] L. F. Giraldo and K. M. Passino, “Dynamics of cooperation in a task completion social dilemma,” Plos one, vol. 12, no. 1, p. e0170604, 2017.

[13] L. F. Giraldo, K. M. Passino, and J. D. Clapp, “Modeling and analysis of group dynamics in alcohol-consumption environments,” IEEE Transactions on Cybernetics, vol. 47, no. 1, pp. 165–176, 2017.

[14] A. F. Zambrano, G. Díaz-García, S. Ramírez, L. F. Giraldo, H. González-Villansisti, M. T. Perdomo, I. D. Hernández, and J. M. Godoy, “Donation networks in underprivileged communities,” IEEE Transactions on Computational Social Systems, vol. 8, no. 1, pp. 76–85, 2020.

[15] 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 Proceedings of the 2017 ACM symposium on information, computer and communications security. ACM, 2011, pp. 355–366.

[16] L. F. Cóbotta, J. Giraldo, A. A. Cárdenas, and N. Quijano, “Response and reconfiguration of cyber-physical control systems: A survey,” in Proceedings of the 2015 IEEE 2nd Colombian Conference on Automatic Control (CCAC). IEEE, 2015, pp. 1–6.

[17] H. S. Foroush and S. Martinez, “On multi-input controllable linear systems under unknown periodic dos jamming attacks,” in Proceedings of the 2013 Conference on Control and its Applications. SIAM, 2013, pp. 222–229.

[18] L. Burbano, L. F. Cóbotta, N. Quijano, and S. Rueda, “Dynamic data integration for resilience to sensor attacks in multi-agent systems,” IEEE Access, vol. 9, pp. 31 236–31 245, 2021.

[19] D. Meng and K. L. Moore, “Studies on resilient control through multiagent consensus networks subject to disturbances,” IEEE Transactions on Cybernetics, vol. 44, no. 11, pp. 2050–2064, 2014.

[20] R. Moghadam and H. Modares, “Resilient autonomous control of distributed multiagent systems in contested environments,” IEEE Transactions on Cybernetics, vol. 49, no. 11, pp. 3957–3967, 2018.

[21] H. Modares, B. Kiumarsi, F. L. Lewis, F. Ferrese, and A. Davoudi, “Resilient and robust synchronization of multiagent systems under attacks on sensors and actuators,” IEEE Transactions on Cybernetics, vol. 50, no. 3, pp. 1240–1250, 2019.

[22] H. J. LeBlanc, H. Zhang, X. Koutsoukos, and S. Sundaram, “Resilient asymptotic consensus in robust networks,” IEEE Journal on Selected Areas in Communications, vol. 31, no. 4, pp. 766–781, 2013.
[23] J. Giraldo and A. A. Cardenas, “Moving target defense for attack mitigation in multi-vehicle systems,” in Proactive and Dynamic Network Defense. Springer, 2019, pp. 163–190.

[24] A. Mustafa, H. Modares, and R. Moghadam, “Resilient synchronization of distributed multi-agent systems under attacks,” Automatica, vol. 115, p. 108869, 2020.

[25] L. Katz, “A new status index derived from sociometric analysis,” Psychometrika, vol. 18, no. 1, pp. 39–43, 1953.

[26] P. Grindrod, M. C. Parsons, D. J. Higham, and E. Estrada, “Communicability across evolving networks,” Physical Review E, vol. 83, no. 4, p. 046120, 2011.

[27] G. Liu, H. Zhang, and T. Huang, “Neural-network-based model-free adaptive control for a class of nonlinear systems,” IEEE Access, vol. 6, pp. 23892–23907, 2018.

[28] Z. Hou and S. Jin, “Data-driven model-free adaptive control for a class of nonlinear systems with dynamic uncertainties,” IEEE Transactions on Neural Networks and Learning Systems, vol. 32, no. 5, pp. 2239–2250, 2020.

[29] Q. Zhu and T. Basar, “Game-theoretic methods for robustness, security, and resilience of cyberphysical control systems: games-in-games principle for optimal cross-layer resilient control systems,” IEEE Control Systems Magazine, vol. 35, no. 1, pp. 46–65, 2015.

[30] J. Zhao, T.-H. Yang, Y. Huang, and P. Holme, “Ranking candidate disease genes from gene expression and protein interaction: a katz-centrality based approach,” PLoS one, vol. 6, no. 9, p. e24306, 2011.

[31] R. Olfati-Saber, J. A. Fax, and R. M. Murray, “Consensus and cooperation in networked multi-agent systems,” Proceedings of the IEEE, vol. 95, no. 1, pp. 215–233, 2007.

[32] N. Chopra, “Output synchronization on strongly connected graphs,” IEEE Transactions on Automatic Control, vol. 57, no. 11, pp. 2896–2901, 2012.

[33] A. Isidori, L. Marconi, and G. Casadei, “Robust output synchronization of a network of heterogeneous nonlinear agents via nonlinear regulation theory,” IEEE Transactions on Automatic Control, vol. 59, no. 10, pp. 2680–2691, 2014.

[34] A. Lalitha, A. Sarwate, and T. Javidi, “Social learning and distributed hypothesis testing,” in Advances in robot control. Springer, 2006, pp. 107–134.

[35] H. Liang, G. Liu, H. Zhang, and T. Huang, “Neural-network-based event-triggered adaptive control of nonaffine nonlinear multiagent systems with dynamic uncertainties,” IEEE Transactions on Neural Networks and Learning Systems, vol. 32, no. 5, pp. 2239–2250, 2020.

[36] A. Mustafa, H. Modares, and R. Moghadam, “Resilient synchronization of distributed multi-agent systems under attacks,” Automatica, vol. 115, p. 108869, 2020.

[37] L. Katz, “A new status index derived from sociometric analysis,” Psychometrika, vol. 18, no. 1, pp. 39–43, 1953.

[38] P. Grindrod, M. C. Parsons, D. J. Higham, and E. Estrada, “Communicability across evolving networks,” Physical Review E, vol. 83, no. 4, p. 046120, 2011.

[39] Z. Hou and S. Jin, “Data-driven model-free adaptive control for a class of nonlinear systems,” IEEE Access, vol. 6, pp. 23892–23907, 2018.

[40] J. Zhao, T.-H. Yang, Y. Huang, and P. Holme, “Ranking candidate disease genes from gene expression and protein interaction: a katz-centrality based approach,” PLoS one, vol. 6, no. 9, p. e24306, 2011.

[41] R. Olfati-Saber, J. A. Fax, and R. M. Murray, “Consensus and cooperation in networked multi-agent systems,” Proceedings of the IEEE, vol. 95, no. 1, pp. 215–233, 2007.

[42] N. Chopra, “Output synchronization on strongly connected graphs,” IEEE Transactions on Automatic Control, vol. 57, no. 11, pp. 2896–2901, 2012.

[43] A. Isidori, L. Marconi, and G. Casadei, “Robust output synchronization of a network of heterogeneous nonlinear agents via nonlinear regulation theory,” IEEE Transactions on Automatic Control, vol. 59, no. 10, pp. 2680–2691, 2014.

[44] A. Lalitha, A. Sarwate, and T. Javidi, “Social learning and distributed hypothesis testing,” in Advances in robot control. Springer, 2006, pp. 107–134.

[45] H. Liang, G. Liu, H. Zhang, and T. Huang, “Neural-network-based event-triggered adaptive control of nonaffine nonlinear multiagent systems with dynamic uncertainties,” IEEE Transactions on Neural Networks and Learning Systems, vol. 32, no. 5, pp. 2239–2250, 2020.

[46] A. Mustafa, H. Modares, and R. Moghadam, “Resilient synchronization of distributed multi-agent systems under attacks,” Automatica, vol. 115, p. 108869, 2020.

[47] L. Katz, “A new status index derived from sociometric analysis,” Psychometrika, vol. 18, no. 1, pp. 39–43, 1953.

[48] P. Grindrod, M. C. Parsons, D. J. Higham, and E. Estrada, “Communicability across evolving networks,” Physical Review E, vol. 83, no. 4, p. 046120, 2011.

[49] Z. Hou and S. Jin, “Data-driven model-free adaptive control for a class of nonlinear systems,” IEEE Access, vol. 6, pp. 23892–23907, 2018.

[50] J. Zhao, T.-H. Yang, Y. Huang, and P. Holme, “Ranking candidate disease genes from gene expression and protein interaction: a katz-centrality based approach,” PLoS one, vol. 6, no. 9, p. e24306, 2011.

[51] R. Olfati-Saber, J. A. Fax, and R. M. Murray, “Consensus and cooperation in networked multi-agent systems,” Proceedings of the IEEE, vol. 95, no. 1, pp. 215–233, 2007.

[52] W. H. Sandholm, “Population games and deterministic evolutionary dynamics,” in Handbook of game theory with economic applications. Elsevier, 2015, vol. 4, pp. 703–778.

[53] A. Lalitha, A. Sarwate, and T. Javidi, “Social learning and distributed hypothesis testing,” in 2014 IEEE International Symposium on Information Theory. IEEE, 2014, pp. 551–555.

[54] B. Wu, S. Carr, S. Bharadwaj, Z. Xu, and U. Topcu, “Byzantine-resilient distributed hypothesis testing with time-varying network topology,” arXiv preprint arXiv:2008.00164, 2020.

[55] C. M. Bishop, Pattern recognition and machine learning. springer, 2006.

[56] Z. Hou and S. Jin, “Data-driven model-free adaptive control for a class of mimo nonlinear discrete-time systems,” IEEE transactions on neural networks, vol. 22, no. 12, pp. 2173–2188, 2011.

[57] H. Liang, X. Guo, Y. Pan, and T. Huang, “Event-triggered fuzzy bipartite tracking control for network systems based on distributed reduced-order observers,” IEEE Transactions on Fuzzy Systems, 2020.

[58] H. Liang, G. Liu, H. Zhang, and T. Huang, “Neural-network-based event-triggered adaptive control of nonaffine nonlinear multiagent systems with dynamic uncertainties,” IEEE Transactions on Neural Networks and Learning Systems, vol. 32, no. 5, pp. 2239–2250, 2020.

[59] A. Tahoun and M. Arafa, “Cooperative control for cyber–physical multi-agent networked control systems with unknown false data injection and replay cyber-attacks,” ISA transactions, vol. 110, pp. 1–14, 2021.

[60] Y. Yang, H. Modares, D. C. Wunsch, and Y. Yin, “Optimal containment control of unknown heterogeneous systems with active leaders,” IEEE Transactions on Control Systems Technology, vol. 27, no. 3, pp. 1228–1236, 2019.

[61] Y. Yang, H. Modares, K. G. Vamvoudakis, Y. Yin, and D. C. Wunsch, “Dynamic intermittent feedback design for infinite containment control on a directed graph,” IEEE transactions on cybernetics, vol. 50, no. 8, pp. 3752–3765, 2019.

[62] S. Bouabdallah, P. Murrieri, and R. Siegwart, “Design and control of an indoor micro quadrotor,” in IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA ’04. 2004, vol. 5. IEEE, 2004, pp. 4393–4398.

GILBERTO DÍAZ-GARCÍA (S’15) is a PhD student in Electrical and Computer Engineering at Universidad de los Andes, Bogotá, Colombia. His current research interests include design and analysis of interconnected dynamical systems and game theory in control engineering.

DANIEL GUATIBONZA is a student in Electrical and Computer Engineering at Universidad de los Andes, Bogotá, Colombia. His current research interests include robotics and machine learning.

LUIS FELIPE GIRALDO (S’08, M’16) received the Ph.D. degree in electrical and computer engineering from The Ohio State University, Columbus, OH, USA, in 2016. He has been an Assistant Professor with the Universidad de los Andes, Bogotá, Colombia, since 2016. His current research interests include design and analysis of interconnected dynamical systems and machine learning.

VOLUME 4, 2016

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3119488, IEEE Access