Received: 2 June 2020 | Revised: 14 July 2020 | Accepted: 5 August 2020
DOI: 10.1002/aic.17013

PROCESS SYSTEMS ENGINEERING

Machine learning-based distributed model predictive control of nonlinear processes

Scarlett Chen[1] | Zhe Wu[1] | David Rincon[1] | Panagiotis D. Christofides[2]

[1]Department of Chemical and Biomolecular Engineering, University of California, Los Angeles, California

[2]Department of Chemical and Biomolecular Engineering and the Department of Electrical and Computer Engineering, University of California, Los Angeles, California

Correspondence
Panagiotis D. Christofides, Department of Chemical and Biomolecular Engineering and the Department of Electrical and Computer Engineering, University of California, Los Angeles, CA 90095, USA.
Email: pdc@seas.ucla.edu

Abstract
This work explores the design of distributed model predictive control (DMPC) systems for nonlinear processes using machine learning models to predict nonlinear dynamic behavior. Specifically, sequential and iterative DMPC systems are designed and analyzed with respect to closed-loop stability and performance properties. Extensive open-loop data within a desired operating region are used to develop long short-term memory (LSTM) recurrent neural network models with a sufficiently small modeling error from the actual nonlinear process model. Subsequently, these LSTM models are utilized in Lyapunov-based DMPC to achieve efficient real-time computation time while ensuring closed-loop state boundedness and convergence to the origin. Using a nonlinear chemical process network example, the simulation results demonstrate the improved computational efficiency when the process is operated under sequential and iterative DMPCs while the closed-loop performance is very close to the one of a centralized MPC system.

KEYWORDS
distributed computation, model predictive control, nonlinear processes, recurrent neural networks

1 | INTRODUCTION

With the rise of big data analytics, machine learning methodologies have gained increasing recognition and demonstrated successful implementation in many traditional engineering fields. One exemplar use of machine learning techniques in chemical engineering is the identification of process models using recurrent neural networks (RNN), which has shown effectiveness in modeling nonlinear dynamic systems. RNN is a class of artificial neural networks that, by using feedback loops in its neurons and passing on past information derived from earlier inputs to the current network, can represent temporal dynamic behaviors. Neural network methods have shown effectiveness in solving both classification and regression problems. For example, feed-forward neural network (FNN) models have shown effectiveness in detecting and distinguishing standard types of cyber-attacks during process operation under model predictive control as demonstrated in Reference 1. Long short-term memory (LSTM) networks, which is a type of RNN, was used in Reference 2 to predict short-term traffic flow in intelligent transportation system management and a supervised LSTM model was used to learn the hidden dynamics of nonlinear processes for soft sensor applications in Reference 3. Furthermore, a single-layer RNN called the simplified dual neural network utilizing dual variables was shown to be Lyapunov stable and globally convergent to the optimal solution of any strictly convex quadratic programming problem.[4]

For many large industrial processes and/or novel processes, developing an accurate and comprehensive model that captures the dynamic behavior of the system can be difficult. Even in the case where a deterministic first-principles model is developed based on fundamental understanding, there may be inherent simplifying assumptions involved. Furthermore, during process operation, the model that is employed in model-based control systems to predict the future evolution of the system state may not always remain accurate as time progresses due to unforeseen process changes or large disturbances, causing plant model mismatch that degrades the performance of the control algorithm.[5] Given these considerations, the model identification of a nonlinear process is crucial for safe and robust model-based feedback control, and given sufficient training data, RNN is an effective tool to develop accurate process models from data.

On the other hand, chemical process operation has extensively relied on automated control systems, and the need of accounting for multivariable interactions and input/state constraints has motivated the development of model predictive control (MPC). Moreover, augmentation in sensor information and network-based communication increases the number of decision variables, state variables, and measurement data, which in turn increases the complexity of the control problem and the computation time if a centralized MPC is used. With these considerations in mind, distributed control systems have been developed, where multiple controllers with inter-controller communication are used to cooperatively calculate the control actions and achieve closed-loop plant objectives. In this context, MPC is a natural control framework to implement due to its ability to account for input and state constraints while also considering the actions of other control actuators. In other words, the controllers communicate with each other to calculate their distinct set of manipulated inputs that will collectively achieve the control objectives of the closed-loop system. Many distributed MPC methods have been proposed in the literature addressing the coordination of multiple MPCs that communicate to calculate the optimal input trajectories in a distributed manner (the reader may refer to References 6-8 for reviews of results on distributed MPC, and to Reference 9 for a review of network structure-based decomposition of control and optimization problems). A robust distributed control approach to plant-wide operations based on dissipativity was proposed in References 10 and 11. Depending on the communication network, that is, whether is one-directional or bi-directional, two distributed architectures, namely sequential and iterative distributed MPCs, were proposed in Reference 12. Furthermore, distributed MPC method was also used in Reference 13 to address the problem of introducing new control systems to pre-existing control schemes. In a recent work,[14] a fast and stable non-convex constrained distributed optimization algorithm was developed and applied to distributed MPC. As distributed MPC systems also depend on an accurate process model, the development and implementation of RNN models in distributed MPCs is an important area yet to be explored. In the present work, we introduce distributed control frameworks that employ a LSTM network, which is a particular type of RNN. The distributed control systems are designed via Lyapunov-based model predictive control (LMPC) theory. Specifically, we explore both sequential distributed LMPC systems and iterative distributed LMPC systems, and compare the closed-loop performances with that of a centralized LMPC system.

The remainder of the paper is organized as follows. Preliminaries on notation, the general class of nonlinear systems, and the stabilizing Lypuanov-based controller for the nonlinear process are given in Section 2. The structure and development of RNN and specifically LSTM, as well as Lyapunov-based control using LSTM models are outlined in Section 3. In Section 4, the formulation and proof for recursive feasibility and closed-loop stability of the distributed LMPC systems using an LSTM model as the prediction model are presented. Lastly, Section 5 includes the application to a two-CSTR-in-series process, demonstrating guaranteed closed-loop stability and enhanced computational efficiency of the proposed distributed LMPC systems with respect to the centralized LMPC.

2 | PRELIMINARIES

2.1 | Notation

For the remainder of this manuscript, the notation x^T is used to denote the transpose of x. $|\cdot|$ is used to denote the Euclidean norm of a vector. $L_fV(x)$ denotes the standard Lie derivative $L_fV(x){:=}\frac{\partial V(x)}{\partial x}f(x)$. Set subtraction is denoted by "\", that is, $A\backslash B := \{x \in \mathbf{R}^n \mid x \in A, x \notin B\}$. $\emptyset$ signifies the null set. The function $f(\cdot)$ is of class $\mathcal{C}^1$ if it is continuously differentiable in its domain. 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.

2.2 | Class of systems

In this work, we consider a general class of nonlinear systems in which several distinct sets of manipulated inputs are used, and each distinct set of manipulated inputs is responsible for regulating a specific subsystem of the process. For the simplicity of notation, throughout the manuscript, we consider two subsystems, subsystem 1 and subsystem 2, consisting of states x_1 and x_2, respectively, which are regulated by only u_1 and u_2, respectively. However, extending the analysis to systems with more than two sets of distinct manipulated input vectors (i.e., having more than two subsystems, with each one regulated by one distinct input vector u_j, j = 1, …, M, M > 2) is conceptually straightforward. The class of continuous-time nonlinear systems considered is represented by the following system of first-order nonlinear ordinary differential equations:

$$\dot{x} = Fx,u_1,u_2,w := fx + g_1xu_1 + g_2xu_2 + vxw,\ xt_0 = x_0 \tag{1}$$

where $x \in \mathbf{R}^n$ is the state vector, $u_1 \in \mathbf{R}^{m_1}$ and $u_2 \in \mathbf{R}^{m_2}$ are two separate sets of manipulated input vectors, and $w \in W$ is the disturbance vector with $W := \{w \in \mathbf{R}^r \mid |w| \le w_m, w_m \ge 0\}$. The control action constraints are defined by $u_1 \in U_1 := \left\{u_{1_i}^{\min} \le u_{1_i} \le u_{1_i}^{\max}, i = 1,\ldots,m_1\right\} \subset \mathbf{R}^{m_1}$, and $u_2 \in U_2 := \left\{u_{2_i}^{\min} \le u_{2_i} \le u_{2_i}^{\max}, i = 1,\ldots,m_2\right\} \subset \mathbf{R}^{m_2}$. $f(\cdot)$, $g_1(\cdot)$, $g_2(\cdot)$, and $v(\cdot)$ are sufficiently smooth vector and matrix functions of dimensions $n \times 1$, $n \times m_1$, $n \times m_2$, and $n \times r$, respectively. Throughout the manuscript, the initial time t_0 is taken to be zero (t_0 = 0), and it is assumed that $f(0)$ = 0, and thus, the origin is a steady state of the nominal (i.e., $w(t) \equiv 0$) system of Equation (1) (i.e., (x_s, u_{1s}, u_{2s}) = (0,0,0), where x_s, u_{1s}, and u_{2s} represent the steady state and input vectors).

2.3 | Stability assumptions

We assume that there exist stabilizing control laws $u_1 = \Phi_1(x) \in U_1$, $u_2 = \Phi_2(x) \in U_2$ (e.g., the universal Sontag control law[15]) such that the origin of the nominal system of Equation (1) with $w(t) \equiv 0$ is rendered exponentially stable in the sense that there exists a $\mathcal{C}^1$ Control Lyapunov function $V(x)$ such that the following inequalities hold for all x in an open neighborhood D around the origin:

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

$$\frac{\partial V(x)}{\partial x}F(x,\Phi_1(x),\Phi_2(x),0) \leq -c_3|x|^2, \tag{2b}$$

$$\left|\frac{\partial V(x)}{\partial x}\right| \leq c_4\,|x| \tag{2c}$$

where c_1, c_2, c_3, and c_4 are positive constants. $F(x, u_1, u_2, w)$ represents the nonlinear system of Equation (1). A set of candidate controllers $\Phi_1(x) \in \mathbf{R}^{m_1}$ and $\Phi_2(x) \in \mathbf{R}^{m_2}$, both denoted by $\Phi_k(x)$ where k = 1, 2, is given in the following form:

$$\phi_{k_i}(x) = \begin{cases} -\dfrac{p+\sqrt{p^2+q^4}}{q^Tq}q & \text{if } q \neq 0 \\ 0 & \text{if } q = 0 \end{cases} \tag{3a}$$

$$\Phi_{k_i}(x) = \begin{cases} u_{k_i}^{min} & \text{if } \phi_{k_i}(x) < u_{k_i}^{min} \\ \phi_{k_i}(x) & \text{if } u_{k_i}^{min} \leq \phi_{k_i}(x) \leq u_{k_i}^{max} \\ u_{k_i}^{max} & \text{if } \phi_{k_i}(x) > u_{k_i}^{max} \end{cases} \tag{3b}$$

where k = 1, 2 represents the two candidate controllers, p denotes $L_fV(x)$ and q denotes $L_{g_{k_i}}V(x)$, $f = [f_1 \cdots f_n]^T$, $g_{k_i} = [g_{k_{i1}},...,g_{k_{in}}]^T$, ($i$ = 1, 2, ..., m_1 for k = 1 corresponding to the vector of control actions $\Phi_1(x)$, and i = 1, 2, ..., m_2 for k = 2 corresponding to the vector of control actions $\Phi_2(x)$). $\phi_{k_i}(x)$ of Equation (3a) represents the i^{th} component of the control law $\phi_k(x)$. $\Phi_{k_i}(x)$ of Equation (3) represents the i_{th} component of the saturated control law $\Phi_k(x)$ that accounts for the input constraints $u_k \in U_k$.

Based on Equation (2), we can first characterize a region where the time-derivative of V is rendered negative under the controller $\Phi_1(x) \in U_1$, $\Phi_2(x) \in U_2$ as $D = \left\{x \in \mathbf{R}^n \mid \dot{V}(x) = L_fV + L_{g_1}Vu_1 + L_{g_2}Vu_2 < -c_3|x|^2, u_1 = \Phi_1(x) \in U_1, u_2 = \Phi_2(x) \in U_2\right\} \cup \{0\}$. Then the closed-loop stability region Ω_ρ for the nonlinear system of Equation (1) is defined as a level set of the Lyapunov function, which is inside D: $\Omega_\rho := \{x \in D \mid V(x) \leq \rho\}$, where $\rho > 0$ and $\Omega_\rho \subset D$. Also, the Lipschitz property of $F(x, u_1, u_2, w)$ combined with the bounds on u_1, u_2, and w implies that there exist positive constants M, $L_x, L_w, L_x^{'}, L_w^{'}$ such that the following inequalities hold $\forall x,\ x^{'} \in \Omega_\rho,\ u_1 \in U_1,\ u_2 \in U_2$ and $w \in W$:

$$|F(x,u_1,u_2,w)| \leq M \tag{4a}$$

$$|F(x,u_1,u_2,w) - F(x',u_1,u_2,0)| \leq L_x|x-x'| + L_w|w| \tag{4b}$$

$$\left|\frac{\partial V(x)}{\partial x}F(x,u_1,u_2,w) - \frac{\partial V(x')}{\partial x}F(x',u_1,u_2,0)\right| \leq L_x^{'}|x-x'| + L_w^{'}|w| \tag{4c}$$

3 | LSTM NETWORK

In this work, we develop an LSTM network model with the following form:

$$\dot{\hat{x}} = F_{nn}(\hat{x},u_1,u_2) := A\hat{x} + \Theta^T y \tag{5}$$

where $\hat{x} \in \mathbf{R}^n$ is the predicted state vector, and $u_1 \in \mathbf{R}^{m_1}$ and $u_2 \in \mathbf{R}^{m_2}$ are the two separate sets of manipulated input vectors. $y^T = [y_1,...,y_n,y_{n+1},...,y_{n+m_1},y_{n+m_1+1}...,y_{n+m_1+m_2},y_{n+m_1+m_2+1}] = \left[H(\hat{x}_1),...,H(\hat{x}_n),u_{1_1},...,u_{1_{m_1}},u_{2_1},...,u_{2_{m_2}},1\right] \in \mathbf{R}^{n+m_1+m_2+1}$ is a vector of the network state $\hat{x}$, where $H(\cdot)$ represents a series of interacting nonlinear activation functions in each LSTM unit, the inputs u_1 and u_2, and the constant 1 which accounts for the bias term. A is a diagonal coefficient matrix, that is, $A = diag\{-\alpha_1, ..., -\alpha_n\} \in \mathbf{R}^{n\times n}$, and $\Theta = [\theta_1,...,\theta_n] \in \mathbf{R}^{(n+m_1+m_2+1)\times n}$ with $\theta_i = \beta_i[\omega_{i1},...,\omega_{i(n+m_1+m_2)},b_i]$, i = 1, ..., n. α_i and β_i are constants, and ω_{ij} is the weight connecting the jth input to the ith neuron where i = 1, ..., n and j = 1, ..., $(n+m_1+m_2)$, and b_i is the bias term for i = 1, ..., n. We use x to represent the state of actual nonlinear system of Equation (1) and use $\hat{x}$ for the state of the LSTM model of Equation (5). Here, α_i is assumed to be positive such that each state $\hat{x}_i$ is bounded-input bounded-state stable.

Instead of having one-way information flow from the input layer to the output layer in a FNN, RNNs introduce feedback loops into the network and allow information exchange in both directions between modules. Unlike FNNs, RNNs take advantage of the feedback signals to store outputs derived from past inputs, and together with the current input information, give a more accurate prediction of the current output. By having access to information of the past, RNN is capable of representing dynamic behaviors of time-series samples, therefore it is an effective method used to model nonlinear processes. Based on the universal approximation theorem, it can be shown that the RNN model with sufficient number of neurons is able to approximate any nonlinear dynamic system on compact subsets of the state-space for finite time.[16,17] However, in a standard RNN model, the problem of vanishing gradient phenomena often arises due to the network's difficulty to capture long term dependencies; this is because of multiplicative gradients that can be exponentially decaying with respect to the number of layers. Therefore, the stored information over extended time intervals is very limited in a short term memory manner. Due to these considerations, Hochreiter and Schmidhuber[18] proposed the LSTM network, which is a type of RNN that uses three gated units (the forget gate, the input gate, and the output gate) to protect and control the memory cell state, $c(k)$, where k = 1, ..., T, such that information will be stored and remembered for long periods of time.[18] For these reasons, LSTM networks may perform better when modeling processes where inputs toward the beginning of a long time-series sequence are crucial to the prediction of outputs toward the end of the sequence. This may be more prevalent in large-scale systems where there may exist inherent time delays between subsystems, causing discrepancies in the speed of the dynamics between the subsystems. The basic architecture of an LSTM network is illustrated in Figure 1a. We develop an LSTM network model to approximate the class of continuous-time nonlinear processes

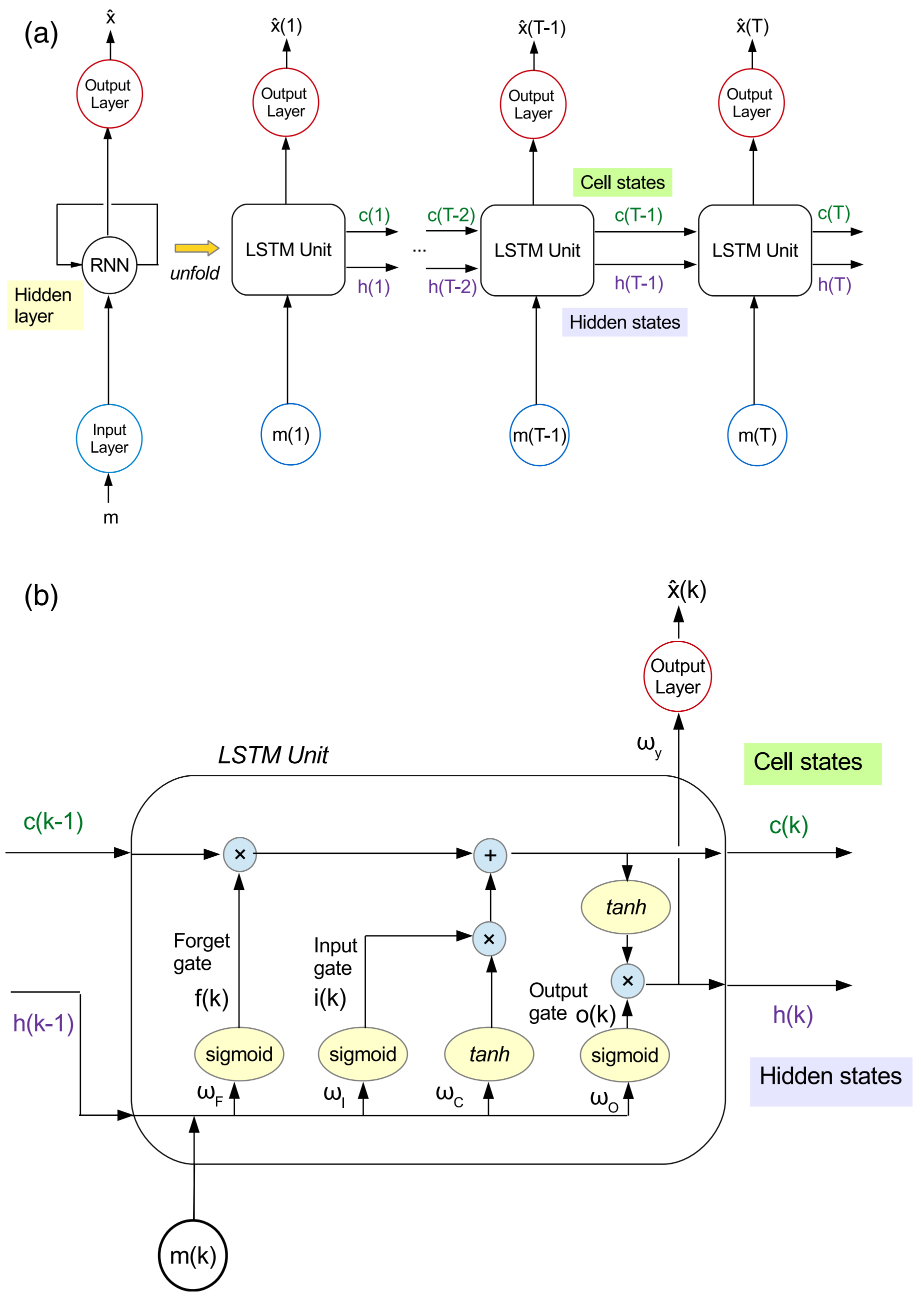

FIGURE 1 Structure of (a) the unfolded LSTM network and (b) the internal setup of an LSTM unit [Color figure can be viewed at wileyonlinelibrary.com]

of Equation (1). We use $m \in \mathbf{R}^{(n+m_1+m_2)\times T}$ to denote the matrix of input sequences to the LSTM network, and $\hat{x} \in \mathbf{R}^{n\times T}$ to denote the matrix of network output sequences. The output from each repeating module that is passed onto the next repeating module in the unfolded sequence is the hidden state, and the vector of hidden states is denoted by *h*. The network output $\hat{x}$ at the end of the prediction period is dependent on all internal states *h*(1), …, *h*(*T*), where the number of internal states *T* (i.e., the number of repeating modules) corresponds to the length of the time-series input sample. The LSTM network calculates a mapping from the input sequence *m* to the output sequence $\hat{x}$ by calculating the following equations iteratively from *k* = 1 to *k* = *T*:

$$i(k) = \sigma\left(\omega_i^m m(k) + \omega_i^h h(k-1) + b_i\right) \tag{6a}$$

$$f(k) = \sigma\left(\omega_f^m m(k) + \omega_f^h h(k-1) + b_f\right) \tag{6b}$$

$$c(k) = f(k)c(k-1) + i(k)\tanh\left(\omega_c^m m(k) + \omega_c^h h(k-1) + b_c\right) \tag{6c}$$

$$o(k) = \sigma\left(\omega_o^m m(k) + \omega_o^h h(k-1) + b_o\right) \tag{6d}$$

$$h(k) = o(k)\tanh(c(k)) \tag{6e}$$

$$\hat{x}(k) = \omega_y h(k) + b_y \tag{6f}$$

where $\sigma(\cdot)$ is the sigmoid function, *tanh*(·) is the hyperbolic tangent function; both of which are activation functions. *h*(*k*) is the internal state, and $\hat{x}(k)$ is the output from the repeating LSTM module with ω_y and b_y denoting 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; correspondingly, 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 vectors 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. Furthermore, $c(k)$ is the cell state which stores information to be passed down the network 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 in the cell state activation function, respectively. The series of interacting nonlinear functions carried out in each LSTM unit, outlined in Equation (6), can be represented by $H(\hat{x})$. The internal structure of a repeating module within an LSTM network where the iterative calculations of Equation (6) are carried out is shown in Figure 1b.

The closed-loop simulation of the continuous-time nonlinear system of Equation (1) is carried out in a sample-and-hold manner, where the feedback measurement of the closed-loop state x is received by the controller every sampling period Δ. Furthermore, state information of the simulated nonlinear process is obtained via numerical integration methods, for example, explicit Euler, using an integration time step of h_c. Since the objective of developing the LSTM model is its eventual utilization in a controller, the prediction period of the LSTM model is set to be the same as the sampling period Δ of the model predictive controller. The time interval between two consecutive internal states within the LSTM can be chosen to be a multiple q_{nn} of the integration time step h_c used in numerical integration of the nonlinear process, with the minimum time interval being $q_{nn} = 1$, that is, $1 \times h_c$. Therefore, depending on the choice of q_{nn}, the number of internal states, T, will follow $T = \frac{\Delta}{q_{nn} \cdot h_c}$. Given that the input sequences fed to the LSTM network are taken at time $t = t_k$, the future states predicted by the LSTM network, $\hat{x}(t)$, at $t = t_k + \Delta$, would be the network output vector at $k = T$, i.e., $\hat{x}(t_k + \Delta) = \hat{x}(T)$. The LSTM learning algorithm is developed to obtain the optimal parameter matrix Γ^*, 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. Under this optimal parameter matrix, the error between the actual state $x(t)$ of the nominal system of Equation (1) (i.e., $w(t) \equiv 0$) and the modeled states $\hat{x}(t)$ of the LSTM model of Equation (5) is minimized. The LSTM model is developed using a state-of-the-art application program interface, that is, Keras, which contains open-source neural network libraries. The mean absolute percentage error between $x(t)$ and $\hat{x}(t)$ is minimized using the adaptive moment estimation optimizer, that is, Adam in Keras, in which the gradient of the error cost function is evaluated using back-propagation. Furthermore, in order to ensure that the trained LSTM model can sufficiently represent the nonlinear process of Equation (1), which in turn ascertains that the LSTM model can be used in a model-based controller to stabilize the actual nonlinear process at its steady-state with guaranteed stability properties, a constraint on the modeling error is also imposed during training, where $|\nu| = |F(x, u_1, u_2, 0) - F_{nn}(x, u_1, u_2)| \leq \gamma |x|$, with $\gamma > 0$. Additionally, to avoid over-fitting of the LSTM model, the training process is terminated once the modeling error falls below the desired threshold and the error on the validation set stops decreasing. One way to assess the modeling error $\nu = F(x(t_k), u_1, u_2, 0) - F_{nn}(x(t_k), u_1, u_2)$ is through numerical approximation using the forward finite difference method. Given that the time interval between internal states of the LSTM model is a multiple of the integration time step $q_{nn} \times h_c$, the time derivative of the LSTM predicted state $\hat{x}(t)$ at $t = t_k$ can be approximated by $\dot{\hat{x}}(t_k) = F_{nn}(x(t_k), u_1, u_2) \approx \frac{\hat{x}(t_k + q_{nn}h_c) - \hat{x}(t_k)}{q_{nn}h_c}$. The time derivative of the actual state $x(t)$ at $t = t_k$ can be approximated by $\dot{x}(t_k) = F(x(t_k), u_1, u_2, 0) \approx \frac{x(t_k + q_{nn}h_c) - x(t_k)}{q_{nn}h_c}$. At time $t = t_k$, $\hat{x}(t_k) = x(t_k)$, the constraint $|\nu| \leq \gamma |x|$ can be written as follows:

$$|\nu| = |F(x(t_k), u_1, u_2, 0) - F_{nn}(x(t_k), u_1, u_2)| \tag{7a}$$

$$\approx \left| \frac{x(t_k + q_{nn}h_c) - \hat{x}(t_k + q_{nn}h_c)}{q_{nn}h_c} \right| \tag{7b}$$

$$\leq \gamma |x(t_k)| \tag{7c}$$

which will be satisfied if $\left| \frac{x(t_k + q_{nn}h_c) - \hat{x}(t_k + q_{nn}h_c)}{x(t_k)} \right| \leq \gamma q_{nn} h_c$. Therefore, the mean absolute percentage error between the predicted states $\hat{x}$ and the targeted states x in the training data will be used as a metric to assess the modeling error of the LSTM model. While the error bounds that the LSTM network model and the actual process should satisfy to ensure closed-loop stability are difficult to calculate explicitly and are, in general, conservative, they provide insight into the key network parameters that will need to be tuned to reduce the error between the two models as well as the amount of data needed to build a suitable LSTM model.

In order to gather adequate training data to develop the LSTM model for the nonlinear process, we first discretize the desired operating region in state-space with sufficiently small intervals as well as discretize the range of manipulated inputs based on the control actuator limits. We run open-loop simulations for the nonlinear process of Equation (1) starting from different initial conditions inside the desired operating region, that is, $x_0 \in \Omega_\rho$, for finite time using combinations of the different manipulated inputs $u_1 \in U_1$, $u_2 \in U_2$ applied in a sample-and-hold manner, and the evolving state trajectories are recorded at time intervals of size $q_{nn} \times h_c$. We obtain enough samples of such trajectories to sweep over all the values that the states and the manipulated inputs (x, u_1, u_2) could take to capture the dynamics of the process. These time-series data can be separated into samples with a fixed length T, which corresponds to the prediction period of the LSTM model, where $\Delta = T \times q_{nn} \times h_c$. The time interval between two time-series data points in the sample $q_{nn} \times h_c$ corresponds to the time interval between two consecutive memory units in the LSTM network. The generated dataset is then divided into training and validation sets.

Remark 1 The actual nonlinear process is a continuous-time model that can be represented using Equation (1); therefore, to characterize the modeling error ν between the LSTM network and the nonlinear process of Equation (1), the LSTM network is represented as a continuous-time model of Equation (5). However, the series of interacting nonlinear operations in the LSTM memory unit is carried out recursively akin to a discrete-time model. The time interval $q_{nn} \times h_c$ between two LSTM memory units is given by the time interval between two consecutive time-series data points in the training samples. Since the LSTM network provides a predicted state at each time interval $q_{nn} \times h_c$ 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, we can use the predicted states from the LSTM network to compare with the predicted states from the nonlinear model of Equation (1) to assess the modeling error. The modeling error is subject to the constraint of Equation (7) to ensure that the LSTM model can be used in the model-based controller with guaranteed stability properties.

3.1 | Lyapunov-based control using LSTM models

Once we obtain an LSTM model with a sufficiently small modeling error, we can design a stabilizing feedback controller $u_1 = \Phi_{nn_1}(x) \in U_1$ and $u_2 = \Phi_{nn_2}(x) \in U_2$ that can render the origin of the LSTM model of Equation (5) exponentially stable in an open neighborhood $\hat{D}$ around the origin 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 $\hat{D}$:

$$\hat{c}_1 |x|^2 \le \hat{V}(x) \le \hat{c}_2 |x|^2, \tag{8a}$$

$$\frac{\partial \hat{V}(x)}{\partial x} F_{nn}(x, \Phi_{nn_1}(x), \Phi_{nn_2}(x)) \le -\hat{c}_3 |x|^2, \tag{8b}$$

$$\left| \frac{\partial \hat{V}(x)}{\partial x} \right| \le \hat{c}_4 \, |x| \tag{8c}$$

where $\hat{c}_1$, $\hat{c}_2$, $\hat{c}_3$, $\hat{c}_4$ are positive constants, and $F_{nn}(x, u_1, u_2)$ represents the LSTM network model of Equation (5). Similar to the characterization method of the closed-loop stability region Ω_ρ for the nonlinear system of Equation (1), we first search the entire state-space to characterize a set of states $\hat{D}$ where the following inequality holds: $\dot{\hat{V}}(x) = \frac{\partial \hat{V}(x)}{\partial x} F_{nn}(x, u_1, u_2) < -\hat{c}_3 |x|^2$, $u_1 = \Phi_{nn_1}(x) \in U_1$, $u_2 = \Phi_{nn_2}(x) \in U_2$. The closed-loop stability region for the LSTM network model of Equation (5) is defined as a level set of Lyapunov function inside $\hat{D}$: $\Omega_{\hat{\rho}} := \left\{ x \in \hat{D} \mid \hat{V}(x) \le \hat{\rho} \right\}$, where $\hat{\rho} > 0$. Starting from $\Omega_{\hat{\rho}}$, the origin of the LSTM network model of Equation (5) can be rendered exponentially stable under the controller $u_1 = \Phi_{nn_1}(x) \in U_1$, and $u_2 = \Phi_{nn_2}(x) \in U_2$. It is noted that the above assumption of Equation (8) is the same as the assumption of Equation (2) for the general class of nonlinear systems of Equation (1) since the LSTM network model of Equation (5) can be written in the form of Equation (1) (i.e., $\dot{\hat{x}} = \hat{f}(\hat{x}) + \hat{g}(\hat{x})u$, where $\hat{f}(\cdot)$ and $\hat{g}(\cdot)$ are obtained from coefficient matrices A and Θ in Equation (5)). However, due to the complexity of the LSTM structure and the interactions of the nonlinear activation functions, $\hat{f}$ and $\hat{g}$ may be hard to compute explicitly. For computational convenience, at $t = t_k$, given a set of control actions $u_1(t_k) \in U_1 \backslash \{0\}$ and $u_2(t_k) \in U_2 \backslash \{0\}$ that are applied in a sample-and-hold fashion for the time interval $t \in [t_k, t_k + h_c)$ (h_c is the integration time step), $\hat{f}$ and $\hat{g}$ can be numerically approximated as follows:

$$\hat{f}(x(t_k)) \approx \frac{\int_{t_k}^{t_k+h_c} F_{nn}(x,0,0)dt - x(t_k)}{h_c} \tag{9a}$$

$$\hat{g}_1(x(t_k)) \approx \frac{\int_{t_k}^{t_k+h_c} F_{nn}(x,u_1(t_k),0)dt - \int_{t_k}^{t_k+h_c} F_{nn}(x,0,0)dt}{h_c u_1(t_k)} \tag{9b}$$

$$\hat{g}_2(x(t_k)) \approx \frac{\int_{t_k}^{t_k+h_c} F_{nn}(x,0,u_2(t_k))dt - \int_{t_k}^{t_k+h_c} F_{nn}(x,0,0)dt}{h_c u_2(t_k)} \tag{9c}$$

The integral $\int_{t_k}^{t_k+h_c} F_{nn}(x, u_1, u_2)dt$ gives the predicted state $\hat{x}(t)$ at $t = t_k + h_c$ under the sample-and-hold implementation of the inputs $u_1(t_k)$ and $u_2(t_k)$; $\hat{x}(t_k + h_c)$ is the first internal state of the LSTM network, given that the time interval between consecutive internal states of the LSTM network is chosen as the integration time step h_c. After obtaining $\hat{f}$, $\hat{g}_1$, and $\hat{g}_2$, the stabilizing control law $\Phi_{nn_1}(x)$ and $\Phi_{nn_2}(x)$ can be computed similarly as in Equation (3), where f, g_1, and g_2 are replaced by $\hat{f}$, $\hat{g}_1$, and $\hat{g}_2$, respectively. Subsequently, $\dot{\hat{V}}$ can also be computed using the approximated $\hat{f}$, $\hat{g}_1$, and $\hat{g}_2$. The assumptions of Equations (2) and (8) are the stabilizability requirements of the first-principles model of Equation (1) and the LSTM network model of Equation (5), respectively. Since the dataset for developing the LSTM network model is generated from open-loop simulations for $x \in \Omega_\rho$, $u_1 \in U_1$, and $u_2 \in U_2$, the closed-loop stability region of the LSTM system is a subset of the closed-loop stability region of the actual nonlinear system, $\Omega_{\hat{\rho}} \subseteq \Omega_\rho$. Additionally, there exist positive constants M_{nn} and L_{nn} such that the following inequalities hold for all $x, x' \in \Omega_{\hat{\rho}}$, $u_1 \in U_1$ and $u_2 \in U_2$:

$$|F_{nn}(x, u_1, u_2)| \le M_{nn} \tag{10a}$$

$$\left| \frac{\partial \hat{V}(x)}{\partial x} F_{nn}(x, u_1, u_2) - \frac{\partial \hat{V}(x')}{\partial x} F_{nn}(x', u_1, u_2) \right| \le L_{nn} \, |x - x'| \tag{10b}$$

4 | DISTRIBUTED LMPC USING LSTM NETWORK MODELS

To achieve better closed-loop control performance, some level of communication may be established between the different controllers. In a distributed LMPC framework, we design two separate LMPCs—LMPC 1 and LMPC 2—to compute control actions u_1 and u_2, respectively; the trajectories of control actions computed by LMPC 1 and LMPC 2 are denoted by u_{d_1} and u_{d_2}, respectively. We consider two types of distributed control architectures: sequential and iterative distributed MPCs. Having only one-way communication, the sequential distributed MPC architecture carries less computational load with transmitting inter-controller signals. However, it must assume the

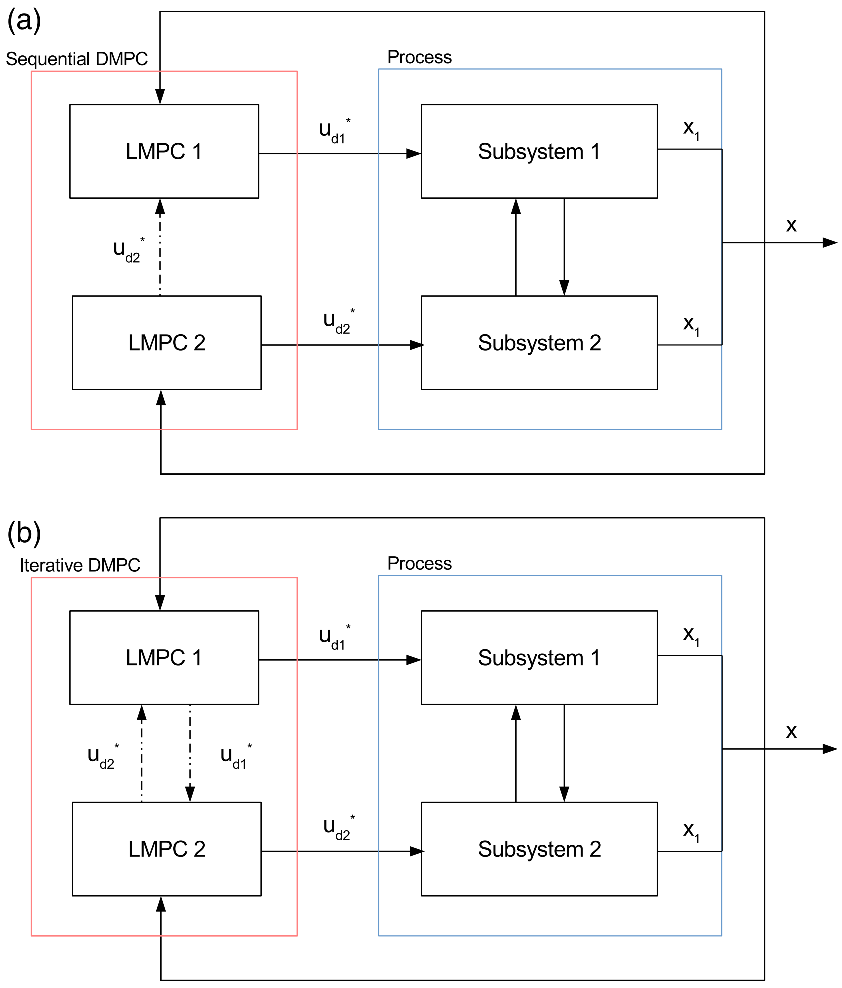

FIGURE 2 Schematic diagrams showing the flow of information of (a) the sequential distributed LMPC and (b) the iterative distributed LMPC systems with the overall process [Color figure can be viewed at wileyonlinelibrary.com]

input trajectories along the prediction horizon of the other controllers downstream of itself in order to make a decision. The iterative distributed MPC system allows signal exchanges between all controllers, thereby allowing each controller to have full knowledge of the predicted state evolution along the prediction horizon and yielding better closed-loop performance via multiple iterations at the cost of more computational time.

4.1 | Sequential distributed LMPC using LSTM network models

The communication between two LMPCs in a sequential distributed LMPC framework is one-way only; that is, the optimal control actions obtained from solving the optimization problem of one LMPC will be relayed to the other LMPC, which will use this information to carry on with its own optimization problem. A schematic diagram of the structure of a sequential distributed LMPC system is shown in Figure 2a. In a sequential distributed LMPC system, the following implementation strategy is used:

1. At each sampling instant $t = t_k$, both LMPC 1 and LMPC 2 receive the state measurement $x(t)$, $t = t_k$ from the sensors.

2. LMPC 2 evaluates the optimal trajectory of u_{d_2} based on the state measurement $x(t)$ at $t = t_k$, sends the control action of the first sampling period $u^*_{d_2}(t_k)$ to the corresponding actuators, and sends the entire optimal trajectory to LMPC 1.

3. LMPC 1 receives the entire optimal input trajectory of u_{d_2} from LMPC 2, and evaluates the optimal trajectory of u_{d_1} based on state measurement $x(t)$ at $t = t_k$ and the optimal trajectory of u_{d_2}. LMPC 1 then sends $u^*_{d_1}(t_k)$, the optimal control action over the next sampling period to the corresponding actuators.

4. When a new state measurement is received ($k \leftarrow k + 1$), go to Step 1.

We first define the optimization problem of LMPC 2, which uses the LSTM network model as its prediction model. LMPC 2 depends on the latest state measurement, but does not have any information on the value that u_{d_1} will take. Thus, to make a decision, LMPC 2 must assume a trajectory for u_{d_1} along the prediction horizon. An explicit nonlinear control law, $\Phi_{nn_1}(x)$, is used to compute the assumed trajectory of u_{d_1}. To inherit the stability properties of $\Phi_{nn_j}(x), j = 1,2$, u_{d_2} must satisfy a Lyapunov-based contractive constraint that guarantees a minimum decrease rate of the Lyapunov function $\hat{V}$. The optimization problem of LMPC 2 is given as follows:

$$\mathcal{J} = \min_{u_{d_2} \in S(\Delta)} \int_{t_k}^{t_{k+N}} L\left(\tilde{x}(t), \Phi_{nn_1}(\tilde{x}(t)), u_{d_2}(t)\right) dt \tag{11a}$$

$$\text{s.t.}\;\; \dot{\tilde{x}}(t) = F_{nn}\left(\tilde{x}(t), \Phi_{nn_1}(\tilde{x}(t)), u_{d_2}(t)\right) \tag{11b}$$

$$u_{d_2}(t) \in U_2, \forall t \in [t_k, t_{k+N}) \tag{11c}$$

$$\tilde{x}(t_k) = x(t_k) \tag{11d}$$

$$\begin{aligned}&\frac{\partial \hat{V}(x(t_k))}{\partial x}\left(F_{nn}(x(t_k), \Phi_{nn_1}(x(t_k)), u_{d_2}(t_k))\right)\\ &\le \frac{\partial \hat{V}(x(t_k))}{\partial x}\left(F_{nn}(x(t_k), \Phi_{nn_1}(x(t_k)), \Phi_{nn_2}(x(t_k)))\right),\\ &\text{if } x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_{nn}}\end{aligned} \tag{11e}$$

$$\hat{V}(\tilde{x}(t)) \le \rho_{nn}, \forall t \in [t_k, t_{k+N}), \text{if } x(t_k) \in \Omega_{\rho_{nn}} \tag{11f}$$

where $\tilde{x}$ 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. The optimal input trajectory computed by this LMPC 2 is denoted by $u^*_{d_2}(t)$, which is calculated over the entire prediction horizon $t \in [t_k, t_{k+N})$. This information is sent to LMPC 1. The control action computed for the first sampling period of the prediction horizon $u^*_{d_2}(t_k)$ is sent by LMPC 2 to its control actuators to be applied over the next sampling period. In the optimization problem of Equation (11), the objective function of Equation (11a) is the integral of $L(\tilde{x}(t), \Phi_{nn_1}(t), u_{d_2}(t))$ over the prediction horizon. Note that $L(x, u_1, u_2)$ is typically in a quadratic form, that is, $L(x, u_1, u_2) = x^T Q x + u_1^T R_1 u_1 + u_2^T R_2 u_2$, where Q, R_1, and R_2 are positive definite matrices, and the minimum of the objective function of Equation (11a) is achieved at the origin. The constraint of Equation (11b) is the LSTM network model of Equation (5) that is used to predict the states of the closed-loop system. Equation (11c) defines the input constraints on u_{d_2} applied over the entire prediction horizon. Equation (11d) defines the initial condition $\tilde{x}(t_k)$ of Equation (11b), which is the state measurement at $t = t_k$. The constraint of Equation (11e) forces the closed-loop state to move toward the origin 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 network model of Equation (11b) will be maintained in $\Omega_{\rho_{nn}}$ for the entire prediction horizon.

The optimization problem of LMPC 1 depends on the latest state measurement as well as the control action computed by LMPC 2 (i.e., $u^*_{d_2}(t), \forall t \in [t_k, t_{k+N})$). This allows LMPC 1 to compute a control action u_{d_1} such that the closed-loop performance is optimized while guaranteeing the stability properties of the Lyapunov-based controllers using LSTM network models, $\Phi_{nn_j}(x), j = 1, 2$, are preserved. Specifically, LMPC 1 uses the following optimization problem:

$$\mathcal{J} = \min_{u_{d_1} \in S(\Delta)} \int_{t_k}^{t_{k+N}} L\left(\tilde{x}(t), u_{d_1}(t), u^*_{d_2}(t)\right) dt \tag{12a}$$

$$\text{s.t.}\;\; \dot{\tilde{x}}(t) = F_{nn}\left(\tilde{x}(t), u_{d_1}(t), u^*_{d_2}(t)\right) \tag{12b}$$

$$u_{d_1}(t) \in U_1, \forall t \in [t_k, t_{k+N}) \tag{12c}$$

$$\tilde{x}(t_k) = x(t_k) \tag{12d}$$

$$\begin{aligned}&\frac{\partial \hat{V}(x(t_k))}{\partial x}\left(F_{nn}\left(x(t_k), u_{d_1}(t_k), u^*_{d_2}(t_k)\right)\right)\\ &\le \frac{\partial \hat{V}(x(t_k))}{\partial x}\left(F_{nn}\left(x(t_k), \Phi_{nn_1}(x(t_k)), u^*_{d_2}(t_k)\right)\right),\\ &\text{if } x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_{nn}}\end{aligned} \tag{12e}$$

$$\hat{V}(\tilde{x}(t)) \le \rho_{nn}, \forall t \in [t_k, t_{k+N}), \text{if } x(t_k) \in \Omega_{\rho_{nn}} \tag{12f}$$

where $\tilde{x}$ 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. The optimal input trajectory computed by LMPC 1 is denoted by $u^*_{d_1}(t)$, 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 $u^*_{d_1}(t_k)$ is sent by LMPC 1 to be applied over the next sampling period. In the optimization problem of Equation 12, the objective function of Equation (12a) is the integral of $L\left(\tilde{x}(t), u_{d_1}(t), u^*_{d_2}(t)\right)$ over the prediction horizon. The constraint of Equation (12b) is the LSTM model of Equation (5) that is used to predict the states of the closed-loop system. Equation (12c) defines the input constraints on u_{d_1} applied over the entire prediction horizon. Equation (12d) defines the initial condition $\tilde{x}(t_k)$ of Equation (12b), which is the state measurement at $t = t_k$. The constraint of Equation (12e) forces the closed-loop state to move toward the origin 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 Equation (12b) will be maintained in $\Omega_{\rho_{nn}}$ for the entire prediction horizon. Since the execution of LMPC 1 depends on the results of LMPC 2, the total computation time to execute the sequential distributed LMPC design would be the sum of the time taken to solve each optimization problem in LMPC 1 and LMPC 2, respectively.

4.2 | Iterative distributed LMPC using LSTM network models

In an iterative distributed LMPC framework, both controllers communicate with each other to cooperatively optimize the control actions. The controllers solve their respective optimization problems independently in a parallel structure, and solutions to each control problem are exchanged at the end of each iteration. The schematic diagram of an iterative distributed LMPC system is shown in Figure 2b. More specifically, the following implementation strategy is used:

1. At each sampling instant t_k, both LMPC 1 and LMPC 2 receive the state measurement $x(t)$ at $t = t_k$ from the sensors.

2. At iteration $c = 1$, LMPC 1 evaluates future trajectories of $u_{d_1}(t)$ assuming $u_2(t) = \Phi_{nn_2}(t), \forall t \in [t_k, t_{k+N})$. LMPC 2 evaluates future trajectories of $u_{d_2}(t)$ assuming $u_1(t) = \Phi_{nn_1}(t), \forall t \in [t_k, t_{k+N})$. The LMPCs exchange their future input trajectories, calculate and store the value of their own cost function.

3. At iteration $c > 1$:

(a) Each LMPC evaluates its own future input trajectory based on state measurement $x(t_k)$ and the latest received input trajectories from the other LMPC.

(b) The LMPCs exchange their future input trajectories. Each LMPC calculates and stores the value of the cost function.

4. If a termination criterion is satisfied, each LMPC sends its entire future input trajectory corresponding to the smallest value of the cost function to its actuators. If the termination criterion is not satisfied, go to Step 3 ($c \leftarrow c + 1$).

5. When a new state measurement is received, go to Step 1 ($k \leftarrow k + 1$).

To preserve the stability properties of the Lyapunov-based controllers $\Phi_{nn_j}(x), j = 1,2$, the optimized u_{d_1} and u_{d_2} must satisfy the contractive constraint that guarantees a minimum decrease rate of the Lyapunov function $\hat{V}$ given by $\Phi_{nn_j}(x), j = 1,2$. Following the same variables and constraints as defined in a sequential distributed LMPC design, the optimization problem of LMPC 1 in an iterative distributed LMPC at iteration $c = 1$ is presented as follows:

$$\mathcal{J} = \min_{u_{d_1} \in S(\Delta)} \int_{t_k}^{t_{k+N}} L(\tilde{x}(t), u_{d_1}(t), \Phi_{nn_2}(\tilde{x}(t)))dt \tag{13a}$$

$$\text{s.t.} \quad \dot{\tilde{x}}(t) = F_{nn}(\tilde{x}(t), u_{d_1}(t), \Phi_{nn_2}(\tilde{x}(t))) \tag{13b}$$

$$u_{d_1}(t) \in U_1, \forall t \in [t_k, t_{k+N}) \tag{13c}$$

$$\tilde{x}(t_k) = x(t_k) \tag{13d}$$

$$\begin{aligned}&\frac{\partial \hat{V}(x(t_k))}{\partial x}(F_{nn}(x(t_k), u_{d_1}(t_k), \Phi_{nn_2}(x(t_k))))\\ &\leq \frac{\partial \hat{V}(x(t_k))}{\partial x}(F_{nn}(x(t_k), \Phi_{nn_1}(x(t_k)), \Phi_{nn_2}(x(t_k)))),\end{aligned}$$

$$\text{if } x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_{nn}} \tag{13e}$$

$$\hat{V}(\tilde{x}(t)) \leq \rho_{nn}, \forall t \in [t_k, t_{k+N}), \text{if } x(t_k) \in \Omega_{\rho_{nn}} \tag{13f}$$

At iteration $c = 1$, the optimization problem of LMPC 2 is shown as follows:

$$\mathcal{J} = \min_{u_{d_2} \in S(\Delta)} \int_{t_k}^{t_{k+N}} L(\tilde{x}(t), \Phi_{nn_1}(\tilde{x}(t)), u_{d_2}(t))dt \tag{14a}$$

$$\text{s.t.} \quad \dot{\tilde{x}}(t) = F_{nn}(\tilde{x}(t), \Phi_{nn_1}(\tilde{x}(t)), u_{d_2}(t)) \tag{14b}$$

$$u_{d_2}(t) \in U_2, \forall t \in [t_k, t_{k+N}) \tag{14c}$$

$$\tilde{x}(t_k) = x(t_k) \tag{14d}$$

$$\begin{aligned}&\frac{\partial \hat{V}(x(t_k))}{\partial x}(F_{nn}(x(t_k), \Phi_{nn_1}(x(t_k)), u_{d_2}(t)))\\ &\leq \frac{\partial \hat{V}(x(t_k))}{\partial x}(F_{nn}(x(t_k), \Phi_{nn_1}(x(t_k)), \Phi_{nn_2}(x(t_k)))),\end{aligned}$$

$$\text{if } x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_{nn}} \tag{14e}$$

$$\hat{V}(\tilde{x}(t)) \leq \rho_{nn}, \forall t \in [t_k, t_{k+N}), \text{if } x(t_k) \in \Omega_{\rho_{nn}} \tag{14f}$$

At iteration $c > 1$, following the exchange of the optimized input trajectories $u^*_{d_1}(t)$ and $u^*_{d_2}(t)$ between the two LMPCs, the optimization problem of LMPC 1 is modified as follows:

$$\mathcal{J} = \min_{u_{d_1} \in S(\Delta)} \int_{t_k}^{t_{k+N}} L\left(\tilde{x}(t), u_{d_1}(t), u^*_{d_2}(t)\right)dt \tag{15a}$$

$$\text{s.t.} \quad \dot{\tilde{x}}(t) = F_{nn}\left(\tilde{x}(t), u_{d_1}(t), u^*_{d_2}(t)\right) \tag{15b}$$

$$u_{d_1}(t) \in U_1, \forall t \in [t_k, t_{k+N}) \tag{15c}$$

$$\tilde{x}(t_k) = x(t_k) \tag{15d}$$

$$\begin{aligned}&\frac{\partial \hat{V}(x(t_k))}{\partial x}\left(F_{nn}\left(x(t_k), u_{d_1}(t_k), u^*_{d_2}(t_k)\right)\right.\\ &\leq \frac{\partial \hat{V}(x(t_k))}{\partial x}(F_{nn}(x(t_k), \Phi_{nn_1}(x(t_k)), \Phi_{nn_2}(x(t_k)))),\end{aligned}$$

$$\text{if } x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_{nn}} \tag{15e}$$

$$\hat{V}(\tilde{x}(t)) \leq \rho_{nn}, \forall t \in [t_k, t_{k+N}), \text{if } x(t_k) \in \Omega_{\rho_{nn}} \tag{15f}$$

And the optimization problem of LMPC 2 becomes:

$$\mathcal{J} = \min_{u_{d_2} \in S(\Delta)} \int_{t_k}^{t_{k+N}} L\left(\tilde{x}(t), u^*_{d_1}(t), u_{d_2}(t)\right)dt \tag{16a}$$

$$\text{s.t.} \quad \dot{\tilde{x}}(t) = F_{nn}\left(\tilde{x}(t), u^*_{d_1}(t), u_{d_2}(t)\right) \tag{16b}$$

$$u_{d_2}(t) \in U_2, \forall t \in [t_k, t_{k+N}) \tag{16c}$$

$$\tilde{x}(t_k) = x(t_k) \tag{16d}$$

$$\begin{aligned}&\frac{\partial \hat{V}(x(t_k))}{\partial x}\left(F_{nn}\left(x(t_k), u^*_{d_1}(t), u_{d_2}(t)\right)\right)\\ &\leq \frac{\partial \hat{V}(x(t_k))}{\partial x}(F_{nn}(x(t_k), \Phi_{nn_1}(x(t_k)), \Phi_{nn_2}(x(t_k)))),\end{aligned}$$

$$\text{if } x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_{nn}} \tag{16e}$$

$$\hat{V}(\tilde{x}(t)) \leq \rho_{nn}, \forall t \in [t_k, t_{k+N}), \text{if } x(t_k) \in \Omega_{\rho_{nn}} \tag{16f}$$

At each iteration $c \geq 1$, the two LMPCs can be solved simultaneously via parallel computing in separate processors. Therefore, the total computation time required for iterative distributed LMPC would be the maximum solving time out of the two controllers accounting for all the iterations required before the termination criterion is met.

Remark 2 One consideration that applies to any MPC system is that the computation time to calculate the solutions to the MPC optimization problem(s) must be less than the sampling time of the actual nonlinear process of Equation (1). One of the main advantages of distributed MPC systems is the reduced computational complexity of the optimization problems, and thus, reduced total computational time compared to solving the optimization problem in a centralized MPC system. Therefore, running more iterations to achieve a more optimal set of solutions (i.e., lower value of the cost function) should be balanced with reducing total computation time, and there should be an upper bound enforced on the maximum number of iterations at all times to ensure calculation of control actions within the sampling time.

Remark 3 It is important to note that the number of iterations c could vary and will not affect the closed-loop stability of the iterative distributed LMPC system. The number of iterations c depends on the termination conditions, which can be of many forms, for example, c must not exceed a maximum iteration number c_{max} (i.e., $c \le c_{max}$), the computational time for solving each LMPC must not exceed a maximum time period, or the difference in the cost function or of the solution trajectory between two consecutive iterations is smaller than a threshold value. During implementation, when one such criterion is met, the iterations will be terminated.

Remark 4 In general, there is no guaranteed convergence of the optimal cost or solution of an iterative distributed LMPC system to the optimal cost or solution of a centralized LMPC. This is due to the non-convexity of the MPC optimization problems. However, the proposed implementation strategy guarantees that the optimal cost of the distributed optimization is upper bounded by the cost of the Lyapunov-based control laws $\Phi_{nn_1}(x) \in U_1, \Phi_{nn_2}(x) \in U_2$.

4.3 | Sample-and-hold implementation of distributed LMPC

Once both optimization problems of LMPC 1 and LMPC 2 are solved, the optimal control actions of the proposed distributed LMPC design (both sequential and iterative distributed LMPC systems) are defined as follows:

$$\begin{aligned} u_1(t) &= u^*_{d_1}(t_k),\ \forall t \in [t_k, t_{k+1} \\ u_2(t) &= u^*_{d_2}(t_k),\ \forall t \in [t_k, t_{k+1} \end{aligned} \tag{17}$$

The control actions computed by each LMPC will be applied in a sample-and-hold manner to the process, which may be subject to bounded disturbances (i.e., $|w(t)| \le w_m$). In this section, we present the stability properties of the distributed LMPC design, accounting for sufficiently small bounded modeling error of the LSTM network and bounded disturbances. Following Lyapunov arguments, this property will guarantee practical stability of the closed-loop system, that is, the closed-loop state $x(t)$ of the nominal process of Equation (1) is bounded in $\Omega_{\hat{\rho}}$ at all times, and ultimately driven to a small neighborhood $\Omega_{\rho_{min}}$ around the origin under the control actions in the distributed LMPC design of Equation (17) implemented in a sample-and-hold manner. First, we will present propositions demonstrating the existence of an upper bound on the state error $|e(t)| = |x(t) - \hat{x}(t)|$ provided that the modeling error $|\nu|$ and process disturbances $|w|$ are bounded, followed by propositions that demonstrate the boundedness and convergence of the LSTM system of Equation (5) and of the actual nonlinear system of Equation (1) under the sample-and-hold implementation of $u_1 = \Phi_{nn_1}(x) \in U_1$ and $u_2 = \Phi_{nn_2}(x) \in U_2$. Both propositions have been previously proved in.[19] Then, we will extend the proof to show the boundedness and convergence of the nonlinear system of Equation (1) under the sample-and-hold implementation of $[u_1\ u_2] = \left[u^*_{d_1}\ u^*_{d_2}\right]$ from the distributed LMPC design of Equation (17) in the presence of sufficiently small bounded disturbances and modeling error.

Proposition 1 Consider the nonlinear system $\dot{x} = F(x, u_1, u_2, w)$ of Equation (1) in the presence of bounded disturbances $|w(t)| \le w_m$ and the LSTM model $\dot{\hat{x}} = F_{nn}(\hat{x}, u_1, u_2)$ of Equation (5) with the same initial condition $x_0 = \hat{x}_0 \in \Omega_{\hat{\rho}}$ and sufficiently small modeling error $|\nu| \le \nu_m$. There exists a class $\mathcal{K}$ function $f_w(\cdot)$ and a positive constant κ such that the following inequalities hold $\forall x, \hat{x} \in \Omega_{\hat{\rho}}$ and $w(t) \in W$:

$$|e(t)| = |x(t) - \hat{x}(t)| \le f_w(t) := \frac{L_w w_m + \nu_m}{L_x}\left(e^{L_x t} - 1\right) \tag{18a}$$

$$\hat{V}(x) \le \hat{V}(\hat{x}) + \frac{\hat{c}_4\sqrt{\hat{\rho}}}{\sqrt{\hat{c}_1}}|x - \hat{x}| + \kappa|x - \hat{x}|^2 \tag{18b}$$

It has also been established that under the controller $u_1(t) = \Phi_{nn_1}(x) \in U_1$, $u_2(t) = \Phi_{nn_2}(x) \in U_2$ implemented in a sample-and-hold fashion, the closed-loop state $x(t)$ of the actual process of Equation (1) and the closed-loop state $\hat{x}(t)$ of the LSTM system of Equation (5) are bounded in the stability region and ultimately driven to a small neighborhood around the origin, given that the conditions of Equation 8 are satisfied, and the modeling error $|\nu| \le \gamma|x| \le \nu_m$, where γ is chosen to satisfy $\gamma < \hat{c}_3/\hat{c}_4$. This is shown in the following proposition. The full proof of the following proposition can be found in Reference 19.

Proposition 2 Consider the system of Equation (1) under the controllers $u_j = \Phi_{nn_j}(\hat{x}) \in U_j$, j = 1, 2, which meet the conditions of Equation (8). The controllers $u_j = \Phi_{nn_j}(\hat{x}) \in U_j$, j = 1, 2 are designed to stabilize the LSTM system of Equation (5), developed with a modeling error $|\nu| \le \gamma|x| \le \nu_m$, where $\gamma < \hat{c}_3/\hat{c}_4$.

The control actions are implemented in a sample-and-hold fashion, that is, $u_j(t) = \Phi_{nn_j}(\hat{x}(t_k)), j = 1,2$, $\forall t \in [t_k, t_{k+1})$, where $t_{k+1} := t_k + \Delta$. Let ϵ_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

$$-\frac{\hat{c}_3}{\hat{c}_2}\rho_s + L_{nn}M_{nn}\Delta \le -\epsilon_s \tag{19a}$$

$$-\frac{\tilde{c}_3}{\hat{c}_2}\rho_s + L_x^{'}M\Delta + L_w^{'}w_m \le -\epsilon_w \tag{19b}$$

and

$$\rho_{nn} := \max\left\{\hat{V}(\hat{x}(t+\Delta)) \mid \hat{x}(t) \in \Omega_{\rho_s}, u_1 \in U_1, u_2 \in U_2\right\} \tag{20a}$$

$$\rho_{min} \ge \rho_{nn} + \frac{\hat{c}_4\sqrt{\hat{\rho}}}{\sqrt{\hat{c}_1}} f_w(\Delta) + \kappa(f_w(\Delta))^2 \tag{20b}$$

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

$$\hat{V}(\hat{x}(t)) \le \hat{V}(\hat{x}(t_k)),\ \forall\ t \in [t_k, t_{k+1}) \tag{21a}$$

$$\hat{V}(x(t)) \le \hat{V}(x(t_k)),\ \forall\ t \in [t_k, t_{k+1}) \tag{21b}$$

and if $x_0 \in \Omega_{\hat{\rho}}$, the state $\hat{x}(t)$ of the LSTM modeled system of Equation (5) 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 Equation (1) is bounded in $\Omega_{\hat{\rho}}$ for all times and ultimately bounded in $\Omega_{\rho_{min}}$.

Proposition 2 demonstrates that, if $x(t_k) = \hat{x}(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_s}$, the closed-loop state of the LSTM system of Equation (5) and of the actual nonlinear process of Equation (1) are both bounded in the stability region $\Omega_{\hat{\rho}}$ and they move toward the origin under $u_1(t) = \Phi_{nn_1}(x) \in U_1$ and $u_2(t) = \Phi_{nn_2}(x) \in U_2$ implemented in a sample-and-hold fashion. If $x(t_k) = \hat{x}(t_k) \in \Omega_{\rho_s}$, the closed-loop state of the LSTM model is maintained in $\Omega_{\rho_{nn}}$ within one sampling period for all $t \in [t_k, t_{k+1})$, and the closed-loop state of the actual nonlinear system is maintained in $\Omega_{\rho_{min}}$ within one sampling period.

In the following theorem, we will prove that the optimization problem of LMPC 1 and of LMPC 2 in the distributed LMPC network can be solved with recursive feasibility, and the closed-loop stability of the nonlinear system of Equation (1) is guaranteed under the sample-and-hold implementation of the optimal control actions $[u_1\ u_2] = \left[u_{d_1}^*\ u_{d_2}^*\right]$ given by the distributed LMPC design of Equation (17).

Theorem 1 *Consider the closed-loop system of Equation (1) under* $[u_1\ u_2] = \left[u_{d_1}^*\ u_{d_2}^*\right]$ *in the distributed LMPC design of Equation (17), which are calculated based on the controllers* $\Phi_{nn_j}(x), j = 1,2$ *that satisfy Equation (8). Let* $\Delta > 0$, $\epsilon_s > 0$, $\epsilon_w > 0$, *and* $\hat{\rho} > \rho_{min} > \rho_{nn} > \rho_s$ *satisfy Equations (19) and (20). Then, given any initial state* $x_0 \in \Omega_{\hat{\rho}}$, *if the conditions of Proposition 1 and Proposition 2 are satisfied, and the LSTM model of Equation (5) has a modeling error* $|\nu| \le \gamma|x| \le \nu_m$, $0 < \gamma < \hat{c}_3/\hat{c}_4$, *then there always exists a feasible solution for the optimization problem of Equations (11), (12), (15), (16). Additionally, it is guaranteed that under the distributed LMPC design* $[u_1\ u_2] = \left[u_{d_1}^*\ u_{d_2}^*\right]$ *of Equation (17),* $x(t) \in \Omega_{\hat{\rho}}, \forall t \ge 0$, *and* $x(t)$ *ultimately converges to* $\Omega_{\rho_{min}}$ *for the closed-loop system of Equation (1).*

Proof. The proof consists of three parts. In *Part* 1, we first prove that the optimization problem of each LMPC in the distributed LMPC network is feasible for all states $x \in \Omega_{\hat{\rho}}$. In *Part* 2, we prove the boundedness and convergence of the state in $\Omega_{\rho_{nn}}$ for the closed-loop LSTM system of Equation (5) under the distributed LMPC design $[u_1\ u_2] = \left[u_{d_1}^*\ u_{d_2}^*\right]$ in Equation (17). Lastly, in *Part* 3, we prove the boundedness and convergence of the closed-loop state to $\Omega_{\rho_{min}}$ for the actual nonlinear system of Equation (1) under the distributed LMPC design $[u_1\ u_2] = \left[u_{d_1}^*\ u_{d_2}^*\right]$ in Equation (17). The following proof is provided in reference to the formulations of the sequential distributed LMPC of Equations (11) and (12), but the same result also applies to the iterative distributed LMPC of Equations (15) and (16).

Part 1: We prove that the optimization problem of each LMPC in the distributed LMPC network is recursively feasible for all $x \in \Omega_{\hat{\rho}}$. If $x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_{nn}}$, the input trajectories $u_{d_j}(t) = \Phi_{nn_j}(x(t_k))$, j = 1, 2, for $t \in [t_k, t_{k+1}]$ are feasible solutions to the optimization problem of LMPC j since such trajectories satisfy the input constraint on u_{d_j} of Equation (11c) in LMPC 2 and of Equation (12c) in LMPC 1, respectively, as well as the Lyapunov-based contractive constraint of Equation (11e) in LMPC 2 and of Equation (12e) in LMPC 1. Additionally, if $x(t_k) \in \Omega_{\rho_{nn}}$, the control actions given by $\Phi_{nn_j}(\tilde{x}(t_{k+i}))$, i = 0, 1, ..., $N-1$ satisfy the input constraint on u_{d_2} of Equation (11c) and the Lyapunov-based constraint of Equation (11f) in LMPC 2, and the input constraint on u_{d_1} of Equation (12c) and the Lyapunov-based constraint of Equation (12f) in LMPC 1, since it is shown in Proposition 2 that the states predicted by the LSTM model of Equation (11b) and of Equation (12b) remain inside $\Omega_{\rho_{nn}}$ under the controller $\Phi_{nn_j}(\tilde{x}), j = 1,2$. Therefore, for all $x_0 \in \Omega_{\hat{\rho}}$, the optimization problems of both Equations (12) and (11) can be solved with recursive feasibility if $x(t) \in \Omega_{\hat{\rho}}$ for all times.

Part 2: Next, we prove that given any $x_0 = \hat{x}_0 \in \Omega_{\hat{\rho}}$, the state of the closed-loop LSTM system of Equation (5) is bounded in $\Omega_{\hat{\rho}}$ for all times and ultimately converges to a small neighborhood around the origin $\Omega_{\rho_{nn}}$ defined by Equation (20a) under the sample-and-hold implementation of the distributed LMPC design $[u_1\ u_2] = \left[u_{d_1}^*\ u_{d_2}^*\right]$ of Equation (17). First, we consider $x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_{nn}}$ at $t = t_k$, therefore activating the contractive constraints of Equation (11e) and Equation (12e). Based on the definition of ρ_{nn} in Equation (20a), this means $x(t_k)$ also belongs to the region $\Omega_{\hat{\rho}} \backslash \Omega_{\rho_s}$. With the conditions of Equation 8 on $\Phi_{nn_1}(\hat{x}(t_k))$ and $\Phi_{nn_2}(\hat{x}(t_k))$ satisfied, the contractive constraints are activated such that the optimal control actions $u_{d_2}^*$, and sequentially $u_{d_1}^*$, are calculated to decrease the value of the Lyapunov function based on the states predicted by the LSTM model of Equation (11b) and Equation (12b) over the next sampling period, respectively. This is shown as follows:

$$\begin{aligned}\dot{\hat{V}}(\hat{x}(t_k)) = & \frac{\partial \hat{V}(\hat{x}(t_k))}{\partial \hat{x}} F_{nn}\left(\hat{x}(t), u_{d_1}^*(t_k), u_{d_2}^*(t_k)\right) \\ & \leq \frac{\partial \hat{V}(\hat{x}(t_k))}{\partial \hat{x}} F_{nn}\left(\hat{x}(t), \Phi_{nn_1}(\hat{x}(t_k)), u_{d_2}^*(t_k)\right) \\ & \leq \frac{\partial \hat{V}(\hat{x}(t_k))}{\partial \hat{x}} F_{nn}(\hat{x}(t), \Phi_{nn_1}(\hat{x}(t_k)), \Phi_{nn_2}(\hat{x}(t_k))) \\ & \leq -\hat{c}_3|\hat{x}(t_k)|^2\end{aligned} \tag{22}$$

The time derivative of the Lyapunov function along the trajectory of $\hat{x}(t)$ of the LSTM model of Equation (5) in $t \in [t_k, t_{k+1})$ is given by:

$$\begin{aligned}\dot{\hat{V}}(\hat{x}(t)) = & \frac{\partial \hat{V}(\hat{x}(t))}{\partial \hat{x}} F_{nn}\left(\hat{x}(t), u_{d_1}^*(t_k), u_{d_2}^*(t_k)\right) \\ & = \frac{\partial \hat{V}(\hat{x}(t_k))}{\partial \hat{x}} F_{nn}\left(\hat{x}(t_k), u_{d_1}^*(t_k), u_{d_2}^*(t_k)\right) \\ & + \frac{\partial \hat{V}(\hat{x}(t))}{\partial \hat{x}} F_{nn}\left(\hat{x}(t), u_{d_1}^*(t_k), u_{d_2}^*(t_k)\right) \\ & - \frac{\partial \hat{V}(\hat{x}(t_k))}{\partial \hat{x}} F_{nn}\left(\hat{x}(t_k), u_{d_1}^*(t_k), u_{d_2}^*(t_k)\right)\end{aligned} \tag{23}$$

After adding and subtracting $\frac{\partial \hat{V}(\hat{x}(t_k))}{\partial \hat{x}} F_{nn}\left(\hat{x}(t_k), u_{d_1}^*(t_k), u_{d_2}^*(t_k)\right)$, and taking into account the conditions of Equation (8), we obtain the following inequality:

$$\begin{aligned}\dot{\hat{V}}(\hat{x}(t)) \leq & -\frac{\hat{c}_3}{\hat{c}_2}\rho_s + \frac{\partial \hat{V}(\hat{x}(t))}{\partial \hat{x}} F_{nn}\left(\hat{x}(t), u_{d_1}^*(t_k), u_{d_2}^*(t_k)\right) \\ & - \frac{\partial \hat{V}(\hat{x}(t_k))}{\partial \hat{x}} F_{nn}\left(\hat{x}(t_k), u_{d_1}^*(t_k), u_{d_2}^*(t_k)\right)\end{aligned} \tag{24}$$

Based on the Lipschitz condition of Equation (10) and that $\hat{x} \in \Omega_{\hat{\rho}}$, $u_1 \in U_1$, and $u_2 \in U_2$, the upper bound of $\dot{\hat{V}}(\hat{x}(t))$ is derived $\forall t \in [t_k, t_{k+1})$:

$$\begin{aligned}\dot{\hat{V}}(\hat{x}(t)) \leq & -\frac{\hat{c}_3}{\hat{c}_2}\rho_s + L_{nn}\,|\hat{x}(t) - \hat{x}(t_k)| \\ & \leq -\frac{\hat{c}_3}{\hat{c}_2}\rho_s + L_{nn}M_{nn}\Delta\end{aligned} \tag{25}$$

Therefore, if Equation (19a) is satisfied, the following inequality holds $\forall \hat{x}(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_s}$ and $t \in [t_k, t_{k+1})$:

$$\dot{\hat{V}}(\hat{x}(t)) \leq -\epsilon_s \tag{26}$$

By integrating the above equation over $t \in [t_k, t_{k+1})$, it is obtained that $\hat{V}(\hat{x}(t_{k+1})) \leq \hat{V}(\hat{x}(t_k)) - \epsilon_s\Delta$. Therefore, $\hat{V}(\hat{x}(t)) \leq \hat{V}(\hat{x}(t_k))$, $\forall t \in [t_k, t_{k+1})$. We have proved that for all $\hat{x}(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_s}$, the state of the closed-loop LSTM system of Equation (5) is bounded in the closed-loop stability region $\Omega_{\hat{\rho}}$ for all times and moves toward the origin under $[u_1\ u_2] = \left[u_{d_1}^*(t_k)\ u_{d_2}^*(t_k)\right]$ implemented in a sample-and-hold fashion.

Next, we consider when $x(t_k) = \hat{x}(t_k) \in \Omega_{\rho_s}$ and Equation (26) may not hold. According to Equation (20a), $\Omega_{\rho_{nn}}$ is designed to ensure that the closed-loop state $\hat{x}(t)$ of the LSTM model does not leave $\Omega_{\rho_{nn}}$ for all $t \in [t_k, t_{k+1})$, $u_1 \in U_1$, $u_2 \in U_2$, and $\hat{x}(t_k) \in \Omega_{\rho_s}$ within one sampling period. If the state $\hat{x}(t_{k+1})$ leaves Ω_{ρ_s}, the controller $[u_1\ u_2] = \left[u_{d_1}^*(t_k)\ u_{d_2}^*(t_k)\right] \in U$ will drive the state toward Ω_{ρ_s} over the next sampling period since Equation (26) is satisfied again at $t = t_{k+1}$. Therefore, the convergence of the state to $\Omega_{\rho_{nn}}$ for the closed-loop LSTM system of Equation (5) is proved for all $\hat{x}_0 \in \Omega_{\hat{\rho}}$.

Part 3: We have proven that the closed-loop state of the LSTM system of Equation (5) are bounded in $\Omega_{\hat{\rho}}$ and ultimately converge to $\Omega_{\rho_{nn}}$ under the controller $[u_1\ u_2] = \left[u_{d_1}^*(t_k)\ u_{d_2}^*(t_k)\right]$ computed by the distributed LMPC design of Equation (17) for all $\hat{x} \in \Omega_{\hat{\rho}}$. We will now prove that the controllers $[u_1\ u_2] = \left[u_{d_1}^*(t_k)\ u_{d_2}^*(t_k)\right]$ computed by the distributed LMPC design of Equation (17) are able to stabilize the actual nonlinear system of Equation (1) while accounting for bounded modeling error $|\nu|$ and disturbances $|w|$. If there exists a positive real number $\gamma < \hat{c}_3/\hat{c}_4$ that constrains the modeling error $|\nu| = |F(x, u_1, u_2, 0) - F_{nn}(x, u_1, u_2)| \leq \gamma|x|$ for all $x \in \Omega_{\hat{\rho}}$, $u_1 \in U_1$, $u_2 \in U_2$, then the origin of the closed-loop nominal system of Equation (1) can be rendered exponentially stable under the controller $[u_1\ u_2] = \left[u_{d_1}^*(t_k)\ u_{d_2}^*(t_k)\right]$. This is shown by proving that $\dot{\hat{V}}$ for the nominal system of Equation (1) can be rendered negative under $[u_1\ u_2] = \left[u_{d_1}^*(t_k)\ u_{d_2}^*(t_k)\right]$. Based on the conditions on the Lyapunov functions of Equation (22) as derived in *Part* 2, and Equation (8c), the time derivative of the Lyapunov function is derived as follows:

$$\begin{aligned}\dot{\hat{V}}(x) = & \frac{\partial \hat{V}(x)}{\partial x} F\left(x, u_{d_1}^*(t_k), u_{d_2}^*(t_k), 0\right) \\ & = \frac{\partial \hat{V}(x)}{\partial x}\left(F_{nn}\left(x, u_{d_1}^*(t_k), u_{d_2}^*(t_k)\right) + F\left(x, u_{d_1}^*(t_k), u_{d_2}^*(t_k), 0\right) - F_{nn}\left(x, u_{d_1}^*(t_k), u_{d_2}^*(t_k)\right)\right) \\ & \leq -\hat{c}_3|x|^2 + \hat{c}_4\,|x|\left(F\left(x, u_{d_1}^*(t_k), u_{d_2}^*(t_k), 0\right) - F_{nn}\left(x, u_{d_1}^*(t_k), u_{d_2}^*(t_k)\right)\right) \\ & \leq -\hat{c}_3|x|^2 + \hat{c}_4\gamma|x|^2 \\ & \leq -\tilde{c}_3|x|^2\end{aligned} \tag{27}$$

When γ is chosen to satisfy $\gamma < \hat{c}_3/\hat{c}_4$, it holds that $\dot{\hat{V}} \leq -\tilde{c}_3|x|^2 \leq 0$ where $\tilde{c}_3 = -\hat{c}_3 + \hat{c}_4\gamma > 0$. Therefore, the closed-loop state of the nominal system of Equation (1) converges to the origin under $[u_1\ u_2] = \left[u^*_{d_1}(t_k)\ u^*_{d_2}(t_k)\right], \forall x_0 \in \Omega_{\hat{\rho}}$ if the modeling error is sufficiently small, that is, $|\nu| \le \gamma|x|$. Additionally, considering the presence of bounded disturbances (i.e., $|w| \le w_m$), we will now prove that the closed-loop state $x(t)$ of the actual nonlinear system of Equation (1) (i.e., $\dot{x} = F(x,u,w)$) is bounded in $\Omega_{\hat{\rho}}$ and ultimately converges to $\Omega_{\rho_{min}}$ under the sample-and-hold implementation of the control actions $[u_1\ u_2] = \left[u^*_{d_1}(t_k)\ u^*_{d_2}(t_k)\right]$ as computed by the distributed LMPC design of Equation (17).

Similarly, we first consider $x(t_k) = \hat{x}(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_{nn}}$, which means $x(t_k)$ also belongs to the region $\Omega_{\hat{\rho}} \backslash \Omega_{\rho_s}$. We derive the time-derivative of $\hat{V}(x)$ for the nonlinear system of Equation (1) with bounded disturbances as follows:

$$\begin{aligned}\dot{\hat{V}}(x(t)) &= \frac{\partial \hat{V}(x(t))}{\partial x} F\left(x(t), u^*_{d_1}(t_k), u^*_{d_2}(t_k), w\right) \\ &= \frac{\partial \hat{V}(x(t_k))}{\partial x} F\left(x(t_k), u^*_{d_1}(t_k), u^*_{d_2}(t_k), 0\right) \\ &\quad + \frac{\partial \hat{V}(x(t))}{\partial x} F\left(x(t), u^*_{d_1}(t_k), u^*_{d_2}(t_k), w\right) \\ &\quad - \frac{\partial \hat{V}(x(t_k))}{\partial x} F\left(x(t_k), u^*_{d_1}(t_k), u^*_{d_2}(t_k), 0\right)\end{aligned} \tag{28}$$

From Equation (27), we know that $\frac{\partial \hat{V}(x(t_k))}{\partial x} F\left(x(t_k), u^*_{d_1}(t_k), u^*_{d_2}(t_k), 0\right) \le -\tilde{c}_3 |x(t_k)|^2$ holds for all $x \in \Omega_{\hat{\rho}}$. Based on Equation (8a) and the Lipschitz condition in Equation (4), the following inequality is obtained for $\dot{\hat{V}}(x(t))$ $\forall t \in [t_k, t_{k+1})$ and $x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_s}$:

$$\begin{aligned}\dot{\hat{V}}(x(t)) \le &-\frac{\tilde{c}_3}{\hat{c}_2}\rho_s + \frac{\partial \hat{V}(x(t))}{\partial x} F\left(x(t), u^*_{d_1}(t_k), u^*_{d_2}(t_k), w\right) \\ &- \frac{\partial \hat{V}(x(t_k))}{\partial x} F\left(x(t_k), u^*_{d_1}(t_k), u^*_{d_2}(t_k), 0\right) \\ &\le -\frac{\tilde{c}_3}{\hat{c}_2}\rho_s + L'_x |x(t) - x(t_k)| + L'_w |w| \\ &\le -\frac{\tilde{c}_3}{\hat{c}_2}\rho_s + L'_x M\Delta + L'_w w_m\end{aligned} \tag{29}$$

Therefore, if Equation (19b) is satisfied, the following inequality holds $\forall x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_s}$ and $t \in [t_k, t_{k+1})$:

$$\dot{\hat{V}}(x(t)) \le -\epsilon_w \tag{30}$$

Integrating Equation (30) will show that Equation (21b) holds; hence, the closed-loop state of the actual nonlinear process of Equation (1) is maintained in $\Omega_{\hat{\rho}}$ for all times, and can be driven toward the origin in every sampling period under the controller $[u_1\ u_2] = \left[u^*_{d_1}(t_k)\ u^*_{d_2}(t_k)\right]$. Additionally, if $x(t_k) \in \Omega_{\rho_s}$, considering the sample-and-hold implementation of control actions, it has been shown in *Part* 2 that the state of the LSTM model of Equation (5) is maintained in $\Omega_{\rho_{nn}}$ within one sampling period. Considering the bounded error between the state of the LSTM of Equation (5) model and the state of the nonlinear system of Equation (1) given by Equation (18a), there exists a compact set $\Omega_{\rho_{min}} \supset \Omega_{\rho_{nn}}$ that satisfies Equation (20b) such that the state of the actual nonlinear system of Equation (1) does not leave $\Omega_{\rho_{min}}$ during one sampling period if the state of the LSTM model of Equation (5) is bounded in $\Omega_{\rho_{nn}}$. If the state $x(t)$ enters $\Omega_{\rho_{min}} \backslash \Omega_{\rho_s}$, we have shown that Equation (30) holds, and thus, the state will be driven toward the origin again under $[u_1\ u_2] = \left[u^*_{d_1}(t_k)\ u^*_{d_2}(t_k)\right]$ during the next sampling period.

Consider $x(t) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_{nn}}$ at $t = t_k$ where the contractive constraints of Equation (11e) and Equation (12e) are activated. Since $x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_{nn}}$, it follows that $x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_s}$, hence Equation (30) holds, implying that the closed-loop state will be driven toward the origin in every sampling step under $[u_1\ u_2] = \left[u^*_{d_1}(t_k)\ u^*_{d_2}(t_k)\right]$ and can be driven into $\Omega_{\rho_{nn}}$ within finite sampling steps. After the state enters $\Omega_{\rho_{nn}}$, the constraint of Equation (11f) and Equation (12f) are activated to maintain the predicted states of the LSTM model of Equation (11b) and Equation (12b) in $\Omega_{\rho_{nn}}$ over the entire prediction horizon. As we characterize a region $\Omega_{\rho_{min}}$ that satisfies Equation (20b), the closed-loop state $x(t)$ of the nonlinear system of Equation (1), $\forall t \in [t_k, t_{k+1})$ is guaranteed to be bounded in $\Omega_{\rho_{min}}$ if the predicted state by the LSTM model of Equation (11b) and Equation (12b) remains in $\Omega_{\rho_{nn}}$. Therefore, at the next sampling step $t = t_{k+1}$, if the state $x(t_{k+1})$ is still bounded in $\Omega_{\rho_{nn}}$, the constraint of Equation (11f) and Equation (12f) maintains the predicted state $\hat{x}$ of the LSTM model of Equations (11b) and (12b) in $\Omega_{\rho_{nn}}$ such that the actual state x of the nonlinear system of Equation (1) stays inside $\Omega_{\rho_{min}}$. However, if $x(t_{k+1}) \in \Omega_{\rho_{min}} \backslash \Omega_{\rho_{nn}}$, following the proof we have shown for the case that $x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_{nn}}$, the contractive constraint of Equations (11e) and (12e) will be activated instead to drive it toward the origin. This completes the proof of boundedness of the states of the closed-loop system of Equation (1) in $\Omega_{\hat{\rho}}$ and convergence to $\Omega_{\rho_{min}}$ for any $x_0 \in \Omega_{\hat{\rho}}$.

Remark 5 Although there exists approximation error induced by Equation (9), it does not jeopardize the stability analysis. This is because the same numerical approximations are used universally whenever the calculations for $\hat{f}$, $\hat{g}_1$, and $\hat{g}_2$ are invoked. This includes when calculating $\Phi_{nn_1}(x), \Phi_{nn_2}(x)$, when characterizing the stability region, and when calculating the time-derivative of the control Lyapunov function in the contractive constraint for the optimization problems of the distributed MPC system. Therefore, $[u_1\ u_2] = [\Phi_{nn_1}(x)\ \Phi_{nn_2}(x)]$ still remains a feasible solution to the distributed MPC that will render the origin of the nonlinear process exponentially stable for all initial conditions $x_0 \in \Omega_{\hat{\rho}}$.

5 | APPLICATION TO A TWO-CSTR-IN-SERIES PROCESS

A chemical process example is utilized to demonstrate the application of sequential distributed and iterative distributed model predictive control using the proposed LSTM model, the results of which will be

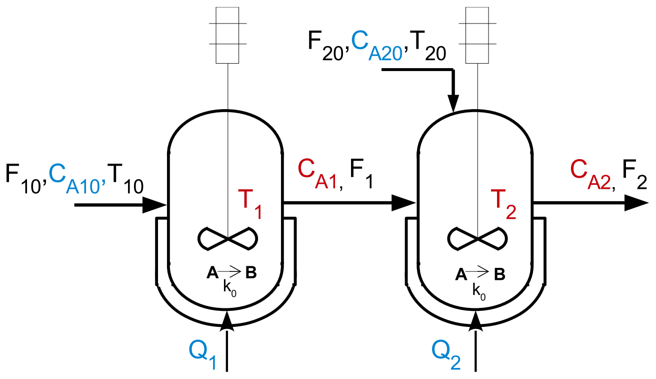

FIGURE 3 Process flow diagram of two CSTRs in series. CSTRs, continuous stirred tank reactors [Color figure can be viewed at wileyonlinelibrary.com]

TABLE 1 Parameter values of the CSTRs

T_{10} = 300 K	T_{20} = 300 K
F_{10} = 5 m^3/hr	F_{20} = 5 m^3/hr
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/hr	Q_{2_s} = 0.0 kJ/hr
$k_0 = 8.46 \times 10^6$ m^3/kmol hr	$\Delta H = -1.15 \times 10^4$ kJ/kmol
C_p = 0.231 kJ/kg K	R = 8.314 kJ/kmol K
ρ_L = 1000 kg/m^3	$E = 5 \times 10^4$ kJ/kmol

Abbreviations: CSTRs, continuous stirred tank reactors.

compared to that of centralized model predictive control. Specifically, two well-mixed, non-isothermal continuous stirred tank reactors (CSTRs) in series are considered where an irreversible second-order exothermic reaction takes place in each reactor as shown in Figure 3. The reaction transforms a reactant A to a product B ($A \to B$). Each of the two reactors are fed with reactant material A with the inlet concentration C_{Aj0}, the inlet temperature T_{j0} and feed volumetric flow rate of the reactor F_{j0}, j = 1, 2, where j = 1 denotes the first CSTR and j = 2 denotes the second CSTR. Each CSTR is equipped with a heating jacket that supplies/removes heat at a rate Q_j, j = 1, 2. The CSTR dynamic models is obtained by the following material and energy balance equations:

$$\frac{dC_{A1}}{dt} = \frac{F_{10}}{V_1}(C_{A10} - C_{A1}) - k_0 e^{\frac{-E}{RT_1}} C_{A1}^2 \tag{31a}$$

$$\frac{dT_1}{dt} = \frac{F_{10}}{V_1}(T_{10} - T_1) + \frac{-\Delta H}{\rho_L C_p} k_0 e^{\frac{-E}{RT_1}} C_{A1}^2 + \frac{Q_1}{\rho_L C_p V_1} \tag{31b}$$

$$\frac{dC_{B1}}{dt} = -\frac{F_{10}}{V_1} C_{B1} + k_0 e^{\frac{-E}{RT_1}} C_{A_1}^2 \tag{31c}$$

$$\frac{dC_{A2}}{dt} = \frac{F_{20}}{V_2} C_{A20} + \frac{F_{10}}{V_2} C_{A1} - \frac{F_{10} + F_{20}}{V_2} C_{A2} - k_0 e^{\frac{-E}{RT_2}} C_{A2}^2 \tag{31d}$$

$$\frac{dT_2}{dt} = \frac{F_{20}}{V_2} T_{20} + \frac{F_{10}}{V_2} T_1 - \frac{F_{10} + F_{20}}{V_2} T_2 + \frac{-\Delta H}{\rho_L C_p} k_0 e^{\frac{-E}{RT_2}} C_{A2}^2 + \frac{Q_2}{\rho_L C_p V_2} \tag{31e}$$

$$\frac{dC_{B2}}{dt} = \frac{F_{10}}{V_2} C_{B1} - \frac{F_{10} + F_{20}}{V_2} C_{B2} + k_0 e^{\frac{-E}{RT_2}} C_{A2}^2 \tag{31f}$$

where C_{Aj}, 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. ΔH, k_0, E, and R represent the enthalpy of the reaction, pre-exponential constant, activation energy, and ideal gas constant, respectively. Process parameter values are listed in Table 1. The manipulated inputs for both CSTRs are the inlet concentration of species A and the heat input rate, 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: $|\Delta C_{Aj0}| \le 3.5$ kmol/m^3 and $|\Delta Q_j| \le 5 \times 10^5$ kJ/hr, j = 1, 2. Therefore, the states of the closed-loop system are $x^T = [C_{A1} - C_{A1_s}\ T_1 - T_{1_s}\ C_{A2} - C_{A2_s}\ T_2 - T_{2_s}]$, 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 the first and the second reactor, such that the equilibrium point of the system is at the origin of the state-space. It is noted that the states of the first CSTR can be separately denoted as $x_1^T = [C_{A1} - C_{A1_s}\ T_1 - T_{1_s}]$ and the states of the second CSTR are denoted as $x_2^T = [C_{A2} - C_{A2_s}\ T_2 - T_{2_s}]$. In a centralized MPC framework, feedback measurement on all states x is received by the controller, and the manipulated inputs for the entire system, $u^T = [\Delta C_{A10}\ \Delta Q_1\ \Delta C_{A20}\ \Delta Q_2]$, are computed by one centralized controller. In a distributed LMPC system, both LMPCs have access to full-state information as well as the overall model of the two-CSTR process. Both LMPC 1 and LMPC 2 receive feedback on $x(t)$; LMPC 1 optimizes u_1^T and LMPC 2 optimizes u_2^T. The common control objective of the model predictive controllers is to stabilize the two-CSTR process at the unstable operating steady-state $x_s^T = [C_{A1_s}\ C_{A2_s}\ T_{1_s}\ T_{2_s}]$, whose values are presented in Table 1. The explicit Euler method with an integration time step of $h_c = 10^{-4}$ hr is used to numerically simulate the dynamic model of Equation (31). The nonlinear optimization problems of the distributed LMPCs of Equations (11) and (12), and of Equations (15) and (16) are solved using the Python module of the IPOPT software package,[20] named PyIpopt with a sampling period $\Delta = 10^{-2}$ hr. The objective function in the distributed LMPC optimization problem has the form $L(x, u_1, u_2) = x^T Q x + u_1^T R_1 u_1 + u_2^T R_2 u_2$, where $Q = diag[2 \times 10^3\ 1\ 2 \times 10^3\ 1]$, $R_1 = R_2 = diag[8 \times 10^{-13}\ 0.001]$; the same objective function is used in both LMPC 1 and LMPC 2 in all distributed LMPC systems. The overall control Lyapunov function is the sum of the control Lyapunov functions for the two CSTRs, that is, $V(x) = V_1(x_1) + V_2(x_2) = x_1^T P_1 x_1 + x_2^T P_2 x_2$, with the following positive definite P matrices:

$$P_1 = P_2 = \begin{bmatrix} 1060 & 22 \\ 22 & 0.52 \end{bmatrix} \tag{32}$$

Given that the modeling error between the LSTM model and the first-principles model is sufficiently small, the control Lyapunov function for the first-principles model of the nonlinear process V can be used as the control Lyapunov function for the LSTM model $\hat{V}$ as well.

5.1 | LSTM network development

Open-loop simulations are conducted for finite sampling steps for various initial conditions inside Ω_ρ, where $\rho = 392$, using the nonlinear system of Equation (1) under various $u_1 \in U_1$, $u_2 \in U_2$ applied in a sample-and-hold manner. These trajectories which consist of training sample points are collected with a time interval of $5 \times h_c$. The LSTM model is then developed to predict future states over one sampling period Δ. This LSTM model captures the dynamics of the overall two-CSTR process of Equation (31), and can be used in all individual distributed LMPCs or in the centralized LMPC. The LSTM network is developed using *Keras* with 1 hidden layer consisting of 50 units, where *tanh* function is used as the activation function, and *Adam* is used as the optimizer. The stopping criteria for the training process include that the mean squared modeling error being less than 5×10^{-7} and the mean absolute percentage of the modeling error being less than 4.5×10^{-4}. After 50 epochs of training, with each epoch taking on average 200 s, the mean squared error between the predicted states of the LSTM network model and of the first-principles models is 4.022×10^{-7} and the mean absolute error is 4.254×10^{-4}. After obtaining an LSTM model with sufficiently small modeling error, the Lyapunov function of the LSTM model, $\hat{V}$, is chosen to be the same as $V(x)$. Subsequently, the set $\hat{D}$ can be characterized using the controllers $[u_1\ u_2] = [\Phi_{nn_1}(x)\ \Phi_{nn_2}(x)]$, from which the closed-loop stability region $\Omega_{\hat{\rho}}$ for the LSTM system can be characterized as the largest level set of $\hat{V}$ in $\hat{D}$ while also being a subset of Ω_ρ. The positive constants $\hat{\rho}_1$ and $\hat{\rho}_2$, which are used to define the largest level sets of the control Lyapunov functions for the first and the second CSTR, respectively, are $\hat{\rho}_1 = \hat{\rho}_2 = 380$. Additionally, the ultimate bounded region $\Omega_{\rho_{nn}}$, and subsequently, $\Omega_{\rho_{min}}$, are chosen to be $\rho_{nn} = 10$ and $\rho_{min} = 12$, determined via extensive closed-loop simulations with $u_1 \in U_1$, $u_2 \in U_2$. Readers interested in more computational details on the development of a recurrent neural network model can refer to Reference 21.

In this study, we simulate three different types of control systems to compare their closed-loop control performances: a centralized LMPC, an iterative distributed LMPC system, and a sequential distributed LMPC system. We develop an LSTM model for the overall two-CSTR process, which is the same model used in both centralized LMPC system and distributed LMPC systems. When this LSTM network model is implemented online during closed-loop simulations, the inputs to the LSTM network are $x(t)$ and $u(t)$ at $t = t_k$, and the outputs are the predicted future states $\hat{x}(t)$ at $t = t_{k+1}$; more examples on this overall LSTM model can be found in Reference 22. 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.

5.2 | Closed-loop model predictive control simulations

To demonstrate the efficacy of the distributed model predictive control network using LSTM models, the following simulations are carried out. First, we simulate a centralized LMPC using the LSTM network for the overall two-CSTR process as its prediction model, where the four manipulated inputs are $u^T = [\Delta C_{A10}\ \Delta Q_1\ \Delta C_{A20}\ \Delta Q_2]$, and it receives feedback on all states $x^T = [C_{A1} - C_{A1_s}\ T_1 - T_{1_s}\ C_{A2} - C_{A2_s}\ T_2 - T_{2_s}]$. Then, we simulate sequential distributed LMPCs and iterative distributed LMPCs, where LMPC 1 and LMPC 2 in both distributed frameworks use the same LSTM model for the overall two-CSTR process as used in a centralized LMPC system. The closed-loop control performances of the aforementioned control networks are compared, the comparison metrics include the computation time of calculating the solutions to the LMPC optimization problem(s), as well as the sum squared error of the closed-loop states $x(t)$ for a total simulation period of $t_p = 0.3$ hr. It should be noted that, since iterative distributed LMPC systems allow parallel computing of the individual controllers, the computation time for obtaining the final solutions to the optimization problems of Equations (15) and (16) should be the maximum time of the two controllers, accounting for all iterations carried out before the termination criterion is reached. The termination criterion used was that the computation time for solving each LMPC must not exceed the sampling period, Δ. On the other hand, in a sequential distributed LMPC system, since the computation of LMPC 1 depends on the optimal trajectory of control action calculated by LMPC 2, the total computation time taken to obtain the solutions to the optimization problems of Equations (11) and (12) must be the sum of the time taken by the two controllers.

Table 2 shows the average computation time for solving the optimization problem(s) of the distributed and centralized LMPC systems, as well as the sum of squared percentage error of all states in the form of $SSE = \int_0^{t_p} \left(\frac{C_{A1} - C_{A1_s}}{C_{A1_s}}\right)^2 + \left(\frac{T_1 - T_{1_s}}{T_{1_s}}\right)^2 + \left(\frac{C_{A2} - C_{A2_s}}{C_{A2_s}}\right)^2 + \left(\frac{T_2 - T_{2_s}}{T_{2_s}}\right)^2 dt$. It is shown in Table 2 that when the two-CSTR process is operated under distributed LMPC systems, the sum squared error and the average computation time are reduced compared to the case where a centralized LMPC system is used. Moreover, it is shown that the iterative

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 iterative distributed, sequential distributed, and centralized LMPC systems using their respective LSTM models with a total simulation time of 0.3 hr

	Average computation time (s)	Sum squared error
Iterative distributed LMPC	26.70	2.85
Sequential distributed LMPC	29.55	3.04
Centralized LMPC	35.26	3.08

Abbreviations: LMPC, Lyapunov-based model predictive control; LSTM, long short-term memory.

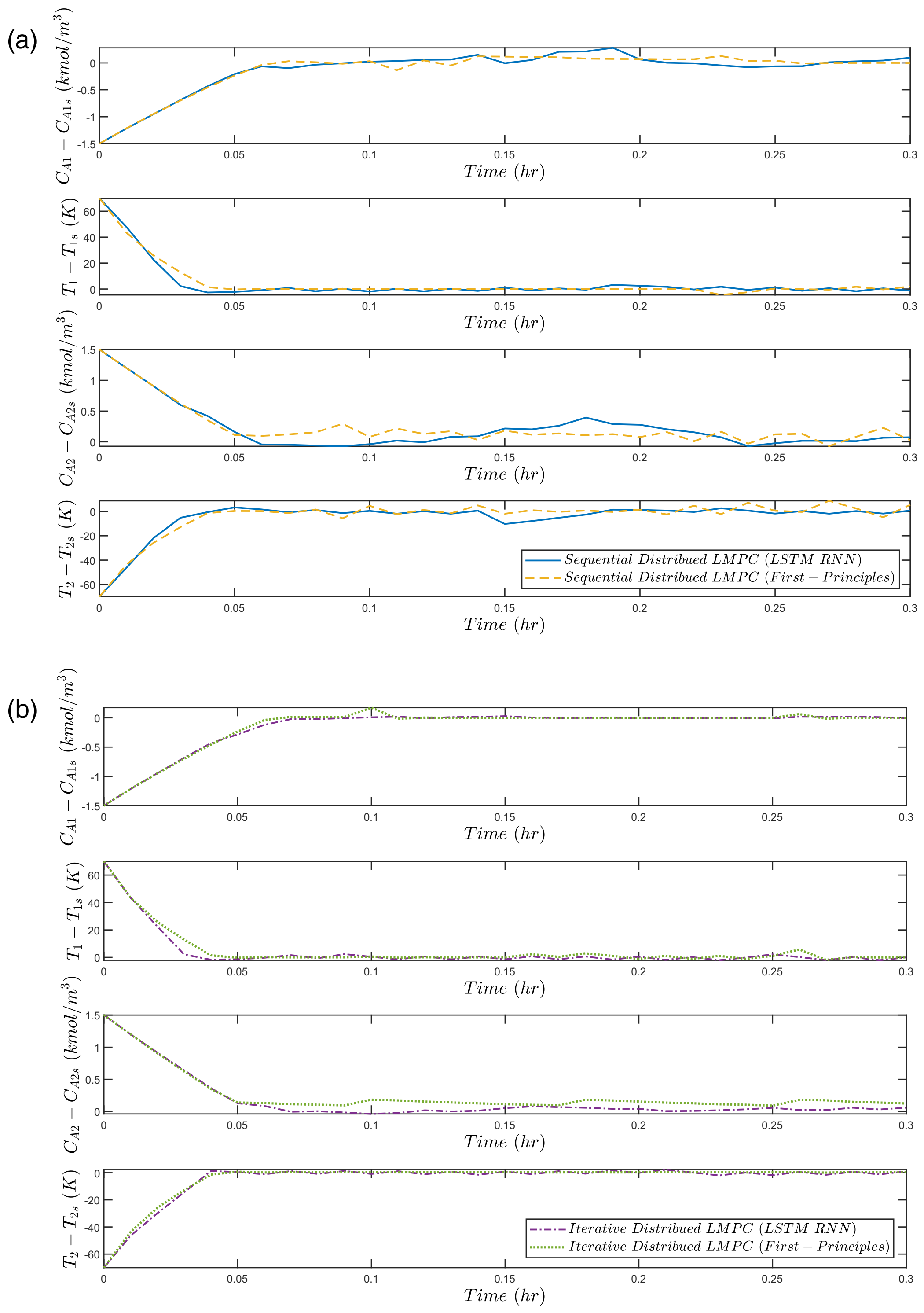

FIGURE 4 Closed-loop state trajectories of (a) sequential and (b) iterative distributed LMPC systems using LSTM models and first-principles models, respectively [Color figure can be viewed at wileyonlinelibrary.com]

distributed MPC using the LSTM model has a lower mean computation time and a lower sum squared error than the sequential distributed MPC, and the distributed MPC in general achieve a lower sum squared error than the centralized MPC. It should also be noted that the computation time of all simulated control systems are lower than the sampling period used in the two-CSTR process such that the proposed control system can be implemented without computational issues.

In this work, we develop machine-learning-based models for the two-CSTR process of Equation (31) assuming that the first-principles model of Equation (31) is unknown. However, in order to have a reasonable baseline for comparison, we show the simulation results of each distributed control framework using the first-principles model of the nonlinear process of Equation (31). Furthermore, in real-life scenarios where the first-principles model of an industrial-scale chemical plant is not available, the comparison of closed-loop control performances using machine-learning-based models can be conducted against plant data. To further illustrate the closed-loop performances of sequential and iterative LMPC systems using the LSTM model, the closed-loop state evolution showing the convergence of closed-loop states from the initial conditions $x_0^T = [-1.5\text{kmol/m}^3\ 70\text{K}\ 1.5\text{kmol/m}^3\ -70\text{K}]$ under the sequential and iterative LMPC using the LSTM model are plotted in Figure 4a,b along with the closed-loop trajectories under the respective distributed LMPCs using the first-principles model as a baseline for comparison. All states converge to $\Omega_{\rho_{min}}$ within 0.10 hr under the sequential and iterative distributed MPCs using the LSTM model. It is reported that using the LSTM model, the sum squared error of an operation period of 0.3 *hr* under iterative distributed LMPC and under sequential distributed LMPC are 2.85 and 3.04, respectively. Using the first-principles model, the sum squared error of the same operation period with the same initial conditions under iterative and sequential distributed LMPC systems are 2.96 and 2.98, respectively, which is on par with the sum squared error achieved using LSTM network, with the iterative distributed LMPC obtaining even a lower sum squared error using LSTM network than using first-principles. Through closed-loop simulations and performance metrics comparisons, we have demonstrated the efficacy of distributed LMPC systems using LSTM network models.

6 | CONCLUSIONS

In this study, we presented the design of distributed model predictive control systems for nonlinear processes using a machine-learning-based model, which was used as the prediction model to capture nonlinear dynamic behavior. Closed-loop stability and performance properties were analyzed for the sequential and iterative distributed model predictive control systems. Extensive open-loop data within the stability region of the nonlinear process characterized by Lyapunov-based control methods were collected to train LSTM network models with a sufficiently small modeling error with respect to the actual nonlinear process model. It was shown that both sequential distributed LMPC system and iterative distributed LMPC system using the LSTM network model were able to achieve efficient real-time computation, and ensure closed-loop state boundedness and convergence to the origin. Working with a nonlinear chemical process network example, the distributed LMPC systems using the LSTM network model were able to obtain similar closed-loop performance as the distributed LMPC systems using the first-principles model, as well as reduced computation time when compared to the closed-loop results of the centralized LMPC system using the LSTM network model.

ACKNOWLEDGMENT

Financial support from the National Science Foundation and the Department of Energy is gratefully acknowledged.

ORCID

Panagiotis D. Christofides https://orcid.org/0000-0002-8772-4348

REFERENCES

1. Chen S, Wu Z, Christofides PD. A cyber-secure control-detector architecture for nonlinear processes. *AIChE J*. 2020;66:e16907.
2. Tian Y and Pan L. Predicting short-term traffic flow by long short-term memory recurrent neural network. Presented at: Proc of the 2015 IEEE Int Conf on Smart City/SocialCom/SustainCom (SmartCity), Chengdu, China, 153-158, 2015.
3. Ke W., Huang D., Yang F., and Jiang Y. Soft sensor development and applications based on lstm in deep neural networks. Presented at: Proc of the IEEE Symp Series on Computational Intelligence (SSCI), Honolulu, HI, USA, 1-6, 2017.
4. Liu S, Wang J. A simplified dual neural network for quadratic programming with its KWTA application. *IEEE Trans Neural Netw*. 2006; 17:1500-1510.
5. Forbes MG, Patwardhan RS, Hamadah H, Gopaluni RB. Model predictive control in industry: challenges and opportunities. *IFAC Papers OnLine*. 2015;48:531-538.
6. Scattolini R. Architectures for distributed and hierarchical model predictive control–a review. *J Process Control*. 2009;19:723-731.
7. Stewart BT, Venkat AN, Rawlings JB, Wright SJ, Pannocchia G. Cooperative distributed model predictive control. *Syst Control Lett*. 2010; 59:460-469.
8. Christofides PD, Scattolini R, de la Pena DM, Liu J. Distributed model predictive control: a tutorial review and future research directions. *Comput Chem Eng*. 2013;51:21-41.
9. Daoutidis P, Tang W, Allman A. Decomposition of control and optimization problems by network structure: concepts, methods, and inspirations from biology. *AIChE J*. 2019;65:e16708.
10. Wang R, Bao J. Distributed plantwide control based on differential dissipativity. *Int J Robust Nonlinear Control*. 2017;27:2253-2274.
11. Yan Y, Wang R, Bao J, Zheng C. Robust distributed control of plantwide processes based on dissipativity. *J Process Control*. 2019;77: 48-60.
12. Liu J, Chen X, Muñoz de la Peña D, Christofides PD. Sequential and iterative architectures for distributed model predictive control of nonlinear process systems. *AIChE J*. 2010;56:2137-2149.
13. Liu J, Muñoz de la Peña D, Christofides PD. Distributed model predictive control of nonlinear process systems. *AIChE J*. 2009;55:1171-1184.
14. Tang W, Daoutidis P. Fast and stable nonconvex constrained distributed optimization: the ELLADA algorithm. *arXiv*. 2020;2004: 1977.
15. Lin Y, Sontag ED. A universal formula for stabilization with bounded controls. *Syst Control Lett*. 1991;16:393-397.
16. Sontag ED. Neural nets as systems models and controllers. Presented at: Proc of the Seventh Yale Workshop on Adaptive and Learning Systems, Yale University, 73–79, 1992.
17. Kosmatopoulos EB, Polycarpou MM, Christodoulou MA, Ioannou PA. High-order neural network structures for identification of dynamical systems. *IEEE Trans Neural Netw*. 1995;6:422-431.
18. Hochreiter S, Schmidhuber J. Long short-term memory. *Neural Comput*. 1997;9:1735-1780.
19. Wu Z, Tran A, Rincon D, Christofides PD. Machine learning-based predictive control of nonlinear processes. Part I: theory. *AIChE J*. 2019;65:e16729.

20. Wächter A, Biegler LT. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Math Program*. 2006;106:25-57.
21. Wu Z, Tran A, Rincon D, Christofides PD. Machine-learning-based predictive control of nonlinear processes. Part II: computational implementation. *AIChE J*. 2019;65:e16734.
22. Wu Z, Rincon D, Christofides PD. Process structure-based recurrent neural network modeling for model predictive control of nonlinear processes. *J Process Control*. 2020;89:74-84.

How to cite this article: Chen S, Wu Z, Rincon D, Christofides PD. Machine learning-based distributed model predictive control of nonlinear processes. *AIChE J*. 2020;66:e17013. https://doi.org/10.1002/aic.17013