Digital Chemical Engineering 6 (2023) 100065

Contents lists available at ScienceDirect

Digital Chemical Engineering

journal homepage: www.elsevier.com/locate/dche

Best practices and methods

Development of directed randomization for discussing a minimal security architecture

Henrique Oyama, Dominic Messina, Keshav Kasturi Rangan, Akkarakaran Francis Leonard, Kip Nieman, Helen Durand *, Katie Tyrrell, Katrina Hinzman, Michael Williamson

Department of Chemical Engineering and Materials Science, Wayne State University, Detroit, MI 48202, United States

ARTICLE INFO

Keywords:
Control system cybersecurity
Active attack detection
Smart manufacturing
Model predictive control
Image-based control

ABSTRACT

Strategies for mitigating the impacts of cyberattacks on control systems using a control-oriented perspective have become of greater interest in recent years. Our group has contributed to this trend by developing several methods for detecting cyberattacks on process sensors, actuators, or both sensors and actuators simultaneously using an advanced optimization-based control strategy known as Lyapunov-based economic model predictive control (LEMPC). However, each technique comes with benefits and limitations, both with respect to one another and with respect to traditional information technology and computer science-type approaches to cybersecurity. An important question to ask, therefore, is what the goal should be of the development of new control-based techniques for handling cyberattacks on control systems, and how we will be able to benchmark these as "successful" compared to other techniques to drive development or signal when the research in this direction has reached maturity. In this paper, we propose that the goal of research in control system cybersecurity for next-generation manufacturing should be the development of a security architecture that provides flexibility and safety with lowest cost, and seek to clarify this concept by re-analyzing some of the security techniques from our prior work in such a context. We also show how new methods can be developed and analyzed within this "minimum security architecture" context by proposing a technique which we term "directed randomization" that may require less sensors to be secured in a system than some of our prior methods, potentially adding flexibility to the system while still maintaining security. Directed randomization seeks to utilize the existence of two possible stabilizing inputs at every sampling time to attempt to create a challenge for an attacker for setting up an arbitrary sensor attack policy without being detected within a finite number of sampling periods. We discuss benefits and limitations of this technique with respect to our prior cybersecurity strategies and also with respect to extended versions of these prior concepts, such as image-based control and distributed control, to provide further insights into the minimum security concept.

1. Introduction

Smart/next-generation manufacturing, which can lead to an increase in automation, enhanced safety, and greater operational efficiency, has received increasing attention in recent years (Davis et al., 2012). A challenge for next-generation manufacturing, due to its focus on computation, communication, and data, is cybersecurity (Ren et al., 2017). An aspect of next-generation manufacturing systems for which cyberattack-resilience is critical is industrial control systems, as control systems are cyberphysical systems for which tampering with any aspect of a control loop could lead to misbehavior of a physical process, potentially resulting in profit losses, equipment degradation, or physical harm to individuals.

Due to its criticality, cybersecurity of control systems has been an active research area (Bhamare et al., 2020; McLaughlin et al., 2016), with research covering topics such as state estimation and control for linear systems in the presence of attacks (Fawzi et al., 2014), using optimization to predict attack behavior (Vamvoudakis et al., 2013), revealing attacks by adding private signals to actuator outputs (Ko et al., 2016), state estimation from corrupted measurements (Hu et al., 2017; Liu et al., 2016), and output feedback control for discrete-time stochastic nonlinear systems with security maintained in probability (Ding et al., 2016). With a stronger relationship to chemical processes, control system cybersecurity was investigated for a simplified Tennessee Eastman

* Corresponding author.
E-mail address: helen.durand@wayne.edu (H. Durand).

https://doi.org/10.1016/j.dche.2022.100065
Received 27 June 2022; Received in revised form 1 November 2022; Accepted 6 November 2022
Available online 23 November 2022

process in Cárdenas et al. (2011) or for distillation in Budiawan et al. (2018). Cybersecurity has also been considered with respect to process design concerns, such as during a hazard and risk analysis (Cormier and Ng, 2020) or in its relationship to equipment (Nieman et al., 2020). Cyberattacks are different from faults in that cyberattacks may involve coordinated effort to mask problems in the system through taking over multiple components and falsifying data in a consistent way across the board (Rangan et al., 2022). This means that methods which seek to locate process faults through data (e.g., Bhadriraju et al., 2021a; Bhadriraju et al., 2021b; Hassanpour et al., 2020; Kumari et al., 2022; Qin and Yin, 2022; Yin et al., 2022) may be insufficient, in many cases, for locating complex cyberattacks without further investigation into how it can be ensured that the data received by the monitoring algorithms is correct.

Model predictive control (MPC) (Qin and Badgwell, 2003) is an attractive control law for use in next-generation manufacturing due to its ability to select control actions for a process which optimize an objective function subject to constraints, and it therefore is an important control law to explore from a cyberattack-resilience perspective. A variant of MPC known as economic model predictive control (EMPC) (Ellis et al., 2014a; Rawlings et al., 2012) utilizes a potentially economics-based objective function in the controller optimization problem, which makes this control law interesting for next-generation manufacturing applications. The question of how to secure processes operated under EMPC has received attention in the context of Lyapunov-based economic model predictive control (LEMPC) (Heidarinejad et al., 2012), a formulation of EMPC with strong closed-loop stability and feasibility properties in the presence of sufficiently small bounded disturbances and measurement noise. For example, machine learning detection strategies (Wu et al., 2018) have been combined with LEMPC implemented in both centralized (Chen et al., 2020) and distributed (Chen et al., 2021) fashions for maintaining closed-loop stability during normal process operation, with the possibility of maintaining closed-loop stability after an attack if the attack is detected by the machine learning-based detection strategy in time (Wu et al., 2020). In our recent work, we have analyzed cybersecurity for control systems from a nonlinear systems perspective (Durand, 2018), which led to the development of detection strategies for handling sensor measurement cyberattacks with safety guarantees, where the detection strategies are derived from control-theoretic considerations for LEMPC both when process dynamics are constant (Oyama and Durand, 2020) and when they are changing over time (Oyama et al., 2021; Rangan et al., 2021). We have also explored handling cyberattacks on actuators or on sensors and actuators simultaneously (Oyama et al., 2022b).

Our prior works have not attempted to characterize what the goal of control-theoretic cybersecurity research should be. For example, from these prior works, one might infer that the goal is to add on new detection policies to existing attempts to provide security to control systems. Though there may be value in such an approach when an existing security architecture is deemed unsuitable for preventing attacks, we view such a goal as limiting, suggesting that the goal of cybersecurity research should only be to add more cost to systems for enhanced security. Instead, our vision is that cybersecurity research can be used to help indicate the extent to which manufacturing could be made open if the right types of new techniques for blocking attacks were developed at the intersection of the computer science approaches and dynamic systems/control theory. It is our premise that control-theoretic cybersecurity research should be probing the limits of agility that processes can obtain, and how lean the security protocols can be without compromising important operating goals such as safety, rather than stopping at adding layers of security to existing protocols.

Our goal in this work is to provide initial steps toward articulating such a vision (which we refer to as the search for a "minimal security architecture") as a foundation for future works to seek to further refine the concept. We proceed to propose this notion in several steps. First, we discuss several aspects of cybersecurity in the traditional information technology/computer science framework, and then begin to discuss our prior control-theoretic cybersecurity techniques (using the naming convention introduced in Oyama et al., 2022b) from the vantage point of seeking to understand how they may or may not lead to greater flexibility in security setups when compared with the traditional cybersecurity notions. We next use the limitations of these prior methods to motivate the development of a framework for coupled control and attack detection termed "directed randomization" that, inspired by cryptography and active attack detection concepts (e.g., Ghaderi et al., 2020; Satchidanandan and Kumar, 2016; Weerakkody et al., 2017), utilizes random selection between two potential stabilizing inputs at every sampling time to attempt to make it more difficult for an attacker to come up with a problematic sensor attack policy that could not be flagged by the detection strategy. We discuss benefits and limitations of this strategy with respect to our prior strategies, and then also deepen the discussion of how to understand, define, and search for minimum cybersecurity architectures by extending the discussion of our cybersecurity techniques beyond the traditional centralized control framework into the domains of image-based control and distributed control.

2. Preliminaries

2.1. Notation

The Euclidean norm of a vector x is denoted by $|\cdot|$, and the transpose of x is denoted by x^T. A class $\mathscr{K}$ function $\alpha : [0, a) \to [0, \infty)$ is strictly increasing with $\alpha(0) = 0$. The transpose of a vector x is denoted by x^T. Set subtraction is signified by " / " such that $x \in A/B := \{x \in R^n : x \in A, x \notin B\}$. A level set of a positive definite function V is denoted by $\Omega_\rho := \{x \in R^n : V(x) \leq \rho\}$. R_+ signifies the set of non-negative real numbers. Process state measurements are assumed to be available synchronously and each is separated by a sampling period of length Δ (i.e., a state measurement is available at every $t_k := k\Delta$, where $k = 0, 1, \ldots$). diag(x) represents a diagonal matrix with the elements of x on the diagonal.

2.2. Class of systems

The class of nonlinear systems considered is as follows:

$$\dot{x}(t) = f(x(t), u(t), w(t)) \tag{1}$$

where $x \in X \subset R^n$, $u \in U \subset R^m$, and $w \in W \subset R^z$ are the state, input, and disturbance vectors, respectively, f is locally Lipschitz on $X \times U \times W$, and $W := \{w \in R^z : |w| \leq \theta_w, \theta_w > 0\}$.

It is assumed that there exists a sufficiently smooth Lyapunov function $V : R^n \to R_+$, functions $\alpha_j(\cdot), j = 1, \ldots, 4$, of class $\mathscr{K}$, and a controller $h(x) = [\overline{h}_1(x) \ \ldots \ \overline{h}_i(x)]^T$ that is capable of asymptotically stabilizing the closed-loop system to the origin of Eq. (1) in the absence of disturbances such that the following inequalities are satisfied:

$$\alpha_1(|x|) \leq V(x) \leq \alpha_2(|x|) \tag{2a}$$

$$\frac{\partial V(x)}{\partial x} f(x, h(x), 0) \leq -\alpha_3(|x|) \tag{2b}$$

$$\left|\frac{\partial V(x)}{\partial x}\right| \leq \alpha_4(|x|) \tag{2c}$$

$$h(x) \in U \tag{2d}$$

$\forall x \in D \subset R^n$ and D is an open neighborhood of the origin. $\Omega_\rho \subset D$ is considered to be the stability region of the nominal closed-loop system under the controller $h(x)$ where $x \in X$, $\forall x \in \Omega_\rho$.

Furthermore, we consider that the components of $h(x)$ satisfy:

$$|\overline{h}_i(x) - \overline{h}_i(\widehat{x})| \leq L_h |x - \widehat{x}| \tag{3}$$

for all $x, \widehat{x} \in \Omega_\rho$, with $L_h > 0$, and $i = 1, \ldots, m$. The smoothness of V and

local Lipschitz property of f give:

$$|f(x_1,u_1,w)-f(x_2,u_2,0)| \leq L_x|x_1-x_2|+L_u|u_1-u_2|+L_w|w| \tag{4a}$$

$$\left|\frac{\partial V(x_1)}{\partial x} f(x_1,u,w)-\frac{\partial V(x_2)}{\partial x} f(x_2,u,0)\right| \leq L_x^{'}|x_1-x_2|+L_w^{'}|w| \tag{4b}$$

$$|f(x,u,w)| \leq M_f \tag{5}$$

$\forall x_1, x_2 \in \Omega_\rho$, $u, u_1, u_2 \in U$ and $w \in W$, where $L_x, L_x^{'}, L_u, L_w, L_w^{'}$, and M_f are positive constants.

We also assume that there are M sets of measurements $y_i \in R^{q_i}$, $i = 1, \ldots, M$, available at t_k as follows:

$$y_i(t) = k_i(x(t)) + v_i(t) \tag{6}$$

where k_i is a vector-valued function, and v_i represents the measurement noise associated with the measurements y_i. We assume that the measurement noise is bounded (i.e., $v_i \in V_i := \{v_i \in R^{q_i} \mid |v_i| \leq \theta_{v,i},\ \theta_{v,i} > 0\}$) and that measurements of each y_i are continuously available. For each of the M sets of measurements, we assume that there exists a deterministic observer (e.g., a high-gain observer Ahrens and Khalil, 2009) described by the dynamic equation:

$$\dot{z}_i = F_i(\epsilon_i, z_i, y_i) \tag{7}$$

where z_i is a process state estimate from the ith observer, $i = 1,\ldots,M$, F_i is a vector-valued function, and $\epsilon_i > 0$. When a controller $h(z_i)$ with Eq. (7) is used to control the closed-loop system of Eq. (1), we consider that Assumptions 1 and 2 below hold.

Assumption 1. ((*Ellis et al., 2014b; Lao et al., 2015*)) There exist positive constants θ_w^*, $\theta_{v,i}^*$, such that for each pair $\{\theta_w, \theta_{v,i}\}$ with $\theta_w \leq \theta_w^*$, $\theta_{v,i} \leq \theta_{v,i}^*$, there exists $0 < \rho_{1,i} < \rho$, $e_{m0i} > 0$ and $\epsilon_{Li}^* > 0$, $\epsilon_{Ui}^* > 0$ such that if $x(0) \in \Omega_{\rho_{1,i}} := \{x \in D | V(x) \leq \rho_{1,i}\}$, $|z_i(0) - x(0)| \leq e_{m0i}$ and $\epsilon_i \in (\epsilon_{Li}^*, \epsilon_{Ui}^*)$, the trajectories of the closed-loop system are bounded in Ω_ρ, $\forall\ t \geq 0$.

Assumption 2. ((*Ellis et al., 2014b; Lao et al., 2015*)) There exists $e_{mi}^* > 0$ such that for each $e_{mi} \geq e_{mi}^*$, there exist $t_{bi}(\epsilon_i)$ such that $|z_i(t) - x(t)| \leq e_{mi}$, $\forall\ t \geq t_{bi}(\epsilon_i)$.

2.3. Lyapunov-based economic model predictive control (LEMPC)

In this work, we utilize an optimization-based control design known as LEMPC (Heidarinejad et al., 2012), which is formulated as follows:

$$\min_{u(t)\in S(\Delta)} \int_{t_k}^{t_{k+N}} L_e(\tilde{x}(\tau), u(\tau))\, d\tau \tag{8a}$$

$$\text{s.t.}\quad \dot{\tilde{x}}(t) = f(\tilde{x}(t), u(t), 0) \tag{8b}$$

$$\tilde{x}(t_k) = x(t_k) \tag{8c}$$

$$\tilde{x}(t) \in X,\ \forall\, t \in [t_k, t_{k+N}) \tag{8d}$$

$$u(t) \in U,\ \forall\, t \in [t_k, t_{k+N}) \tag{8e}$$

$$V(\tilde{x}(t)) \leq \rho_e,\quad \forall\, t \in [t_k, t_{k+N}),\quad \text{if } x(t_k) \in \Omega_{\rho_e} \tag{8f}$$

$$\frac{\partial V(x(t_k))}{\partial x} f(x(t_k), u(t_k), 0) \leq \frac{\partial V(x(t_k))}{\partial x} f(x(t_k), h(x(t_k)), 0),\ \text{if } x(t_k) \in \Omega_\rho \Big/ \Omega_{\rho_e} \tag{8g}$$

where $u(t) \in S(\Delta)$ signifies that the optimal solution is a piecewise-constant input vector. N is the length of the prediction horizon, where each sampling period has a duration of Δ. The objective function is the time-integral of the economic stage cost L_e of Eq. (8a), evaluated throughout the prediction horizon. The predictions $\tilde{x}(t)$ are obtained from the nominal model of Eq. (8b). The constraints of Eqs. (8d) and (8e) are state and input constraints, respectively. The two Lyapunov-based stability constraints are given by Eqs. (8f) and (8g) where $\Omega_{\rho_e} \subset \Omega_\rho$.

3. The concept of a minimal security architecture

As noted in Section 1, there are many existing methods today for cybersecurity that relate to control systems, and this raises the question of why additional methods, such as those developed in our prior works from a control-theoretic perspective, are needed, or what value they would add to industry. In the chemical process industries, it is typical to add a variety of safety features to a system as a result of, for example, a Hazard and Operability (HAZOP) analysis, to ensure that the risk of a harmful accident occurring is acceptably low. However, after it is acceptably low, there would not be a business case for continuing to add further layers of protection. For cybersecurity, the same would be expected to hold; there would be a point at which adding more detection strategies to the system would no longer have an adequate business case. One could therefore question whether an advanced control-theoretic technique, such as those derived in our prior works, for detecting and handling an attack would gain traction compared to enhancing more traditional information technology defenses or simpler strategies based on the process dynamics such as recording the difference between a state prediction and state measurement without involving control theory in an extensive fashion.

Our premise in this work is that the goal of the development of new cyberattack-handling policies for control systems, while considering control design and dynamics, is not only to provide "more security," but to search for new policies that are as open and flexible as possible with control theory and design opening new avenues that could not be considered when looking at the problem from a simplified process or purely computer science perspective. For example, one of our prior methods for handling sensor measurement attacks in Oyama and Durand (2020) guarantees that an undetected attack would not be able to compromise safety of a system, as long as at least one of the redundant observers used to reconstruct the system state is not impacted by an attack. If then the sensors corresponding to one of the observers were tightly secured, others may be able to be more flexibly added or removed, and no safety issues could occur without an attack being detected as long as the tightly secured sensors remained uncompromised. This "back-off" in the level of security required for all sensors is something that can be revealed by control theory specifically. This discussion indicates that an appreciation for the intersection of control and information technology can aid in better understanding the concept of a "minimal security architecture" that is as flexible, cost-effective, and secure as possible.

Prior to advancements in Internet connection and remote access, industrial control systems were more isolated Ani et al. (2017). Through the advancement of the Internet, and more specifically Industry 4.0 and the Internet of Things (IoT), industrial control systems are becoming a part of a wider telecommunications network where data can be shared between devices. Much critical infrastructure now depends on Internet connections and remote accessibility (Ani et al., 2017) because this connectivity can increase productivity and accessibility while creating a significant opportunity for cyberattacks via vulnerable networks and/or software/hardware, malicious actions on the part of employees, or accidental actions carried out by an employee that could cause unauthorized access to the network. Companies value the ability to increase connectivity and accessibility of their manufacturing systems, so that the concept of exploring how much flexibility might be introduced to operations through rigorous policies for handling cyberattacks at the control system level is warranted.

Risk assessment is vital to industrial control systems to effectively create a security infrastructure tailored to the specific control system (Francia et al., 2012). The information system should be documented

thoroughly with details of every connection port noted to create a well-developed security protocol. The security controls should be implemented and then monitored during operation and then re-assessed after an allotted time period. The security protocol of an industrial control system may change with time (Ani et al., 2017; Francia et al., 2012), and companies should ensure that software and guidelines for employees for interacting with networked and computing systems are kept up-to-date. Updates should be done in a routine manner while adhering to production protocol and limiting down time. In general, having a robust cybersecurity policy can require a good deal of effort not only in setting up the system, but in maintaining it in the face of frequent changes. This is another case where having a type of "minimal security architecture" could aid in prioritizing updates and potentially reducing the ongoing effort for keeping up with changes in software, software patches, and attacker methods.

Today, there is a notion that a security infrastructure should be designed to promote confidentiality, integrity, and availability (Ani et al., 2017). For confidentiality, control system data should not be accessible to malicious actors to prevent them from gaining critical information that could be exploited for stealing proprietary information or potentially being used later in attacking the control system. The integrity of the system refers to the data being unaltered as it is transmitted between parts of the closed loop to maintain reliable operation of the control system. Lastly, availability needs to be maintained to avoid down time and loss of profits. Attacks could be performed which impact confidentiality, integrity, or availability, and attacks could result in safety issues, loss of life, or loss of physical property or production time if security is left unchecked. However, it is currently not known whether creative operating policies and/or process designs could be developed that make the need to rigorously achieve confidentiality and integrity less important for achieving availability.

There are two main systems in a process facility: the information technology (IT) system and the operational technology (OT) system. The IT system includes all hardware, software, and equipment needed for storing, retrieving, and transmitting data, such as servers, computers, data servers, and firewalls. The OT system provides the direct monitoring and control of industrial equipment, processes, and events, and includes processing equipment, pipes, sensors, and controllers (Iaiani et al., 2021). Supervisory Control and Data Acquisition (SCADA) systems are an example of a system for monitoring and control (Nicola et al., 2018) that utilizes security strategies such as firewalls for protecting processes and devices connected to the network of the plant. Patel and Zaveri (2010) addresses the impact of cyberattacks on SCADA systems and has assessed cyberattacks based on their impact on the profits made or costs incurred by the plant. Some of the more traditional techniques for seeking to protect physical systems are to use a viable network topology and firewall, and to provide a personnel protocol with limited access points, as well as intrusion detection software. The network topology (Smith, 2016) at a plant should keep parts of the networked system independent of each other. Establishing a thorough understanding of the system and its connectivity can reduce vulnerabilities. Limiting the personnel access to only certain parts of the system can reduce the chance of unintended consequences. Firewalls can be software or hardware, and they analyze network traffic based on a set of rules and are configured to accept only certain connections, IP addresses, or sources to prevent malicious data from traveling past the firewall (Stewart, 2013; Cybersecurity & Infrastructure Security Agency, 2019). There are different types of firewalls (e.g., packet filtering firewalls (Trabelsi et al., 2018) and proxy firewalls (Aziz et al., 2012)) that evaluate incoming data packets in different ways. These information technology means for attempting to protect networked control systems are not exhaustive, but provide an indication of the many different decisions at an IT level which lead to technologies that then must be maintained and reviewed with changes over time, reducing the flexibility for rapid adoption of new advances.

4. Initial analyses of control-theoretic cybersecurity policies within a minimal security architecture

To provide clarity on the notions in the prior section, this section seeks to provide initial thoughts on how strategies for cyberattack detection from our prior work (Oyama and Durand, 2020) might fit within the concept of the "minimal security architecture." Specifically, in Oyama and Durand (2020), three policies for integrating control and sensor measurement cyberattack detection are developed that motivate much of the work in the remainder of this manuscript. These have been reviewed in Oyama et al. (2022b), so our review here is somewhat brief, and we refer readers to those prior works for greater detail. The first strategy (to be referred to as Detection Strategy 1-S) seeks to perform active detection by modifying the LEMPC control law of Eq. (8) to remove the constraint of Eq. (8f) and design the LEMPC around an auxiliary steady-state at random times $t_{s,i}$ to force the closed-loop state toward the auxiliary steady-state contained within Ω_ρ. The LEMPC is switched back to the form of Eq. (8) a sampling period later (at $t_{e,i}$). When the state measurement $\widetilde{x}(t_k)$ is within an appropriate subset of the steady-state of the auxiliary steady-state, the Lyapunov function associated with the auxiliary steady-state (termed the ith steady-state with its ith Lyapunov function V_i and associated parameters, functions, and controllers of the form in Section 2.2 but with a subscript i) should decrease between the beginning and end of the next sampling period. If it does not (or if the state measurement leaves Ω_{ρ_1} at any time in the subsequent sampling period), an attack is flagged. We denote the LEMPC designed around the original steady-state as the 1-LEMPC ($i = 1$), and that for the ith auxiliary steady-state as the i-LEMPC, $i > 1$. The theoretical result for this detection strategy is presented below.

Proposition 1. ((*Ellis et al., 2014b; Lao et al., 2015*)) *Consider the systems below*

$$\dot{x}_i = f_i(x_i(t), u_i(t), w(t)) \tag{9a}$$

$$\dot{\widetilde{x}}_i = f_i(\widetilde{x}_i(t), u_i(t), 0) \tag{9b}$$

where $|x_i(t_0) - \widetilde{x}_i(t_0)| \leq \delta$ *where* $t_0 = 0$. *If* $x_i(t), \widetilde{x}_i(t) \in \Omega_{\rho_i}$ *for* $t \in [0, T]$, *then there exists a function* $f_{w,i}(\cdot,\cdot)$ *such that:*

$$|x_i(t) - \widetilde{x}_i(t)| \leq f_{w,i}(\delta, t - t_0) \tag{10}$$

for all $x_i(t), \widetilde{x}_i(t) \in \Omega_{\rho_i}$, $u_i \in U_i$, *and* $w \in W$, *with*

$$f_{w,i}(s,\tau) := \left(s + \frac{L_{w,i}\theta_w}{L_{x,i}}\right) e^{L_{x,i}\tau} - \frac{L_{w,i}\theta_w}{L_{x,i}} \tag{11}$$

Proposition 2. ((*Ellis et al., 2014b*)) *Let* $V_i(\cdot)$ *represent the Lyapunov function of the nominal system of Eq. (1), in deviation form from the ith steady-state, under the controller* $h_i(\cdot)$ *that satisfies Eqs. (2a)–(2d) and Eq. (3) for the system of Eq. (1) when it is in deviation variable form from the ith steady-state. Then there exists a function* f_{V_i} *such that:*

$$V_i(\bar{x}) \leq V_i(\bar{x}') + f_{V_i}(|\bar{x} - \bar{x}'|) \tag{12}$$

$\forall \bar{x}, \bar{x}' \in \Omega_{\rho_i}$ *where* $f_{V_i}(\cdot)$ *is given by:*

$$f_{V_i}(s) := \alpha_{4,i}\left(\alpha_{1,i}^{-1}(\rho_i)\right)s + M_{V_i}s^2 \tag{13}$$

where M_{V_i} *is a positive constant.*

Theorem 1. ((*Oyama, Durand, 2020*)) *Consider the closed-loop system of Eq. (1) under the implementation strategy described above, where the steady-state input for the ith steady-state is within the input bounds and* $\widetilde{x}_i(t_k) \in \Omega_{\rho_{h,i}}/\Omega_{\rho_{s,i}}$, *in the absence of a false sensor measurement cyberattack where each controller* $h_i(\cdot)$, $i \geq 1$, *used in each i-LEMPC meets the inequalities in Eqs. (2a)–(2d) and (3) with respect to the ith dynamic model.*

Let $\epsilon_{W_i} > 0$, $\Delta > 0$, $N \geq 1$, $\Omega_{\rho_i} \subset \Omega_{\rho_{samp2,1}} \subset \Omega_{\rho_1} \subset X_1$ *for* $i > 1$, $\rho_i > \rho_{h,i} > \rho_{\min,i} > \rho_{s,i} > \rho'_{s,i} > 0$, *where* $\Omega_{\rho_{h,i}}$ *is defined as the smallest level set of* Ω_{ρ_i} *that guarantees that if* $V_i(\widetilde{x}_i(t_k)) \leq \rho_{h,i}$, $V_i(x_i(t_k)) \leq \rho_i$, *and* $\rho_1 > \rho_{samp2,1} > \rho_{samp,1} > \rho_{e,1} > \rho_{\min,1} > \rho_{s,1} > \rho'_{s,1} > 0$ *(where* $\Omega_{\rho_{samp,1}}$ *is defined as a level set of* Ω_{ρ_1} *that guarantees that if* $x_1(t_k) \in \Omega_{\rho_1}/\Omega_{\rho_{samp,1}}$, *then* $\widetilde{x}_1(t_k) \in \Omega_{\rho_1}/\Omega_{\rho_{e,1}}$*) satisfy:*

$$-\alpha_{3,i}\left(\alpha_{2,i}^{-1}\left(\rho'_{s,i}\right)\right) + L'_{x,i}M_{f,i}\Delta \leq -\epsilon_{w,i}/\Delta,\ i = 1, 2, \ldots, \tag{14}$$

$$\rho_{e,1} + f_{V,1}\left(f_{W,1}(\delta, \Delta)\right) \leq \rho_{samp2,1} \tag{15}$$

$$-\alpha_{3,1}\left(\alpha_{2,1}^{-1}\left(\rho_{e,1}\right)\right) + L'_{x,1}M_{f,1}\Delta + L'_{x,1}\delta + L'_{w,1}\theta_w \leq -\epsilon'_{w,1}/\Delta \tag{16}$$

$$-\alpha_{3,i}\left(\alpha_{2,i}^{-1}\left(\rho_{s,i}\right)\right) + L'_{x,i}M_{f,i}\Delta + L'_{x,i}\delta + L'_{w,i}\theta_w \leq -\epsilon'_{w,i}/\Delta,\ i = 1, 2, 3, \ldots, \tag{17}$$

$$\rho_{\min,i} = \max\left\{V_i(x_i(t)) : x_i(t_k) \in \Omega_{\rho'_{s,i}},\ t \in [t_k, t_{k+1}),\ u_i \in U_i\right\},\ i = 1, 2, \ldots, \tag{18}$$

$$\rho_{samp2,1} \geq \max\left\{V_1(x_1(t)) : x_1(t_k) \in \Omega_{\rho_{samp,1}} / \Omega_{\rho_{e,1}},\ t \in [t_k, t_{k+1}),\ u_1 \in U_1\right\} \tag{19}$$

$$\rho_1 \geq \max\left\{V_1(\widetilde{x}_1(t_k)) : x_1(t_k) \in \Omega_{\rho_{samp2,1}}\right\} \tag{20}$$

$$\rho_i = \max\left\{V_i(x_i(t_k)) : \widetilde{x}_i(t_k) \in \Omega_{\rho_{h,i}}\right\},\ i = 2, 3, \ldots, \tag{21}$$

$$\rho'_{s,i} < \min\left\{V_i(x_i(t_k)) : \widetilde{x}_i(t_k)\ \in \Omega_{\rho_i} / \Omega_{\rho_{s,i}}\right\},\ i = 1, 2, \ldots, \tag{22}$$

$$\begin{aligned}\epsilon'_{w,i}\ > \max_{\widetilde{x}_i(t_k) \in \Omega_{\rho_{h,i}}/\Omega_{\rho_{s,i}}} \Big|&\min\left\{V_i(\widetilde{x}_i(t_k)) : \widetilde{x}_i(t_k) \in \Omega_{\rho_{h,i}}/\Omega_{\rho_{s,i}}\right\} \\ &-\max\left\{V_i(\widetilde{x}_i(t_{k+1})) : \widetilde{x}_i(t_k) \in \Omega_{\rho_{h,i}}/\Omega_{\rho_{s,i}},\ u_i \in U_i,\ \left|x_i(t_p) - \widetilde{x}_i(t_p)\right| \leq \theta_{v,1},\ p = k, k+1\right\}\Big|\end{aligned} \tag{23}$$

If $\widetilde{x}_1(t_0) \in \Omega_{\rho_{samp2,1}}$, $x_1(t_0) \in \Omega_{\rho_{samp2,1}}$, *and* $|\widetilde{x}_i(t_k) - x_i(t_k)| \leq \delta$, $k = 0, 1\ldots$, *then the closed-loop state is maintained in* $\Omega_{\rho_{samp2,1}}$ *and the state measurement is in* Ω_{ρ_1} *when the 1-LEMPC is activated at* t_0 *and for* $t_{e,i-1} \leq t < t_{s,i}$ *or when the i-LEMPC is activated for* $t_{s,i} \leq t < t_{e,i}$ *under the implementation strategy described above, and the closed-loop state and the state measurement are maintained within* Ω_{ρ_1} *for* $t \geq 0$. *Furthermore, in the sampling period after* $t_{s,i}$, *if* $\widetilde{x}_i(t_k) \in \Omega_{\rho_i}/\Omega_{\rho_{s,i}}$, V_i *decreases and* $x_i(t) \in \Omega_{\rho_i}$ *for* $t \in [t_k, t_{k+1})$. This detection strategy does not guarantee stability and feasibility at every instance of process operation as the detection of cyberattacks is active only over randomly selected sampling periods and it is possible for fake sensor measurements to indicate a decrease in the process states without matching the actual trajectory followed by the process resulting in the cyberattack detection strategy being ineffective against such cyberattacks. In terms of a minimal security architecture, this strategy would likely not be a very economical choice for handling sensor attacks, both because it disrupts process operation when it probes for attacks compared to what might be done if this strategy was not used, but still does not provide a guarantee of attack detection so that there is not necessarily much gained by adding this detection policy. There are no guarantees regarding optimality of profits compared to a case where no probing occurs or where attacks occur.

The second sensor attack detection strategy from Oyama and Durand (2020) (Detection Strategy 2-S) involves controlling the process using only the 1-LEMPC but checking at every sampling time whether a state prediction for t_k based on a measurement made at time t_{k-1} (denoted by $x(t_k|t_{k-1})$) is "close" (in a norm sense) to the measurement $x(t_k)$ at t_k (where closeness is expressed using ν as a threshold on the prediction error). The implementation strategy is:

1. If $|x(t_k|t_{k-1}) - x(t_k)| > \nu$ at a sampling time, flag that a cyberattack is happening and apply mitigating action. Else, go to Step 1a.
   (a) Operate the process under the 1-LEMPC. $t_k \leftarrow t_{k+1}$. Go to Step 1.

The theoretical result for this strategy is presented below.

Theorem 2. ((Oyama and Durand, 2020)) *Consider the system of Eq. (1) in closed-loop under the implementation strategy described above based on a controller* $h_1(\cdot)$ *that satisfies the assumptions of Eqs. (2a)–(2d) and (3). Let the conditions of Theorem 1 hold with* $t_{s,j} = \infty$, $j = 2, 3, \ldots$, *and* $\delta \geq f_{W,1}(\theta_{v,1}, \Delta) + \nu$. *If* $\widetilde{x}_1(t_0) \in \Omega_{\rho_{samp2,1}} \subset \Omega_{\rho_1}$ *and* $x_1(t_0) \in \Omega_{\rho_{samp2,1}}$, *then* $x_1(t) \in \Omega_{\rho_{samp2,1}}$ *and the state measurement at each sampling time is in* Ω_{ρ_1} *for all times before a sampling time* t_A *that a cyberattack falsifies a state measurement, and* $x_1(t) \in \Omega_{\rho_{samp2,1}}$ *for* $t \in [t_A, t_A + \Delta)$, *if the attack is not detected at* t_A.

This detection strategy guarantees stability and feasibility for at least a sampling period after an undetected attack. Because this strategy adds cost but cannot guarantee detection of an attack, one might consider that it is not desirable from a minimal security architecture perspective unless another policy can be found that is able to provide detection guarantees when this procedure fails.

The final strategy (Detection Strategy 3-S) from (Oyama and Durand, 2020) uses an LEMPC of the form of Eq. (8) but with the state measurement replaced by a state estimate z_1. M observers are constructed that all estimate the state of the system, and these cross-check one another by comparing $|z_r(t_k) - z_l(t_k)|$ with an upper bound $\epsilon_{\max} := \max\{e_r^* + e_l^*\}$, $r = 1, \ldots, M$, $l = 1, \ldots, M$. An assumption of this method is that no more than $M - 1$ estimators can be impacted by the attack, and that it occurs after all of the estimates have converged (i.e., after t_{bp}, $p = 1, \ldots, M$). The implementation strategy is as follows:

1. If $|z_i(t_k) - z_j(t_k)| > \epsilon_{\max}$, $i = 1, \ldots, M$, $j = 1, \ldots, M$, at any sampling time, or $z_1(t_k) \notin \Omega_\rho$, flag that a cyberattack is occurring and perform mitigating actions. Else, go to Step 1a.
   (a) Operate using the LEMPC with $\widetilde{x}(t_k) = z_1(t_k)$. $t_k \leftarrow t_{k+1}$. Go to Step 1.

Defining $\epsilon_M^* := \epsilon_{\max} + \max\{e_{mj}^*\}$, $j = 1, \ldots, M$, the following theorem describes the theoretical results for this method.

Theorem 3. *Consider the system of Eq. (1) in closed-loop under the output feedback LEMPC of Eq. (8) based on an observer and controller pair satisfying Assumptions 1 and 2 and formulated with respect to the* $i = 1$ *measurement vector, and formulated with respect to a controller* $h(\cdot)$ *that meets Eqs. (2a)–(2d) and (3). Let at least one state estimator be non-impacted by an attack, and* $\theta_w \leq \theta_w^*$, $\theta_{v,i} \leq \theta_{v,i}^*$, $\epsilon_i \in (\epsilon_{Li}^*, \epsilon_{Ui}^*)$, *and* $|z_i(t_0) - x(t_0)| \leq e_{m0i}$, *for* $i = 1, \ldots, M$. *Also, let* $\epsilon_{W,1} > 0$, $\Delta > 0$, $\Omega_{\rho_1} \subset X$, *and* $\rho_1 > \rho_{\max} > \rho_{1,1} > \rho_{e,1} > \rho_{\min,1} > \rho_{s,1} > 0$, *satisfy:*

$$\rho_{e,1} \leq \rho_{\max} - \max\{f_V(f_W(\epsilon_M^*, \Delta)), M_f \max\{t_{z1}, \Delta\}\alpha_4(\alpha_1^{-1}(\rho_{\max}))\} \quad (24)$$

$$\rho_{e,1} \leq \rho_1 - f_V(f_W(\epsilon_M^*, \Delta)) - f_V(\epsilon_M^*) \quad (25)$$

$$-\alpha_3(\alpha_2^{-1}(\rho_{s,1})) + L_x'(M_f\Delta + \epsilon_M^*) + L_w'\theta_w \leq -\epsilon_{W,1}/\Delta \quad (26)$$

$$\rho_{\min,1} = \max\{V(x(t)) | V(x(t_k)) \leq \rho_{s,1},\ t \in [t_k, t_{k+1}),\ u \in U\} \quad (27)$$

$$\rho_{\min,1} + f_V(f_W(\epsilon_M^*, \Delta)) \leq \rho_1 \quad (28)$$

$$\rho_{\max} + f_V(\epsilon_M^*) \leq \rho_1 \quad (29)$$

where t_{z1} is the first sampling time after t_{b1}, and f_V and f_W are defined as in Propositions 1 and 2 for $i = 1$ but with the subscripts dropped. Then, if $x(t_0) \in \Omega_{\rho_{e,1}}$, $x(t) \in \Omega_{\rho_{\max}}$ for all $t \geq 0$ and $z_1(t_h) \in \Omega_{\rho_1}$ for $t_h \geq \max\{\Delta, t_{z1}\}$ until a cyberattack is detected according to the implementation strategy above, if the attack occurs after t_q. It may seem that Detection Strategy 3-S, which is able to guarantee safety in the presence of attacks as long as some of the sensors are not attacked, suggests a minimal security architecture could be one in which only the sensors corresponding to a state estimator that requires the least sensors of all of the redundant estimators are secured (i.e., the others may be less secure; this assumes that others are able to be found). However, one could then ask why the others are required at all, if one set is fully secured (i.e., if one estimate is always correct and it is known which estimate that is, it would be preferable to use that estimator all the time without the added cost of the redundancy). In this strategy then, the minimum number of sensors that could be used to reconstruct a state would be limited by fundamental control-theoretic properties such as observability.

Though this detection strategy guarantees stability and feasibility at every instance of process operation in the presence of undetected cyberattacks on process sensors by keeping state measurements and estimates within a bound of each other, it still does not guarantee that profits are close to what would be obtained without an attack. This is because the undetected attacks could be present for a long time period, causing the state to continue to differ over time from the trajectory that it otherwise would have taken, and thereby causing profits to deviate from what they otherwise would have been. This also depends on the disturbances and measurement noise.

The concept of a minimal security architecture brings up the question of how one might evaluate the cost tradeoffs between different technologies in the future. Different cybersecurity products will cost different amounts for different size organizations and from different sellers with different features. In chemical engineering, it is typical to perform cost analyses for production using cost analysis procedures based on applying different factors to estimated base costs. However, a difference between analysis for cybersecurity costs and chemical process costs has traditionally been that whereas the process must be built (making a cost analysis for chemical process equipment "mandatory"), cost analyses for cybersecurity may require some sense of the tradeoffs between paying extra for cybersecurity and the potential that nothing may happen if no extra is paid. Gordon et al. (2020), for example, presents a framework for attempting to quantify benefits from cybersecurity that account for both losses from attacks as well as the difference between cybersecurity costs and the benefits from protecting against the loss. We might envision that after developing a number of possible frameworks for control system cybersecurity, these types of analyses might be used to compare different ideas and clarify which control and cybersecurity combinations are least costly. A model study might be chosen in such a case (e.g., a distillation column as was selected for a cybersecurity study in Ahmad, 2020), and then the system can be run under the different control policies both with and without attacks to analyze how the strategies affect profits during normal operation and to what extent they prevent catastrophes during operation that would be costly. An analysis of how easy it is to make changes to the system in all cases can also be considered, as can the difficulty of setting up the security systems (e.g., if a control-theoretic policy is used, this may require expert knowledge from a control engineer that may not be required with more standard information technology-based approaches). This can help to make a comparison between different strategies for securing a system from many angles, which may aid with developing a final policy.

Remark 1. Though it was noted above that redundant observers may not be needed if some sensors are secured and are sufficient for providing state estimates to a controller, they may have utility for seeking to entice attackers to attack systems for the purpose of better understanding their attack methods and motives (a "honeypot").

5. Minimum security architecture for inspiring new cyberattack detection methods: developing directed randomization

In this section, we seek to demonstrate the utility of the minimum security architecture for guiding the development of new cyberattack detection strategies through the example of a strategy which we term "directed randomization." This policy is motivated by the fact that even the strongest cyberattack detection guarantees reviewed above (i.e., under Detection Strategy 3-S, attacks can be detected or otherwise not cause safety issues) require assumptions such as that at least one of the state estimators cannot be impacted by an attack. Within the minimum security architecture notion, this begs the question of whether requiring some sensor measurements to be secured would impose restrictions on flexibility of some systems, and if there is any way to detect cyberattacks even if all sensors might be compromised. To probe this question, we develop an active attack detection scheme that takes a hint from cryptography (which is aligned with the thought process in Teixeira et al., 2012) and is similar in spirit to active detection methods such as Ko et al. (2019) (which is based on private inputs from actuators that are used to detect that the sensor measurements do not match what should be occurring in the absence of an attack). The goal of directed randomization is to provide a means to make it unlikely that past state measurements could be incorrect by the way that the control actions for the process behave (assuming the attacker follows a specific attack model). This section is organized as follows: first, we clarify what a technique that attempts to flag incorrect state measurements should achieve. Then, we provide an intuitive concept for attempting to achieve these goals, and show that this method does not have all required properties. This inspires the development of the directed randomization strategy, which utilizes randomness in a targeted fashion to attempt to thwart sensor cyberattacks occurring according to a specific policy.

5.1. Handling full sensor attacks: how the strategy should work

The first question in attempting to develop a strategy that may not need any sensors to be secured is to ask what kinds of properties such a controller might have. A strategy is needed for attempting to force attackers to reveal themselves even if they were to gain hold of all sensors. We would like to add some type of unexpected action to the control input, where the attacker could not guess what we would do any better than a random guess (which is reminiscent of a one-time pad (Darup and Jager, 2019)). Because the process should subsequently behave according to its dynamics, we would then like to use a process model to predict changes in a state measurement, which would presumably follow a unique pattern based on the changes to the inputs, but this could be complicated by sensor noise and plant/model mismatch. If a cyberattack occurs, the proper pattern would no longer be detected in the state measurements and an attacker could not know the pattern because it is a one-time pad (similar to strategies such as dynamic watermarking Satchidanandan and Kumar, 2016). What is needed to detect an attack in a finite time is for the trajectory under the attack to be distinguishable with a high probability from the trajectory under the expected pattern in finite time, so that it would be highly improbable for

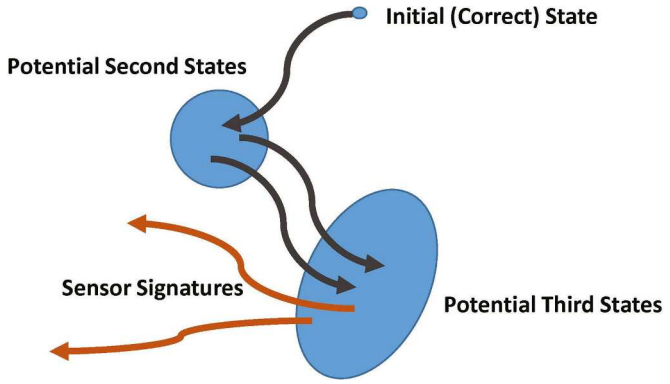

Fig. 1. Concept of regions in which the state measurement could not be verified to be falsified. The size of the regions is dictated by the magnitude of the measurement noise and plant/model mismatch. The initial condition is the potential state for the process at time t_0, just as each point in the second set of states is a potential state for the process at time t_1, and the points in the third set of states are potential states for the process at time t_2. If we do not receive a measurement at t_1, we expect that the region corresponding to the third set of states is larger than that for the second set of states due to the measurement noise and disturbances broadening the set of possible states at the end of a sampling period based on allowable states at the beginning of the sampling period. However, if we receive a measurement at t_1, this can help to prevent as much broadening of the possible states at the next sampling time.

someone to falsify the state measurements. However, we would also like to guarantee closed-loop stability while doing this.

Our prior work has looked at strategies incorporating randomness (e.g., randomly selecting steady-states in Detection Strategy 2-S or randomly selecting control laws in Durand, 2018); however, neither could always guarantee detection of an attack or that it could not cause safety issues before it was detected. This indicates that randomness on its own is not sufficient to prevent attacks from being successful; this indicates that a rigorous investigation of how this strategy can be designed to ensure that attacks even on all sensors and all actuators at once are found before they cause safety issues is important. It is also important to recognize the challenge of dealing with noise and disturbances. For example, consider Fig. 1. If there is a state measurement impacted by noise only (i.e., no attack), then the state is in a ball around the measurement according to the upper bound on the noise. Under an input, in the presence of plant/model mismatch and measurement noise, there will be a set of all possible states that could be reached and measured from the original set of possible states. If an attacker was to present a state measurement outside of the expected set, it would be known to be incorrect. However, how to ensure that there is a bounded time before they either show themselves or reveal they are not there is an important question.

Sample-and-hold also causes issues, since an attacker might be able to back-solve for the input if they knew the state measurements. Specifically, if the attacker receives the actual measurements over a time period when the input is kept constant, they may be able to determine which input might have been applied to the system and thereby to determine which states should be in the measurement trajectory. This could help them to provide state measurements that appear correct despite being false. To avoid this, the proposed strategy has to affect the state without giving the attacker enough time to react (i.e., an operator or engineer should be able to tell that an attack was performed before the attacker is able to provide state measurements that appear correct). In addition, the control actions applied while probing for attacks should guarantee closed-loop stability, ensure there are no time periods when attacks are not being probed for, avoid significant impacts on profits, and make it highly unlikely for an attacker to have falsified data after it has been validated (and highly likely that the attacker did falsify the data (or that a fault is occurring) if it is not validated).

It is necessary to assess whether there are any circumstances under which these types of manipulations can be performed. To do this, consider the extreme case in which there are no disturbances and sensor measurements are perfect. In this case, the dynamics of the same system under two different inputs could not be exactly the same, so that measurements should reveal when an input applied is not exactly what is expected. As sensor noise and disturbances (plant/model mismatch) are added, we would expect there to be inputs that are close enough to one another where the allowable measured states at the end of the sampling period could match (and that the set of inputs causing this overlap becomes larger as the bound on the noise and disturbances increases, so that the extent of noise and disturbances is limited by whether it is desired to make certain inputs cause distinguishable trajectories).

To move from the above discussion of desired properties of a detection procedure to providing details of that procedure, we will use a process example to showcase why a rough intuitive strategy for attempting to meet these goals is insufficient, and that more care is needed in defining the controller. Then, we will provide an idea for a detection strategy that meets the goals above when the attacker has a specific strategy.

Remark 2. The concept that we need to ensure that the attacker does not receive measurements while an input is held constant that would enable them to tell what input was applied could be thought of with (loose) cryptographic wording as considering the process dynamics as a public key and the input as a private key. The concept that it is easier to compute the outcome of the decryption by an entity with both keys, but hard without both keys (creating a one-way function) is a loose description of what is sought to be achieved. Specifically, the process dynamics evolve faster under the unknown input than an attacker would be able to falsify without knowing the input, so that the measurement will show this effect (i.e., the dynamics under the input are "decrypted" quickly through measurement) whereas the attacker would be lucky to provide the correct state measurement (there is a nonzero chance of this, but it is more challenging).

5.1.1. Chemical process example: attempting to detect sensor attacks with signature signals

In this section, a rough attempt to meet the goals of the prior section is developed in a process example in which a bias is placed in the control action applied to the system to cause the sensor measurements to be different than what would be applied if the bias does not occur. In this case, even if an attack is attempting to be stealthy in the sense that it falsifies state measurements so that they look like the process is operating normally, the attack may be able to be noticed by a failure of the sensor data to follow the trajectory that it would take under the bias. The choice of the bias can depend on the dynamics of the system, and would need to cause the differences between many attacked and non-attacked conditions to be readily revealed.

The concept is illustrated with a CSTR controlled by a Lyapunov-Based Economic Model Predictive Controller (LEMPC) (Heidarinejad et al., 2012), with the dynamics and parameters from Alanqar et al. (2015). This CSTR has two states (concentration C_A of reactant in the reactor and temperature T of fluid in the reactor) and two inputs (a heat rate Q being supplied to the reactor and a feed concentration of reactant C_{A0}). The attack takes place as follows:

1. For $t \in [0, 0.33)$ h: No attack is performed.
2. For $t \in [0.33, 0.5)$ h: A bias is applied to the heat rate to increase its value by 5×10^4 kJ/h, when the state measurements are not falsified. This causes the shift in the input to be observed in the state measurements.
3. $t \geq 0.5$ h: An attack occurs in which the sensor measurements appear as if they were coming from control of the process using LEMPC, with no bias applied.

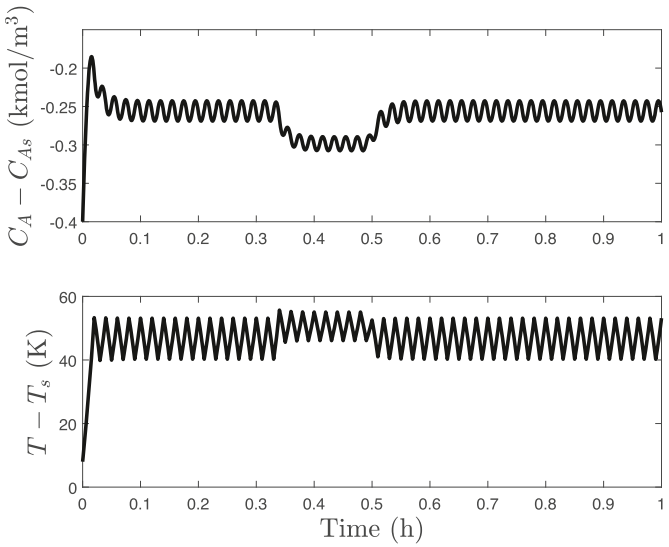

Fig. 2. Process state trajectories under the detection policy attempt of Section 5.1.1.

Fig. 2 indicates the response of the closed-loop system to the operating policy just outlined. The disadvantage of this *ad hoc* approach is that if an attacker learned the bias that would be applied to the inputs, they could easily modify their sensor falsification strategy to include an effect that appeared to suggest the bias was added. This indicates that adding a bias alone is not enough to showcase an attack; the manner in which the bias is applied is critical. The next section will use this conclusion to develop a modified concept.

Remark 3. The idea in this example was to introduce a signal on top of the state of the system that has a human-readable impact on the output signal for aiding in the diagnosis of the attacks. Though this bias does not appear to have affected the ability of the closed-loop state to be maintained in a defined region of state-space before the attack, that is not necessarily guaranteed with a general bias.

5.1.2. *Directed randomization protocol*

As noted in the prior section, though a bias can be added to a control action to attempt to show that this bias is present in the sensor measurements, if the attacker knows this bias, they would be able to fake its presence. We could ask whether we could modify the idea such that, inspired by Detection Strategy 1-S (where probing for attacks occurs at random times), we add the bias to the control action at random times to attempt to catch the attacker unaware. However, it would somehow need to be assessed whether closed-loop stability (in the sense of boundedness of the closed-loop state within an expected region of state-space) can still be maintained at all times when the bias is randomly applied. To handle these various considerations, we consider a strategy that we term "directed randomization." To describe this strategy, we will discontinue referring to "biases" in the control action and instead consider that the full control action (which might be comprised of a baseline value plus a bias) constitutes the control action.

In directed randomization, every possible state measurement is associated *a priori* with two possible control actions (to avoid the number of potential sensor measurements and control actions being infinite, we will consider that the sensing device has a limited resolution, so that there are a finite, though potentially large, number of possible state measurements). These control actions will be selected to achieve certain goals with respect to what we will call "reachable sets," which we take to mean sets of all potential state measurements that could be obtained by the end of a sampling period if: 1) the actual initial state is in a region around the initial state measurement that is consistent with the sensor noise bound and 2) the state measurement at the beginning of the next sampling period must be consistent with the sensor noise bound and the bound on the plant/model mismatch. Specifically, the reachable set after a sampling period contains all possible final states after one sampling period that might have been able to be measured given that the measurement at the beginning of the sampling period and at the end are both subject to the sensor noise, and there are many possible state trajectories between two sampling periods that start at a given initial condition due to the many different realizations of plant/model mismatch that could occur (similar to the notion between the initial state and second set of potential states in Fig. 1). The goals that the inputs must achieve with respect to the reachable sets are: Input Requirement 1) they must ensure that the reachable sets are not overlapping as in Fig. 3 and Input Requirement 2) they must ensure that the two potential reachable sets (and the state as it evolves with time to end up in the reachable sets) never leave a bounded and safe region of state-space Ω_ρ when the process is operated without attacks. In the remainder, we will assume that it is possible to find such inputs, and will not investigate the conditions under which this is possible for given system dynamics. The need for Input Requirement 1 comes from the desire to integrate the control action selection policy with sensor measurement attack detection. Specifically, at every sampling time, someone who knows the two control actions that might have been applied could provide a state measurement consistent with one of these two control actions, but not with both at once. Input Requirement 2 is required to ensure that this strategy does not lead to loss of closed-loop stability under non-attacked operation.

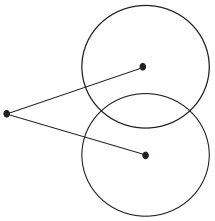

Fig. 3. Overlapping reachable sets.

Input Requirement 1 leads to the design of an integrated detection and control policy for sensor attacks in which one of the two allowable control actions is randomly selected at every sampling time. The selection can occur based on a string of random binary digits (inspired by a one-time pad), where identical copies of this string are available at both the detection device and the actuators, and a 0 represents one of the control actions while a 1 represents the other. Because of Input Requirement 1, the inputs that might be applied at t_k will cause the state measurement to be expected to end up in one of two disjoint sets $R_{1,k+1}$ or $R_{2,k+1}$ at t_{k+1} if there is no attack. If there is an attack, an attacker who only receives state measurements at every sampling time and does not have access to the input signal may know what $R_{1,k+1}$ or $R_{2,k+1}$ are; however, they would not know which of these two was expected by the detection policy at a given t_{k+1}. This makes it "harder" for them to evade detection.

At t_{k+1}, an attacker might provide some state measurement outside of either $R_{1,k+1}$ or $R_{2,k+1}$ (in which case they would be flagged), or they might provide one within either $R_{1,k+1}$ or $R_{2,k+1}$. We might consider that it "makes sense" that if the attacker knew the two inputs which might be applied, they might provide one in either $R_{1,k+1}$ or $R_{2,k+1}$. However, as there was a 50/50 chance of either of these two inputs being selected, they essentially have a 50/50 chance of guessing which of the two regions they think they should select to provide a state measurement within. This means that at t_{k+1}, one of two outcomes is achieved: 1) the attacker provides the false state measurement in the wrong region with a 50% probability or 2) the attacker provides the false state measurement in the right region with a 50% probability. In the first case, a detection policy that checks if they provided the state measurement in the correct region would flag them. However, this detection policy would not catch

them in the second case. Because they have a 50% probability of guessing the "correct" region to provide a false state measurement within at t_{k+1}, there is a large chance that the detection policy "misses them." While this may provide an improvement in plant security compared to not performing a check at all, it may also fail to detect attacks a significant portion of the time.

We can improve on this strategy, however, by recognizing that since the random number string is dictating the selected control action, and each number in the string is independent from the others, the analysis just performed holds at the next sampling time also. Specifically, consider that there was an attack on the state measurement at t_{k+1}, but that it was not flagged because it was in the correct region $R_{c,k}$ (which is either $R_{1,k+1}$ or $R_{2,k+1}$, depending on whether the random number selected at t_k was 0 or 1). At t_{k+1}, there is no basis for believing the state measurement to be false (and indeed, since it is in the right region, the state measurement without the falsification would have also been within $R_{c,k}$ as well, which bounds how far "off" the false value can be from the actual value without the attack being caught after one sampling period). At t_{k+1}, because the state measurement has not been flagged as false, it is "trusted" for providing the control input to apply to the process (i.e., if we call the falsified state measurement $x_f(t_{k+1})$, then the set of two control inputs which might be applied is the set of two inputs corresponding to a measurement of $x_f(t_{k+1})$). Now, $R_{1,k+2}$ and $R_{2,k+2}$ are computed by assuming that the measurement at t_{k+1} is "correct" (within the bound on the measurement noise). At t_{k+2}, the state measurement will be expected by the detector to be in one of these two regions. Just as at t_{k+1}, the attacker has a 50/50 chance of guessing which of the two inputs was applied and therefore which of the two regions $R_{1,k+2}$ and $R_{2,k+2}$ is the region that they should place the next state measurement within to avoid detection. In this case, $R_{1,k+2}$ and $R_{2,k+2}$ are computed based on the expected value of the actual state in the region around $x_f(t_{k+1})$ based on the bound on the measurement noise, and based on the possible disturbance profiles between t_{k+1} and t_{k+2} as well as potential measurement noise at t_{k+2}. Though the predictions are tied to $x_f(t_{k+1})$, the two control actions that are known to be potentially applied at t_{k+1} are selected independently from $x_f(t_{k+1})$ (and rather according to the string of random numbers). This means that at t_{k+2}, the attacker is again faced with a 50% probability of selecting the expected $R_{c,k+2}$ for providing a new false state measurement.

Though between t_{k+1} and t_{k+2}, the attacker may have a 50% chance of evading detection, their likelihood of evading detection twice (from t_k to t_{k+2}) is lower. Specifically, as noted above, the probability of the attacker selecting $R_{c,k+1}$ given that the state measurement at t_k was not flagged as an attack is 50%, and the probability of the attacker selecting $R_{c,k+2}$ given that the state measurement at t_{k+1} was not flagged as an attack is 50%. Therefore, the probability that the attacker succeeds at evading detection twice in a row is 25%. We can imagine that if the attack is not flagged at t_{k+2} (i.e., the attacker guessed the correct region a second time out of pure luck), then at the next time, the likelihood that they guess $R_{c,k+3}$ correctly is again 50% so that the likelihood that they evaded detection three times in a row is 12.5%. We could continue like this to obtain lower and lower probabilities of getting the correct result multiple times in a row, which is suggested by Fig. 3, in which the first node (node 0) represents the state at t_0, at which time we consider that the state measurement is accurate. In a case with no disturbances and measurement noise, the two possible end states branching from node 0 could be achieved from this initial node. If there is no attack, the state measurement should read the value at node 1 at sampling time t_1. If there is an attack at t_1, the state will be at one of the two locations that branch off from node 0 if it is stealthy, or it will be at some other point entirely if it is not stealthy. If the state measurement is not at node 1, an attack will be flagged. The likelihood of the attacker giving a false state measurement multiple times in a row (i.e., from t_1 on) are noted near each node in the figure. The two potential end states from every node are also shown.

This logic forms the basis of the detection aspect of directed randomization. Specifically, at the end of every sampling period, we consider that we have a 50% chance that the last state measurement was falsified if the state measurement at the end of the sampling period was in the expected region (as calculated based on the last state measurement). However, we have a 25% chance that both the state measurement before that and the last state measurement were both falsified. If we look back $\overline{n}$ sampling times, if no attack has been flagged, we have a $\frac{100}{2^{\overline{n}}}$ percentage chance that all of the last $\overline{n}$ state measurements were falsified. Therefore, one can specify a likelihood ϵ with which some number of the last state measurements have been falsified, and this will fix $\overline{n} = \frac{-1}{\ln 2}\ln\frac{\epsilon}{100}$. The detector therefore must check whether each falsified state measurement is in $R_{c,p}$, $p = k-\overline{n}+1, \cdots, k$ (considering t_k to be the most recent time at which a state measurement was taken), and if all are in the correct regions, it states that it is unlikely that all of the last $\overline{n}$ state measurements have been attacked. If we assume that the attacker would provide false state measurements at every sampling time after they initiated an attack, we could think of this as a statement that it is unlikely that $t_{k-\overline{n}}$ was falsified if all of the subsequent measurements were not flagged as attacked measurements. For example, in Fig. 4 at the fourth node, there is a 12.5% chance that all measurements from t_1 to t_3 have been falsified if the measurement at t_0 was not flagged as an attack, but a 25% chance that only t_2 and t_3 were falsified if the measurement at t_1 was not flagged, and a 50% chance that the measurement at t_3 was falsified if that at t_2 was not flagged. If the 12.5% likelihood of falsification is deemed acceptably low, for example, then we can consider that the first scenario is sufficiently unlikely so that it is not the case (i.e., it would be considered unlikely that the attacker provided a false state measurement at every time after t_0 and was never caught if the measurement at t_0 was not flagged as an attack). This suggests that the measurement at t_0 was likely correct, or else the attacker would likely have been caught at some time since that point (since it would have been hard for them to "fake it" for so long). Therefore, the detector will apply a moving window strategy where at t_k, if no attack has been flagged since $t_{k-\overline{n}+1}$, we consider that the measurement at $t_{k-\overline{n}}$ was likely correct and stop including it in the set of measurements for which we are uncertain if they were falsified or not. Colloquially, we can think of this as that the detector uses the outcomes of the last $\overline{n}$ predictions of $R_{c,p}$ to back-validate the measurement at $t_{k-\overline{n}}$. The idea is that the detector determines that it is unlikely that someone "faked" the measurements for so long (i.e., since $t_{k-\overline{n}+1}$), meaning that at some point in the past the measurement must have been correct, and that time is likely the farthest point back in time in the window being considered (i.e., $t_{k-\overline{n}}$). In other words, since an attacker probably could not have gotten the guess of the input that was applied (and thus $R_{c,p}$) right $\overline{n}$ times in a row, and since no attack has been flagged yet (meaning that if there had been an attack, then the attacker must have guessed the input correctly $\overline{n}$ times in a row or else the attack would already have been flagged), there probably was not an attacker providing false state measurements to the system and guessing the input correctly $\overline{n}$ times in a row. The detector's policy assumes that 1) an attacker may have started an attack in which they were trying to guess $R_{c,p}$ at every sampling time and 2) if they did start such an

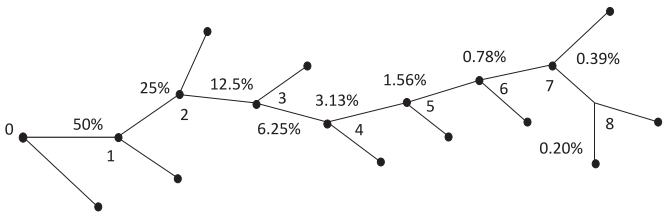

Fig. 4. Concept of random selection between two possible inputs at every sampling time, where each would provide a distinct state measurement at the end of the sampling period. The likelihood that the attacker provided a false state measurement every time since t_1 is noted near each node.

attack, they would continue to employ this attack at every sampling time thereafter. Under these assumptions, if it is unlikely that an attacker provided false state measurements $\overline{n}$ times in a row, then $t_{k-\overline{n}}$ probably did not come from an attack (or else the attacker would have provided false state measurements $\overline{n}$ times in a row since the detector's policy is based on the assumption that if the attacker started this type of attack in which they seek to guess $R_{c,p}$ at every sampling time, they did not stop it since then).

The implementation strategy for directed randomization as described above is as follows:

1. Generate a random string of 1's and 0's to indicate which of two control actions will be selected at every sampling time. Share this string only between the actuator and the attack detection system. Go to Step 2.
2. At t_k, obtain the state measurement $\widetilde{x}(t_k)$. Select one of the two control actions corresponding to $\widetilde{x}(t_k)$, assuming that the state measurement $\widetilde{x}(t_k)$ is correct within the bound on the measurement noise. The detector uses this input to compute $R_{c,k+1}$. Go to Step 3.
3. Control the process using the control action selected at a Step 3. Go to Step 4.
4. At t_{k+1}, the detector checks whether the state measurement is within $R_{c,k+1}$ and if it is within Ω_ρ. If not, flag an attack. If yes, consider the measurement at $t_{k-\overline{n}+1}$ is validated. Go to Step 5.
5. $t_k \leftarrow t_{k+1}$. Go to Step 2.

Because directed randomization seeks to make it more challenging for an attacker to evade detection for times longer than $\overline{n}$ sampling periods, one might consider that it is attempting to detect attacks in a finite time and thereby to put a bound on the profit that could be lost in the time interval of $\overline{n}$ sampling periods. Below we place a bound on the profit lost in $\overline{n}$ sampling periods (which requires the assumption that the attacker continues the attack for $\overline{n}$ sampling periods in a row and is expected to be detected before $\overline{n}$ sampling periods pass).

Proposition 3. ((*Durand and Wegener, 2020*)) *Consider the systems below*

$$\dot{x}_1 = f(x_1(t), u_1(t), w(t)) \tag{30a}$$

$$\dot{x}_2 = f(x_2(t), u_2(t), w(t)) \tag{30b}$$

with initial states $x_1(t_0) = x_2(t_0), u_1$ *developed based on* $x_1(t_0)$ *and* u_2 *developed based on a measurement* $\widetilde{x}_2(t_0)$ *such that* $|x_1(t_0) - \widetilde{x}_2(t_0)| \leq \delta$ *with* $t_0 = 0$. *If* $x_1(t), \quad x_2(t) \in \Omega_\rho, \quad |f(x_1, u_1, w) - f(x_1, u_2, w)| \leq L_u|u_1 - u_2|, |u_1(t) - u_2(t)| \leq B(\delta), \; B(\delta) > 0$ *for* $t \in [0, T]$, *then there exists a function* $f_W(\cdot, \cdot)$ *such that:*

$$|x_1(t) - x_2(t)| \leq f_W(\delta, t - t_0) \tag{31}$$

for all $x_1(t), x_2(t) \in \Omega_\rho$, $u_1, u_2 \in U$, *and* $w \in W$, *with*

$$f_W(s, \tau) := \left(\frac{L_u B(s)}{L_x}\right)\left(e^{L_x \tau} - 1\right) \tag{32}$$

Proposition 4. *Consider the following systems:*

$$\dot{x}_a(t) = f(x_a(t), u_a(t), w(t)) \tag{33}$$

$$\dot{x}_b(t) = f(x_b(t), u_b(t), w(t)) \tag{34}$$

where $x_a(t_0) = x_b(t_0)$, $x_a(t)$ *represents the state trajectory when the sample-and-hold input trajectory* $u_a(t)$ *applied to the process is one that would have been applied with no attack on the sensors, and* $x_b(t)$ *represents the state trajectory when the sample-and-hold input trajectory* $u_b(t)$ *applied to the process is one that is applied with a sensor measurement attack. Assume that* $|u_a(t) - u_b(t)| \leq C, C > 0, \forall \; t \in [t_0, t_0 + \overline{n})$ *Then, there exist* $M > 0$, $\epsilon_1 > 0$, *and* $\epsilon_2 > 0$ *such that over* $\overline{n}$ *sampling periods:*

$$\int_{t_0}^{t_0+\overline{n}} [l_e(x_a(\tau), u_a(\tau)) - l_e(x_b(\tau), u_b(\tau))] d\tau \leq (\epsilon_1 + \epsilon_2)\overline{n}\Delta \tag{35}$$

Proof. Denote $x_a(t)$ as the closed-loop state under input policy $u_1(t)$ (associated with the state following a trajectory determined in the absence of an attack), and $x_b(t)$ as the closed-loop state under an alternative input policy $u_2(t)$ (associated with the state following an input trajectory determined in the presence of an attack). Because l_e is a continuous function of the closed-loop state when the input is fixed, there exist $\epsilon_1 > 0$, $\delta_1 > 0$ such that $|l_e(x_a(t), u_a(t_0)) - l_e(x_b(t), u_a(t_0))| \leq \epsilon_1$ whenever $|x_a(t) - x_b(t)| \leq \delta_1$ and $|l_e(x_b(t), u_a(t_0)) - l_e(x_b(t), u_b(t_0))| \leq \epsilon_2$, whenever $|u_a(t) - u_b(t)| \leq \delta_2$. Over $\overline{n}$ sampling periods, there is an upper bound on $|x_a(t) - x_b(t)|$ that can be obtained from Proposition 3 over the sampling period from t_0 to t_1 and extensions that can be made over sampling periods after the first. Over $\overline{n}$ sampling periods, this pattern will repeat to give an upper bound on $|x_a(t) - x_b(t)|$ that can be considered to be δ_1. There is also an upper bound on $|u_a(t) - u_b(t)|$ in the assumption of the theorem that can be considered to be δ_2 Then:

$$\begin{aligned} |l_e(x_a(t), u_a(t_0)) - l_e(x_b(t), u_b(t_0))| &\leq |l_e(x_a(t), u_a(t_0)) - l_e(x_b(t), u_a(t_0)) + l_e(x_b(t), u_a(t_0)) - l_e(x_b(t), u_b(t_0))| \\ &\leq |l_e(x_a(t), u_a(t_0)) - l_e(x_b(t), u_a(t_0))| + |l_e(x_b(t), u_a(t_0)) - l_e(x_b(t), u_b(t_0))| \\ &\leq \epsilon_1 + \epsilon_2 \end{aligned} \tag{36}$$

Taking the time integral of the difference between the stage costs $l_e(x_a(t), u_a(t_0))$ and $l_e(x_b(t), u_b(t_0))$ for $t \in [t_0, t_1)$ gives:

$$\int_{t_0}^{t_1} [l_e(x_a(\tau), u_a(t_0)) - l_e(x_b(\tau), u_b(t_0))] d\tau \leq (\epsilon_1 + \epsilon_2)\Delta \tag{37}$$

Applying this recursively indicates that the maximum profit loss over $\overline{n}$ sampling periods when $x_a(t_0) = x_b(t_0)$ and no attack occurs before t_0 is $\sum_{i=0}^{\overline{n}} (\epsilon_1 + \epsilon_2)\Delta = (\epsilon_1 + \epsilon_2)\overline{n}\Delta$.□

Remark 4. It is critical that $R_{1,k}$ and $R_{2,k}$ do not overlap because if they were to overlap, then a stealthy cyberattack can be formulated where an attacker can provide a state measurement within the overlap, and it would not be able to be distinguished which of the two potential inputs they thought was creating that state measurement. In this case, they would remain stealthy despite that they did not know the applied input.

Remark 5. The fact that an attacker who wants to provide a guess of the state within the correct expected region cannot do better than randomly guess between the two potential regions provides a potential to detect attacks, even if all sensors are compromised, in a finite time. A benefit of this strategy is that when setting up the potential control inputs for every state measurement *a priori*, time can be spent to analyze which two inputs would satisfy Input Requirements 1 and 2 but be most profitable for the system. This may help to reduce some of the profit loss from a probing strategy by enabling the probing strategy to be set up with economics in mind from the start and then incorporating random selection between two potentially economically viable alternatives at

each sampling time. However, further analysis would be needed to better understand impacts on profits.

Remark 6. Given that one of the assumptions of the detection strategy above is that the attacker continues to provide false state measurements after starting to do so, it is reasonable to ask whether this is a significant limitation (as the attacker may choose to stop providing false state measurements, for example, after a short time). To analyze this case, we note that at any sampling time after an attacker starts to fake the state measurements, they can either stop faking them, or continue to fake them. In the case that they stop faking them, the implementation strategy above will flag an attack if the closed-loop state measurement is outside of either $R_{c,k+1}$ or Ω_ρ. This means that before an attacker can stop faking the measurements, to avoid detection, they must ensure that the (non-attacked) closed-loop state measurement is within Ω_ρ (i.e., the measurement $x_m(t_{k+1})$ is within a safe operating region; Ω_ρ should be selected such that this implies that the actual closed-loop state $x_a(t_{k+1})$ is also in a safe operating region $\Omega_{\rho_{safe}}$ (i.e., $V(x_m(t_{k+1})) \leq \rho$ implies that when $|x_m(t_{k+1}) - x_a(t_{k+1})| \leq \theta_v$, where θ_v represents the measurement noise bound, then $V(x_a(t_{k+1})) \leq \rho_{safe}$)). In addition, since $R_{c,k+1}$ is computed from the last state measurement (which was fake), the only way for the attacker to avoid detection is if the set of possible (actual) state measurements at t_{k+1} (given the measurement noise around the actual state) intersects with $R_{c,k+1}$ that was computed based on the fake state measurement. This means that both some possible fake state ($x_f(t_{k+1})$) within $R_{c,k+1}$ and the actual state ($x_a(t_{k+1})$) cannot be far from one another at t_{k+1} (i.e., $|x_a(t_{k+1}) - x_f(t_{k+1})| \leq \theta_v$, for some potential fake states, where θ_v represents the radius of the ball around $x_a(t_{k+1})$ in which all potential non-attacked state measurements would fall given the bound on the measurement noise). This then also prevents the attacker from doing "too much" with the state if they would like to avoid being detected when they remove the attack before $\overline{n}$ sampling periods pass. The conclusion of this is that the attacker either needs to continue to provide false state measurements at every sampling time after they start doing so (in which case, if $\overline{n}$ is large enough, it becomes challenging for them to continue to do this without being detected), or they would need to ensure that if they stop applying false state measurements before $\overline{n}$ sampling periods pass, the non-attacked state measurement at the time they remove the attack is not too far off (according to the equations above) from what they would have had to propose in the event of an attack. This would suggest that if the attacker does not plan to continue to provide state measurement profiles after they start attacking, they lose flexibility in how much they can make the attacked process measurements deviate from the actual process measurements. This can work against them if they are trying to destabilize the process. For example, if the set of two control actions that are defined at every possible state measurement in the directed randomization strategy is selected such that when the state measurements are within the noise bound around the actual closed-loop state (i.e., $|x_m(t_{k+1}) - x_a(t_{k+1})| \leq \theta_v$), then the control actions will still be stabilizing for the actual process, this can help to make it more challenging for an attacker to effectively disrupt a process without detection. Specifically, if they can only provide state measurements that are "close" to the actual state (i.e., $|x_a(t_{k+1}) - x_f(t_{k+1})| \leq \theta_v$) before $\overline{n}$ sampling periods pass, then if the control actions would be stabilizing even for such imperfect measurements, the attacker is not able to destabilize the process when they stop attacking. We can also add the requirement that the safe operating region $\Omega_{\rho_{safe}}$ should be a sufficiently large superset of Ω_ρ such that if a non-attacked state measurement is obtained at t_0 and is in Ω_ρ (with the actual closed-loop state in a neighborhood of that measurement defined by $|x_m(t_{k+1}) - x_a(t_{k+1})| \leq \theta_v$), then the actual closed-loop state and state measurement cannot leave $\Omega_{\rho_{safe}}$ within $\overline{n}$ sampling periods. This helps to make it more challenging for an attacker to evade detection long enough to cause the actual closed-loop state to leave $\Omega_{\rho_{safe}}$. This is because it is unlikely that the attacker will evade detection for $\overline{n}$ sampling periods if they keep attacking (and the closed-loop state will still be in Ω_ρ for $\overline{n}$ sampling periods after the non-attacked state measurement was obtained), or if the attacker stops the attack, they must ensure that some of the false control actions they could have provided would not have been far from a state measurement within Ω_ρ (given $|x_a(t_{k+1}) - x_f(t_{k+1})| \leq \theta_v$), meaning that the actual closed-loop state cannot be far from that either ($|x_a(t_{k+1}) - x_f(t_{k+1})| \leq \theta_v$) and they would not have been able to drive the closed-loop state from an accurate state measurement within Ω_ρ out of $\Omega_{\rho_{safe}}$ if they attacked for less than $\overline{n}$ sampling periods. However, profits could still be impacted by strategies that switch between actual and falsified state measurements before $\overline{n}$ sampling periods pass.

Remark 7. In the prior remark, there was an assumption that the attacker is providing state measurements which seek to evade the detection policy by guessing state measurements in either $R_{1,k}$ or $R_{2,k}$ at sampling time t_k. In general, an attacker could attempt a different type of policy altogether; in this case, it is more difficult to make conclusions because the attacker's policy is not specified. However, it seems reasonable to expect that it would be unlikely for an attacker doing something that is not intended to be "stealthy," when inputs are being randomly selected at the actuator, to do much better than the "stealthy" attacker (i.e., intuitively, it seems that it would be difficult for an attacker to accidentally choose a strategy that has a motive behind it but matches a policy that the process is applying that has randomness in it).

Remark 8. Despite some of the conceptual benefits of this strategy, this method would be challenging to implement in practice without the development of a strategy for forming the set of allowable state measurement-control action pairs without needing to enumerate every one before the start of the operation of the process (this would scale poorly and could require significant memory unless a functional relationship was learned to store the data).

Remark 9. Above, we focused on two possible biases. This gives the stealthy attacker a 50/50 chance of getting the reachable set correct at every sampling time if they are operating according to the described attack model. It is possible to use more than two biases as long as the regions continue to not overlap. This could reduce the size of $\overline{n}$ for obtaining a low chance of the attacker guessing correctly multiple times in a row (e.g., if there are four biases, the likelihood of the attacker guessing correctly at a single sampling time is 25%, so that over $\overline{n}$ sampling periods, the likelihood that they would have guessed correctly every time reduces to $(0.25)^{\overline{n}}$; in general, if there are p biases, the likelihood is reduced after $\overline{n}$ sampling periods to $\left(\frac{1}{p}\right)^{\overline{n}}$). This provides a strategy for attempting to reduce the likelihood more quickly with potential to reduce profit by requiring more possibilities to be able to be selected at a given sampling time and potentially making the setup of the strategy more challenging as it requires more control actions to be selected at more sampling times.

Remark 10. We note that we only need to ensure that the two regions $R_{1,k}$ and $R_{2,k}$ do not overlap; it is not necessary that they do not overlap with prior regions (e.g., with $R_{1,k-1}$ or $R_{2,k-1}$). There also is no requirement about how different the control actions selected for two different sampling times need to be from one another, and we assume that the attacker can know all of the control action-state measurement relationships. The only information they do not know is which of the two control actions was actually selected for a given state measurement.

Remark 11. If, despite a lower probability of the attacker succeeding in evading detection for $\overline{n}$ sampling periods, the attacker does evade detection for $\overline{n}$ sampling periods, there are no stability guarantees.

Remark 12. It was stated above that this method is intended for aiding with detecting sensor attacks even if all sensors are compromised. However, to handle arbitrary numbers of compromised sensors up to all sensors being compromised, it must be carefully designed because the

control actions applied must create two potential and non-overlapping regions $R_{1,k+1}$ and $R_{2,k+1}$ where the next set of state measurements must lie to avoid detection. For the two regions to be non-overlapping requires that the two different inputs cause every state being measured to differ significantly from the values they would have taken under another control action after only a sampling period. This is a requirement on the process dynamics and sampling period length that could limit when this method can be used for a given process.

5.2. Directed randomization: discussing concepts through an image-based control example

One of the key challenges behind the directed randomization strategy discussed above is the question of how to design such a strategy practically, since it requires multiple inputs to be selected at different points in state-space. In this section, we use an example of level control for a tank to showcase one idea for selecting different inputs, which is by designing two proportional-integral (PI) controllers with different step changes in the set-point. This discussion is an extended version of work on image-based control simulations (Oyama et al., 2022a), including when the image-based sensors may be cyberattacked (Oyama et al., 2022c). After reviewing the work in Oyama et al. (2022a) and (Oyama et al., 2022c) with more details on the simulation setup, this section will provide additional discussion of the relationship of the control-theoretic approaches to cybersecurity discussed in Section 4 to image-based control and explore the idea for selecting multiple inputs for a given state measurement with two PI control laws.

In image-based control, the camera is the sensor, so that a false measurement would be an incorrect image. Adversarial image-based attacks have been explored in the context of several different fields, including medical imaging (Mahler et al., 2018), autonomous driving (Sun et al., 2020), and neural network-based image recognition (Chen et al., 2019). Image-based attacks can take the form of disruptions in image data between the sensor or camera and the site of image processing. Such disruptions can include alterations to real images (local forgery) or the insertion of entirely fabricated images (global forgery). Attacks can be categorized as either white box or black box, depending on the amount of information the attacker has about the involved image processing algorithms (Chen et al., 2019). White-box attacks assume sufficient knowledge of the process, whereas black-box attacks involve only minimal information about the process and sensors involved.

We first review a level control example employed in Oyama et al., 2022, 2022c in the context of image-based control. The process dynamics are given by:

$$\frac{dh}{dt} = \left(u - c\sqrt{h}\right) \Big/ A \tag{38}$$

where h is the level in the tank and the manipulated input u is the volumetric flow rate entering the tank. The flow rate exiting the tank is taken to be $c\sqrt{h}$ (where $c = 0.008333$ m$^{5/2}$/s is the outlet resistance coefficient). $A = 0.23$ m^2 denotes the cross-sectional area of the tank. The tank level can vary between 0 m and 0.5184 m, and the input can vary between $u_{\min} = 0$ m^3/s and $u_{\max} = 0.6$ m^3/s. The process is not subject to disturbances. The resulting process dynamics are simulated using the explicit Euler numerical integration method with an integration step of 10^{-3} s.

A proportional-integral (PI) controller was designed to drive the tank level to its set-point h_{sp} over the 7 s of operation using measurements obtained from a fixed camera. The controller had the following form:

$$\frac{d\epsilon}{dt} = h_{sp} - \widetilde{h}, \ \epsilon(0) = 0 \tag{39}$$

$$u = u_s + K_c\left(h_{sp} - \widetilde{h}\right) + K_c\epsilon/\tau_I \tag{40}$$

where u_s is the steady-state value of u that corresponds to the initial tank level (0.0026 m^3/s) and $\widetilde{h}$ corresponds to the level measurement from the camera. ϵ is the dynamic state of the PI controller. The PI tuning parameters were selected to be $K_c = 0.6$ and $\tau_I = 43.2$. For the image-based measurements, which are sent to the controller every sampling period $\Delta = 0.1$ s, a fixed camera was positioned facing one side of the tank. A render from the camera in Blender showing the initial level is presented on the left in Fig. 5 (the black part of the image represents the fluid in the tank, and the green part represents the environment; the tank is considered to be transparent). As can be seen in this figure, the camera was positioned in a manner that causes the level in the tank to not take a large fraction of the image for any height of the fluid in the tank. We would expect that if the camera was moved closer to the tank (providing less of a view of the green environment above the tank and greater focus on the black fluid in the tank) that the distance between two pixels would represent less actual distance at the plant, giving the camera sensor higher resolution in measuring the level.

The changes in the level in the tank in Blender were simulated by adjusting the location of the top edge of the tank. The tank was modeled as a vertical plane in Blender. Blender has several modes of operation for objects. For the tank, the two important modes are Object Mode (where the tank can be created and added to the scene) and Edit Mode (where changes in individual vertices can be performed). Movement of the tank level with time was performed in Blender Edit Mode. Initially, the bottom left edge of the tank was positioned at $(-1, 0, -1.57$ m), with the bottom right edge at $(1, 0, -1.57$ m) (the bottom center is then at $(0, 0, -1.57$ m). Because the tank level is at $h = 0.1$ m, the top center of the tank is at $(0, 0, -1.47$ m). This enables the upper edge of the tank in Blender to be adjusted in the Python programming interface in Edit Mode using

Table 1
Arguments for bpy.ops.transform.translate.

Argument	Value
value	(0, 0, Delta_level)
orient_type	'GLOBAL'
orient_matrix	((1, 0, 0), (0, 1, 0), (0, 0, 1))
orient_matrix_type	'GLOBAL'
constraint_axis	(False, False, True)
mirror	True
use_proportional_edit	False
proportional_edit_falloff	'SMOOTH'
proportional_size	1
use_proportional_connected	False
use_proportional_projected	False

(a) Original camera image of the level of a tank.

(b) Modified image of the level of a tank based on Fig. 9a.

Fig. 5. Renders of the level of the tank at $t = 0$ (left picture) and when $h = 0.36$ m (right picture) using Blender (Oyama et al., 2022a).

the command bpy.ops.transform.translate with the modification in the level set to the change in the level (from Eq. (38)) over the last sampling period. Further details of the command used are presented in Table 1.

The simulation was designed with the two different colors for the tank and background to set up an image processing model in which the level was determined from the images rendered with Blender by using changes in the RGB color of the pixels at the top of the tank to indicate where the transition occurs between the level and the background. Specifically, using Blender's Python programming interface, the Python Imaging Library (Pillow) (Clark, 2015) was imported into Blender. The filepath for the render was set, and the render was performed at the end of each sampling period after $t = 0$ using the command bpy.ops.render.render(write_still = True). This was used to create a.png image of the camera's view. The image generated was then loaded into Blender with the Pillow commands Image.open(filepath) and load to load the image pixels. Then, a counter was decremented from 1079 starting from the pixel in the bottom middle (the array element [960,1079] of the load command). This index was decremented by 1 each time that the RGB values for this pixel did not meet certain requirements that would have corresponded to the switch in color from black to green (signifying that the top of the tank was reached). Specifically, it was checked whether the red channel value was either 29 or 30, whether the green channel value was 65 or 66, and whether the blue channel value was 19 or 20. If the red, green, and blue channels all took one of these values, the index was incremented by 1 (to reflect that the final height of the tank was one index before this since this new index corresponded to the environment and no longer to the tank) and set to I_p. Factors that play a role in what values to expect for the red, green, and blue channels, besides the colors specified for the tank and environment, are the position of the light and the camera with respect to the object. The light in the scene was a point light (HSV values of 0, 0, 1) and a power of 1000 W, positioned at (4.0762, 1.0055, 5.9039) m in the global coordinates. The camera was positioned at (0.004103, − 6.1636, 0.49802) m in the global coordinates.

A linear relationship was used to relate I_p to the level in the tank. This was developed by fitting a line using the values of I_p corresponding to the bottom of the tank and the initial level at $t = 0$ (no attempt was made to ensure that the initial height of the tank or the bottom of the tank were exactly where a pixel would be located in an image). The developed conversion is given by Eq. (41) below:

$$\widetilde{h} = -3.8462 \times 10^{-3} \times I_p + 4.1577 \tag{41}$$

At every sampling time t_k, I_p is computed according to the procedure above, and the corresponding measured level of the tank is sent to the PI controller.

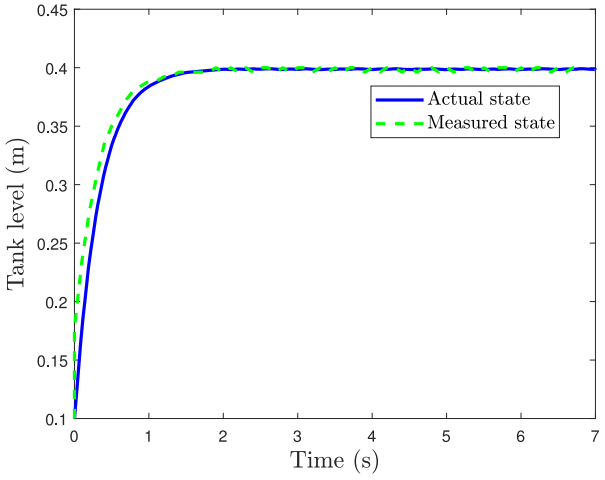

Fig. 6. Closed-loop response of the tank level under the IBC based on camera sensor (Oyama et al., 2022c).

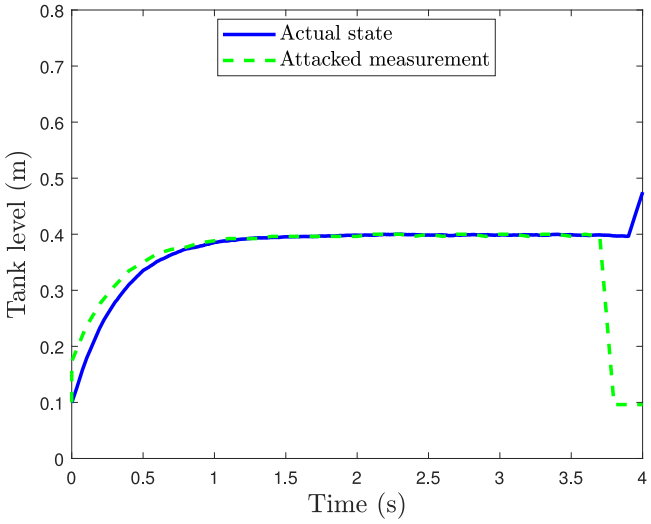

Fig. 7. Attack policy and closed-loop trajectory over time under the IBC system (Oyama et al., 2022c).

A set-point change to $h_{sp} = 0.4$ m occurred at $t = 0$, and the process was simulated for 7 s of operation, initialized at $x_{init} = x(t_0) = 0.1$ m in Blender 2.93. Another render of the tank level, which is closer to the set-point (Fig. 6), is shown in Fig. 5 on the right. The maximum difference between the actual and measured tank level obtained in the closed-loop simulation was 0.00272 m. The maximum difference between two pixels in the tank image is equivalent to 0.00384 m (anything between two pixels will give a value difference smaller than 0.00384 m). This is consistent with the result obtained from the closed-loop simulation.

In (Oyama et al., 2022c), we presented two attack cases that we will now review. In the first, we consider a case where the tank dynamics are not fully described by Eq. (38) (specifically, a random variable is added to the right-hand side of Eq. (38) with zero mean and a standard deviation of 0.08 m/s, and bound of 0.1 m/s). In this case, we will perform an attack corresponding to image replacement on the system after 4 s of operation. In this case, the tank image from $t = 0$ (left picture in Fig. 5) is developed before the process is run and stored to be pulled in after 4 s of operation using Pillow instead of the image that is the correct image at that time. Since the PI controller was driving the level toward 0.4 m from 0.1 m until that time, the swapping of the images creates state

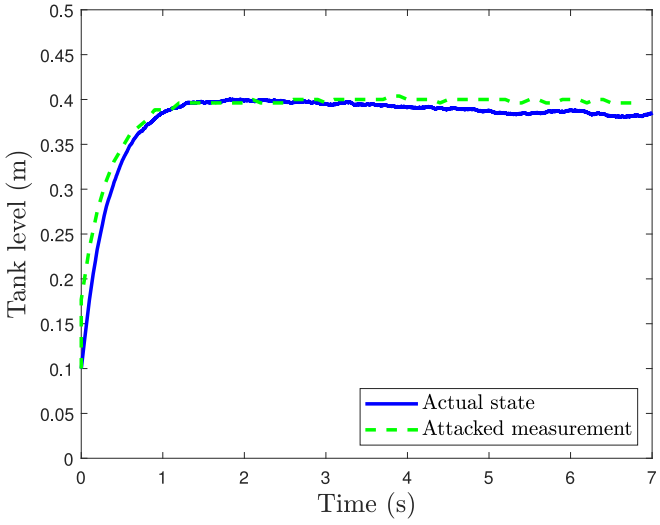

Fig. 8. Stealthy attack policy and closed-loop trajectory over time for the process under small disturbance (Oyama et al., 2022c).

measurements lower than the state measurement at 0.4 h. This results in control actions being computed that continuously increase the level and result in overflow of the tank (Fig. 7).

The second attack from Oyama et al. (2022c) that we now review is one in which the attack is "stealthy" (Oyama et al., 2021) such that the state measurements follow a plausible trajectory (specifically, the falsified images follow a plausible trajectory where they are obtained by generating fake images where the level appears to move according to Eq. (38), with bounded disturbances added to the right-hand side, but where the disturbances have different realizations compared to the actual process). In this case, the disturbance has zero mean and a standard deviation of 1 m/s, and bound of 0.1 m/s. The stealthy attack is initiated after 1 s of operation. Fig. 8 shows the stealthy attack trajectory and the actual state over 7 s of operation and indicates that the stealthy attack by the end of that time has caused some deviation between the actual state and the state measurements being received by the controller based on the false images.

In these examples, in the absence of an attack, an image is provided where there is a direct translation from the image to a state measurement, making these image-based controllers similar to more traditional controllers (with some noise in the measurements governed by the distance between pixels in the images). The methods in Oyama and Durand (2020) also consider that measurements of the state are available. This indicates that when the image provides a representation of the state, a straightforward extension of the results in Oyama and Durand (2020) can be made to understand how control-theoretic security guarantees can be made in the presence of image-based control. However, in practice, a key difference is that there are many factors which can prevent an image-based sensor from accurately representing the process state. One of these is that the image-based sensing algorithm must be set up to avoid unexpected errors in the image processing (e.g., not recognizing the top of the tank correctly due to failing to check which range of RGB values would correspond to the top of the tank in the presence of lighting and with the given camera angle). This is an example of where the use of a simulation platform such as Blender may help to reduce the number of unknown vulnerabilities in an image-based control system.

Another issue, however, is that in image-based control, other visuals that are not related to the state could obscure the measurement of the state. For example, if something was to fall in the tank or to stand in front of the camera, measurements of the state would no longer be obtained. One could argue that these might be then considered in a similar framework to cyberattacks in Oyama and Durand (2020), helping the results in Oyama and Durand (2020) to have a broader applicability as a means for probing and locating not only cyberattacks but also image abnormalities (equivalent to sensor faults in a traditional process). If one wanted, for example, to attempt to detect abnormal behavior before it began to impact the level measurements taken in the middle of the frame, one could consider expanding the monitoring to look at the top of the level over all of the pixels and then if it did not follow an expected trajectory (e.g., if it was not flat), to flag the image as abnormal, attempting to locate objects moving into the camera view before they obscure the measurements. One of the benefits of this small-scale image-based control example is that it allows some of the challenges of the strategies from Oyama and Durand (2020) from an economics standpoint to be assessed. For example, if LEMPC is used to control this system and Detection Strategy 1-S is considered for this system, then at times, the level in the tank would need to be varied. Because the level can only go in one of two directions, it would make the most sense to seek to move it, at a random time, in a direction opposite to the direction in which it is going at that time (e.g., if it is increasing, it could be decreased). It would need to be ensured that the tank could be emptied quickly enough so that a decrease or increase in level is observed between sampling periods, in the spirit of Eq. (23).

If Detection Strategy 2-S is used instead, then safety is not guaranteed. With the simple dynamics of the level, it would not be overly challenging for an attacker to provide reasonable state trajectories that are falsified, similar to the stealthy attack described above. Finally, for Detection Strategy 3-S, only one state is available, and therefore there are not other states from which to reconstruct the measurements. However, one benefit of images is that they can be used to provide additional checks in place of reconstructing measurements (for example, it could be checked whether all of the pixels at the top of the tank have the same coordinate, or whether there are any other black pixels outside the domain of the box). However, unlike the case where the state estimate is based on process dynamics (e.g., Assumptions 1 and 2), the checks using the pixels are based on logic and therefore attackers may find ways to undermine them if they are known.

In the above examples, it was assumed that images could be directly replaced and that the only way to attempt to tell this would be based on process dynamics as in Oyama and Durand (2020). However, another method for dealing with falsified images focuses on digital image forensics, specifically the analysis of sensor-based noise. This type of detection policy could also be analyzed in Blender. In (Farid, 2012), it is noted that images taken with a digital camera contain two types of noise: additive and multiplicative. This noise results from slight imperfections in the manufacture of camera sensors and as a result is consistent over time. For images taken with any given camera, the multiplicative noise is consistent enough to develop a noise profile, termed the photo-response non-uniformity noise (PRNU). An image taken with this camera will exhibit a correlation involving its noise and the PRNU (Farid, 2012).

To demonstrate that Blender with Pillow would be able to be used in manipulating images (which would be needed to attempt to add noise to an image after it is generated using Blender to represent a desired noisy camera profile for use in assessing how the noise profile might contribute to detection of attacks with Blender), consider the camera image of Fig. 9(a), which is an image of the level of a tank taken from a fixed camera positioned facing one side of the tank. The RGBA values at each image pixel could be modified. This is shown in Fig. 9(b), in which the RGBA values of Fig. 9(a) in the tank have been altered using the Pillow package in Python. In particular, a noise matrix, in which each element follows a normal distribution (mean 0 and standard deviation 10), with the same dimension as the tank image has been added to the matrix of the original tank image after each element in the noise matrix is rounded to the smallest integer greater than or equal to it and the "alpha" (A) value set to zero (Fig. 9(a)). For this operation, each element in the matrices is a data type numpy.uint8 (8-bit unsigned integer (0 to 255), which is used for matrices that represent images.

We close by discussing how we can transform the image-based control example described above to one which provides insights on how choosing different control laws for a given state measurement as part of the directed randomization policy might work. In particular, we select two PI controllers with different set-points. For these simulations, the process was initialized at $h(t_0) = 0.1$ m. Process disturbances were added to the right-hand side of Eq. (38) with zero mean and standard deviation of 1 m/s, and bound of 0.1 m/s. The simulation was performed over 10 s of operation in Blender 2.93 using its embedded Python interpreter. A proportional-integral (PI) controller was formulated to drive the tank level to its set-point h_{sp} within 5 s of a set-point change by selecting PI tuning parameters of $K_c = 0.6$ and $\tau_I = 43.2$. Two different versions of this PI controller were used in computing inputs that would result in different state trajectories. Specifically, the first PI controller (PI-1) used a set-point of $h = 0.23$ m, whereas the second PI controller (PI-2) used a set-point of $h = 0.27$ m at the initial time t_0. Both used a sampling period of $\Delta = 0.1$ s. At t_0, one of these two control laws is randomly selected to compute an input. In directed randomization, a new set of two control actions would be selected between at the next sampling time. However, in this example, we will not follow the directed randomization protocol rigorously, but instead focus on showcasing concepts behind the random selection between two different control laws at certain points in time. Therefore, in this example, the controller selected at t_0 is utilized for a number of sampling times until the closed-

(a) Original camera image of the level of a tank. (b) Modified image of the level of a tank based on Fig. 9a.

Fig. 9. Original and modified tank images.

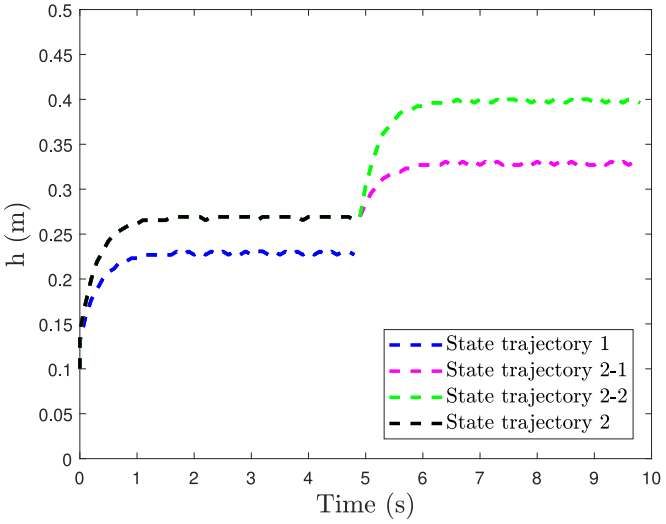

Fig. 10. Expected closed-loop state trajectories under different inputs computed by PID controllers over 10 s.

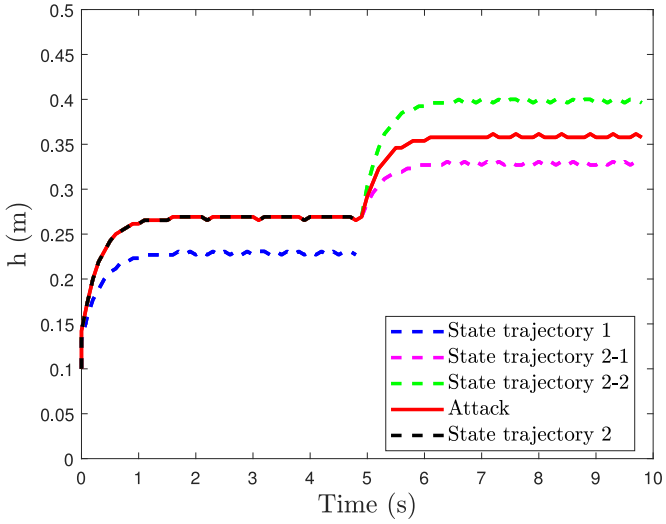

Fig. 11. Illustration of the directed randomization-inspired method when an attack policy is applied at t = 0.

loop state reaches the set-point for the controller that was selected and is maintained there. After being held at that value until 5 s, another opportunity is provided for random selection between two PI controllers (PI-2/1 and PI-2/2) that again have different set-points (where a goal in selecting such set-points from a closed-loop stability standpoint should be to not overflow the tank).

Fig. 10 helps to visualize the variation on directed randomization just described. Specifically, from 0 to 5 s of operation, two different trajectories are shown: one corresponding to the level when the set-point for the controller is changed from $h = 0.10$ m to $h = 0.23$ m (blue dashed line), and one corresponding to the level when the set-point for the controller is changed from $h = 0.10$ m to $h = 0.27$ m (black dashed line). One of these two control laws would be selected at t_0 and then used for 5 s; in Fig. 10, to visualize what happens after 5 s, the plot from 5 s to 10 s assumes that the controller with a set-point of $h = 0.27$ m was selected at t_0. Then at $t = 5$ s, we assume that a new set of two controllers are available to be randomly selected between, which have set-points of either $h = 0.33$ m (which would create a level trajectory given by the magenta dashed line) or $h = 0.40$ m (which would create a level trajectory given by the green dashed line). The two trajectories plotted from 5 s to 10 s aid with visualizing the two potential results of the random control law selection at $t = 5$ s.

While this strategy lacks the constant changing of the inputs to attempt to make it difficult for an attacker to avoid detection, it provides a visualization of the impacts of having two PI control laws for a given state measurement as the means for selecting between two random control actions in this example. The results suggest that with the PI tuning chosen, the closed-loop level response does not appear to have significant overshoot, indicating that a steady-state analysis may be able to be used in this case to aid in locating two different set-points for each state measurement that might be stabilizing (i.e., those which do not result in a steady-state where the tank level is above the overflow point). However, Fig. 10 also indicates that two potential challenges with the method are: 1) selecting the set-points so that in the short sampling period, they create state trajectories that are sufficiently different from one another after a sampling period to tell which of the inputs was applied and 2) the disruption to normal operation. Regarding the first point, for example, though the steady-state values in Fig. 10 after 5 s of operation are not close to one another, it can be seen that closer to 0 s,

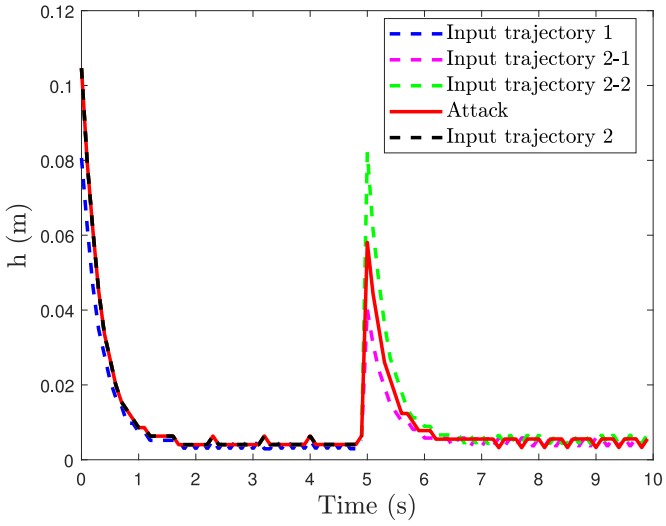

Fig. 12. Expected control action trajectories using the direct randomization method versus attacked inputs over 10 s.

the state trajectories appear more similar, indicating that care would need to be taken to ensure that the two possible values of the set-point considered at t_0 provide state measurements that differ sufficiently after Δ to enable discernment of which input was applied. One idea might be to provide one set-point above the initial level and one below. However, we can see that already the strategy in Fig. 10 is somewhat disruptive to normal operation (i.e., steady-state tracking), and further actions to attempt to further distinguish between trajectories could further impact steady-state tracking objectives.

Though this strategy does not rigorously follow the directed randomization implementation strategy, we consider an attack on the system. To illustrate how an attack might be detected using this approach, consider an attack policy in which the images provided to the image processing algorithm follow the control law of PI-2 from 0 to 5 s of operation and then change the set-point used from 5 s to 10 s. Figs. 11–12 show the expected state and input trajectories from the different PI controllers and the attack policy over 10 s of operation. We can see that from 0 to 5 s of operation, the attack policy chose the correct input trajectory to apply to the system, which resulted in the expected state trajectory under PI-2 control actions. However, from 5 to 10 s, the inputs from different PI controllers were expected to drive the closed-loop state to either $h = 0.33$ m (PI-2/1) or $h = 0.40$ m (PI-2/2). Under the attack policy, the state reaches $h = 0.36$ m within 10 s of operation. This was not the expected outcome and thus leads to the attack being flagged. It is notable that this attack does not produce a final level within the range of errors that might be expected in the camera sensor itself. For example, there is an error in the measurements of the level due to the error conversion between the pixel index and tank level (which is in the order of 10^{-3} m).

6. A minimal security architecture: further insights through distributed MPC

The directed randomization protocol was motivated by the desire to move toward a minimal security architecture, as discussed above. We close this work with several further insights on minimal security architecture through consideration of distributed (rather than centralized) control. Distributed control systems have been considered particularly within the context of large-scale production systems with numerous process states and inputs where centralized frameworks may be limited by computation time, but they have also been noted for their potential for greater fault-tolerance by enabling parts of the system to be operated independently (Liu et al., 2010). It might be asked whether greater fault-tolerance corresponds to greater attack resilience or detection capabilities, and therefore how distributed control might fit within the minimal security architecture framework. Distributed control systems have already been considered in cyberattack contexts within, for example, information exchange (Ananduta et al., 2018) and machine learning-based detection (Chen et al., 2021). A challenge for distributed control is that the physical manifestation of a distributed control system yields a greater potential surface for cyberattacks due to the increased number of controller units and communication links. Therefore, we must first extend the discussion of theoretical resilience properties from centralized to distributed control. We will do this within the context of the three LEMPC-based Detection Strategies 1-S, 2-S, and 3-S described in Section 4. Theoretical results for distributed LEMPC have been previously explored in Albalawi et al. (2017); Christofides et al., 2013; Liu et al. (2010), and the closed-loop stability results strongly parallel those for a centralized LEMPC. Therefore, we expect to be able to translate the results of the three detection frameworks developed for centralized LEMPC to a distributed LEMPC context. However, the structure of the distributed controllers should be considered for its abilities to both open new attacks and add a level of redundancy for attack detection (such as the chance that controllers which are behaving oddly may be symptomatic of an attack in other parts of the system which may or may not be displaying normal behavior). To move toward discussing distributed LEMPC in a minimal security architecture framework, in this section, we first present two distributed LEMPC (DLEMPC) formulations (Albalawi et al., 2017; Christofides et al., 2013; Liu et al., 2010) and subsequently discuss their properties in relation to cyberattack detection and post-detection operating capabilities.

6.1. Class of systems for distributed LEMPC

For the discussion of distributed LEMPC, we consider the input-affine subset of the class of nonlinear process systems of Eq. (1) as follows:

$$\dot{x}(t) = f(x(t)) + \sum_{j=1}^{m} g_j(x(t))u_j(t) + k(x(t))w(t) \tag{42}$$

where $x \in X \subset R^n$ is the state vector of the system, $u_j \in U \subset R^{z_j}\ \forall\, j = 1, \cdots, m$ are the input vectors, and $w \in W \subset R^z$ are the disturbance vectors. The index $j = 1, \ldots, m$ represents components of a subsystem (i.e., inputs to be computed in different distributed LEMPC's). We assume f, g, and k are locally Lipschitz functions, and $f(0) = 0$ when $u_j(t) = 0$ for all $j = 1, \ldots, m$ and $w(t) = 0$ for all t. As in Section 2.2, we assume the existence of a sufficiently smooth Lyapunov function V, functions $\alpha_k(\cdot)$, $k = 1, \ldots, 4$, and controller $h(x) = [\overline{h}_1(x) \ldots \overline{h}_m(x)]^T$ where now $u_j = \overline{h}_j(x), j = 1, \ldots, m$. Furthermore, Eqs. (2)–(7) are modified to be reflective of the process model of Eq. (42). Examples of the Lipschitz constraints modified from Eq. (4) are shown below for clarity:

$$\left|\frac{\partial V}{\partial x} f(x) - \frac{\partial V}{\partial x'} f(x')\right| \leq \overline{L}_x |x - x'| \tag{43}$$

$$\left|\frac{\partial V}{\partial x} g_j(x) - \frac{\partial V}{\partial x} g_j(x')\right| \leq L'_{u_j} |x - x'|,\ j = 1, \ldots, m \tag{44}$$

6.2. Sequential distributed Lyapunov-based economic model predictive control

One type of distributed LEMPC is said to have a "sequential" architecture. In a sequential distributed Lyapunov-based economic model predictive controller in which information flows in one direction from distributed controllers 1 through m, where $m > 1$, each controller j (with $j = 1, \ldots, m$) minimizes the global cost function for its respective future input trajectories based on input trajectories received from controllers prior in the sequence, and assuming an input trajectory of $h_j(x)$ for each subsequent controller. The jth actuators apply the first input step trajectory, and the jth LEMPC sends the computed input trajectories for the distributed controllers 1 to j to the $j+1$th controller. Full state feedback is assumed to be available to each distributed controller at each sampling time.

The design of the jth sequential DLEMPC is as follows:

$$\min_{u_j(t) \in S(\Delta)} \int_{t_k}^{t_{k+N}} \left[L_e\left(\widetilde{x}(\tau), u_j(\tau)\right)\right] d\tau \tag{45a}$$

$$\text{s.t. } \dot{\widetilde{x}}(t) = f(\widetilde{x}(t)) + \sum_{j=1}^{m} g_j(\widetilde{x}(t))u_j(t) \tag{45b}$$

$$\widetilde{x}(t_k) = x(t_k) \tag{45c}$$

$$x(t) \in X,\ \forall\, t \in [t_k, t_{k+N}) \tag{45d}$$

$$u_j(t) \in U_j,\ \forall\, t \in [t_k, t_{k+N}) \tag{45e}$$

$$u_r(t) = \overline{h}_r(\widetilde{x}(t_{k+l})), r = j+1, \cdots, m,\ \forall\, t \in [t_{k+l}, t_{k+l+1}), l = 0, \cdots, N-1 \tag{45f}$$

$$u_p(t) = u_p^*(t|t_k), p = 1, \ldots, j-1,\ t \in [t_k, t_{k+N}) \tag{45g}$$

$$V(\tilde{x}(t)) \le \rho_e, \quad \forall t \in [t_k, t_{k+N}), \text{ if } x(t_k) \in \Omega_{\rho_e} \tag{45h}$$

$$\frac{\partial V(x(t_k))}{\partial x}\sum_{j=1}^{m} g_j(x(t_k))u_j(t_k) \le \frac{\partial V(x(t_k))}{\partial x}\sum_{j=1}^{m} g_j\left(x(t_k), \overline{h}_j(x(t_k))\right) \text{ if } x(t_k) \notin \Omega_{\rho_e} \tag{45i}$$

where $\dot{\tilde{x}}(t)$ is the predicted state trajectory of Eq. (42) within the jth sequential DLEMPC with $w \equiv 0$ (Eq. (45b)). The model is initialized by a state measurement at time t_k, denoted as $x(t_k)$ (Eq. (45c)). Eqs. (45d) and (45e) denote the state and input constraints, respectively. Eq. (45f) defines the input values assumed for the distributed controllers computed after the jth DLEMPC in the sequence, while Eq. (45g) defines the input values of the distributed controllers evaluated before the jth DLEMPC. The constraints of Eqs. (45h) and (45i) are Lyapunov-based stability constraints. The vector of optimal inputs for the jth subsystem is denoted $u_j^*(\tau|t_k)$, where $\tau \in [t_k, t_{k+N})$, and $\Omega_{\rho_e} \subset \Omega_\rho$ is a subset of the stability region under which the state is allowed to evolve freely according to Eq. (45h).

The implementation strategy of the sequential distributed controller is as follows (Albalawi et al., 2017; Liu et al., 2010):

1. At a sampling time t_k, each distributed controller receives a state measurement $x(t_k)$.
2. For $j = 1$ to $j = m$:
   (a) The jth DLEMPC receives the entire set of input trajectories $u_p(\tau|t_k)$, $p = 1, \ldots, j-1$, $\tau \in [t_k, t_{k+N})$ from the $j-1$th DLEMPC and assumes $u_r(\tau) = h_r(\tilde{x}(t_{k+l}))$, $r = j+1, \ldots, m$, $\tau \in [t_{k+l}, t_{k+l+1})$, $l = 0, \ldots, N-1$ for the subsystem inputs which have not yet been calculated. It then solves Eq. (45) to find the optimal input trajectory $u_j^*(\tau|t_k)$, $\tau \in [t_k, t_{k+N})$.
   (b) The jth DLEMPC applies $u_j^*(t_k|t_k)$ to the process, and sends $u_p(\tau|t_k)$, $p = 1, \ldots, j$, $\tau \in [t_k, t_{k+N})$, to the $j+1$ DLEMPC.
3. At the beginning of the next sampling period, return to Step 1 ($k \leftarrow k+1$)

6.3. Iterative distributed Lyapunov-based economic model predictive control

A second type of distributed LEMPC is iterative DLEMPC. Iterative DLEMPC differs from sequential DLEMPC in that each controller communicates with all other controllers after every controller solves for the optimal inputs, instead of a controller receiving information from the controllers ahead of them, making a decision, and passing it along to the next in line. Instead, each controller solves for optimal inputs simultaneously. Then, these inputs are all shared among the distributed controllers, the optimization problem is re-solved, and new inputs are calculated. Each final input is passed simultaneously to its respective actuator when a termination condition has been reached, typically after a number of iterations (c) or when the inputs computed by the actuators are not changing much between iterations (i.e., the difference in their values between two iterations is within a convergence bound). The formulation of an iterative DLEMPC is as follows:

$$\min_{u_j(t) \in S(\Delta)} \int_{t_k}^{t_{k+N}} \left[L_e\left(\tilde{x}(\tau), u_j(\tau)\right)\right] d\tau \tag{46a}$$

$$\text{s.t.} \quad \dot{\tilde{x}}(t) = f(\tilde{x}(t)) + \sum_{j=1}^{m} g_j(\tilde{x}(t))u_j(t) \tag{46b}$$

$$\tilde{x}(t_k) = x(t_k) \tag{46c}$$

$$\tilde{x}(t) \in X, \forall t \in [t_k, t_{k+N}) \tag{46d}$$

$$u_i(t) = \overline{h}_i(\tilde{x}(t_{k+l})), \ i \in \{1, \cdots, m\}, \ i \ne j, \ t \in [t_{k+l}, t_{k+l+1}), \ l = 0, \cdots, N-1, c = 1 \tag{46e}$$

$$u_i(t) = u_{i,c-1}^*(t|t_k), \ i \in \{1, \ldots, m\}, \ i \ne j, \ t \in [t_{k+l}, t_{k+l+1}), \ l = 0, \ldots, N-1, \ c \ge 2 \tag{46f}$$

$$u_j(\tau) \in U_j, \forall t \in [t_k, t_{k+N}) \tag{46g}$$

$$V(\tilde{x}(t)) \le \rho_e, \forall \ t \in [t_k, t_{k+N}) \tag{46h}$$

$$\frac{\partial V(\tilde{x}(t_k))}{\partial x} g_j\left(\tilde{x}(t_k), u_j(t_k)\right) \le \frac{\partial V(\tilde{x}(t_k))}{\partial x} g_j\left(\tilde{x}(t_k), \overline{h}_j(t_k)\right), \text{if } x(t_k) \notin \Omega_{\rho_e} \tag{46i}$$

An implementation strategy for iterative DLEMPC is described as follows (Albalawi et al., 2017; Liu et al., 2010):

1. At a sampling time t_k, all of the controllers receive a state measurement $x(t_k)$. Set $c = 1$.
2. At iteration c, $c \ge 1$
   (a) If $c = 1$, each of the j-DLEMPC's, $j = 1, \ldots, m$, assumes that the inputs u_i, $i \ne j$, which it is not calculating are set to $h_i(\tilde{x}(t_{k+l}))$, $\forall \ t \in [t_{k+l}, t_{k+l+1})$, $l = 0, \ldots, N-1$. If $c > 1$, each of the j-DLEMPC's, $j = 1, \ldots, m$, assumes that the inputs u_i which it is not calculating are set to the values $u_{i,c-1}^*(t|t_k)$, for $i = 1, \ldots, m$ but $i \ne j$, $\forall \ t \in [t_{k+l}, t_{k+l+1})$, $l = 0, \ldots, N-1$, which were computed by all of the other DLEMPC's at the prior iteration.
   (b) Each DLEMPC shares its future input trajectory $u_{j,c}^*(t|t_k)$, $\forall \ t \in [t_{k+l}, t_{k+l+1})$, $l = 0, \ldots, N-1$, with all of the other DLEMPC's.
   (c) Iteration termination conditions are checked, such as whether a maximum bound on the number of allowable iterations has been reached or if the profit has substantially changed between the last two iterations of the DLEMPC. If the termination criteria do not force termination, it is checked that stability termination conditions are satisfied, which are that $V(\tilde{x}^{tot}) \le \rho_e$ if $x(t_k) \in \Omega_{\rho_e}$ or that $V(\tilde{x}^{tot}) \le V(x(t_k))$ when the constraint of Eq. (46i) is applied, where $\tilde{x}^{tot}$ represents the prediction of the nominal system of Eq. (42) if the m input vectors $u_{j,c}^*(t_k|t_k)$ from all of the j-DLEMPC's, $j = 1, \ldots, m$, are implemented. If the iteration termination conditions and the stability termination conditions indicate termination is needed, $u_{j,c}^*(t_k|t_k)$, $j = 1, \ldots, m$ is applied; go to Step 3. If the iteration termination conditions indicate termination is needed but the stability termination conditions are not satisfied, $h_i(x(t_k))$, $i = 1, \ldots, m$, is applied to the process; go to Step 3. If the iteration termination conditions do not indicate termination is needed, return to Step 2 ($c \leftarrow c+1$).
3. At the beginning of the next sampling period, return to Step 1 ($k \leftarrow k+1$)

6.4. Detection of cyberattacks using distributed LEMPC

This section provides discussion and theory indicating that Detection Strategies 1-S, 2-S, and 3-S readily extend to a distributed LEMPC framework, showing that using these controllers to control a process and participate in cyberattack detection will provide similar guarantees on safety as obtained when a centralized LEMPC is used. Specifically, the first detection strategy (1-S) is a method for verifying controller performance by checking how the process operates under a specific control strategy with an expected trajectory in the absence of a cyberattack. To extend the method in Detection Strategy 1-S to the case of a distributed control framework, we consider analyzing the behavior of the Lyapunov function V_i over a sampling period in which the DLEMPC for every subsystem is switched to be designed with respect to the ith steady-state.

V_i is expected to decrease towards a region near the origin when the probing for cyberattacks has been initialized and if conditions similar to those of Theorem 1 hold for the final control actions implemented (which is aided through the checks performed in the iterative DLEMPC in Step 2c). The Lyapunov function for the system under the full set of inputs to be applied is calculated after the last controller in a sequential framework calculates inputs, or after the last iteration in an iterative framework, in order to utilize the inputs calculated by every controller in evaluating the Lyapunov function along the predicted state under the final control actions. By utilizing an algorithm which checks the trajectory of the Lyapunov function of the system under the full set of inputs to be applied, it may be possible to detect an attack on the system when a positive time derivative of the Lyapunov function is calculated over a sampling period, as in the centralized case.

We will refer to the baseline LEMPC for the $(1,j)$-DLEMPC, and the jth DLEMPC with the ith steady-state as the (i,j)-DLEMPC, $i > 1$. It may be possible for an attacker to exploit the distributed design of the control system by providing false sensor measurements to only, for example, a subset of distributed controllers that may cause V_i to decrease under the total set of control actions computed, but may not cause it to decrease under the inputs computed and assumed in a given subsystem. To avoid this, the implementation strategy of the sequential DLEMPC can include a check not only on $\dot{V}_i$ for the full system, but also within individual distributed LEMPC's. Assuming that the communication links between the controllers are not attacked, the role of the distributed framework here is to add a level of redundancy. Furthermore, the DLEMPC's always have feasible inputs when there is no cyberattack (the feasible input is the Lyapunov-based controller in each subsystem in the case of a sequential DLEMPC framework, or is the last set of inputs that caused the stability conditions of Step 2c to be satisfied (or the Lyapunov-based controller if the inputs after the first iteration fail to do this)). This means that infeasibility of any distributed controller is indicative of a cyberattack (this aids in detecting cases in which a cyberattack reprograms a controller to calculate local inputs which cause the remaining controllers to calculate infeasible inputs (e.g., it does not pass a solution that met constraints in the prior DLEMPC)).

We now analyze whether DLEMPC could aid with controller diagnosis when using Detection Strategy 1-S. To analyze this, one may take advantage of the sequential and iterative designs to check the value of the Lyapunov function over a sampling period in every controller to determine its trajectory. In both DLEMPC frameworks, each controller will predict a slightly different Lyapunov function over a sampling period due to assumptions made about the inputs of other controllers in the system (i.e. controllers in the middle of the sequential framework will receive calculated inputs for controllers prior in the sequence while assuming that the Lyapunov-based controller is applied for all other controllers, which will result in a slightly different predicted state compared to controllers later in the sequence which have received calculated optimal inputs from the other controllers). An iterative framework similarly will produce a discrepancy between the Lyapunov function each controller calculates before convergence; specifically, at the first iteration, each controller attempts to optimize its respective input while assuming all others are the Lyapunov-based control actions, then each controller recalculates its input at each iteration, assuming the calculated inputs from the last iteration for every other controller. This indicates that differences in state predictions between the controllers are not necessarily expected to signify an attack. Furthermore, false sensor measurements may affect one, many, or all of the controllers (as well as one or more states at a time), and some or all controllers could be receiving false inputs as the inputs from the other distributed controllers. As a result, there is a possibility that it could be difficult to distinguish which sensor or communication link between controllers is being attacked even if the trajectory of the Lyapunov function using the state predictions from each controller is checked. For example, if the Lyapunov function increases over time in one controller, it may not be clear what the source of the issue is, and if it is affecting other controllers as well but that because they are all providing different predictions from one another, the others' Lyapunov functions are not showing an increase. One could attempt to use the distributed architecture to give strategies for potential diagnosis. For example, in the sequential framework, if $\dot{V}$ for the jth controller is not negative under the control actions computed in that controller, an attack may have occurred at that controller or higher in the sequence; if the jth controller in an iterative framework has a value of $\dot{V}$ that is not negative under the control actions it computed, this may suggest that this controller is subject to some type of attack.

In Detection Strategy 2-S, state predictions are utilized to detect potential cyberattacks while providing closed-loop stability guarantees for one sampling period if a cyberattack is not detected but conditions similar to those of Theorem 2 hold for the final control actions applied to the system. The state predictions should utilize the final set of all control actions to be applied to the process. If the deviation between the prediction and the measured state exceeds a threshold, then a cyberattack may be flagged on the system. We can ask whether checking state predictions with the inputs used in each distributed controller could be used for diagnosis of attacks (i.e., the computed input and the assumed values of the other inputs). However, because each distributed controller is computing state predictions with different input assumptions, each controller would create state predictions using potentially different inputs than the final values that will be applied to the process by using some inputs computed by each of the individual controllers. Therefore, any threshold used to check if an individual controller was potentially under attack would need to set a threshold for comparison of the prediction with the measurement that accounts for these input discrepancies.

In Detection Strategy 3-S, state estimates are used for providing the state estimates to each DLEMPC and for detecting attacks. Though every controller could also have an associated estimator so that each estimate at every distributed controller could be compared with the others, this is a high degree of redundancy compared to having a single estimator and broadcasting its result to every DLEMPC. Distributed state estimators (Zhang, 2014) may be considered with distributed control instead of centralized state estimators. Overall, the distributed versions of the detection strategies can be readily implemented (the proofs of closed-loop stability and feasibility for each method extend from the centralized versions discussed in Section 4 when the inputs actually applied to the process meet conditions similar to those in Section 4 . However, it is not obvious that this extension has large benefits for attack detection or diagnosis beyond what the centralized architecture could achieve. Though there are more potential attack surfaces due the increased levels of communication required to send input and state measurements to and between controllers, there are also opportunities for adding additional checks at each controller (either state measurements or predictions) to seek to catch if any of the controllers in particular is acting abnormally. This implies that the direct extension of the three detection strategies to a distributed control framework does not clearly have benefits from a minimal security architecture perspective, as it suggests that these strategies can add additional redundant checks but does not provide a clear avenue for reducing the cost for securing the system.

Remark 13. In blockchain (Rouhani and Deters, 2019), consensus among distributed elements can help to verify data. Validating data in a blockchain can be computationally intensive, whereas controllers in a distributed architecture would want to validate data received from one another without excessive computation time. Methods for reducing computation time of data verification with blockchain have also been explored (Choi et al., 2020), and it is possible that blockchain technologies can be useful toward security for data in manufacturing and engineering contexts (Joannou et al., 2020).

Remark 14. As noted above, one of the challenges for distributed control may be the larger attack surface opened by having additional communication channels. However, one could imagine adding new requirements to the communication protocols to attempt to stop this from adding too much vulnerability to the system. For example, in Bellink (2019), a check is made by a robot that when it receives data from another system, it has a secondary number that has been encrypted and passed between the robot and the other system to verify its identity before accepting its information. A similar strategy could be used with the distributed control strategies here as well (including encrypting data sent between parties to protect it) to attempt to reduce the attack surface offered by greater communication between control laws and provide the benefits of greater redundancy and cross-checking if desired. However, it is expected that redundancy and cross-checking could increase costs and computational overhead.

Remark 15. Another type of attack that can occur on distributed systems could be to attempt to mess up the profits under the controllers determined by the set of controllers by attacking one or a few of the distributed controllers (Velarde et al., 2017). In general for nonlinear systems, unless the problem has a special structure (Hammami et al., 2020) such that it can be decomposed so that the original is guaranteed to have the same optimal solution as the distributed version, it is hard to make claims about how far off such an attack could bring profits compared to the centralized case (because it is harder in that case to even clarify how far off the profits should be for the distributed case compared to the centralized case). However, attacking profits is a potential motive of an attacker who is going after a distributed control system.

Remark 16. Though above it was not clear that the redundancy of the controllers was helpful within a minimum security architecture for detection, it might be asked whether the ability to remove a subset of controllers from a network without removing all controllers by using distributed control could be a useful characteristic for cybersecurity of control systems. One challenge, however, is that this implies a desire to keep a process operational after some units have been under attack. This may be challenging to want to attempt to do if there is a possibility that the other units not known to be attacked could also have been compromised. If it was desired to do so, however, (or to otherwise take controllers offline to reprogram them, where the concept of periodic network refreshment has some precedent in Griffioen et al. (2019)) then attacks would need to be able to be diagnosed to shut down the appropriate controller. If a controller/input is taken offline for a given process, then a backup control policy could be applied in its place while it is offline while the other controllers solve the distributed LEMPC problem modified to expect that the isolated controllers are applying the backup control law inputs. This could impact feasibility of the other distributed controllers that are still in operation when one is taken offline, unless the backup control law is the Lyapunov-based controller on which the LEMPC is based. From a stability perspective, this strategy of controller removal would be similar to Lao et al. (2014), in which actuators were taken offline for preventative maintenance and the closed-loop state was driven before the time that they are taken offline into a region around the origin within which the closed-loop state could be stabilized with the remaining controllers. Though this was implemented in a centralized control framework, the distributed control framework could be designed to achieve a similar goal. However, in the case that attacks could cause certain controllers to be diagnosed as rogue and taken offline at unexpected times, it may be necessary to operate within the intersection of all stability regions that could be left if any of the controllers are suddenly removed. Fig. 13 illustrates the concept of controller removal. On the left in Fig. 13 are initially three distributed controllers ($m = 3$) which communicate in sequence. At some time, the first of the controllers is removed. The control law used in its place is operated in a decentralized fashion (the case on the right in Fig. 13; i.e., it is not aware of what the other distributed controllers are doing).

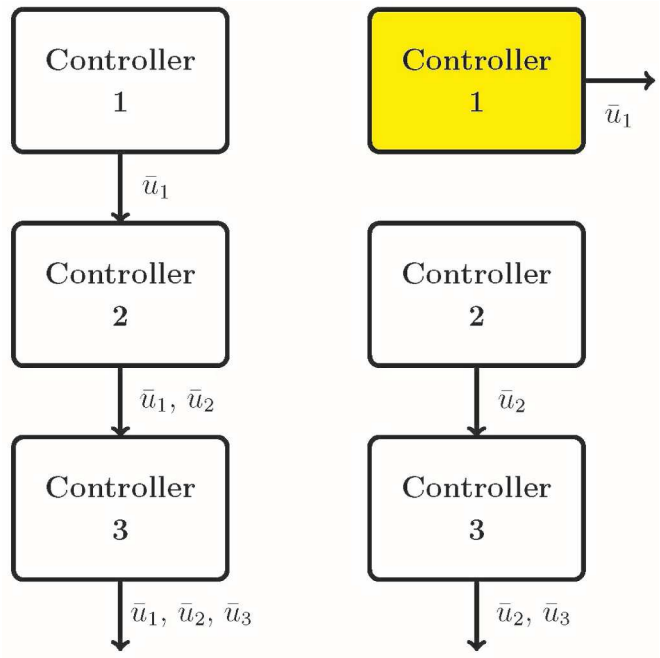

Fig. 13. Distributed (left) and decentralized/distributed (right) architectures.

However, in the decentralized-distributed case, not all controllers communicate, and therefore the combined effects of $\bar{u}_1$, $\bar{u}_2$, and $\bar{u}_3$ are not assessed by any of the controllers before these control actions are implemented on the process. This is the motivation for instead causing the controllers to be unified by causing the backup policy to correspond to an appropriate stabilizing control law which all remaining distributed controllers can be updated to use. However, one could imagine a strategy that an attacker could use if systems were to be removed due to an anomalous behavior of one distributed controller; the attacker might deliberately attack one controller to get it offline to then attack the backup control policy, or attack some of the remaining controllers. One could imagine trying to make the situation harder for the attacker by having a tradeoff between what is computed. For example, perhaps at random times the backup control policy could change to a constant control action, in sync with the other controllers adjusting their assumed control action for that controller, to attempt to prevent the attacker from having as much authority on the system with one of the distributed controllers taken offline, particularly if taking a controller offline is inspired by a suspected attack on the system indicating infiltration.

7. Conclusion

This work discussed the goal of chemical process control cybersecurity studies as being the understanding of what the cheapest and most flexible security architecture might be for process systems by considering both the traditional security approaches along with newer control-theoretic approaches in seeking to find the best path forward for industry in a next-generation manufacturing setting. We presented discussion of this "minimal security architecture" concept with respect to detection methods which we had previously introduced and their distributed versions. We also showed how this concept motivates the creation of new strategies, such as the directed randomization protocol that was discussed and for which the concept of creating different control actions for the same state measurement was demonstrated using an image-based level control example in Blender.

Declaration of Competing Interest

The authors declare that they have no known competing financial

interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

Financial support from the National Science Foundation CNS-1932026 and CBET-1839675, Air Force Office of Scientific Research (Award number FA9550-19-1-0059), and Wayne State University is gratefully acknowledged. We would like to thank Arlan Bonislawski for his work in explaining elliptic curve cryptography. We would like to thank Minhazur Rahman, Govanni Gjonaj, and Michael Williamson for their work in building a foundation of image-based control in Blender that helped us to complete this work.

References

Ahmad, H., 2020. Cyber Attack Detection in Nonlinear Binary Distillation Column. Master's thesis.
Ahrens, J.H., Khalil, H.K., 2009. High-gain observers in the presence of measurement noise: a switched-gain approach. Automatica 45, 936–943.
Alanqar, A., Ellis, M., Christofides, P.D., 2015. Economic model predictive control of nonlinear process systems using empirical models. AlChE J. 61, 816–830.
Albalawi, F., Durand, H., Christofides, P.D., 2017. Distributed economic model predictive control for operational safety of nonlinear processes. AlChE J. 63 (8), 3404–3418.
Ananduta, W., Barreiro-Gomez, J., Ocampo-Martinez, C., Quijano, N., 2018. Resilient information-exchange protocol for distributed model predictive control schemes. 2018 Annual American Control Conference (ACC). IEEE, pp. 1286–1291.
Ani, U.P.D., He, H., Tiwari, A., 2017. Review of cybersecurity issues in industrial critical infrastructure: manufacturing in perspective. J. Cyber Secur. Technol. 1 (1), 32–74.
Aziz, M.Z.A., Ibrahim, M.Y., Omar, A.M., Ab Rahman, R., Zan, M.M.M., Yusof, M.I., 2012. Performance analysis of application layer firewall. 2012 IEEE Symposium on Wireless Technology and Applications (ISWTA). IEEE, pp. 182–186.
Bellink, K., 2019. Secure and Private Formation Control of the Nexus Robot Using Fully Homomorphic Encryption. Master Thesis.. University of Groningen.
Bhadriraju, B., Kwon, J.S.-I., Khan, F., 2021. OASIS-P: Operable Adaptive Sparse Identification of Systems for fault Prognosis of chemical processes. J. Process Control 107, 114–126.
Bhadriraju, B., Kwon, J.S.-I., Khan, F., 2021. Risk-based fault prediction of chemical processes using operable adaptive sparse identification of systems (OASIS). Comput. Chem. Eng. 152, 107378.
Bhamare, D., Zolanvari, M., Erbad, A., Jain, R., Khan, K., Meskin, N., 2020. Cybersecurity for industrial control systems: a survey. Comput. Secur. 89, 101677 https://doi.org/10.1016/j.cose.2019.101677.
Budiawan, I., Pranoto, H.R., Hidayat, E.M., Arief, S.R., 2018. Design and implementation of cyber-physical system-based automation on plant chemical process: study case mini batch distillation column. 2018 6th International Conference on Information and Communication Technology (ICoICT). IEEE, pp. 360–365.
Cárdenas, A.A., Amin, S., Lin, Z.-S., Huang, Y.-L., Huang, C.-Y., Sastry, S., 2011. Attacks against process control systems: risk assessment, detection, and response. Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security. Association for Computing Machinery, New York, NY, USA, pp. 355–366.
Cybersecurity & Infrastructure Security Agency, 2019. Security tip (ST04-004): understanding firewalls for home and small office use. https://www.cisa.gov/uscert/ncas/tips/ST04-004. Accessed: 2022-02-07.
Chen, L., Zhu, G., Li, Q., Li, H., 2019. Adversarial example in remote sensing image recognition. arXiv preprint arXiv:1910.13222.
Chen, S., Wu, Z., Christofides, P.D., 2020. A cyber-secure control-detector architecture for nonlinear processes. AlChE J. 66 (5), e16907.
Chen, S., Wu, Z., Christofides, P.D., 2021. Cyber-security of centralized, decentralized, and distributed control-detector architectures for nonlinear processes. Chem. Eng. Res. Des. 165, 25–39.
Choi, M.K., Yeun, C.Y., Seong, P.H., 2020. A novel monitoring system for the data integrity of reactor protection system using blockchain technology. IEEE Access 8, 118732–118740.
Clark, A., 2015. Pillow (PIL fork) documentation. https://buildmedia.readthedocs.org/media/pdf/pillow/latest/pillow.pdf.
Christofides, P.D., Scattolini, R., Muñoz de la Peña, D., Liu, J., 2013. Distributed model predictive control: a tutorial review and future research directions. Comput. Chem. Eng. 51, 21–41.
Cormier, A., Ng, C., 2020. Integrating cybersecurity in hazard and risk analyses. J. Loss Prev. Process Ind. 64, 104044.
Darup, M.S., Jager, T., 2019. Encrypted cloud-based control using secret sharing with one-time pads. 2019 IEEE 58th Conference on Decision and Control (CDC). IEEE, pp. 7215–7221.
Davis, J., Edgar, T., Porter, J., Bernaden, J., Sarli, M., 2012. Smart manufacturing, manufacturing intelligence and demand-dynamic performance. Comput. Chem. Eng. 47, 145–156.
Ding, D., Wang, Z., Han, Q.-L., Wei, G., 2016. Security control for discrete-time stochastic nonlinear systems subject to deception attacks. IEEE Trans. Syst., Man, Cybern. 48 (5), 779–789.
Durand, H., 2018. A nonlinear systems framework for cyberattack prevention for chemical process control systems. Mathematics 6 (9), 169.
Durand, H., Wegener, M., 2020. Mitigating safety concerns and profit/production losses for chemical process control systems under cyberattacks via design/control methods. Mathematics 8, 499. https://doi.org/10.3390/math8040499.
Ellis, M., Durand, H., Christofides, P.D., 2014. A tutorial review of economic model predictive control methods. J. Process Control 24, 1156–1178.
Ellis, M., Zhang, J., Liu, J., Christofides, P.D., 2014. Robust moving horizon estimation based output feedback economic model predictive control. Syst. Control Lett. 68, 101–109.
Farid, H., 2012. Digital image forensics: lecture notes, exercise, and matlab code for a survey course in digital image and video forensics. Accessed: 2021-12-10.
Fawzi, H., Tabuada, P., Diggavi, S., 2014. Secure estimation and control for cyber-physical systems under adversarial attacks. IEEE Trans. Autom. Control 59 (6), 1454–1467.
Francia III, G.A., Thornton, D., Dawson, J., 2012. Security best practices and risk assessment of SCADA and industrial control systems. Proceedings of the International Conference on Security and Management (SAM). The Steering Committee of The World Congress in Computer Science, Computer Engineering, and Applied Computing, p. 1.
Ghaderi, M., Gheitasi, K., Lucia, W., 2020. A blended active detection strategy for false data injection attacks in cyber-physical systems. IEEE Trans. Control Netw. Syst. 8 (1), 168–176.
Gordon, L.A., Loeb, M.P., Zhou, L., 2020. Integrating cost–benefit analysis into the NIST cybersecurity framework via the Gordon–Loeb model. J. Cybersecur. 6 (1), tyaa005.
Griffioen, P., Romagnoli, R., Krogh, B.H., Sinopoli, B., 2019. Secure networked control via software rejuvenation. 2019 IEEE 58th Conference on Decision and Control (CDC). IEEE, pp. 3878–3884.
Hammami, D.E.H., Maraoui, S., Bouzrara, K., 2020. Nonlinear distributed model predictive control with dual decomposition and event-based communication approach. Trans. Inst. Meas. Control 42 (15), 2929–2940.
Hassanpour, H., Mhaskar, P., House, J.M., Salsbury, T.I., 2020. A hybrid modeling approach integrating first-principles knowledge with statistical methods for fault detection in HVAC systems. Comput. Chem. Eng. 142, 107022.
Heidarinejad, M., Liu, J., Christofides, P.D., 2012. Economic model predictive control of nonlinear process systems using Lyapunov techniques. AlChE J. 58, 855–870.
Hu, Q., Fooladivanda, D., Chang, Y.H., Tomlin, C.J., 2017. Secure state estimation and control for cyber security of the nonlinear power systems. IEEE Trans. Control Netw. Syst. 5 (3), 1310–1321.
Iaiani, M., Tugnoli, A., Bonvicini, S., Cozzani, V., 2021. Analysis of cybersecurity-related incidents in the process industry. Reliab. Eng. Syst. Saf. 209, 107485.
Joannou, D., Kalawsky, R., Martínez-García, M., Fowler, C., Fowler, K., 2020. Realizing the role of permissioned blockchains in a systems engineering lifecycle. Systems 8 (4), 41.
Ko, W.-H., Satchidanandan, B., Kumar, P., 2016. Theory and implementation of dynamic watermarking for cybersecurity of advanced transportation systems. 2016 IEEE Conference on Communications and Network Security (CNS). IEEE, pp. 416–420.
Ko, W.-H., Satchidanandan, B., Kumar, P., 2019. Dynamic watermarking-based defense of transportation cyber-physical systems. ACM Trans. Cyber-Physical Syst. 4 (1), 1–21.
Kumari, P., Bhadriraju, B., Wang, Q., Kwon, J.S.-I., 2022. A modified Bayesian network to handle cyclic loops in root cause diagnosis of process faults in the chemical process industry. J. Process Control 110, 84–98.
Lao, L., Ellis, M., Christofides, P.D., 2014. Smart manufacturing: handling preventive actuator maintenance and economics using model predictive control. AlChE J. 60, 2179–2196.
Lao, L., Ellis, M., Durand, H., Christofides, P.D., 2015. Real-time preventive sensor maintenance using robust moving horizon estimation and economic model predictive control. AlChE J. 61, 3374–3389.
Liu, J., Chen, X., Muñoz de la Peña, D., Christofides, P.D., 2010. Sequential and iterative architectures for distributed model predictive control of nonlinear process systems. AlChE J. 56 (8), 2137–2149.
Liu, S., Wei, G., Song, Y., Liu, Y., 2016. Extended Kalman filtering for stochastic nonlinear systems with randomly occurring cyber attacks. Neurocomputing 207, 708–716.
Mahler, T., Nissim, N., Shalom, E., Goldenberg, I., Hassman, G., Makori, A., Kochav, I., Elovici, Y., Shahar, Y., 2018. Know your enemy: characteristics of cyber-attacks on medical imaging devices. arXiv preprint arXiv:1801.05583.
McLaughlin, S., Konstantinou, C., Wang, X., Davi, L., Sadeghi, A.-R., Maniatakos, M., Karri, R., 2016. The cybersecurity landscape in industrial control systems. Proc. IEEE 104 (5), 1039–1057.
Nicola, M., Nicola, C.-I., Duţă, M., Sacerdoţianu, D., 2018. SCADA systems architecture based on OPC and web servers and integration of applications for industrial process control. Int. J. Control Sci. Eng. 8 (1), 13–21.
Nieman, K., Oyama, H.C., Wegener, M., Durand, H., 2020. Predict the impact of cyberattacks on control systems. Chem. Eng. Prog. 116 (9), 52–57.
Oyama, H., Durand, H., 2020. Integrated cyberattack detection and resilient control strategies using Lyapunov-based economic model predictive control. AIChE J. 66, e17084.
Oyama, H., Leonard, A.F., Rahman, M., Gjonaj, G., Williamson, M., Durand, H., 2022. On-line process physics tests via Lyapunov-based economic model predictive control and simulation-based testing of image-based process control. Proceedings of the American Control Conference, Atlanta, Georgia.
Oyama, H., Messina, D., Durand, H., Rangan, K.K., 2022b. Lyapunov-based economic model predictive control for detecting and handling actuator and simultaneous

sensor/actuator cyberattacks on process control systems. Front. Chem. Eng. 4, 810129 https://doi.org/10.3389/fceng.2022.810129.
Oyama, H., Messina, D., O'Neill, R., Cherney, S., Rahman, M., Rangan, K.K., Gjonaj, G., Durand, H., 2022c. Test methods for image-based information in next-generation manufacturing. Proceedings of the IFAC Symposium on Dynamics and Control of Process Systems, including Biosystems (DYCOPS), Busan, Republic of Korea, 55. Elsevier, pp. 73–78.
Oyama, H., Rangan, K.K., Durand, H., 2021. Handling of stealthy sensor and actuator cyberattacks on evolving nonlinear process systems. J. Adv. Manuf. Process. 3 (3), e10099.
Patel, S., Zaveri, J., 2010. A risk-assessment model for cyber attacks on information systems. J. Comput. 5 (3), 352–359.
Qin, S.J., Badgwell, T.A., 2003. A survey of industrial model predictive control technology. Control Eng. Pract. 11, 733–764.
Qin, Y., Yin, X., 2022. Start-up monitoring for intermittent manufacturing based on hierarchical stationarity analysis. Chem. Eng. Res. Des. 185, 26–36.
Rangan, K.K., Oyama, H., Durand, H., 2021. Integrated cyberattack detection and handling for nonlinear systems with evolving process dynamics under Lyapunov-based economic model predictive control. Chem. Eng. Res. Des. 170, 147–179.
Rangan, K.K., Oyama, H., Durand, H., 2022. Actuator cyberattack handling using Lyapunov-based economic model predictive control. 13th IFAC Symposium on Dynamics and Control of Process Systems, including Biosystems (DYCOPS 2022).
Rawlings, J.B., Angeli, D., Bates, C.N., 2012. Fundamentals of economic model predictive control. Proceedings of the IEEE Conference on Decision and Control, Maui, Hawaii, pp. 3851–3861.
Ren, A., Wu, D., Zhang, W., Terpenny, J., Liu, P., 2017. Cyber security in smart manufacturing: Survey and challenges. IIE Annual Conference Proceedings. Institute of Industrial and Systems Engineers (IISE), pp. 716–721.
Rouhani, S., Deters, R., 2019. Blockchain based access control systems: state of the art and challenges. IEEE/WIC/ACM International Conference on Web Intelligence, pp. 423–428.
Satchidanandan, B., Kumar, P.R., 2016. Dynamic watermarking: active defense of networked cyber–physical systems. Proc. IEEE 105 (2), 219–240.
Smith, R.E., 2016. Elementary Information Security. Jones & Bartlett Learning.
Stewart, J.M., 2013. Network Security, Firewalls and VPNs. Jones & Bartlett Publishers.
Sun, J., Cao, Y., Chen, Q.A., Mao, Z.M., 2020. Towards robust Lidar-based perception in autonomous driving: general black-box adversarial sensor attack and countermeasures. 29th Security Symposium ({USENIX} Security 20), pp. 877–894.
Teixeira, A., Shames, I., Sandberg, H., Johansson, K.H., 2012. Revealing stealthy attacks in control systems. 2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton). IEEE, pp. 1806–1813.
Trabelsi, Z., Zeidan, S., Shuaib, K., Salah, K., 2018. Improved session table architecture for denial of stateful firewall attacks. IEEE Access 6, 35528–35543.
Vamvoudakis, K.G., Hespanha, J.P., Kemmerer, R.A., Vigna, G., 2013. Formulating cyber-security as convex optimization problems. Control of Cyber-Physical Systems. Springer, pp. 85–100.
Velarde, P., Maestre, J.M., Ishii, H., Negenborn, R.R., 2017. Vulnerabilities in Lagrange-based DMPC in the context of cyber-security. 2017 IEEE International Conference on Autonomic Computing (ICAC). IEEE, pp. 215–220.
Weerakkody, S., Ozel, O., Griffioen, P., Sinopoli, B., 2017. Active detection for exposing intelligent attacks in control systems. 2017 IEEE Conference on Control Technology and Applications (CCTA). IEEE, pp. 1306–1312.
Wu, Z., Albalawi, F., Zhang, J., Zhang, Z., Durand, H., Christofides, P.D., 2018. Detecting and handling cyber-attacks in model predictive control of chemical processes. Mathematics 6 (10), 173.
Wu, Z., Chen, S., Rincon, D., Christofides, P.D., 2020. Post cyber-attack state reconstruction for nonlinear processes using machine learning. Chem. Eng. Res. Des. 159, 248–261.
Yin, X., Qin, Y., Chen, H., Du, W., Liu, J., Huang, B., 2022. Community detection based process decomposition and distributed monitoring for large-scale processes. AlChE J. e17826.
Zhang, J., 2014. Distributed Moving Horizon State Estimation of Nonlinear Systems. University of Alberta. Master's thesis.