Contents lists available at ScienceDirect

Chemical Engineering Research and Design

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

Decentralized machine-learning-based predictive control of nonlinear processes

Scarlett Chen[a], Zhe Wu[a], Panagiotis D. Christofides[a,b,*]

[a] Department of Chemical and Biomolecular Engineering, University of California, Los Angeles, CA 90095-1592, USA
[b] Department of Electrical and Computer Engineering, University of California, Los Angeles, CA 90095-1592, USA

ARTICLE INFO

Article history:
Received 18 June 2020
Received in revised form 14 July 2020
Accepted 16 July 2020
Available online 31 July 2020

Keywords:
Recurrent neural networks
Model predictive control
Nonlinear processes
Decentralized control

ABSTRACT

This work focuses on the design of decentralized model predictive control (MPC) systems for nonlinear processes, where the nonlinear process is broken down into multiple, yet coupled subsystems and the dynamic behavior of each subsystem is described by a machine learning model. One decentralized MPC is designed and used to control each subsystem while accounting for the interactions between subsystems through feedback of the entire process state. The closed-loop stability of the overall nonlinear process network and the performance properties of the decentralized model predictive control system using machine-learning prediction models are analyzed. More specifically, multiple recurrent neural network models suited for each different subsystem need to be trained with a sufficiently small modeling error from their respective actual nonlinear process models to ensure closed-loop stability. These recurrent neural network models are subsequently used as the prediction model in decentralized Lyapunov-based MPCs to achieve efficient real-time computation time while ensuring closed-loop state boundedness and convergence to the origin. The simulation results of a nonlinear chemical process network example demonstrate the effective closed-loop control performance when the process is operated under the decentralized MPCs using the independently-trained recurrent neural network models, as well as the improved computational efficiency compared to the closed-loop simulation of a centralized MPC system.

1. Introduction

Many industrial systems, such as power distribution grids, mechanical systems and chemical processes, supply chains, and urban traffic networks, are large-scale systems that cannot be handled by using a centralized controller because the system to be controlled is too large and the control problem to be solved is too complex (Bakule, 2008). These challenges cannot be simply solved by using faster computers with larger memory. Decentralized control systems have been proposed to address the concerns associated with control of large-scale systems, including challenges posed by high dimensionality, information structure constraints, uncertainties and delays in the system Bakule (2008). The analysis of the overall system is divided into independent sub-problems that are weakly related to each other to be regulated by separate controllers, which together constitute a decentralized control structure. As such, decentralized control structures provide a practical solution to decoupling large-scale processes and reducing the computational complexity of a centralized control problem. Many recent works have been done on the subject of decentralized control (Hudon and Bao, 2012; Liu et al., 2014; Hioe et al., 2014). For example, augmented estimators for subsystems were designed in Yin et al. (2018) to form a distributed state estimation network from decentralized estimators. A quasi-decentralized control framework was developed in Sun and El-Farra (2008) where the network utilization and communication costs were minimized. With respect to earlier works, in Magni and Scattolini (2006), a stabilizing decentralized model predictive control (MPC) algorithm was presented for nonlin-

* Corresponding author.
E-mail address: pdc@seas.ucla.edu (P.D. Christofides).
https://doi.org/10.1016/j.cherd.2020.07.019

ear discrete-time systems subject to decaying disturbances. Moreover, a dynamically-coupled decentralized MPC system for large-scale linear processes subject to input constraints was presented in Alessio et al. (2011), where the global model of the process was approximated by multiple smaller subsystem models and the degree of decoupling was a tunable parameter in the design. Furthermore, a decentralized control strategy using the overlapping decomposition method and using H_∞ controllers was applied to a web transport system in Knittel et al. (2002). By only acquiring local output measurements and deciding local control inputs, the main advantages of a decentralized control scheme are the reduced communications and parallel computations.

In many practical problems, a mathematical model based on physical first-principles is not fully known or too complicated to be used as the internal model for model-based control. Nonlinear system identification and an efficient approach to such is a crucial step to designing nonlinear model predictive controllers. While there is no systematic way to parameterize general nonlinear dynamic systems, among existing techniques, the universal approximation properties of neural networks make them a powerful method for modeling nonlinear systems using data (Funahashi and Nakamura, 1993). There are many different structures of neural networks, such as feedforward neural networks (FNN), convolutional neural networks (CNN), and recurrent neural networks (RNN). RNN is itself a class of artificial neural networks (ANNs) that has proven its effectiveness in representing temporal dynamic behaviors of sequential data by taking advantage of feedback signals stored in its network units (Funahashi and Nakamura, 1993; Jin et al., 1995). Many classes and variants of RNNs have been proposed and utilized in various applications, such as model identification, speech recognition, signal processing, and intrusion detection, etc. (Debar and Dorizzi, 1992; Graves et al., 2013; Chung et al., 2014). A continuous-time recurrent neural network is developed in Al Seyab and Cao (2008) and used in the context of nonlinear model predictive control where automatic differentiation techniques are used in both training the neural network model as well as in solving the online optimization problem in the controller. A combination of FNN and RNN is used in Yan and Wang (2012) where unmodeled dynamics as well as unknown higher-order terms from the decomposition of the nonlinear system are modeled by an FNN via supervised learning, and the optimization problem of the nonlinear model predictive control is iteratively solved using a single-layer RNN called the simplified dual network. Three deep neural network structures are trained and evaluated in Ogunmolu et al. (2016) to demonstrate their effectiveness in modeling underlying characteristics of dynamical systems from input-output data. Moreover, in Wu and Christofides (2019), an ensemble of RNN models is developed and used in the design of a Lyapunov-based economic model predictive control to address economic optimality of nonlinear systems. The integration of machine-learning-based modeling methods and various advanced control architectures is a broad field with expanding research scope. Using state-of-the-art machine-learning methods to address the issues of model uncertainties in a decentralized control structure highlights the research interest of this study.

In this work, we introduce decentralized model-based control frameworks, where each decentralized controller employs a Long-Short-Term-Memory (LSTM) network – a particular class of RNN. One decentralized controller is designed and designated to one subsystem of the overall process, and each decentralized controller is designed via Lyapunov-based model predictive control (LMPC) theory (Mhaskar et al., 2006). We analyze the stability properties of the decentralized LMPC system that uses LSTM network models as the prediction model for each subsystem, and then compare the closed-loop performances of the decentralized LMPCs with those using first-principles models as the prediction model, and lastly compare with the closed-loop performance of a centralized LMPC system. The remainder of the paper is organized as follows. Section 2 presents preliminaries on notation, the general class of nonlinear systems considered, and the stabilizability assumptions. An introduction on RNN, specifically the structure and development of LSTM networks, as well as Lyapunov-based control using LSTM networks are presented in Section 3. The formulation and stability proofs of the decentralized LMPC systems using LSTM models are outlined in Section 4. In Section 5, closed-loop simulations of a two-CSTR-in-series process under the decentralized LMPC system are presented.

2. Preliminaries

2.1. Notation

Throughout the manuscript, the notation x^T is used to denote the transpose of x. $|\cdot|$ is used to denote the Euclidean norm of a vector. Set subtraction is denoted by "\", i.e., $A\backslash B:=\{x \in \mathbf{R}^n | x \in A, x \notin B\}$. A continuous function $\alpha : [0, a) \to [0, \infty)$ is said to belong to class $\mathcal{K}$ if it is strictly increasing and is zero only when evaluated at zero. The function $f(\cdot)$ is of class $\mathcal{C}^1$ if it is continuously differentiable in its domain. $L_f V(x)$ denotes the standard Lie derivative $L_f V(x) := \frac{\partial V(x)}{\partial x} f(x)$.

2.2. Class of systems

Consider a general class of continuous-time nonlinear systems in which several distinct sets of manipulated inputs are used, with each set of manipulated inputs regulating a specific subsystem of the process. The class of the overall continuous-time nonlinear system is described as follows:

$$\dot{x} = F(x, u, w) := f(x) + g(x)u + v(x)w, \quad x(t_0) = x_0 \tag{1}$$

where $x \in \mathbf{R}^n$ is the vector of all states of the nonlinear system, $u \in \mathbf{R}^m$ is the vector of all manipulated inputs of the system, and $w \in \mathbf{R}^l$ is the disturbance vector for the entire system. $f(\cdot)$, $g(\cdot)$, and $v(\cdot)$ are sufficiently smooth vector and matrix functions of dimensions $n \times 1$, $n \times m$, and $n \times l$, respectively. We refer to each subpart of the process as a subsystem. Throughout the manuscript, we consider $j = 1, \ldots, N_{sys}$ subsystems, with each subsystem j consisting of states x_j which are regulated by only u_j and potentially impacted by states in other subsystems due to coupling between subsystems. The continuous-time nonlinear dynamics of the subsystem j is described as follows:

$$\dot{x}_j = F_j(x, u_j, w) := f_j(x) + g_j(x)u_j + v_j(x)w, \quad x_j(t_0) = x_{j_0}, \quad \forall j = 1, \ldots, N_{sys} \tag{2}$$

where N_{sys} represents the number of subsystems, $x_j \in \mathbf{R}^{n_j}$ represents the state vector for subsystem j, and x represents the vector of all states $x = [x_1^T \ldots x_{N_{sys}}^T]^T \in \mathbf{R}^n$, where $n = \sum_{j=1}^{N_{sys}} n_j$.

$u_j \in \mathbf{R}^{m_j}$ is the set of manipulated input vectors for each subsystem j, which together constitute the vector of all manipulated inputs $u \in \mathbf{R}^m$ with $m = \sum_{j=1}^{N_{sys}} m_j$. The manipulated input vector constraints are defined by $u_j \in U_j := \{u_{j_i}^{min} \le u_{j_i} \le u_{j_i}^{max}, i = 1, \ldots, m_j\} \subset \mathbf{R}^{m_j}, \forall j = 1, \ldots, N_{sys}$. Therefore, the set that bounds the manipulated input vector u for the overall system is denoted by U, which is the union of all U_j, $j = 1, \ldots, N_{sys}$. $w \in W$ is the disturbance vector with $W := \{w \in \mathbf{R}^l | |w| \le w_m, w_m \ge 0\}$. $f_j(\cdot)$, $g_j(\cdot)$, and $v_j(\cdot)$ are sufficiently smooth vector and matrix functions of dimensions $n_j \times 1$, $n_j \times m_j$, and $n_j \times l$, respectively. The initial time t_0 is taken to be zero ($t_0 = 0$). We assume that $f_j(0) = 0$, $\forall j = 1, \ldots, N_{sys}$, thus, the origin is a steady-state of the nominal system of Eq. (1) (i.e., $u(t) \equiv 0, w(t) \equiv 0$). Therefore, we have $(x_s, u_s) = (0, 0)$, where x_s and u_s are the steady-state state and input vectors, respectively.

2.3. *Stability assumptions*

Depending on the partitioning of the overall large-scale system, there may exist interacting dynamics between the subsystems, where the states of one subsystem may be impacted by the states of other subsystems. Accounting for the coupling effects between these subsystems, we assume that there exist stabilizing control laws $u_j = \Phi_j(x) \in U_j$ which regulate the individual subsystems $j = 1, \ldots, N_{sys}$ and will be applied to the control actuators in the respective subsystems such that the origin of the overall system of Eq. (1) with $w(t) \equiv 0$ is rendered exponentially stable. This implies that there exists a $\mathcal{C}^1$ control Lyapunov function $V(x)$ such that the following inequalities hold for all $x \in \mathbf{R}^n$ in an open neighborhood D around the origin:

$$c_1|x|^2 \le V(x) \le c_2|x|^2, \tag{3a}$$

$$\frac{\partial V(x)}{\partial x} F(x, \boldsymbol{\Phi}(x), 0) \le -c_3|x|^2, \tag{3b}$$

$$\left| \frac{\partial V(x)}{\partial x} \right| \le c_4|x| \tag{3c}$$

where c_1, c_2, c_3 and c_4 are positive constants. $F(x, u, w)$ represents the overall nonlinear system of Eq. (1). $\boldsymbol{\Phi}(x) = [\Phi_1(x)^T \cdots \Phi_{N_{sys}}(x)^T]^T$ represents the vector containing the candidate controllers for each subsystem j, i.e., $\Phi_j(x) \in \mathbf{R}^{m_j}$, for $j = 1, \ldots, N_{sys}$. The candidate controller for each subsystem j is given in the following form:

$$\phi_{j_i}(x) = \begin{cases} -\dfrac{p + \sqrt{p^2 + q^4}}{q^T q} q & \text{if} \quad q \neq 0 \\ 0 & \text{if} \quad q = 0 \end{cases} \tag{4a}$$

$$\Phi_{j_i}(x) = \begin{cases} u_{j_i}^{min} & \text{if} \quad \phi_{j_i}(x) < u_{j_i}^{min} \\ \phi_{j_i}(x) & \text{if} \quad u_{j_i}^{min} \le \phi_{j_i}(x) \le u_{j_i}^{max} \\ u_{j_i}^{max} & \text{if} \quad \phi_{j_i}(x) > u_{j_i}^{max} \end{cases} \tag{4b}$$

where p denotes $\frac{\partial V(x)}{\partial x_j} f_j(x)$ and q denotes $\frac{\partial V(x)}{\partial x_j} g_{j_i}(x)$. Here, $f_j \in \mathbf{R}^{n_j}$ and $g_{j_i} \in \mathbf{R}^{n_j \times m_j}$ for $j = 1, \ldots, N_{sys}$, and $i = 1, \ldots, m_j$ for subsystem j corresponding to the vector of control actions $\Phi_j(x) \in \mathbf{R}^{m_j}$. It should be noted that the control Lyapunov function $V(x)$ can be a linear combination of multiple control Lyapunov functions V_j, where each V_j is designated for the subsystem j and is a function of x_j only, i.e., $V(x) = \sum_{j=1}^{N_{sys}} V_j(x_j)$. Thus, $\frac{\partial V(x)}{\partial x_j} = \frac{\partial V_j(x_j)}{\partial x_j}$, $\forall j = 1, \ldots, N_{sys}$, and the time-derivative of V can be represented as $\dot{V}(x) = L_f V + L_g V u = \sum_{j=1}^{N_{sys}} \frac{\partial V(x)}{\partial x_j} (f_j + \sum_{i=1}^{m_j} g_{j_i} u_{j_i})$. $\phi_{j_i}(x)$ of Eq. (4a) represents the ith component of the control law $\phi_j(x)$. $\Phi_{j_i}(x)$ of Eq. (4) represents the ith component of the saturated control law $\Phi_j(x)$ that accounts for the input constraints $u_j \in U_j$ for subsystem j. Note that the candidate control law $\Phi_j(x)$ is calculated based on the nonlinear dynamics of the subsystem j of Eq. (2), and the set of candidate control laws for the overall system is denoted as $\boldsymbol{\Phi}(x) = [\Phi_1(x)^T \ldots \Phi_{N_{sys}}(x)^T]^T \in U$, which together can render the overall system of Eq. (1) with $w \equiv 0$ exponentially stable.

Based on Eq. (3), we can first characterize a region where the time-derivative of the Control Lyapunov function V is rendered negative definite under the controllers $\boldsymbol{\Phi}(x) = [\Phi_1(x)^T \cdots \Phi_{N_{sys}}(x)^T]^T \in U$ as $D = \{x \in \mathbf{R}^n | \dot{V}(x) = L_f V + L_g V u \le -c_3|x|^2, u = \boldsymbol{\Phi}(x) \in U\} \cup \{0\}$. Subsequently, the closed-loop stability region Ω_ρ for the nonlinear system of Eq. (1) is defined as a level set of V, which is inside D: $\Omega_\rho := \{x \in D | V(x) \le \rho\}$, where $\rho > 0$ and $\Omega_\rho \subset D$. Also, the Lipschitz property of $F(x, u, w)$ combined with the bounds on u and w imply that there exist positive constants M, L_x, L_w, L_x', L_w' such that the following inequalities hold for all $x, x' \in \Omega_\rho, u \in U$ and $w \in W$:

$$|F(x, u, w)| \le M \tag{5a}$$

$$|F(x, u, w) - F(x', u, 0)| \le L_x|x - x'| + L_w|w| \tag{5b}$$

$$\left| \frac{\partial V(x)}{\partial x} F(x, u, w) - \frac{\partial V(x')}{\partial x} F(x', u, 0) \right| \le L_x'|x - x'| + L_w'|w| \tag{5c}$$

Note that the controller $\Phi_{j_i}(x)$, $i = 1, \ldots, m_j$ of Eq. (4) is a candidate stabilizing control law for the $u_j \in \mathbf{R}^{m_j}$ inputs regulating subsystem j accounting for the interacting dynamics between subsystems, and the nominal system of Eq. (1) is rendered exponentially stable under the set of all such stabilizing control laws for N_{sys} subsystems, $\boldsymbol{\Phi}(x) \in U \subset \mathbf{R}^m$.

Remark 1. Before we proceed, it is important to elaborate on the structure and differences between decentralized and distributed control systems. Both decentralized and distributed control systems aim to alleviate the computational complexity and effort of a centralized control problem used to regulate multiple subsystems. More specifically, there exists intercontroller communication in a distributed control system, sharing information on the control actions calculated by each local controller. This means that additional communication channels need to be established between the local controllers, which contributes to additional communication network traffic. Correspondingly, the computation time may increase (e.g., in a sequential distributed control system, the controllers need to wait for the calculated results from the precedent controllers and are executed in a sequential manner; in an iterative distributed control system, while the calculated control actions are exchanged simultaneously past the first iteration, the additional iterations also contribute to longer computation time). On the other hand, in a decentralized control system, each local controller concurrently calculates the optimal control actions based on state feedback measurements, thereby significantly reducing the computational time for control action calculation. However, without information on the control actions taken by the other controllers, the cost function minimized by each local controller only includes

the relevant control actions and the corresponding predicted states of the respective subsystem, thereby yielding inferior closed-loop performance.

Remark 2. It is important to note that, regardless of the open-loop stability property of the equilibrium point in which the process is operated at, the stabilizability assumption ensures the existence of a controller that achieves desired decay rate of the closed-loop state to the origin and the Lyapunov stability constraints in MPC ensure closed-loop stability and closed-loop state convergence to the origin at a rate that is as fast or faster than the one achieved by a Lyapunov-based controller. If the process steady-state of operation is open-loop stable, the closed-loop stability region may be bigger given the limitation imposed in the control actions by the control actuator constraints.

3. Long short-term memory neural network

It has been demonstrated in numerous examples in the literature that recurrent neural network (RNN) models are capable of modeling dynamic behaviors of time-series data and are an effective method to represent nonlinear processes (Su et al., 1992; Funahashi and Nakamura, 1993; Jin et al., 1995). With the use of feedback loops in the network, RNNs store outputs derived from past inputs, and use these previous output information together with current inputs to obtain a more accurate prediction of the current output. With sufficient number of neurons in the RNN model, it can be shown that RNNs are capable of approximating any nonlinear dynamic systems on compact subsets of the state-space for finite time based on the universal approximation theorem (Sontag, 1992; Kosmatopoulos et al., 1995).

There are many types and variations of recurrent neural networks suited for different purposes. Classic RNNs have neurons which pass on historical information across time, and they are prevalent in the fields of natural language processing and speech recognition (Chung et al., 2014). However, they face difficulty of accessing information from a long time ago due to the vanishing/exploding gradient phenomena, and they cannot consider any future input for the prediction of the current state since the feedback loops are in the forward direction in time only. To this end, many variations of RNNs have been proposed. For example, in order to address the vanishing/exploding gradient problem, specific gates are used in each repeating network unit to assess and regulate the information transferred down the network. More specifically, Long Short-Term Memory (LSTM) networks use three gates (the forget gate, the input gate, and the output gate) to protect and control the memory cell state, thus information will be stored and remembered for long periods of time, making LSTM networks well suited for data sequences that exhibit long time lags between relevant data points (Hochreiter and Schmidhuber, 1997; Schmidhuber, 2015). Gated Recurrent Unit (GRU), which is a variation of the LSTM unit, combines the gating functions of the input gate and the forget gate in an LSTM unit, thereby significantly reducing the training and execution time (Dey and Salem, 2017). On the other hand, Bidirectional RNN (BRNN) was proposed in Schuster and Paliwal (1997) where a hidden layer with forward connections in time and a hidden layer with backward connections in time were used together such that both past outputs and future outputs were used along with current inputs to predict the current output. Moreover, Deep RNN (DRNN) stacks multiple hidden recurrent layers on top of each other, where each hidden state is continuously passed to both the next time step of the current layer, as well as the current time step of the next layer (Schmidhuber, 2015). LSTM networks could also adopt the configurations of BRNN and DRNN for more complex applications. For each subsystem j, $j = 1, \ldots, N_{sys}$, we develop a classic LSTM model to approximate the continuous-time nonlinear processes of each subsystem of Eq. (2). Combining N_{sys} such LSTM network models, the nonlinear process of Eq. (1) can be represented as follows:

$$\dot{\hat{x}} = F_{nn}(\hat{x}, u) = [\dot{\hat{x}}_1^T \cdots \dot{\hat{x}}_{N_{sys}}^T]^T \tag{6}$$

and the LSTM network model for each subsystem j, modeling the nonlinear dynamics of Eq. (2), is represented in the following form:

$$\dot{\hat{x}}_j = F_{nn_j}(\hat{x}, u_j) := \lambda_j A_j \hat{x} + \Theta_j^T y_j, \quad j = 1, \ldots, N_{sys} \tag{7}$$

where $\hat{x} \in \mathbf{R}^n$ is the predicted state vector for the overall system, $\hat{x}_j \in \mathbf{R}^{n_j}$ is the predicted state vector given by the LSTM model for subsystem j, and $u_j \in \mathbf{R}^{m_j}$ is the manipulated input vector for subsystem j. $y_j = [y_1 \cdots y_{n_j+m_j+1}]^T = [H(\hat{x}_1) \cdots H(\hat{x}_{n_j}) \;\; u_1 \ldots u_{m_j} \;\; 1]^T \in \mathbf{R}^{n_j+m_j+1}$ is a vector of both the network states $\hat{x}_j$ and the inputs u_j, where $H(\cdot)$ represents a series of interacting nonlinear activation functions in each LSTM unit. $\lambda_j \in \mathbf{R}^{n_j \times n}$ is a coefficient matrix representing the interacting dynamic behavior between the states of the overall system x and the states of the subsystem j, x_j. A_j is a diagonal coefficient matrix, where $A_j = diag\{-\alpha_{j_1} \cdots -\alpha_{j_n}\} \in \mathbf{R}^{n \times n}$, and $\Theta_j = [\theta_1 \cdots \theta_{n_j+m_j+1}] \in \mathbf{R}^{(n_j+m_j+1) \times n_j}$ with $\theta_i = \beta_{j_i}[\omega_{i1} \ldots \omega_{i(n_j+m_j)} \;\; b_{j_i}]$, $i = 1, \ldots, n_j$. α_{j_i} and β_{j_i} are constants, and ω_{ik} is the weight connecting the kth input to the ith neuron where $i = 1, \ldots, n_j$ and $k = 1, \ldots, (n_j + m_j)$, and b_{j_i} is the bias term for $i = 1, \ldots, n_j$. We assume that α_{j_i} are positive constants such that each state in the vector $\hat{x}_j$ is bounded-input bounded-state stable. We use x_j to represent the state of the actual nonlinear subsystem of Eq. (2) and use $\hat{x}_j$ for the state of the subsystem j modeled by the LSTM network of Eq. (7). The basic architecture of an LSTM network is illustrated in Fig. 1. The matrix of input sequences to the LSTM network is denoted by $m \in \mathbf{R}^{(n+m_j) \times T}$ containing the measured states $x \in \mathbf{R}^n$ at each model predictive controller (MPC) sampling period Δ and the manipulated inputs $u_j \in \mathbf{R}^{m_j}$ to be optimized over the next sampling period Δ, both with a sequence length of T. The matrix of the network output sequences is the predicted states for subsystem j denoted by $\hat{x}_j \in \mathbf{R}^{n_j \times T}$. From each repeating LSTM unit, the network state that is passed onto the next LSTM unit in the unfolded sequence is the hidden state, denoted by $h(1), \ldots, h(T)$, where the number of internal states T (i.e., the number of repeating LSTM units) corresponds to the length of the input sequence. The LSTM network calculates the predicted output sequence $\hat{x}_j$ from the input sequence m by computing the following calculations iteratively from $k = 1$ to $k = T$:

$$i(k) = \sigma(\omega_i^m m(k) + \omega_i^h h(k-1) + b_i) \tag{8a}$$

$$f(k) = \sigma(\omega_f^m m(k) + \omega_f^h h(k-1) + b_f) \tag{8b}$$

$$c(k) = f(k)c(k-1) + i(k)\tanh(\omega_c^m m(k) + \omega_c^h h(k-1) + b_c) \tag{8c}$$

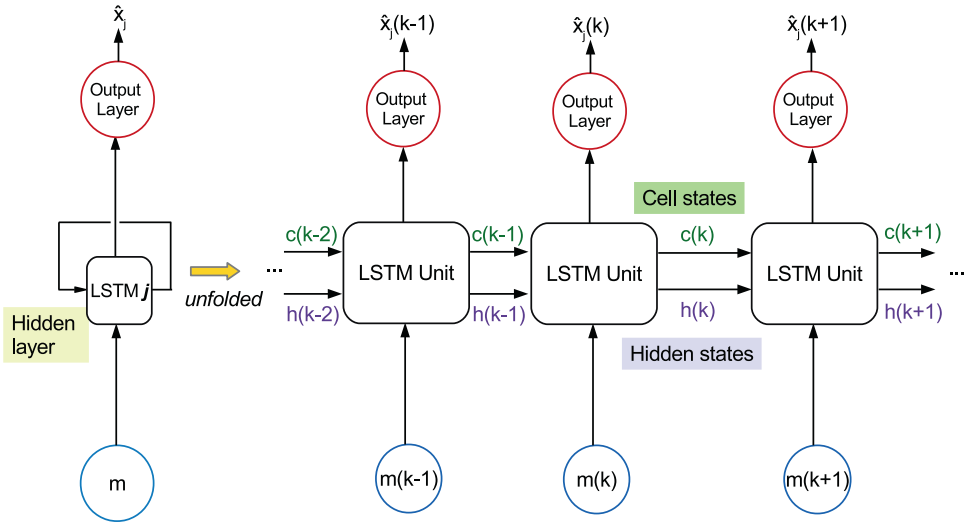

Fig. 1 – A long short-term memory (LSTM) recurrent neural network for subsystem *j* and the series of its unfolded structure, where *m* is the input vector consisting of the state measurement *x* at each MPC sampling period and the control action u_j to be optimized over the next sampling period, *c* is the cell state vector, *h* is the hidden state vector, and $\hat{x}_j$ is the output vector.

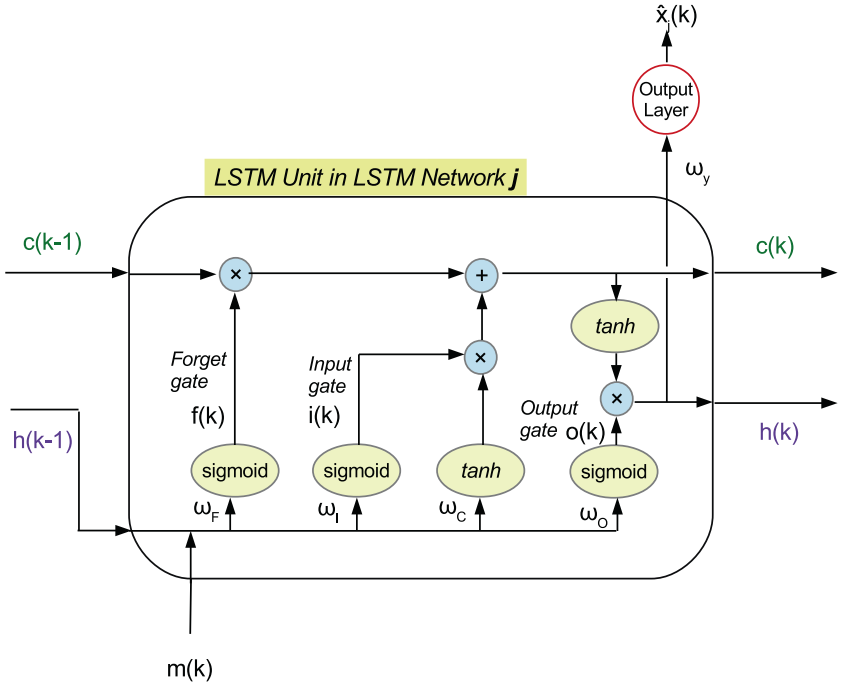

Fig. 2 – The internal structure of an LSTM unit inside the LSTM network *j* where the past cell state vector $c(k-1)$, past hidden state vector $h(k-1)$, and the current input vector $m(k)$ are used to obtain $c(k)$, $h(k)$, and the network output vector $\hat{x}_j(k)$ for subsystem *j* via the input gate, the forget gate, the output gate, and an output layer.

$$o(k) = \sigma(\omega_o^m m(k) + \omega_o^h h(k-1) + b_o) \tag{8d}$$

$$h(k) = o(k)\tanh(c(k)) \tag{8e}$$

$$\hat{x}_j(k) = \omega_y h(k) + b_y \tag{8f}$$

where $m(k)$ denotes the kth element in the input sequence, $h(k)$ and $\hat{x}_j(k)$ are the internal state and the output computed by the kth LSTM unit in the unfolded sequence, respectively. $\sigma(\cdot)$ is the sigmoid activation function, $\tanh(\cdot)$ is the hyperbolic tangent activation function. ω_y and b_y denote the weight matrix and bias vector for the output, respectively. The outputs from the input gate, the forget gate, and the output gate are represented by $i(k), f(k), o(k)$, respectively; ω_i^m, ω_i^h, ω_f^m, ω_f^h, ω_o^m, ω_o^h are the weight matrices for the input vector m and the hidden state vector h within the input gate, the forget gate, and the output gate respectively, and b_i, b_f, b_o represent the bias vectors within each of the three gates, respectively. $c(k)$ is the cell state which stores memory and passes information down the LSTM units, with ω_c^m, ω_c^h and b_c representing the weight matrices for the input and hidden state vectors, and the bias vector, respectively. The series of interacting nonlinear functions carried out in each LSTM unit, outlined in Eq. (8), can be represented by $H(\cdot)$. The internal structure of an LSTM unit showing the gating functions is shown in Fig. 2.

Remark 3. Since the class of nonlinear systems we consider in this work is continuous-time, the stability analysis

of the data-based LSTM model is also conducted using the continuous-time representation. However, given the inherent internal structure of the LSTM model and that the LSTM model is constructed based on data arranged in sequences of uniform intervals, the LSTM model works as a discrete-time model with a uniform sampling time. While the LSTM model and the first-principles model of the process are both represented as continuous-time models, the computation of the LSTM model occurs at discrete time instants, similar to how the first-principles continuous-time model in the form of ordinary differential equations can be numerically integrated via explicit Euler method, but with a different integration time step.

The continuous-time nonlinear system of Eq. (1) is operated under the proposed decentralized Lyapunov-based model predictive control (LMPC) system in a sample-and-hold manner, where the feedback measurement of the closed-loop state x for the nonlinear system is received by the designated local controller, LMPC j, every sampling period Δ. Using numerical integration methods such as the explicit Euler method, we can obtain the state information of the simulated nonlinear process with an integration time step of h_c. We develop the LSTM network model for each subsystem j such that it can be used as the prediction model in the decentralized LMPC; to this end, the total prediction period of the LSTM network model is also set to be Δ (i.e., the last output state vector is obtained at the end of every Δ), and the time interval between two consecutive internal states h in the LSTM network for subsystem j can be chosen as a multiple q_{nn} of the integration time step h_c, with the minimum time interval being the integration time step with $q_{nn} = 1$. Depending on the choice of q_{nn}, the number of internal states T will follow $T = \frac{\Delta}{q_{nn} \cdot h_c}$. Therefore, given that the input sequence is fed to the LSTM network at $t = t_k$, the LSTM network provides a sequence of T future predicted states following Eq. (8), with the last network output vector corresponding to the predicted state $\hat{x}_j(t)$ at $t = t_k + \Delta$. Since input sequences with a length of T are needed to produce T number of internal states, the input time-series samples will be of a fixed length of T. Fig. 3 illustrates the different time steps used in this work, i.e., the integration time step for the numerical simulation (h_c), the time interval between the internal states in the developed LSTM network models ($q_{nn} \times h_c$), and the sampling period in the model predictive control algorithm (Δ).

To collect training data for developing an LSTM network for each subsystem j, we first discretize the targeted region in state-space with sufficiently small intervals; then, open-loop simulations are carried out where the first-principles model of Eq. (2) is numerically integrated at a fixed integration time step of h_c, and these simulations are conducted for various initial conditions $x_0 \in \Omega_\rho$ under various input sequences $u_j \in U_j, \quad j = 1, \ldots, N_{sys}$ implemented in a sample-and-hold manner with a sampling period of Δ. We obtain enough samples of trajectories for finite time to sweep over all the values that (x, u) can take in the targeted region in state-space. Then, we extract the state measurements every $q_{nn} \times h_c$ interval to use as the target predicted internal states during training. As there are T such internal states within each controller sampling period $\Delta = T \times q_{nn} \times h_c$, the last internal state corresponds to the output predicted state at the end of every Δ. The time interval between two data points in the sample of time-series sequence is $q_{nn} \times h_c$, and it corresponds to the time interval between two consecutive LSTM units in the LSTM network. We separate the dataset into the corresponding input and output vectors for each subsystem j, i.e., the input vector includes data on all states of the overall system x and the distinct set of manipulated inputs for the subsystem, u_j, and the output vector includes the states of the subsystem, x_j. The generated dataset is then divided into training and validation sets.

Remark 4. We could use all the numerically integrated state values every h_c step (i.e., $q_{nn} = 1$) as the internal states in the LSTM network, however, this would increase the computational load associated with training and implementing the model. Therefore, we choose a q_{nn} value such that there are sufficient state values within each sampling period of the controller Δ to capture the dynamic state evolution while not overburdening the computational effort. While the control actions are computed every sampling period Δ (i.e., the model predictive controllers are executed once every Δ), the objective function of the optimization problem of the model predictive controller includes the integral of all the predicted states over the prediction horizon of $N \times \Delta$, which includes all the predicted internal states intervaled at $q_{nn} \times h_c$ given by the LSTM model.

With sufficient data in the training dataset, the LSTM network is developed using an application program interface which contains open-source neural network libraries, e.g., *Keras*. The optimal parameter matrix of the LSTM network, $\Gamma*$, which includes the network parameters ω_i, ω_f, ω_c, ω_o, ω_y, b_i, b_f, b_c, b_o, b_y, is optimized by minimizing the mean absolute percentage error between $x_j(t)$ and $\hat{x}_j(t)$ for each subsystem j. The minimization of the state error is carried out using the adaptive moment estimation optimizer, i.e., *Adam* in Keras, in which the gradient of the error cost function is calculated using back-propagation. A constraint on the modeling error ν_j for each LSTM network j is imposed during training, $|\nu_j| = |F_j(x, u_j, 0) - F_{nn_j}(x, u_j)| \leq \gamma_j |x_j|$, with $\gamma_j > 0$, such that the modeling error for the overall nonlinear system is sufficiently small, i.e., $|\nu| = \sum_{j=1}^{N_{sys}} |\nu_j| \leq \gamma |x|$, $\gamma > 0$. With a sufficiently small modeling error, the trained LSTM network model can adequately represent the nonlinear process of subsystem j and can be used in the proposed model predictive controller to stabilize the actual nonlinear process of Eq. (1) at its steady-state with guaranteed stability properties.

The modeling error can be numerically approximated using the forward finite difference method. Note that the time interval $q_{nn} \cdot h_c$ between two LSTM memory units is given by the time interval between two consecutive time-series data points in the training data sequences. Since the LSTM network provides a predicted state at each time interval $q_{nn} \cdot h_c$ calculated by each LSTM memory unit, similarly to how we can use numerical integration methods to obtain the state at the same time instance using the continuous-time model of Eq. (1), we can use the predicted states from the LSTM network to compare with the predicted states from the nonlinear model of Eq. (1) to assess the modeling error. The time derivative of the LSTM predicted state $\hat{x}_j(t)$ at $t = t_k$ can be approximated by $\dot{\hat{x}}_j(t_k) = F_{nn_j}(x(t_k), u_j) \approx \frac{\hat{x}_j(t_k + q_{nn} \cdot h_c) - \hat{x}_j(t_k)}{q_{nn} \cdot h_c}$. The time derivative of the actual state $x_j(t)$ at $t = t_k$ can be approximated by $\dot{x}_j(t_k) = F_j(x(t_k), u_j, 0) \approx \frac{x_j(t_k + q_{nn} \cdot h_c) - x_j(t_k)}{q_{nn} \cdot h_c}$. At time $t = t_k$, $\hat{x}_j(t_k) = x_j(t_k)$, the constraint $|\nu_j| \leq \gamma |x_j|$ for the LSTM network of subsystem j can be written as follows:

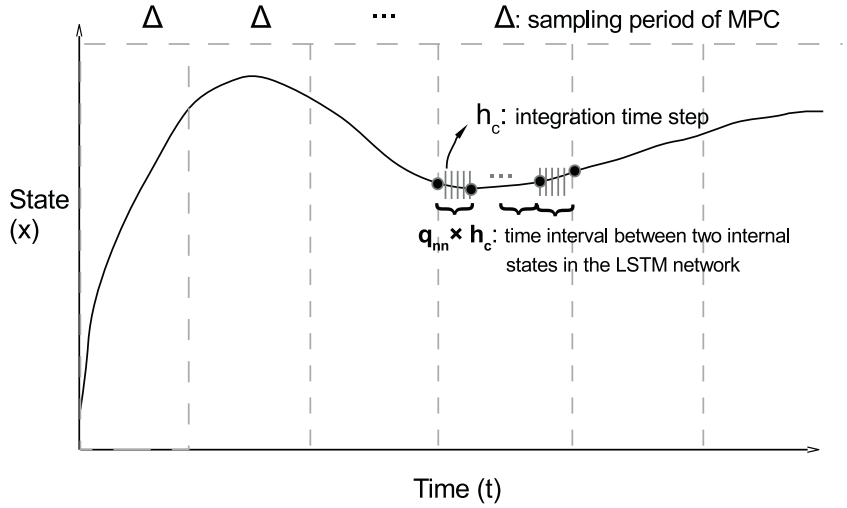

Fig. 3 – An illustration showing the integration time step h_c used in the numerical integration of the process states x, the time interval between internal states in the LSTM network $q_{nn} \times h_c$, and the sampling period for the model predictive controller.

$$|\nu_j| = |F_j(x(t_k), u_j, 0) - F_{nn_j}(x(t_k), u_j)| \tag{9a}$$

$$\approx |\frac{x_j(t_k + q_{nn} \cdot h_c) - \hat{x}_j(t_k + q_{nn} \cdot h_c)}{q_{nn} \cdot h_c}| \tag{9b}$$

$$\leq \gamma_j |x_j(t_k)| \tag{9c}$$

which will be satisfied if $|\frac{x_j(t_k+q_{nn} \cdot h_c) - \hat{x}_j(t_k+q_{nn} \cdot h_c)}{x_j(t_k)}| \leq \gamma_j \cdot q_{nn} \cdot h_c$. Therefore, we can use the mean absolute percentage error between the predicted states $\hat{x}_j$ and the targeted states x_j in the training data to assess the modeling error of the LSTM model for subsystem j. To avoid over-fitting of the LSTM models, the training process is terminated once the modeling error falls below the desired threshold and the error on the validation set stops decreasing. The modeling error of the overall system of Eq. (6) can be represented as a sum of the modeling errors of the individual LSTM networks, i.e., $|\nu| = \sum_{j=1}^{N_{sys}} |\nu_j|$.

Remark 5. For the theoretical analysis and the construction of the LSTM model in this work, we assume that a first-principles model for the nominal process is available in the form of ordinary differential equations. In the case that such models are not available, the modeling error approximation in Eq. (9) can be calculated based on real plant data.

3.1. *Lyapunov-based control using LSTM models*

After obtaining a sufficiently small modeling error between the trained LSTM network and the actual nonlinear model of Eq. (1) for subsystem j, we assume that there exists a set of stabilizing feedback controllers $u = \boldsymbol{\Phi}_{nn}(x) \in U$, where $\boldsymbol{\Phi}_{nn}(x) = [\Phi_{nn_1}(x)^T \ldots \Phi_{nn_{N_{sys}}}(x)^T]^T$, that can render the origin of the LSTM model of Eq. (6) exponentially stable in the sense that there exists a $\mathcal{C}^1$ control Lyapunov function $\hat{V}(x)$ such that the following inequalities hold for all $x \in \mathbf{R}^n$ in an open neighborhood around the origin $\hat{D}$:

$$\hat{c}_1|x|^2 \leq \hat{V}(x) \leq \hat{c}_2|x|^2, \tag{10a}$$

$$\frac{\partial \hat{V}(x)}{\partial x} F_{nn}(x, \boldsymbol{\Phi}_{nn}(x)) \leq -\hat{c}_3|x|^2, \tag{10b}$$

$$\left|\frac{\partial \hat{V}(x)}{\partial x}\right| \leq \hat{c}_4|x| \tag{10c}$$

where $\hat{c}_1$, $\hat{c}_2$, $\hat{c}_3$, $\hat{c}_4$ are positive constants, and $F_{nn}(x, u)$ is represented by the system of LSTM models Eq. (6). While there are many forms that the Lyapunov function $\hat{V}$ can take, it is a function that captures all the states of the overall system of Eq. (6). Similar to the Lyapunov function $V(x)$ for the nonlinear system of Eq. (1) introduced in Section 2.3, one example of $\hat{V}$ is a linear combination of the Lyapunov functions of the states of individual subsystems, i.e., $\hat{V}(x) = \sum_{j=1}^{N_{sys}} \hat{V}_j(x_j)$. Here, we assume that $\hat{V}_j(\cdot)$ is a function of x_j only.

Similar to the characterization method of the closed-loop stability region Ω_ρ for the nonlinear subsystem of Eq. (1), we first search the entire state-space to characterize a set of states $\hat{D}$ where the following inequality holds: $\dot{\hat{V}}(x) = \frac{\partial \hat{V}(x)}{\partial x} F_{nn}(x, u) \leq -c_3|x|^2$, $u = \boldsymbol{\Phi}_{nn}(x) \in U$. The closed-loop stability region for the LSTM model of Eq. (6) is defined as a level set of Lyapunov function inside $\hat{D}$: $\Omega_{\hat{\rho}} := \{x \in \hat{D} | \hat{V}(x) \leq \hat{\rho}\}$, where $\hat{\rho} > 0$. Starting from $\Omega_{\hat{\rho}}$, the origin of the LSTM model of Eq. (6) can be rendered exponentially stable under the set of controllers $u_j = \Phi_{nn_j}(x) \in U_j$, for $j = 1, \ldots, N_{sys}$. The assumptions of Eq. (10) are the stabilizability conditions for the LSTM model of Eq. (6), and they are analogous to the ones of Eq. (3) for the general class of nonlinear systems of Eq. (1) since the LSTM model of Eq. (7) for each subsystem can be written in the form of Eq. (2) (i.e., for each subsystem, $\dot{\hat{x}}_j = \hat{f}_j(\hat{x}) + \hat{g}_j(\hat{x})u_j$, where $\hat{f}_j(\cdot)$ and $\hat{g}_j(\cdot)$ are obtained from coefficient matrices λ_j, A_j and Θ_j in Eq. (7)). However, due to the complexity of the LSTM structure and the interactions of the nonlinear activation functions, $\hat{f}_j$ and $\hat{g}_j$ may be hard to compute explicitly. For computational convenience, we will numerically approximate $\hat{f}_j$ and $\hat{g}_j$ using the forward finite difference method similar to Eq. (9). At $t = t_k$, we compute two sets of predicted states $\hat{x}_j(t)$ at $t = t_k + q_{nn} \cdot h_c$ using the LSTM network under the sample-and-hold implementation of two sets of inputs respectively: first, we use $u_{j_i} = 0, \quad \forall i = 1, \ldots, m_j$, applied in a sample-and-hold manner for the time interval $[t_k, t_k + q_{nn} \cdot h_c)$ to obtain the predicted

state $\hat{x}_j(t)$ at $t = t_k + q_{nn} \cdot h_c$ under zero inputs, denoted as $\int_{t_k}^{t_k+q_{nn}\cdot h_c} F_{nn_j}(x, 0)dt$; then, we use $u_j(t_k) = [0 \ldots u_{j_i}(t_k) \ldots 0] \in \mathbf{R}^{m_j}$, where $u_{j_i}^{\min} \le u_{j_i}(t_k) \le u_{j_i}^{\max}$, $u_{j_i}(t_k) \neq 0$, applied in a sample-and-hold manner for the interval $[t_k, t_k + q_{nn} \cdot h_c)$ to obtain the predicted state $\hat{x}_j(t)$ at $t = t_k + q_{nn} \cdot h_c$ under a non-zero input $u_{j_i}(t_k)$, denoted as $\int_{t_k}^{t_k+q_{nn}\cdot h_c} F_{nn_j}(x, u_j(t_k))dt$. Therefore, given the state measurement $x(t_k)$ at $t = t_k$, $\hat{f}_j$ and $\hat{g}_{j_i}$ can be numerically approximated as follows:

$$\hat{f}_j(t_k) = \frac{\int_{t_k}^{t_k+q_{nn}\cdot h_c} F_{nn_j}(x, 0)dt - x(t_k)}{q_{nn} \cdot h_c} \tag{11a}$$

$$\hat{g}_{j_i}(t_k) = \frac{\int_{t_k}^{t_k+q_{nn}\cdot h_c} F_{nn_j}(x, u_j(t_k))dt - \int_{t_k}^{t_k+q_{nn}\cdot h_c} F_{nn_j}(x, 0)dt}{q_{nn} \cdot h_c \cdot u_{j_i}(t_k)} \tag{11b}$$

Here, $q_{nn} \cdot h_c$ is the time interval between two internal states within the LSTM network; thus, the predicted state $\hat{x}(t_k + q_{nn} \cdot h_c)$ represented by the integral $\int_{t_k}^{t_k+q_{nn}\cdot h_c} F_{nn_j}(x, \cdot)dt$ in Eq. (11), is the first internal state of the LSTM network. After obtaining $\hat{f}_j$ and $\hat{g}_{j_i}$, $i = 1, \ldots, m_j$, the stabilizing control law $\Phi_{nn_j}(x_j)$ can be computed similarly as in Eq. (4) and repeated for all subsystems $j = 1, \ldots, N_{sys}$, with $\hat{f}_j$ and $\hat{g}_{j_i}$ replacing f_j and g_{j_i} respectively. Subsequently, $\dot{\hat{V}}$ can also be computed using the approximated values of $\hat{f}_j$ and $\hat{g}_{j_i}$.

Since the dataset for developing the LSTM model for subsystem j is generated from open-loop simulations for $x \in \Omega_\rho$, $u_j \in U_j$, the closed-loop stability region of the LSTM system is a subset of the closed-loop stability region of the actual nonlinear system, i.e., $\Omega_{\hat{\rho}} \subseteq \Omega_\rho$. Moreover, there exist positive constants M_{nn} and L_{nn} such that the following inequalities hold for all $x, x' \in \Omega_{\hat{\rho}}$, $u \in U$:

$$|F_{nn}(x, u)| \le M_{nn} \tag{12a}$$

$$\left| \frac{\partial \hat{V}(x)}{\partial x} F_{nn}(x, u) - \frac{\partial \hat{V}(x')}{\partial x} F_{nn}(x', u) \right| \le L_{nn}|x - x'| \tag{12b}$$

Remark 6. Depending on how the overall system of Eq. (1) is partitioned, the extent of the coupling effect between the different subsystems j may be different. The Lyapunov-based control law $\Phi_{nn_j}(x(t))$ for each subsystem j are designed accounting for the coupling effect between the partitioned subsystems, and $\boldsymbol{\Phi}_{nn}(x)$, which is the vector containing all subsystem control laws, can render the origin of the LSTM model of Eq. (6) exponentially stable for all $\hat{x}_0 = x_0 \in \Omega_{\hat{\rho}}$.

4. Decentralized LMPC using LSTM models

When the optimization problem of a centralized MPC is too complex to solve within a reasonable time period (i.e., the sampling period), the control problem may be decoupled into smaller local optimization problems that are solved in separate processors/controllers to achieve improved computational efficiency. In a decentralized LMPC system, there is no communication between the different local controllers, therefore each controller does not have any knowledge on the control actions calculated by the other controllers. A schematic diagram showing the decentralized LMPC architecture of a process consisting of N_{sys} subsystems is shown in Fig. 4. Consider $j = 1, \ldots, N_{sys}$ subsystems, each with distinct sets of states x_j and manipulated inputs u_j. We design separate LMPCs, each designated for one subsystem to compute the respective control actions. The trajectories of control actions computed by LMPC j are denoted by u_{d_j}. Each decentralized LMPC may receive full-state feedback measurements, but they only have information on the dynamic behavior of their respective subsystem. Therefore, separate LSTM models are developed, one for each subsystem j. For instance, LMPC j uses the LSTM network model developed for subsystem j as its prediction model to predict the states of subsystem j (i.e., x_j), and calculates the control actions u_{d_j} which are applied to the corresponding control actuators in subsystem j. To inherit the stability properties of Φ_{nn_j}, $j = 1, \ldots, N_{sys}$, the optimized control actions u_{d_j} must satisfy a Lyapunov-based contractive constraint that guarantees a minimum decrease rate of the Lyapunov function $\hat{V}$. The mathematical formulation of each decentralized LMPC j, $j = 1, \ldots, N_{sys}$, is shown as follows:

$$\mathcal{J}_j = \min_{u_{d_j} \in S(\Delta)} \int_{t_k}^{t_{k+N}} L(\tilde{x}_j(t), u_{d_j}(t))dt \tag{13a}$$

$$\text{s.t.} \quad \dot{\tilde{x}}_j(t) = F_{nn_j}(\bar{x}(t), u_{d_j}(t)) \tag{13b}$$

$$\bar{x}(t) = [x_1(t_k)^T \ldots x_{j-1}(t_k)^T \tilde{x}_j(t)^T x_{j+1}(t_k)^T \ldots x_{N_{sys}}(t_k)^T]^T \tag{13c}$$

$$u_{d_j}(t) \in U_j, \quad \forall \ t \in [t_k, t_{k+N}) \tag{13d}$$

$$\tilde{x}_j(t_k) = x_j(t_k) \tag{13e}$$

$$\frac{\partial \hat{V}(x(t_k))}{\partial x_j}(F_{nn_j}(x(t_k), u_{d_j}(t_k))) \le \frac{\partial \hat{V}(x(t_k))}{\partial x_j}(F_{nn_j}(x(t_k), \Phi_{nn_j}(x(t_k)))), \quad \text{if} \quad x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_{nn}} \tag{13f}$$

$$\hat{V}(\bar{x}(t)) \le \rho_{nn}, \quad \forall \ t \in [t_k, t_{k+N}), \quad \text{if} \quad x(t_k) \in \Omega_{\rho_{nn}} \tag{13g}$$

where $\tilde{x}_j$ is the predicted state trajectory, $S(\Delta)$ is the set of piecewise constant functions with period Δ, and N is the number of sampling periods in the prediction horizon for subsystem j. The optimal input trajectory computed by this LMPC j is denoted by $u_{d_j}^*(t)$, which is calculated over the entire prediction horizon $t \in [t_k, t_{k+N})$. The control action computed for the first sampling period of the prediction horizon $u_{d_j}^*(t_k)$ is sent by LMPC j to its control actuators to be applied over the next sampling period. In the optimization problem of Eq. (13), the objective function of Eq. (13a) is the integral of $L(\tilde{x}_j(t), u_{d_j}(t))$ over the prediction horizon. Note that $L(x_j, u_j)$ is typically in a quadratic form, i.e., $L(x_j, u_j) = x_j^T Q_j x_j + u_j^T R_j u_j$, where Q_j and R_j are the weighting matrices of the states and the inputs of subsystem j respectively, such that each subsystem will be steered towards the origin by minimizing the objective function. The constraint of Eq. (13b) is the LSTM model of Eq. (7) that is used to predict the states of the closed-loop subsystem j using $\bar{x}(t)$ and $u_{d_j}(t)$ as the input vector to the LSTM model. $\bar{x}(t)$ is a vector containing the predicted states of subsystem j, $\tilde{x}_j(t)$, and the measured states of all other subsystems at $t = t_k$, $x_i(t_k)$, $\forall i = 1, \ldots, N_{sys}$, and $i \neq j$. Eq. (13d) defines the input

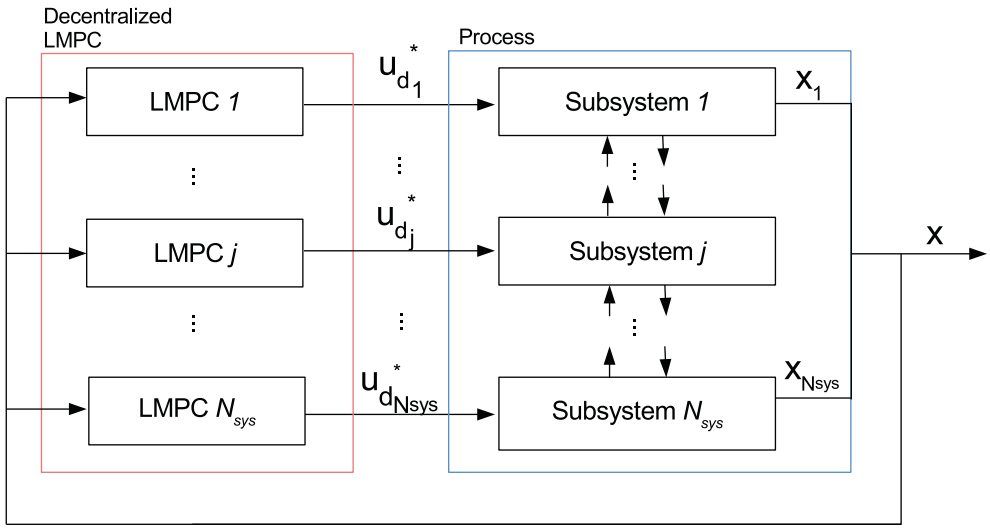

Fig. 4 – A schematic showing the flow of information of N_{sys} number of decentralized LMPCs with the overall process subdivided into N_{sys} number of subsystems.

constraints on u_{d_j} applied over the entire prediction horizon. Eq. (13e) defines the initial condition $\tilde{x}_j(t_k)$ of Eq. (13b), which is the state measurement $x_j(t)$ at $t = t_k$. The constraint of Eq. (13f) forces the closed-loop state to move towards the origin at a minimum rate characterized by the Lyapunov function $\hat{V}$ and the Lyapunov-based control law $\Phi_{nn_j}(x(t_k))$ if $x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_{nn}}$. However, if $x(t_k)$ enters $\Omega_{\rho_{nn}}$, the states predicted by the LSTM model of Eq. (13b) will be maintained in $\Omega_{\rho_{nn}}$ for the entire prediction horizon.

Since the decentralized LMPCs examine different optimization problems specific to their respective subsystems and they are computed in separate processors in parallel, the computation time for solving one iteration of the decentralized LMPC design in one sampling period (assuming that the feedback measurements are available to both controllers at synchronous intervals) will be the maximum time out of the two LMPCs.

Remark 7. The computation time to calculate the solutions to any MPC optimization problem(s) must be less than the sampling time of the nonlinear process. One of the main advantages of decentralized MPC systems is the reduced computational complexity of the optimization problems where a large-scale system is broken down into local subsystems, and thus, reduced total computational time is needed compared to solving the optimization problem in a centralized MPC system.

Remark 8. In the case that full-state feedback is not available to the controllers, decentralized estimators can be used to reconstruct state information. Provided that the state estimation error is bounded and is rendered sufficiently small in a short time, it can be considered as part of the bounded process disturbance, and the closed-loop stability analysis of the estimator-based control system may be established.

4.1. *Sample-and-hold implementation of Decentralized LMPC*

Once the optimization problems of all subsystems $j = 1, \ldots, N_{sys}$ are solved, the optimal control actions of the proposed decentralized LMPC design are defined as follows:

$$u_j(t) = u^*_{d_j}(t_k), \quad j = 1, \ldots, N_{sys}, \quad \forall t \in [t_k, t_{k+1}) \tag{14}$$

The stability of the overall system of Eq. (1) is ensured by the inclusion of a contractive constraint Eq. (13f) in the formulation of each of the decentralized LMPCs, which ascertains that the optimized control actions of Eq. (14) guarantee a minimum decrease rate of the Lyapunov function $\hat{V}$ characterized by the Lyapunov-based control law using LSTM models $\Phi_{nn_j}(x(t))$. We have previously proven the stability properties of a centralized MPC using RNN models where the Lyapunov-based control actions are applied in a sample-and-hold manner, given that the process disturbances and the modeling error are sufficiently small; detailed proof can be found in Wu et al. (2019a). More specifically, given that the stabilizability conditions of Eq. (10) are satisfied and the modeling error $|\nu| \leq \gamma|x| \leq \nu_m$, where γ is chosen to satisfy $\gamma < \hat{c}_3/\hat{c}_4$ ($\hat{c}_3$ and $\hat{c}_4$ are the positive constants in Eq. (10)), the closed-loop state $x(t)$ of the actual process of Eq. (1) and the closed-loop state $\hat{x}(t)$ of the LSTM system of Eq. (6) are bounded in the stability region and driven to a small neighborhood around the origin under the set of Lyapunov-based controllers $\boldsymbol{\Phi}_{nn}(x) \in U$ implemented in a sample-and-hold manner. This is established in the following proposition:

Proposition 1. *Consider the system of Eq. (1) under the Lyapunov-based controllers $u_j = \Phi_{nn_j}(\hat{x}) \in U_j$, $j = 1, \ldots, N_{sys}$, which meet the conditions of Eq. (10), and are designed to stabilize the LSTM system of Eq. (6) and implemented in a sample-and-hold fashion, i.e., $u_j(t) = \Phi_{nn_j}(\hat{x}(t_k))$, $j = 1, \ldots, N_{sys}$, $\forall t \in [t_k, t_{k+1})$, where $t_{k+1} := t_k + \Delta$. The LSTM system is developed with an overall modeling error $|\nu| \leq \gamma|x| \leq \nu_m$, where $\gamma < \hat{c}_3/\hat{c}_4$. Let $\epsilon_s, \epsilon_w > 0$, $\Delta > 0$, $\tilde{c}_3 = -\hat{c}_3 + \hat{c}_4\gamma > 0$, and $\hat{\rho} > \rho_{\min} > \rho_{nn} > \rho_s$ satisfy*

$$-\frac{\hat{c}_3}{\hat{c}_2}\rho_s + L_{nn}M_{nn}\Delta \leq -\epsilon_s \tag{15a}$$

$$-\frac{\tilde{c}_3}{\hat{c}_2}\rho_s + L'_x M\Delta + L'_w w_m \leq -\epsilon_w \tag{15b}$$

and

$$\rho_{nn} := \max\{\hat{V}(\hat{x}(t+\Delta)) \;|\; \hat{x}(t) \in \Omega_{\rho_s}, u \in U\} \tag{16a}$$

$$\rho_{\min} \geq \rho_{nn} + \frac{\hat{c}_4\sqrt{\hat{\rho}}}{\sqrt{\hat{c}_1}} f_w(\Delta) + \kappa(f_w(\Delta))^2 \tag{16b}$$

Then, for any $x(t_k) = \hat{x}(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_s}$, *the following inequality holds:*

$$\dot{\hat{V}}(\hat{x}(t)) \leq \dot{\hat{V}}(\hat{x}(t_k)), \quad \forall t \in [t_k, t_{k+1}) \tag{17a}$$

$$\dot{\hat{V}}(x(t)) \leq \dot{\hat{V}}(x(t_k)), \quad \forall t \in [t_k, t_{k+1}) \tag{17b}$$

and if $x_0 \in \Omega_{\hat{\rho}}$, *the state* $\hat{x}(t)$ *of the LSTM modeled system of Eq. (6) is bounded in* $\Omega_{\hat{\rho}}$ *for all times and ultimately bounded in* $\Omega_{\rho_{nn}}$, *and the state* $x(t)$ *of the nonlinear system of Eq. (1) is bounded in* $\Omega_{\hat{\rho}}$ *for all times and ultimately bounded in* $\Omega_{\rho_{\min}}$.

Consider each closed-loop subsystem of Eq. (2) under $u_j = u_{d_j}^*$ in the decentralized LMPC design of Eq. (14), which are calculated based on the controllers $\Phi_{nn_j}(x)$, $j = 1, \ldots, N_{sys}$ that collectively satisfy Eq. (10) for the overall system. The control actions computed by each LMPC will be applied in a sample-and-hold manner to the process. The proof for recursive feasibility of each of the decentralized LMPCs, and the closed-loop stability of the overall nonlinear process of Eq. (1) under the sample-and-hold implementation of the optimal control actions $u_j = u_{d_j}^*$, $j = 1, \ldots, N_{sys}$ of Eq. (14) will be established in the following theorem.

Theorem 1. *Consider the nonlinear process of Eq. (1) and the LSTM network system of Eq. (6). In the presence of bounded disturbances* $|w(t)| \leq w_m$ *and a sufficiently small modeling error* $|\nu| \leq \gamma|x| \leq \nu_m$, $\gamma > 0$, *there exists a class* $\mathcal{K}$ *function* $f_w(\cdot)$ *and a positive constant* κ *such that* $\forall x, \hat{x} \in \Omega_{\hat{\rho}}$ *and* $w(t) \in W$, $|e(t)| = |x(t) - \hat{x}(t)| \leq f_w(t) := \frac{L_w w_m + \nu_m}{L_x}(e^{L_x t} - 1)$. *Let* $\Delta > 0$, $\epsilon_s > 0$, $\epsilon_w > 0$, $\tilde{c}_3 = -\hat{c}_3 + \hat{c}_4\gamma > 0$, *and* $\hat{\rho} > \rho_{\min} > \rho_{nn} > \rho_s$ *satisfy the conditions of Eqs. (15a)–(16b), then given any initial state* $x_0 \in \Omega_{\hat{\rho}}$, *there always exists a feasible solution for the optimization problems of Eq. (13) for each decentralized LMPC, and it is guaranteed that under the optimized control actions* $u_j = u_{d_j}^*$, $j = 1, \ldots, N_{sys}$, $x(t) \in \Omega_{\hat{\rho}}$, $\forall t \geq 0$, *and* $x(t)$ *ultimately converges to* $\Omega_{\rho_{\min}}$ *for the closed-loop system of Eq. (1).*

Proof. This proof consists of three parts. First, we will prove the recursive feasibility of the optimization problem in each decentralized LMPC in *Part* 1. Then, under the optimized control actions of the decentralized LMPC design of Eq. (14), we will prove the boundedness and convergence of the closed-loop state of the LSTM network system of Eq. (6) in a compact set $\Omega_{\rho_{nn}}$ in *Part* 2. And lastly, in *Part* 3, we will prove the boundedness and convergence of the closed-loop state of the nonlinear system of Eq. (1) in a compact set $\Omega_{\rho_{\min}}$ under the sample-and-hold implementation of the control actions from the decentralized LMPC design of Eq. (14).

Part 1: In this part, we prove that the optimization problem in Eq. (13) of each decentralized LMPC for subsystem j is feasible for all states $x \in \Omega_{\hat{\rho}}$. First, we consider $x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_{nn}}$. The input trajectories $u_{d_j}(t) = \Phi_{nn_j}(x(t_k))$, $j = 1, \ldots, N_{sys}$ for $t \in [t_k, t_{k+1})$ are feasible solutions to the optimization problem of Eq. (13) as the input constraint of Eq. (13d) and the Lyapunov-based contractive constraint of Eq. (13f) are both satisfied. Now we consider $x(t_k) \in \Omega_{\rho_{nn}}$. The input trajectories $u_{d_j}(t_{k+i}) = \Phi_{nn_j}(\hat{x}(t_{k+i}))$, $i = 0, 1, \ldots, N-1$, $j = 1, \ldots, N_{sys}$ satisfy the constraints on the inputs in Eq. (13d) and the Lyapunov-based constraint of Eq. (13g). This is because it has been shown in Proposition 1 that the states predicted by the LSTM model of Eq. (13b) under the Lyapunov-based controllers $\Phi_{nn_j}(\tilde{x})$, $j = 1, \ldots, N_{sys}$ remain inside $\Omega_{\rho_{nn}}$. Therefore, we have proven that for all $x_0 \in \Omega_{\hat{\rho}}$, the optimization problem of each decentralized LMPC can be solved with recursive feasibility.

$$\frac{\partial \hat{V}(\hat{x}(t_k))}{\partial \hat{x}_j} F_{nn_j}(\hat{x}(t_k), u_{d_j}^*(t_k)) \leq \frac{\partial \hat{V}(\hat{x}(t_k))}{\partial \hat{x}_j} F_{nn_j}(\hat{x}(t_k), \Phi_{nn_j}(\hat{x}(t_k))), \quad \forall j = 1, \ldots, N_{sys} \tag{18}$$

Since the Lyapunov function for the overall system $\hat{V}(x)$ can be a linear combination of the Lyapunov functions for each subsystem, $\hat{V}(x) = \sum_{j=1}^{N_{sys}} \hat{V}_j(x_j)$, where $\hat{V}_j$ is assumed to be a function of x_j only, Eq. (18) can be extended to the overall system of Eq. (6) to give the following inequality:

$$\sum_{j=1}^{N_{sys}} \frac{\partial \hat{V}(\hat{x}(t_k))}{\partial \hat{x}_j} F_{nn_j}(\hat{x}(t_k), u_{d_j}^*(t_k)) \leq \sum_{j=1}^{N_{sys}} \frac{\partial \hat{V}(\hat{x}(t_k))}{\partial \hat{x}_j} F_{nn_j}(\hat{x}(t_k), \Phi_{nn_j}(\hat{x}(t_k))) \tag{19a}$$

$$\Rightarrow \frac{\partial \hat{V}(\hat{x}(t_k))}{\partial \hat{x}} F_{nn}(\hat{x}(t_k), u_d^*(t_k)) \leq \frac{\partial \hat{V}(\hat{x}(t_k))}{\partial \hat{x}} F_{nn}(\hat{x}(t_k), \Phi_{nn}(\hat{x}(t_k))) \leq -\hat{c}_3|\hat{x}(t_k)|^2 \tag{19b}$$

where Eq. (19b) is given by the stabilizability condition of Eq. (10b). By the definition of ρ_{nn} in Eq. (16a), $x(t_k) = \hat{x}(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_{nn}}$ also belongs to the set $\Omega_{\hat{\rho}} \backslash \Omega_{\rho_s}$. Considering Eq. (19b), the conditions of Eq. (10), and the Lipschitz conditions of Eq. (12), the time derivative of the Lyapunov function along the trajectory of $\hat{x}(t)$ of the LSTM model of Eq. (6) in $t \in [t_k, t_{k+1})$ is given by:

$$\begin{aligned}
\frac{\partial \hat{V}(\hat{x}(t))}{\partial \hat{x}} F_{nn}(\hat{x}(t), u_d^*(t_k)) &= \frac{\partial \hat{V}(\hat{x}(t_k))}{\partial \hat{x}} F_{nn}(\hat{x}(t_k), u_d^*(t_k)) \\
&\quad + \frac{\partial \hat{V}(\hat{x}(t))}{\partial \hat{x}} F_{nn}(\hat{x}(t), u_d^*(t_k)) \\
&\quad - \frac{\partial \hat{V}(\hat{x}(t_k))}{\partial \hat{x}} F_{nn}(\hat{x}(t_k), u_d^*(t_k)) \\
&\leq -\frac{\hat{c}_3}{\hat{c}_2}\rho_s + \frac{\partial \hat{V}(\hat{x}(t))}{\partial \hat{x}} F_{nn}(\hat{x}(t), u_d^*(t_k)) \\
&\quad - \frac{\partial \hat{V}(\hat{x}(t_k))}{\partial \hat{x}} F_{nn}(\hat{x}(t_k), u_d^*(t_k)) \\
&\leq -\frac{\hat{c}_3}{\hat{c}_2}\rho_s + L_{nn}|\hat{x}(t) - \hat{x}(t_k)| \\
&\leq -\frac{\hat{c}_3}{\hat{c}_2}\rho_s + L_{nn}M_{nn}\Delta \\
&\leq -\epsilon_s
\end{aligned} \tag{20}$$

By integrating the above equation over $t \in [t_k, t_{k+1})$, it is obtained that $V(\hat{x}(t_{k+1})) \leq V(\hat{x}(t_k)) - \epsilon_s\Delta$, hence $V(\hat{x}(t)) \leq$

$V(\hat{x}(t_k)), \forall t \in [t_k, t_{k+1})$. Thus, for all $\hat{x}(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_s}$, the state of the closed-loop LSTM system of Eq. (6) is bounded in the closed-loop stability region $\Omega_{\hat{\rho}}$ for all times and moves towards the origin under the sample-and-hold implementation of $u_j = u_{d_j}^*, \quad j = 1, \ldots, N_{sys}$. Next, we consider when $x(t_k) = \hat{x}(t_k) \in \Omega_{\rho_s}$. $\Omega_{\rho_{nn}}$ as defined in Eq. (16a) ensures that when $\hat{x}(t_k) \in \Omega_{\rho_s}$, under $u \in U$, the closed-loop state $\hat{x}(t)$ of the LSTM model does not leave $\Omega_{\rho_{nn}}$ for all $t \in [t_k, t_{k+1})$. If $\hat{x}(t_{k+1})$ leaves $\Omega_{\rho_{nn}}$, then Eq. (20) is satisfied at $t = t_{k+1}$, and the control actions of the decentralized LMPC design of Eq. (14) will drive $\hat{x}$ towards the origin over the next sampling period. If $\hat{x}(t_k)$ is outside of Ω_{ρ_s} but inside of $\Omega_{\rho_{nn}}$, the constraint of Eq. (13g) will ensure that $\hat{x}(t_{k+i}), \quad i = 1, \ldots, N-1$ will also remain inside $\Omega_{\rho_{nn}}$. This concludes *Part* 2 of the proof showing the convergence of the closed-loop state $\hat{x}$ of the LSTM system of Eq. (6) to $\Omega_{\rho_{nn}}$ for all $\hat{x}_0 \in \Omega_{\hat{\rho}}$.

$$\begin{aligned}
\frac{\partial \hat{V}(x(t_k))}{\partial x} F(x(t_k), u_d^*(t_k), 0) = & \frac{\partial \hat{V}(x(t_k))}{\partial x} (F_{nn}(x(t_k), u_d^*(t_k)) \\
& + F(x(t_k), u_d^*(t_k), 0) - F_{nn}(x(t_k), u_d^*(t_k))) \\
\leq & -\hat{c}_3 |x(t_k)|^2 + \hat{c}_4 |x(t_k)| |(F(x(t_k), u_d^*(t_k), 0) \\
& - F_{nn}(x(t_k), u_d^*(t_k))) \\
\leq & -\hat{c}_3 |x(t_k)|^2 + \hat{c}_4 \gamma |x(t_k)|^2 \\
\leq & -\tilde{c}_3 |x(t_k)|^2
\end{aligned} \tag{21}$$

where $\tilde{c}_3 = -\hat{c}_3 + \hat{c}_4 \gamma > 0$. Therefore, the closed-loop state of the nominal system of Eq. (1) with $w \equiv 0$ converges to the origin under $u_j = u_{d_j}^*, \quad j = 1, \ldots, N_{sys}, \quad \forall x_0 \in \Omega_{\hat{\rho}}$ if the modeling error $|\nu| \leq \gamma |x|$. Considering bounded disturbances and the effect of sample-and-hold implementation, based on Eq. (21), Eq. (10a), and the Lipschitz condition in Eq. (5), the following inequality is obtained for the time-derivative of $\hat{V}(x(t))$ $\forall t \in [t_k, t_{k+1})$ and $x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_s}$:

$$\begin{aligned}
\frac{\partial \hat{V}(x(t))}{\partial x} F(x(t), u_d^*(t_k), w) = & \frac{\partial \hat{V}(x(t_k))}{\partial x} F(x(t_k), u_d^*(t_k), 0) \\
& + \frac{\partial \hat{V}(x(t))}{\partial x} F(x(t), u_d^*(t_k)), w) \\
& - \frac{\partial \hat{V}(x(t_k))}{\partial x} F(x(t_k), u_d^*(t_k), 0) \\
\leq & -\frac{\tilde{c}_3}{\hat{c}_2} \rho_s + \frac{\partial \hat{V}(x(t))}{\partial x} F(x(t), u_d^*(t_k), w) \\
& - \frac{\partial \hat{V}(x(t_k))}{\partial x} F(x(t_k), u_d^*(t_k), 0) \\
\leq & -\frac{\tilde{c}_3}{\hat{c}_2} \rho_s + L_x' |x(t) - x(t_k)| + L_w' |w| \\
\leq & -\frac{\tilde{c}_3}{\hat{c}_2} \rho_s + L_x' M \Delta + L_w' w_m \\
\leq & -\epsilon_w
\end{aligned} \tag{22}$$

Integrating Eq. (22) for $t \in [t_k, t_{k+1})$ will show that Eq. (17b) holds. Therefore, the closed-loop state of the actual nonlinear process of Eq. (1) is maintained in $\Omega_{\hat{\rho}}$ for all $t \geq t_0$, and can be driven towards the origin in every sampling period under the decentralized LMPC control actions $u_j = u_{d_j}^*, \quad j = 1, \ldots, N_{sys}$. Next, we consider the case where $x(t_k) \in \Omega_{\rho_s}$. Given that the state error is bounded by $|e(t)| = |x(t) - \hat{x}(t)| \leq f_w(t) := \frac{L_w w_m + \nu_m}{L_x} (e^{L_x t} - 1)$, there exists a compact set $\Omega_{\rho_{min}} \supset \Omega_{\rho_{nn}}$ satisfying Eq. (16b) such that the state of the actual nonlinear system of Eq. (1) at $t = t_{k+1}$ (i.e., $x(t_{k+1})$) does not leave $\Omega_{\rho_{min}}$ if the predicted state of the LSTM model of Eq. (6) at $t = t_{k+1}$ (i.e., $\hat{x}(t_{k+1})$) is bounded in $\Omega_{\rho_{nn}}$, which is guaranteed if $x(t_k) \in \Omega_{\rho_s}$, as shown in *Part* 2. Therefore, we have proven that the state $x(t)$ of the nonlinear system of Eq. (1) is bounded in $\Omega_{\hat{\rho}}$ for all $x_0 \in \Omega_{\hat{\rho}}, \quad t \geq t_0$ and ultimately bounded in $\Omega_{\rho_{min}}$ under the sample-and-hold implementation of the decentralized LMPC control design of Eq. (14). After the state $x(t_k)$ enters $\Omega_{\rho_{nn}}$, the constraint of Eq. (13g) is activated to maintain the predicted states $\hat{x}$ of the LSTM model of Eq. (13b) inside $\Omega_{\rho_{nn}}$ over the entire prediction horizon, such that the closed-loop state $x(t)$ of the nonlinear system of Eq. (1) remains in $\Omega_{\rho_{min}}$ under the sample-and-hold implementation of the optimized decentralized control action $u_j = u_{d_j}^*, \quad j = 1, \ldots, N_{sys}$ for $t \in [t_k, t_{k+1})$. At the next sampling step, if $x(t_{k+1}) \in \Omega_{\rho_{min}} \backslash \Omega_{\rho_{nn}}$, then Eq. (22) holds, and the contractive constraint of Eq. (13f) will be activated instead to drive the state towards the origin during the next sampling period. Therefore, with the overall system stabilized under the control actions $u_j = u_{d_j}^*, \quad j = 1, \ldots, N_{sys}$ of the decentralized LMPC design of Eq. (14), we have proven that the overall system can be stabilized under the control actions of the decentralized LMPC design of Eq. (14) implemented in a sample-and-hold fashion. □

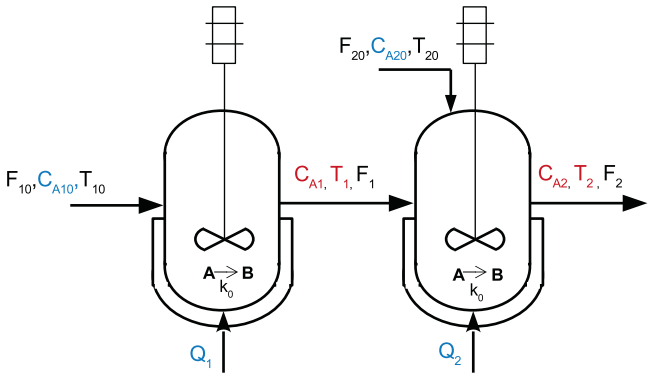

Fig. 5 – Process flow diagram of two CSTRs in series.

5. Application to a two-CSTR-in-series process

We use a chemical process example to demonstrate the closed-loop simulation of the nonlinear process under the decentralized Lyapunov-based model predictive control (LMPC) system using the trained LSTM models, and the simulation results will be compared to that of decentralized LMPC system using first-principles models, as well as a centralized model predictive control system using LSTM models. More specifically, two well-mixed, non-isothermal continuous stirred tank reactors (CSTRs) that operate in series are considered, where an irreversible second-order exothermic reaction transforming a reactant A to a product B ($A \rightarrow B$) takes place in each reactor. A schematic diagram of the process is shown in Fig. 5. Reactant material A with inlet concentration C_{Aj0}, inlet temperature T_{j0} and feed volumetric flow rate of the reactor F_{j0}, are fed into each of the two reactors $j = 1, 2$, where $j = 1$ denotes the first CSTR and $j = 2$ denotes the second CSTR. A heating jacket is installed on each CSTR to supply/remove heat

Table 1 – Parameter values of the two CSTRs in series.

$T_{10} = 300\,K$	$T_{20} = 300\,K$
$F_{10} = 5\,m^3/h$	$F_{20} = 5\,m^3/h$
$V_1 = 1\,m^3$	$V_2 = 1\,m^3$
$T_{1_s} = 401.9\,K$	$T_{2_s} = 401.9\,K$
$C_{A1_s} = 1.954\,kmol/m^3$	$C_{A2_s} = 1.954\,kmol/m^3$
$C_{A10_s} = 4\,kmol/m^3$	$C_{A20_s} = 4\,kmol/m^3$
$Q_{1_s} = 0.0\,kJ/h$	$Q_{2_s} = 0.0\,kJ/h$
$k_0 = 8.46 \times 10^6\,m^3/kmol\,h$	$\Delta H = -1.15 \times 10^4\,kJ/kmol$
$C_p = 0.231\,kJ/kg\,K$	$R = 8.314\,kJ/kmol\,K$
$\rho_L = 1000\,kg/m^3$	$E = 5 \times 10^4\,kJ/kmol$

at a rate Q_j, $j = 1, 2$. The dynamic models of the two-CSTR-in-series process are obtained by the following material and energy balance equations:

$$\frac{dC_{A1}}{dt} = \frac{F_{10}}{V_1}(C_{A10} - C_{A1}) - k_0 e^{\frac{-E}{RT_1}} C_{A1}^2 \tag{23a}$$

$$\frac{dT_1}{dt} = \frac{F_{10}}{V_1}(T_{10} - T_1) + \frac{-\Delta H}{\rho_L C_p} k_0 e^{\frac{-E}{RT_1}} C_{A1}^2 + \frac{Q_1}{\rho_L C_p V_1} \tag{23b}$$

$$\frac{dC_{A2}}{dt} = \frac{F_{20}}{V_2} C_{A20} + \frac{F_{10}}{V_2} C_{A1} - \frac{F_{10} + F_{20}}{V_2} C_{A2} - k_0 e^{\frac{-E}{RT_2}} C_{A2}^2 \tag{23c}$$

$$\frac{dT_2}{dt} = \frac{F_{20}}{V_2} T_{20} + \frac{F_{10}}{V_2} T_1 - \frac{F_{10} + F_{20}}{V_2} T_2 + \frac{-\Delta H}{\rho_L C_p} k_0 e^{\frac{-E}{RT_2}} C_{A2}^2 + \frac{Q_2}{\rho_L C_p V_2} \tag{23d}$$

where C_{A_j}, V_j, T_j and Q_j, $j = 1, 2$ are the concentration of reactant A, the volume of the reacting liquid, the temperature, and the heat input rate in the first and the second reactor, respectively. The reacting liquid has a constant density of ρ_L and a constant heat capacity of C_p for both reactors. E, R, ΔH, and k_0 represent the activation energy, ideal gas constant, enthalpy of the reaction, and pre-exponential constant, respectively. The process parameter values are listed in Table 1.

For both CSTRs, the manipulated inputs are the inlet concentration of species A and the heat input rate supplied by the heating jacket, which are represented by the deviation variables $\Delta C_{Aj0} = C_{Aj0} - C_{Aj0_s}$, $\Delta Q_j = Q_j - Q_{j_s}$, $j = 1, 2$, respectively. The manipulated inputs are bounded as follows: $|\Delta C_{Aj0}| \leq 3.5\,kmol/m^3$ and $|\Delta Q_j| \leq 5 \times 10^5\,kJ/h$, $j = 1, 2$. The states of the closed-loop system are the concentration of species A and the temperature in the first and the second reactor, which are also represented by their deviation variables such that the equilibrium point of the system is at the origin of the state-space. Therefore, the vector of closed-loop states is $x = [C_{A1} - C_{A1_s}\ T_1 - T_{1_s}\ C_{A2} - C_{A2_s}\ T_2 - T_{2_s}]^T$, where C_{A1_s}, C_{A2_s}, T_{1_s} and T_{2_s} are the steady-state values of concentration of A and temperature in each of the two tanks, respectively.

The states of the first CSTR can be separately denoted as $x_1 = [C_{A1} - C_{A1_s}\ T_1 - T_{1_s}]^T$ and the states of the second CSTR are denoted as $x_2 = [C_{A2} - C_{A2_s}\ T_2 - T_{2_s}]^T$. Correspondingly, the manipulated inputs that regulate the state of the first CSTR can be denoted as $u_1 = [\Delta C_{A10}\ \Delta Q_1]^T$, and the manipulated inputs for the second CSTR are $u_2 = [\Delta C_{A20}\ \Delta Q_2]^T$. We let the states and the manipulate inputs of the first CSTR constitute subsystem 1, and that of the second CSTR constitute subsystem 2.

In a centralized LMPC framework, feedback measurement of all states x is received by the controller, and the manipulated inputs for the entire system, $u = [\Delta C_{A10}\ \ \Delta Q_1\ \ \Delta C_{A20}\ \ \Delta Q_2]^T$,

Table 2 – Average LMPC computation time in one sampling period and the sum of squared percentage error of all states along the closed-loop trajectory under the centralized and the decentralized systems using their respective LSTM models and first-principles models, all with a total operation time of 0.3 h.

	Ave. computation time (s)	Sum of squared percentage error
Decentralized (LSTM)	5.41	2.94
Decentralized (First-Principles)	5.18	2.84
Centralized (LSTM)	35.26	3.08
Centralized (First-Principles)	27.93	2.96

are computed by one centralized controller. In a decentralized LMPC scheme, we design two LMPCs, each one is responsible for one subsystem. The two LMPCs are solved independently in separate processors without any intercommunications. Due to the specific dynamic behavior and interactions between the two CSTRs, the states of the first CSTR play a part in the dynamic evolution of the states of the second CSTR, but not vice versa. Therefore, LMPC 1 receives feedback measurements of x_1, predicts $\hat{x}_1$ in its internal model, and manipulates the inputs for the first CSTR, $u_1 = [\Delta C_{A10}\ \ \Delta Q_1]^T$. We assume that LMPC 2 has access to full-state measurement signals but does not have any knowledge on how x_1 dynamically evolves. Therefore, LMPC 2 receives feedback measurements of x, predicts the states of the second CSTR, $\hat{x}_2$, and only manipulates the inputs for the second CSTR, $u_2 = [\Delta C_{A20}\ \ \Delta Q_2]^T$. Furthermore, we assume that the feedback measurements of all states are available to both controllers at the same synchronous intervals $\Delta = 10^{-2}$ h.

The dynamic model of Eq. (23) is numerically simulated using the explicit Euler method with an integration time step of $h_c = 10^{-4}$ h. The nonlinear optimization problems of each of the decentralized LMPCs of Eq. (13) are solved using the Python module of the IPOPT software package, PyIpopt (Wächter and Biegler, 2006), with the same sampling period $\Delta = 10^{-2}$ h. The objective function in the decentralized LMPC optimization problem for subsystem j has the form $L_j(x_j, u_j) = x_j^T Q_j x_j + u_j^T R_j u_j$, where $Q_j = diag[2 \times 10^3\ \ 1]$, $R_j = diag[8 \times 10^{-13}\ \ 0.001]$, for $j = 1, \ldots, N_{sys}$. Here, we have two subsystems, $N_{sys} = 2$. The control Lyapunov function for each decentralized LMPC j is $V_j(x_j) = x_j^T P_j x_j$, for $j = 1, 2$, with the following positive definite P_j matrices:

$$P_1 = P_2 = \begin{bmatrix} 1060 & 22 \\ 22 & 0.52 \end{bmatrix} \tag{24}$$

5.1. *LSTM network development*

To collect the training data for building the LSTM network models, we carry out open-loop simulations for finite sampling steps for various initial conditions inside Ω_{ρ_j}, where $\rho_j = 392$, $j = 1, 2$, for both subsystems using the nonlinear system of Eq. (23). We apply various $u_1 \in U_1, u_2 \in U_2$ in a sample-and-hold manner to the nonlinear process and collect trajectories with a time interval of $q_{nn} \cdot h_c = 5h_c$. We separate these trajectories of manipulated input vector u_j and the state vector x_j into segments with a fixed length $T = 20$, such that the time period that the training samples cover, $T \times q_{nn} \times h_c = 20 \times 5 \times 10^{-4}$ h, is the same as the prediction period of the

LSTM network $\Delta = 10^{-2}$ h. We also normalize both the input vector and state vector samples with respect to their means and standard deviations. The trajectories for $T = 20$ of the state vectors x_j over the prediction period Δ are used as the target state vectors when training the LSTM networks. Using the normalized samples of input and state vector sequences for each subsystem j, we develop a separate LSTM network j for each subsystem to predict the normalized future states over one sampling period Δ, which are then re-scaled to the actual future states $\hat{x}_1$ and $\hat{x}_2$ as described in the previous section. Each LSTM network j captures the dynamic behavior of its respective subsystem j, but does not have any information on the dynamic behaviors of all other subsystems; the LSTM netowrk j can then be used in the decentralized LMPC j as the prediction model.

It should be noted that, depending on the different architectures of the control systems, the choice of inputs and outputs as well as the structure of the LSTM model used in the control system may be different. In some cases, the nonlinear process, such as the one outlined in Eq. (23), cannot be completely decoupled into two separate subsystems. For instance, the states of the first CSTR x_1 are completely independent from the states and inputs of the second CSTR; however, the evolution of the states of the second CSTR, x_2, depends on the values of the states of the first CSTR, x_1. With knowledge on only the first CSTR (subsystem 1) and the first CSTR being independent from information on the second CSTR, LSTM network 1 receives information on $x_1(t_k)$ and $u_1(t_k)$ as the inputs to the neural network, and is able to predict $\hat{x}_1(t_{k+1})$, where $t_{k+1} := t_k + \Delta$. Although we assume that full-state feedback information is available to all controllers at the same time intervals, due to the unique process dynamics in this case, decentralized LMPC 1 for the first CSTR (subsystem 1) requires feedback measurements of x_1 only. Therefore, LSTM network 1, which is built to be used as the internal prediction model in decentralized LMPC 1, only uses $x_1(t_k)$ and $u_1(t_k)$ as the inputs to the neural network, and does not need information on $x_2(t_k)$. On the other hand, since the prediction of $\hat{x}_2(t_{k+1})$ depends on $x_1(t_k)$ and $x_2(t_k)$, decentralized LMPC 2 will take full advantage of the feedback measurements of all states. Therefore, the inputs to the LSTM network 2 are $x_1(t_k)$, $x_2(t_k)$, and $u_2(t_k)$, and the outputs of LSTM network 2 are the predicted future states of the second CSTR, $\hat{x}_2(t)$ at $t = t_{k+1}$. Note that since LSTM network 2 does not have any information on u_1 and only predicts closed-loop states of the second CSTR x_2, the LSTM network model is only accounting for the impact of $x_1(t_k)$, $x_2(t_k)$, and $u_2(t_k)$ on the future state $x_2(t_{k+1})$ analogous to only having knowledge on the dynamic behavior of subsystem 2.

Both LSTM networks $j = 1, 2$ are developed with 1 hidden layer, where *tanh* function is used as the activation function, and *Adam* is used as the optimizer to minimize the error between the sequences of target states and predicted states. With the normalized training data, the mean squared modeling error needs to be less than 5×10^{-7} to terminate the training process. After 10 epochs of training, each epoch taking on average 700 s, the training mean squared error between the predicted states $\hat{x}_1$ of the LSTM network 1 and the first-principles model of the first CSTR is 6.70×10^{-7}, and the validation mean squared error is 6.43×10^{-4}. Similarly, the mean squared error between the predicted states $\hat{x}_2$ of the LSTM network 2 and the first-principles model of the second CSTR in the training dataset is 7.02×10^{-7}, and the mean squared error in the validation dataset is 1.14×10^{-6}, after 10 epochs of training with each epoch taking on average 720 s. The Lyapunov function of the LSTM network model for both subsystems, $\hat{V}_j$, $j = 1, 2$, is chosen to be the same as V_j. The set $\hat{D}_j$ can be characterized using the stabilizing control laws $u_j = \Phi_{nn_j}(x_j)$, $j = 1, 2$ of Eq. (4), computed using the numerically approximated $\hat{f}_j(t_k), \hat{g}_{j_i}(t_k)$ in Eq. (11). Subsequently, the closed-loop stability region $\Omega_{\hat{\rho}_j}$ for subsystem j can be characterized as the largest level set of $\hat{V}_j$ in $\hat{D}_j$ while also being a subset of Ω_{ρ_j}. The positive constants $\hat{\rho}_j$, $j = 1, 2$, which are used to define the largest level sets of the control Lyapunov functions for subsystem 1 and 2 respectively, are $\hat{\rho}_1 = \hat{\rho}_2 = 380$. Additionally, the ultimate bounded region $\Omega_{\rho_{nn_j}}$, and subsequently, $\Omega_{\rho_{\min_j}}$, are chosen to be $\rho_{nn_j} = 10$ and $\rho_{\min_j} = 12$, for $j = 1, 2$; the positive constants ρ_{nn_j} and $\rho_{\min_j}$ are determined via extensive closed-loop simulations with $u_j \in U_j$. More computational details on the development of a recurrent neural network model can be found in Wu et al. (2019b).

Remark 9. The machine-learning-based model of each subsystem can be of different structures and variants of the recurrent neural network, and can be trained using different algorithms, depending on the complexity and dynamic behavior of the nonlinear process. In general, if the machine learning algorithm is capable of capturing temporal relationship between its input and output variables, then it can be used for model construction from data for dynamic systems. For example, a classic recurrent neural network model is fitted for a single-CSTR process in Wu et al. (2019b), and a recurrent neural network whose design depends on the a priori structural process knowledge is fitted for the two-CSTR-in-series process in Wu et al. (2020). Besides neural networks, hidden Markov models and support vector machines have also been applied in modeling and control of dynamic systems (Liu and Yao, 2005; Chen and Pan, 2009). In this example, we constructed two LSTM network models for the two CSTRs individually. However, similar studies can be done where the first CSTR can be modeled using a classic RNN with simpler network structures to improve computational efficiency during training and execution, and the second CSTR can be modeled using a different class of recurrent neural networks (e.g., LSTM or GRU) with more units (i.e., more nonlinear activation functions and fitted parameters) to account for the relatively higher complexity in the second subsystem.

5.2. *Closed-loop model predictive control simulations*

We run closed-loop simulations on the two-CSTR-in-series process of Eq. (23) operated under a centralized LMPC system and under two decentralized LMPC systems. First, to design a centralized LMPC system, we use an LSTM network model developed for the overall two-CSTR process of Eq. (23) where, at time instant $t = t_k$, the inputs to this overall LSTM model are the four manipulated inputs $u(t) = [\Delta C_{A10} \;\; \Delta Q_1 \;\; \Delta C_{A20} \;\; \Delta Q_2]^T$ applied in a sample-and-hold fashion for $t \in [t_k, t_{k+1})$, as well as the four states of the overall process $x(t) = [C_{A1} - C_{A1_s} \;\; T_1 - T_{1_s} \;\; C_{A2} - C_{A2_s} \;\; T_2 - T_{2_s}]^T$ at $t = t_k$, and the outputs of this LSTM network model are the predicted states $\hat{x}(t_{k+1})$. The centralized LMPC uses this two-CSTR process model as the prediction model to optimize the four manipulated inputs $u^*(t_k)$ while receiving feedback

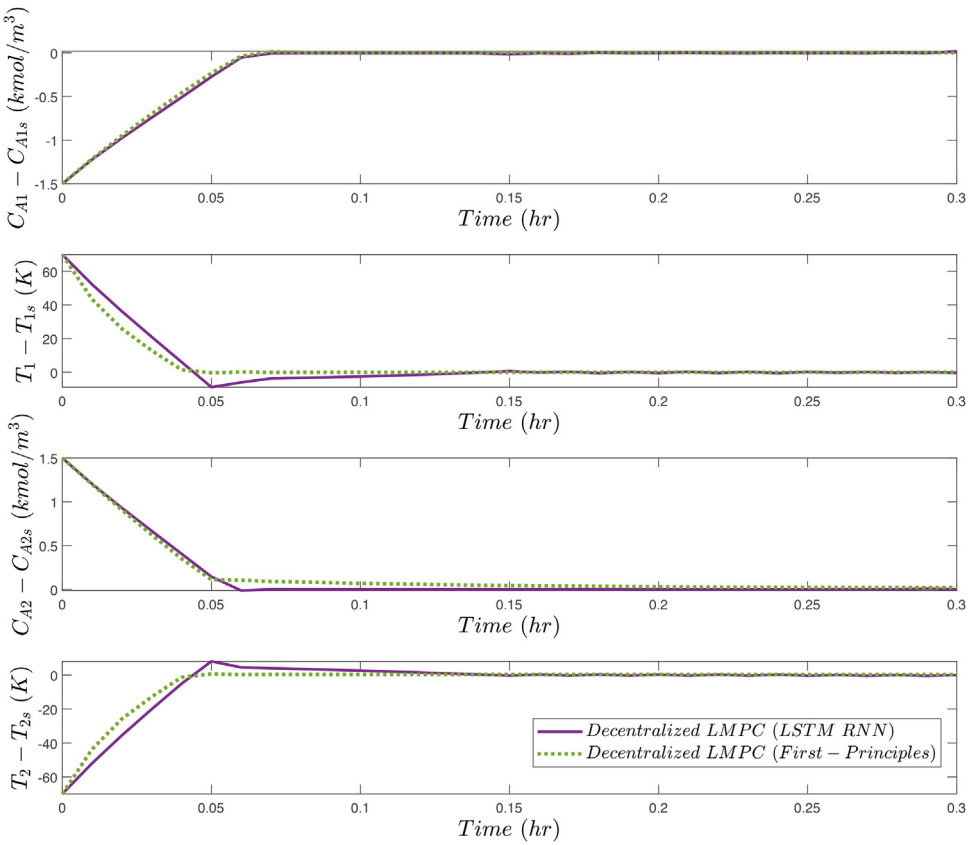

Fig. 6 – Closed-loop state trajectories of the decentralized LMPC systems using LSTM model and first-principles (FP) model, respectively.

of all states x. The closed-loop performance of this centralized LMPC using an LSTM network model will be used as a benchmark for comparison. Next, with the two LSTM networks developed for subsystem 1 and 2 as shown in Section 5.1, we design two decentralized LMPCs, where LMPC 1 uses the LSTM network model for subsystem 1, and LMPC 2 uses the LSTM network model for subsystem 2. LMPC 1 receives feedback of $x_1 = [C_{A1} - C_{A1_s} \quad T_1 - T_{1_s}]^T$ and optimizes $u_1 = [\Delta C_{A10} \quad \Delta Q_1]^T$; LMPC 2 receives feedback of all states x and optimizes $u_2 = [\Delta C_{A20} \quad \Delta Q_2]^T$. The two decentralized LMPCs are independent of each other, and they can be solved simultaneously in separate processors in parallel to save computation time. Therefore, the computation time for solving one LMPC iteration is the maximum time of the two decentralized controllers. To further demonstrate the efficacy of the decentralized LMPC system using LSTM models, we compare the closed-loop performance of the decentralized LMPC system using LSTM models with that of the same decentralized LMPC system using first-principles models. The closed-loop performances of the centralized and the decentralized control networks using their respective LSTM models, as well as using the first-principles models are compared in terms of the computation time of solving one MPC iteration and the sum of squared percentage error of the closed-loop states x for a total simulation period of $t_p = 0.3$ h, which are both shown in Table 2. The sum of squared percentage error is in the form of $SSPE = \int_0^{t_p} \left[\left(\frac{C_{A1} - C_{A1_s}}{C_{A1_s}}\right)^2 + \left(\frac{T_1 - T_{1_s}}{T_{1_s}}\right)^2 + \left(\frac{C_{A2} - C_{A2_s}}{C_{A2_s}}\right)^2 + \left(\frac{T_2 - T_{2_s}}{T_{2_s}}\right)^2 \right] dt$. It is shown that the average computation time is significantly reduced when the process of Eq. (23) is operated under two decentralized LMPCs compared with the result of a centralized LMPC, and both control systems generate comparable sum of squared percentage errors. Furthermore, the decentralized LMPC system using LSTM models achieves similar average computation time and sum of squared percentage error as the decentralized LMPC system using first-principles models. The computation time of all simulated control systems are lower than the sampling period of the process; therefore, the proposed control system using LSTM models can be implemented without computational issues.

The closed-loop state evolution of the two-CSTR-in-series process under the two decentralized LMPCs using the LSTM model and using the first-principles model are shown in Fig. 6, and the corresponding input profiles are shown in Fig. 7. The closed-loop state trajectories in state-space under the centralized LMPC using an LSTM model, the decentralized LMPC system using two LSTM models, and the decentralized LMPC system using the respective first-principles model for each subsystem, are shown in Fig. 8 to show the boundedness and convergence of the closed-loop states. Starting from initial conditions $x_0 = [-1.5 \quad 70 \quad 1.5 - 70]^T$, all states of each subsystem j, $j = 1, 2$, converge to $\Omega_{\rho_{\min_j}}$ within 0.07 h and are bounded in $\Omega_{\hat{\rho}_j}$ under the two decentralized LMPCs using their respective LSTM network models for all time. Therefore, through closed-loop simulations and assessing their performance metrics, we have illustrated the effectiveness of the proposed decentralized LMPC design using separate LSTM network models in stabilizing the overall nonlinear process.

Remark 10. The absence of a first-principles model provides the motivation for developing machine-learning-based models for nonlinear processes. However, we show the simulation results of the control frameworks using first-principles models of the two-CSTR-in-series process of Eq. (23) to adequately compare with the results of the same process operated under the same control frameworks using LSTM network models.

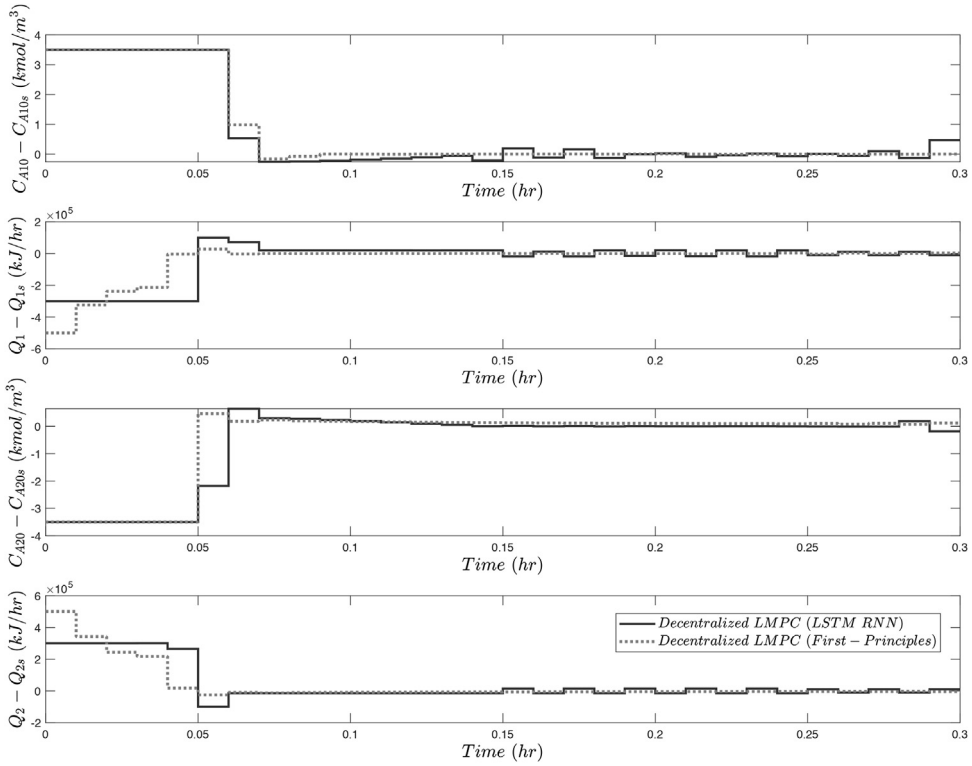

Fig. 7 – Closed-loop input trajectories of the decentralized LMPC systems using LSTM model and first-principles (FP) model, respectively.

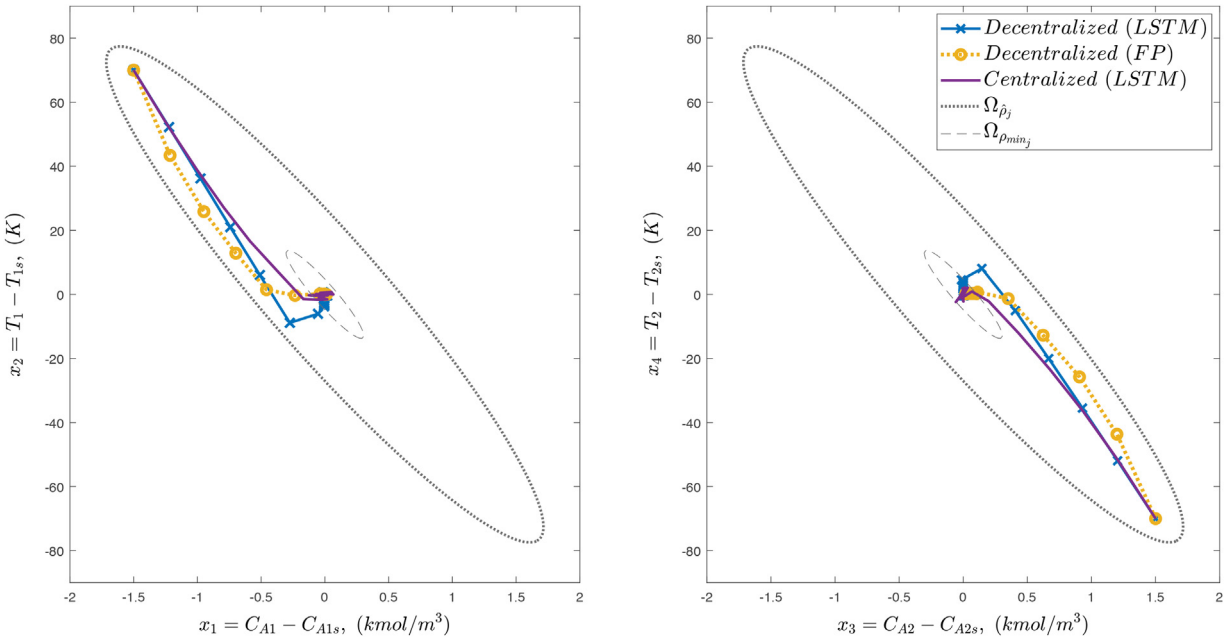

Fig. 8 – Closed-loop state trajectories of the centralized LMPC system using an LSTM model, and the decentralized LMPC systems using LSTM models and first-principles (FP) models in state space, showing the boundedness of the states of each subsystem j, $j = 1, 2$ in $\Omega_{\hat{\rho}_j}$ for all operation time $t_p = 0.03$ h, and the convergence of the states of each subsystem j in $\Omega_{\rho_{min_j}}$ after $t \geq 0.07$ h.

In real-life scenarios where the first-principles model of the studied process is not attainable, the same comparison can be carried out with respect to real plant data.

6. Conclusions

In this work, we presented the design of decentralized model predictive control systems for nonlinear processes using machine-learning models, which were developed separately to capture the nonlinear dynamic behavior of independent subsystems. The nonlinear process was divided into multiple subsystems, and the closed-loop stability and performance properties of the proposed decentralized framework with respect to the overall process were analyzed. Using Lypuanov-based control methods to characterize the stability region of the nonlinear process, extensive open-loop data were collected to train one LSTM network model for each subsystem with a sufficiently small modeling error. It was shown that the

decentralized LMPCs using LSTM models were able to ensure closed-loop state boundedness and convergence to a small neighborhood around the origin for all process states inside the stability region while achieving efficient real-time computation. Using a nonlinear chemical process example, the decentralized LMPCs using LSTM models were able to obtain similar closed-loop performance as the same decentralized LMPCs using first-principles models. Furthermore, they were able to significantly reduce the computation time while ensuring similar closed-loop results when compared to the case of the centralized LMPC using an LSTM model for the overall process.

Declaration of Competing Interest

The authors report no declarations of interest.

References

Al Seyab, R.K., Cao, Y., 2008. Nonlinear system identification for predictive control using continuous time recurrent neural networks and automatic differentiation. J. Process Control 18, 568–581.

Alessio, A., Barcelli, D., Bemporad, A., 2011. Decentralized model predictive control of dynamically coupled linear systems. J. Process Control 21, 705–714.

Bakule, L., 2008. Decentralized control: an overview. Annu. Rev. Control 32, 87–98.

Chen, J., Pan, F., 2009. Dynamic modeling of biotechnical process based on online support vector machine. J. Comput. 4, 251–258.

Chung, J., Gulcehre, C., Cho, K., Bengio, Y., 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In: Proceedings of the NIPS Workshop on Deep Learning and Representation Learning, Montreal, Canada.

Debar, H., Dorizzi, B., 1992. An application of a recurrent network to an intrusion detection system. In: Proceedings of the International Joint Conference on Neural Networks, Beijing, China, pp. 478–483.

Dey, R., Salem, F.M., 2017. Gate-variants of gated recurrent unit (GRU) neural networks. In: Proceedings of the 60th IEEE International Midwest Symposium on Circuits and Systems, Medford, MA, USA, pp. 1597–1600.

Funahashi, K., Nakamura, Y., 1993. Approximation of dynamical systems by continuous time recurrent neural networks. Neural Netw. 6, 801–806.

Graves, A., Mohamed, A., Hinton, G., 2013. Speech recognition with deep recurrent neural networks. In: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, pp. 6645–6649.

Hioe, D., Hudon, N., Bao, J., 2014. Decentralized nonlinear control of process networks based on dissipativity – a hamilton-jacobi equation approach. J. Process Control 24, 172–187.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Comput. 9, 1735–1780.

Hudon, N., Bao, J., 2012. Dissipativity-based decentralized control of interconnected nonlinear chemical processes. Comput. Chem. Eng. 45, 84–101.

Jin, L., Nikiforuk, P.N., Gupta, M.M., 1995. Approximation of discrete-time state-space trajectories using dynamic recurrent neural networks. IEEE Trans. Autom. Control 40, 1266–1270.

Knittel, D., Gigan, D., Laroche, E., 2002. Robust decentralized overlapping control of large scale winding systems. In: Proceedings of the 2002 American Control Conference, Anchorage, AK, USA, pp. 1805–1810.

Kosmatopoulos, E.B., Polycarpou, M.M., Christodoulou, M.A., Ioannou, P.A., 1995. High-order neural network structures for identification of dynamical systems. IEEE Trans. Neural Netw. 6, 422–431.

Liu, S., Liu, J., Feng, Y., Rong, G., 2014. Performance assessment of decentralized control systems: an iterative approach. Control Eng. Pract. 22, 252–263.

Liu, F., Yao, Y., 2005. Modeling and analysis of networked control systems using hidden markov models. Proceedings of the International Conference on Machine Learning and Cybernetics, 928–931.

Magni, L., Scattolini, R., 2006. Stabilizing decentralized model predictive control of nonlinear systems. Automatica 42, 1231–1236.

Mhaskar, P., El-Farra, N.H., Christofides, P.D., 2006. Stabilization of nonlinear systems with state and control constraints using lyapunov-based predictive control. Syst. Control Lett. 55, 650–659.

Ogunmolu, O., Gu, X., Jiang, S., Gans, N., 2016. Nonlinear Systems Identification Using Deep Dynamic Neural Networks (arXiv Preprint) arXiv:1610.01439.

Schmidhuber, J., 2015. Deep learning in neural networks: an overview. Neural Netw. 61, 85–117.

Schuster, M., Paliwal, K.K., 1997. Bidirectional recurrent neural networks. IEEE Trans. Signal Process. 45, 2673–2681.

Sontag, E.D., 1992. Neural nets as systems models and controllers. In: Proceedings of the Seventh Yale Workshop on Adaptive and Learning Systems, Yale University, New Haven, CT, USA, pp. 73–79.

Su, H.T., McAvoy, T.J., Werbos, P., 1992. Long-term predictions of chemical processes using recurrent neural networks: a parallel training approach. Ind. Eng. Chem. Res. 31, 1338–1352.

Sun, Y., El-Farra, N.H., 2008. Quasi-decentralized model-based networked control of process systems. Comput. Chem. Eng. 32, 2016–2029.

Wächter, A., Biegler, L.T., 2006. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Math. Program. 106, 25–57.

Wu, Z., Christofides, P.D., 2019. Economic machine-learning-based predictive control of nonlinear systems. Mathematics 7, 494.

Wu, Z., Rincon, D., Christofides, P.D., 2020. Process structure-based recurrent neural network modeling for model predictive control of nonlinear processes. J. Process Control 89, 74–84.

Wu, Z., Tran, A., Rincon, D., Christofides, P.D., 2019a. Machine learning-based predictive control of nonlinear processes. part I: Theory. AIChE J. 65, e16729.

Wu, Z., Tran, A., Rincon, D., Christofides, P.D., 2019b. Machine-learning-based predictive control of nonlinear processes. part II: Computational implementation. AIChE J. 65, e16734.

Yan, Z., Wang, J., 2012. Model predictive control of nonlinear systems with unmodeled dynamics based on feedforward and recurrent neural networks. IEEE Trans. Ind. Inform. 8, 746–756.

Yin, X., Zeng, J., Liu, J., 2018. Forming distributed state estimation network from decentralized estimators. IEEE Trans. Control Syst. Technol. 27, 2430–2443.