Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 2548319, 13 pages
https://doi.org/10.1155/2020/2548319

Research Article

Reinforcement Learning-Based Backstepping Control for Container Cranes

Xiao Sun and Zhihang Xie

College of Mechanical Engineering, Hunan University of Technology, Zhuzhou 412000, China

Correspondence should be addressed to Zhihang Xie; 1984387635@qq.com

Received 1 August 2019; Revised 1 January 2020; Accepted 27 January 2020; Published 19 February 2020

Academic Editor: Zoran Gajic

A novel backstepping control scheme based on reinforcement fuzzy Q-learning is proposed for the control of container cranes. In this control scheme, the modified backstepping controller can handle the underactuated system of a container crane. Moreover, the gain of the modified backstepping controller is tuned by the reinforcement fuzzy Q-learning mechanism that can automatically search the optimal fuzzy rules to achieve a decrease in the value of the Lyapunov function. The effectiveness of the applied control scheme was verified by a simulation in Matlab, and the performance was also compared with the conventional sliding mode controller aimed at container cranes. The simulation results indicated that the used control scheme could achieve satisfactory performance for step-signal tracking with an uncertain lope length.

1. Introduction

A robotic container crane is a robot that lifts the cargo off the ground with ropes and then carries the cargo to the designated locations. As the robot carries the cargo to the destination, it is necessary to stabilize the swing angle of the rope. Significant swing-related problems may cause the cargo to fall or even roll over. Therefore, an appropriate control strategy is necessary to ensure that the robot responds quickly to the ideal command while suppressing the amplitude of the swaying angle of the rope. The uncertainties of the system, resulted from the uncertain system parameters, can bring up the challenges on the design of the controller. Therefore, some previous control techniques relying on exact models exhibited certain limitations [1–3]. Many present control strategies aimed to the uncertain systems have been proposed for this problem, such as the sliding mode control [4–8], fuzzy control [9, 10], adaptive control [11, 12], and fuzzy PID control [13, 14] strategies.

Reinforcement learning (RL) is a learning method that gradually explores the optimal policy by interacting with the environment [15]. In the reinforcement learning, the target is usually to maximize the cumulative rewards or minimize the cumulative costs over the entire learning process. The entire process of typical reinforcement learning can be described as the following. The learning process starts by the agent adopting an action in the initial state based on the current policy, and the adopted action will transfer the system from the current state to the next state with certain probability. Subsequently, the agent will repeat to adopt an action and then transfer the system from the current state to the next state until the end of the learning. In this process, an action transferring the system from the current state to the next state will be evaluated with the reward or cost that is also called the instant reward or cost. The offered instant rewards/costs of each action from all the visited states can be further used to dynamically explore the optimal policy of adopting actions that can achieve the maximum of rewards or the minimum of costs over the entire process, which can be completed by many temporal difference (TD) methods such as Q-learning [16] and SARSA [17].

When the reinforcement learning is applied in the field of control for continuous systems, it is inevitable to encounter the problem of "curse of dimensionality" that means the number of discrete states that are supposed to be visited by the agent increases to infinity. Therefore, fuzzy logic can

be used to fuzzify the system states, allowing the application of reinforcement learning methods originally used for discrete systems [18].

In recent years, there have been developments in the use of fuzzy reinforcement learning theory to solve control problems of nonlinear systems. In a recent approach [19], for the coordinated control problem of multiple manipulators, a reinforcement learning method was used to deal with the uncertainties of the dynamic models. This approach took into account minimizing both the errors of tracking trajectory and the control quantities for each robot, thereby solving the problem of the inconsistencies between different manipulators. In the literature [20], the control law was outputted from the reinforcement learning mechanism in which the actions corresponding to each state were set to satisfy the stability requirement, and a neural network was used to solve the problem of "the curse of dimensionality."

In this paper, a modified backstepping controller is proposed for the underactuated system of robotic container cranes. The control gain of this controller is important because it influences the convergence of tracking errors. However, it is difficult for the designer to empirically obtain the appropriate values of the control gain because the experience about the appropriate control gains is always expensive or even unavailable in practices. Therefore, fuzzy Q-learning is applied to automatically search the optimal fuzzy rules that can output the appropriate values of the control gain. More precisely, the value of the Lyapunov function is used to judge the applied actions, and the control gains that result in the decrease of the value of Lyapunov function will be given a high-value reward and vice versa. Therefore, the control target can be achieved by the fuzzy Q-learning mechanism that obtains the optimal fuzzy rules outputting the appropriate control gains in the applied controller after the appropriate learning process, which can reduce the value of Lyapunov function and then achieve the convergence of tracking errors. The rest of this paper is organised as follows: in Section 2, a nonlinear dynamics model of robotic container cranes is established by the Lagrangian method. In Section 3, a reinforcement fuzzy Q-learning-based backstepping control scheme is detailed to control the position of the load and stabilize the swaying angle of the rope. The stability proof is also presented in this section. In Section 4, the simulation is conducted to verify the effectiveness of the applied controller, and the performance is compared with the conventional sliding model controller. The conclusion is given in Section 5.

2. Dynamical Model of the Robot

The robotic container crane model is shown in Figure 1, where x is the horizontal displacement of the robot; θ is the load swing angle; m_1 and m_2 are the weights of the robot body and the load, respectively; and L and F are the length of the rope and the driving force of the robot, respectively.

Assuming that the entire system is fiction-free and the ropes have no mass and undergo no elastic deformation, the kinetic (T) and potential (U) energy of the robot system can be, respectively, expressed as follows:

$$\begin{cases} T = \dfrac{1}{2}(m_1 + m_2)\dot{x}^2 + \dfrac{1}{2}m_2\left(\dot{L}^2 + \dot{L}^2\dot{\theta}^2\right) + m_2(\dot{L}\dot{x}\sin\theta + L\dot{x}\dot{\theta}\cos\theta), \\ U = m_2 gL(1 - \cos\theta), \end{cases} \tag{1}$$

where g is the local gravitational acceleration. According to the Lagrangian equation,

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial T}{\partial \dot{q}_i} - \frac{\partial T}{q_i} + \frac{\partial U}{q_i} = \tau_i, \tag{2}$$

where $\tau = [F, 0]^T$ is the control inputs of the system and $q = [x, \theta]^T$. The dynamics equations of the container crane can be achieved:

$$\begin{cases} (m_1 + m_2)\ddot{x} + m_2(\ddot{L}\sin\theta + L\ddot{\theta}\cos\theta) + m_2\left(2\dot{L}\dot{\theta}\cos\theta - L\dot{\theta}^2\sin\theta\right) - F = 0, \\ m_2 gL(\sin\theta) + m_2 L\ddot{x}\cos\theta + m_2 L^2\ddot{\theta} + 2m_2 L\dot{L}\dot{\theta} = 0. \end{cases} \tag{3}$$

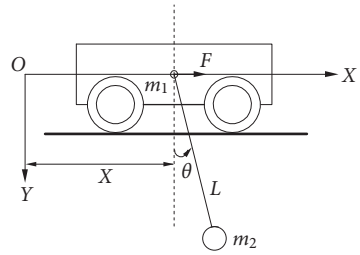


Figure 1: Schematic illustration of a container crane.

The above equation can be rewritten in the state space form

$$\begin{bmatrix} \ddot{x} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} \dfrac{m_2 g \sin\theta\cos\theta - m_2\ddot{L}\sin\theta + m_2 L\dot{\theta}^2\sin\theta}{m_1 + m_2\sin^2\theta} \\ \dfrac{-(m_1+m_2)g\sin\theta + m_2\ddot{L}\sin\theta\cos\theta - m_2 L\dot{\theta}^2\sin\theta\cos\theta}{m_1 L + m_2 L\sin^2\theta} - \dfrac{2\dot{\theta}\dot{L}}{L} \end{bmatrix} + \begin{bmatrix} \dfrac{1}{m_1 + m_2\sin^2\theta} \\ -\dfrac{\cos\theta}{m_1 L + m_2 L\sin^2\theta} \end{bmatrix} \begin{bmatrix} F \\ F \end{bmatrix}. \tag{4}$$

The dynamics of a container crane, which is shown in equation (4), can also be presented in the block diagram as shown in Figure 2.

3. The Design of Reinforcement Learning-Based Backstepping Controller

In this section, a backstepping controller for a crane robot is designed. A fuzzy reinforcement Q-learning mechanism is applied to determine the appropriate parameters of the controller to achieve stability. The control scheme is shown in Figure 3.

First, the state space form of the system dynamics equation can be rewritten as

$$\begin{bmatrix} \ddot{x} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} k_1(\theta) \\ k_2(\theta) \end{bmatrix} + \begin{bmatrix} k_3(\theta) \\ -\dfrac{\cos\theta}{L}k_3(\theta) \end{bmatrix} \begin{bmatrix} F \\ F \end{bmatrix}. \tag{5}$$

To satisfy the above equations,

$$\begin{aligned} k_1(\theta) &= \frac{m_2 g\sin\theta\cos\theta - m_2\ddot{L}\sin\theta + m_2 L\dot{\theta}^2\sin\theta}{m_1 + m_2\sin^2\theta}, \\ k_2(\theta) &= \frac{-(m_1+m_2)g\sin\theta + m_2\ddot{L}\sin\theta\cos\theta - m_2 L\dot{\theta}^2\sin\theta\cos\theta}{m_1 L + m_2 L\sin^2\theta} - \frac{2\dot{\theta}\dot{L}}{L}, \\ k_3(\theta) &= \frac{1}{m_1 + m_2\sin^2\theta}. \end{aligned} \tag{6}$$

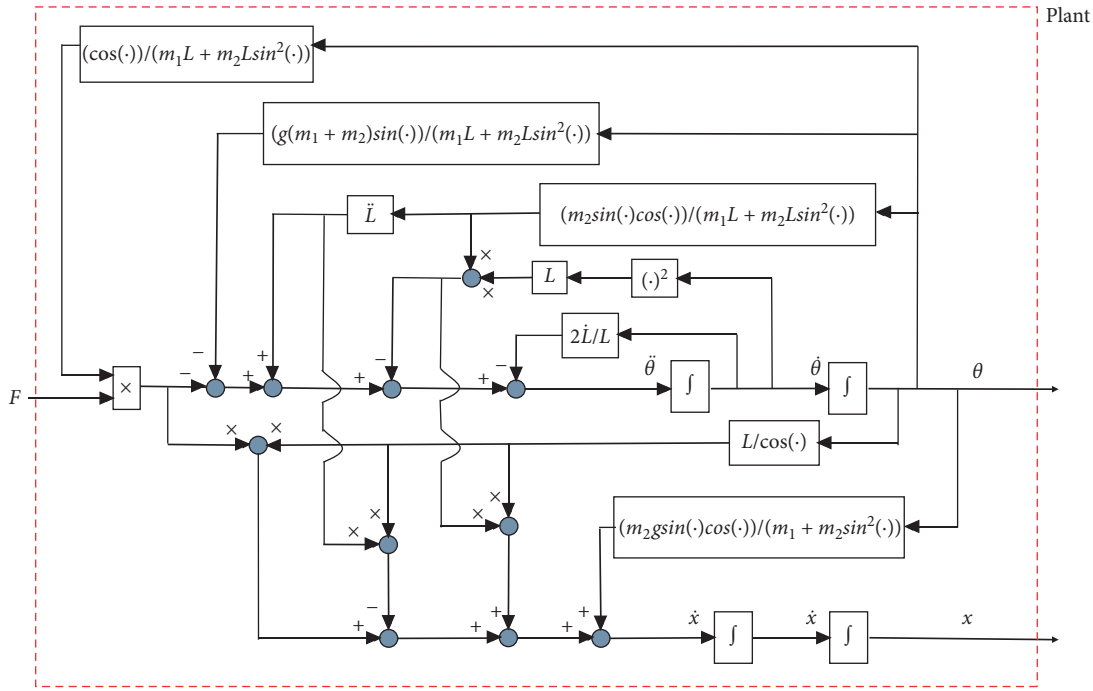


Figure 2: Block diagram of the dynamics of a container crane.

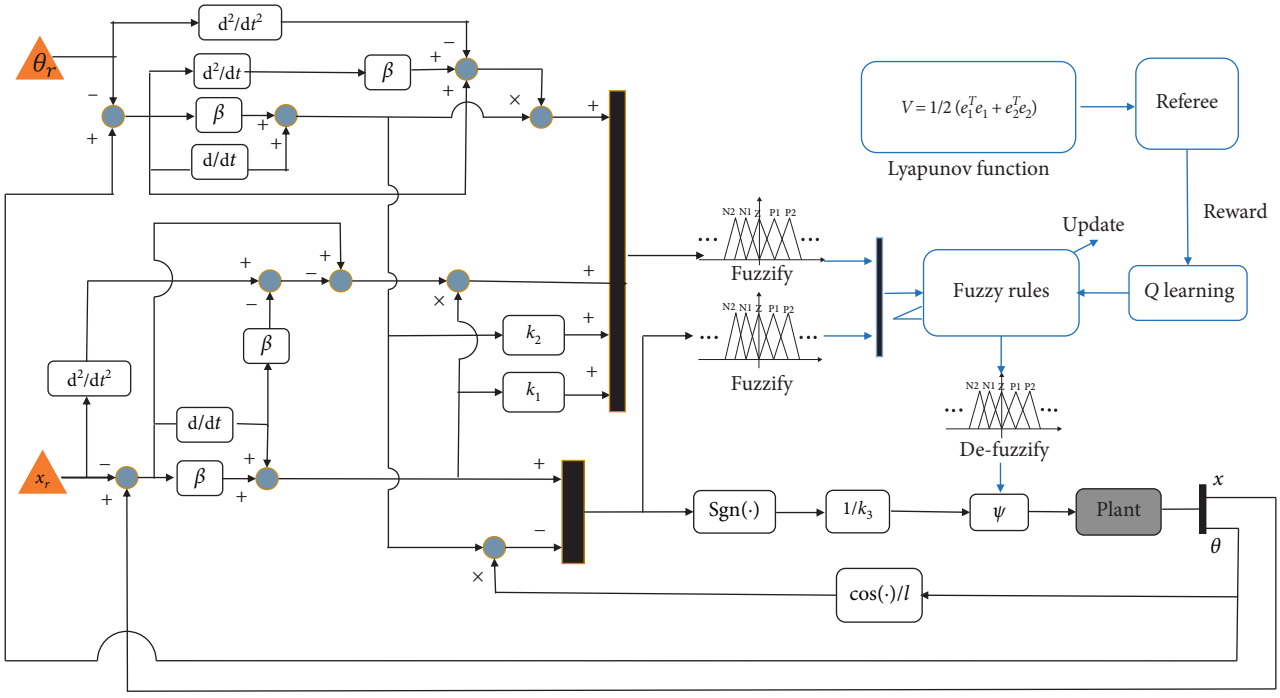


Figure 3: Reinforcement learning-based backstepping control scheme (BSC).

Furthermore,

$$\begin{cases} \ddot{x} = k_1(\theta) + k_3(\theta)F, \\ \ddot{\theta} = k_2(\theta) - \dfrac{\cos\theta}{L}k_3(\theta)F. \end{cases} \tag{7}$$

The dynamics model (equation (7)) is written in a form on which the backstepping control method can be conveniently applied:

$$\begin{cases} \dot{\mathbb{X}}_1 = \mathbb{X}_2, \\ \dot{\mathbb{X}}_2 = \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} + \begin{bmatrix} k_3 \\ -\dfrac{\cos\theta}{L}k_3 \end{bmatrix} F, \end{cases} \tag{8}$$

where $\mathbb{X}_1 = [x, \theta]^T$ and $\mathbb{X}_2 = [\dot{x}, \dot{\theta}]^T$.

The first Lyapunov function of backstepping control is designed as follows:

$$V_1 = \frac{1}{2}e_1^T e_1, \tag{9}$$

where $e_1 = \mathbb{X}_1 - \mathbb{X}_d = [x - x_d, \theta - \theta_d]^T$. x_d and θ_d are the desirable values of trolley's position along the X-axis and the sway angle, respectively. Taking the first derivative for equation (9),

$$\dot{V}_1 = e_1^T\dot{e}_1 = e_1^T\left(\mathbb{X}_2 - \dot{\mathbb{X}}_d\right) = e_1^T\left(\mathbb{X}_2^* - \dot{\mathbb{X}}_d\right) + e_1^T\left(\mathbb{X}_2 - \mathbb{X}_2^*\right). \tag{10}$$

It is noticed that negative $\dot{V}_1 \le 0$ can be obtained by $\mathbb{X}_2 = \mathbb{X}_2^* = \dot{\mathbb{X}}_d - \beta_1 e_1$. $\beta_1 > 0$ is a positive number. Consequently, the second Lyapunov function is designed as

$$V_2 = V_1 + \frac{1}{2}e_2^T e_2, \tag{11}$$

where $e_2 = \mathbb{X}_2 - \mathbb{X}_2^*$. Taking the first derivative for equation (11) yields the following:

$$\begin{aligned} \dot{V}_2 &= e_1^T\left(\mathbb{X}_2 - \dot{\mathbb{X}}_d\right) + e_2^T\left(\begin{bmatrix} k_1 \\ k_2 \end{bmatrix} + \begin{bmatrix} k_3 \\ -\dfrac{\cos\theta}{L}k_3 \end{bmatrix} F - \dot{x}_2^*\right) \\ &= e_1^T(e_2 - \beta_1 e_1) + e_2^T\left(\begin{bmatrix} k_1 \\ k_2 \end{bmatrix} + \begin{bmatrix} k_3 \\ -\dfrac{\cos\theta}{L}k_3 \end{bmatrix} F - \ddot{\mathbb{X}}_r + \beta_1\mathbb{X}_2 - \beta_1\dot{\mathbb{X}}_d\right) \\ &= -e_1^T\beta_1 e_1 + e_2^T\left(\begin{bmatrix} k_1 \\ k_2 \end{bmatrix} + \begin{bmatrix} k_3 \\ -\dfrac{\cos\theta}{L}k_3 \end{bmatrix} F + e_1 - \ddot{\mathbb{X}}_d + \beta_1\mathbb{X}_2 - \beta_1\dot{\mathbb{X}}_d\right) \\ &\le \left[\dot{x} - \dot{x}_d + \beta_1(x - x_d)\right]\left[k_1 + k_3F - \ddot{x}_d + \beta_1(\dot{x} - \dot{x}_d) + (x - x_d)\right] \\ &\quad + \left[\dot{\theta} - \dot{\theta}_d + \beta_1(\theta - \theta_d)\right]\left[k_2 - \frac{\cos\theta}{L}k_3F - \ddot{\theta}_d + \beta_1\left(\dot{\theta} - \dot{\theta}_d\right) + (\theta - \theta_d)\right] \\ &= E_x\left[-\varnothing_x + (x - x_d)\right] + E_\theta\left[-\varnothing_\theta + (\theta - \theta_d)\right] + E_x k_1 + E_\theta k_2 + \left(E_x - \frac{\cos\theta}{L}E_\theta\right)k_3F, \end{aligned} \tag{12}$$

where $E_x = \dot{x} - \dot{x}_d + \beta_1(x - x_d)$, $E_\theta = \dot{\theta} - \dot{\theta}_d + \beta_1(\theta - \theta_d)$, $\varnothing_x = \ddot{x}_d - \beta_1(\dot{x} - \dot{x}_d)$, and $\varnothing_\theta = \ddot{\theta}_d - \beta_1(\dot{\theta} - \dot{\theta}_d)$.

It is worthwhile to notice that the crane robot system (equation (7)) is an underactuated system that has 2 controlled variables (position and swag angle, respectively) and only 1 control input (force F). As a result, the conventional backstepping control law for fully actuated systems is not applicable in this case. Moreover, the control law shown in equation (13) that can necessarily ensure a negative derivative of Lyapunov function V_2 would result in impractically big control signals when its dominator is approximate to zero (i.e., E_x and E_θ are of small values).

$$F = -\frac{E_x\left[-\varnothing_x + (x - x_d)\right] + E_\theta\left[-\varnothing_\theta + (\theta - \theta_d)\right] + E_x k_1 + E_\theta k_2}{(E_x - \cos\theta/LE_\theta)k_3}. \tag{13}$$

Consequently, a novel control law for F that can avoid the issue resulted from a zero value in the dominator is applied as follows:

$$F = -\frac{1}{k_3}\psi \cdot \operatorname{sgn}\left(E_x - \frac{\cos\theta}{L}E_\theta\right), \tag{14}$$

where $\psi \ge 0$ is the control gain. It is noticed that the negative derivative of Lyapunov function (equation (11)) is

maintained as long as the parameter Ψ shown in equation (14) satisfies the following equation:

$$\begin{cases} -\psi\left|E_x - \dfrac{\cos\theta}{L}E_\theta\right| + \delta \le 0, \\ \delta = E_x\left[-\varnothing_x + (x - x_d)\right] + E_\theta\left[-\varnothing_\theta + (\theta - \theta_d)\right] + E_x k_1 + E_\theta k_2. \end{cases} \tag{15}$$

It is difficult to design the parameter δ in a deterministic way to satisfy equation (15) because the small value of $|E_x - (\cos\theta/L)E_\theta|$ would result in the impractically big value of ψ. Consequently, the reinforcement fuzzy Q-learning is applied to search the appropriate values of ψ.

In equations (12) and (15), to ensure the stability of the system, Ψ is adjusted based on the values of $|E_x - (\cos\theta/L)E_\theta|$ and δ. A reasonable linguistical adjustment rule is as follows:

(A) If $|E_x - (\cos\theta/L)E_\theta|$ is big and δ is positive, then ψ is medium

(B) If $|E_x - (\cos\theta/L)E_\theta|$ is big and δ is negative, then ψ is small

(C) If $|E_x - (\cos\theta/L)E_\theta|$ is small and δ is positive, then ψ is very large

(D) If $|E_x - (\cos\theta/L)E_\theta|$ is small and δ is negative, then ψ is large

The terms small, medium, and large in the above equations are all linguistical descriptions. The actual numerical output is obtained by the fuzzy reasoning based on the numerical values of actions and the parameters of the fuzzy structure. The group of actions corresponding to the linguistical description of "large" is $\psi_{\text{large}} = u_1, u_2, \ldots, u_l$. The numerical value of u_i directly affects the performance of the controller. The action among Ψ is selected based on $|E_x - (\cos\theta/L)E_\theta|$ and δ using the fuzzy Q-learning method to achieve the convergence of Lyapunov function. In this control scheme, $E_x - (\cos\theta/L)E_\theta$ and δ are used as the inputs for fuzzy reasoning, i.e., the state in Q-learning. $\Gamma_1 = E_x - (\cos\theta/L)E_\theta$ and $\Gamma_2 = \delta$ are fuzzified by the triangular membership functions shown in Figure 4 with the details of fuzzy sets shown as equations (16) and (17):

$$\text{Lin}(\Gamma_1) = \left\{\tau_{1,1}, \ldots, \tau_{1,a}, \ldots, \tau_{1,A}\right\}, \quad a = 1, 2, \ldots, A, \tag{16}$$

$$\text{Lin}(\Gamma_2) = \left\{\tau_{2,1}, \ldots, \tau_{2,b}, \ldots, \tau_{2,B}\right\}, \quad b = 1, 2, \ldots, B, \tag{17}$$

where A is the number of fuzzy sets $(\tau_{1,a})$ for $\Gamma_1 = E_x - (\cos\theta/L)E_\theta$ and B is the number of fuzzy sets $(\tau_{2,b})$ for $\Gamma_2 = \delta$.

We define the n-th fuzzy rule Rn in the fuzzy Q-learning as follows:

Rn: If s_1^k is L_1^n and s_2^k is L_2^n and...and s_m^k is L_m^n, then, $\psi = U_{nL}$ ($\psi = u_{1L}$ with $q_{1L}(n, 1)$ or $\psi = u_{2L}$ with $q_{2L}(n, 2)$ or $\psi = u_{3L}$ with $q_{3L}(n, 3)$ or...or $\psi = u_{pL}$ with $q_{1L}(p, 1)$), where the set $u \in (u_{1L}, \ldots, u_{pL})$ is the chosen set of parameters Ψ under the n-th rule in the state s at the moment k. The Rn rule corresponding to the input state vector sk = $\{s_1^k, s_2^k, \ldots, s_m^k\}$ yields the membership functions {μ (sk), . . ., μ (sk)} at a given time. Each u in the set UnL has a corresponding q value. Therefore, it is necessary for the reinforcement learning to continually update the q value for each action in all rules based on the membership functions and the rewards to achieve the optimal policy of selecting actions u_{iL} in all rules. Next, the rewards are given according to the variance of the value of Lyapunov function $(\dot{V} < 0)$.

First, the Q-learning mechanism selects the smallest u value corresponding to the q value based on each fuzzy rule:

$$u_{nL}^* = \arg\min_{u_{nL} \in U_{nL}} q_L(n, u_{nL}). \tag{18}$$

To prevent u from falling into local optimum conditions in the selection process, a greedy mechanism is introduced:

$$u_n^\Psi = \begin{cases} u_n^\Gamma, & \text{with probability } \varepsilon, \\ u_{nL}^*, & \text{with probability } 1 - \varepsilon. \end{cases} \tag{19}$$

The numerical value of the parameter Ψ is obtained by defuzzifing the sample selected by the rule:

$$\psi\left(s^k\right) = \frac{\sum_{n=1}^N \mu_n\left(s^k\right) u_n^\Psi}{\sum_{n=1}^N \mu_n\left(s^k\right)}. \tag{20}$$

Under the greedy mechanism, the choice of u is random, which makes the reinforcement learning more globally exploratory in the training process.

The defuzzification of the Q value of the state vector at time k can be expressed by the following equation:

$$Q_{IL}\left(s^k\right) = \frac{\sum_{n=1}^N \mu_n\left(s^k\right) q_{nL}\left(n, u_{nL}^\Psi\right)}{\sum_{n=1}^N \mu_n\left(s^k\right)}. \tag{21}$$

The defuzzification of the target value under state s^k can be expressed as follows:

$$P_{IL}\left(s^k\right) = \frac{\sum_{n=1}^N \mu_n\left(s^k\right) q_{nL}\left(n, u_{nL}^*\right)}{\sum_{n=1}^N \mu_n\left(s^k\right)}. \tag{22}$$

When the state vector s^k of the system enters the next state s^{k+1} under the action of u_{IL}, the generated cost information is c^k, and the time error in the process is

$$\Delta Q_{IL} = c^k + \eta P_{IL}\left(s^{k+1}\right) - Q_{IL}\left(s^k, u_{IL}\left(s^k\right)\right), \tag{23}$$

where $\eta \in (0, 1)$ is the discount factor that reflects the consideration of the future reward and c^k is the reward given at the instant k. In this case, the variance of the Lyapunov function (equation (11)) is used as the reward. More precisely, if the value of Lyapunov function decreases during the period from instant $k-1$ to instant k, a large reward value will be given. Otherwise, a small reward value will be given. The function describing the reward is as follows:

$$c^k = \begin{cases} 0, & \text{if } V(t-1) = 0, \\ \dfrac{V(k-1) - V(k)}{|V(k-1)|}, & \text{otherwise.} \end{cases} \tag{24}$$

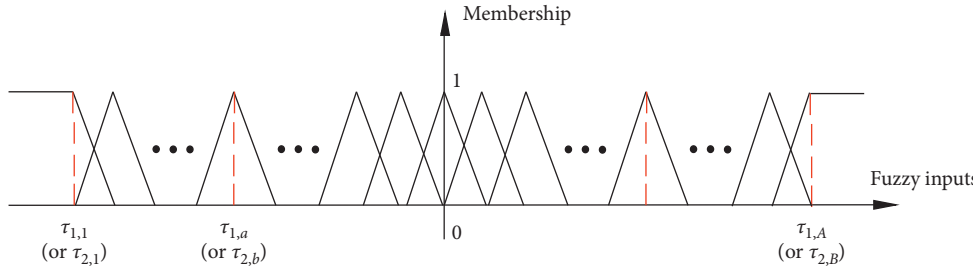


Figure 4: Membership function of fuzzy inputs.

It is clear that a larger value (no greater than 1) will be attained with a more dramatic decrease of the Lyapunov function. The reward with the largest value of 1 will be achieved only if $V(k) = 0$, which means the system achieves the desired steady state.

The iteration equation of the final q value is as follows:

$$q_{IL}\left(n, u_{IL}^{\Psi}\right) \longleftarrow q_{IL}\left(n, u_{IL}^{\Psi}\right) + \lambda \Delta Q_{IL} \frac{u_n\left(s^k\right)}{\sum_{n=1}^{N} \mu_n\left(s^k\right)}, \quad (25)$$

where λ is the learning coefficient between 0 and 1.

The parameters of the proposed controller, which should be determined and tuned by users, consist of the parameter in the backstepping part (the parameter β_1 in the items of E_θ and E_x in equation (14)) and the parameters in the reinforcement learning part (parameters of fuzzy sets $\tau_{1,a}$ in equation (16) and $\tau_{2,b}$ in equation (17), mutation probability ε in equation (19), discount factor η in equation (23), learning rate λ in equation (25), and action group of control gain ψ). Several rules of tuning the parameters of the controller are given in Remarks 1–6 to achieve a satisfying control performance.

Remark 1. In the backstepping part, β_1 is the proportional gain of the first virtual control gain of the backstepping controller. A big value of β_1 can be chosen in order to achieve a fast decreasing value of Lyapunov function, which means the fast convergence of the errors of both the trolley's position and the sway angle. However, an excessively big value of β_1 shows the risk on amplifying the measurement noise of tracking errors (e_1) and the derivatives of tracking errors $(\dot{e}_1)$, which would negatively influence the control performance. On other words, there is a trade-off between the fast convergence of system errors and the immunity to the measurement noise. Hence, we suggest that the trials of selecting the value of β_1 should start at a small value (e.g., 0.01) and then gradually increase the value until the satisfyingly fast decrease of Lyapunov function is achieved.

Remark 2. In the reinforcement learning part, the fuzzy sets of inputs used to do fuzzy reasoning ($\tau_{1,a}$ and $\tau_{2,b}$ shown in equations (16) and (17)) are important because they transform the numerical inputs Γ_1 and Γ_2 into the group of firing rates corresponding to the linguistic description (e.g., small, medium, and big), which are applicable on fuzzy reasoning. Hence, we suggest choosing big values for $\tau_{1,A}$ and $\tau_{2,B}$ and small values for $\tau_{1,1}$ and $\tau_{2,1}$ in order to cover the range of Γ_1 and Γ_2 during the control task. We also suggest that the fuzzy sets $\tau_{1,a}$ and $\tau_{2,b}$ should be distributed evenly among the selected range in order to well present the dynamics of the second Lyapunov function to the fuzzy inference.

Remark 3. In the reinforcement learning part, mutation probability ε reflects the trade-off between the exploration of potentially better solutions and the exploitation of learnt good solutions. It is generally agreed that the good solutions are likely obtained during the later stage of learning; therefore, we suggest offering ε with a big value (e.g., 0.6) during the initial stage of control and a small value (e.g., 0) during the later stage of control.

Remark 4. In the reinforcement learning part, discount factor η shows the attention on the control performance in future steps. In our case, although the variance of Lyapunov function is influenced by the previous control signals, the current variance of Lyapunov function is mainly determined by the current control signal, which means the current control gain on stabilizing the system should be judged mainly by the current performance (the current variance of Lyapunov function). Therefore, we suggest discount factor η should be offered a small value (e.g., 0.1).

Remark 5. In the reinforcement learning part, learning rate λ reflects the efficiency of remembering new knowledge and forgetting the old knowledge. A big value of λ could achieve a fast convergence of q, which means a high learning efficiency. However, it is desirable for reinforcement learning to keep the old knowledge to certain extent because of the risk on learning the false knowledge (e.g., the data used to learn are contaminated by measurement noise or unknown disturbances). Therefore, we suggest learning rate λ should be offered a medium value (e.g., 0.4~0.6).

Remark 6. Action group of control gains $(u_{1L}, \ldots, u_{pL})$ can be regarded as the most imperative part in determining the parameters of the controller because the actual control gains are calculated by equation (20) based on the members of this group. The minimum value of the member (u_{1L}) in this group should be small enough while the maximum value of the member (u_{pL}) in this group should be big enough, which ensure the optimal control gains satisfying equation (15) (meaning the convergence of Lyapunov function) are inside the set of calculated actual control gains. However, excessive

big values of control gain ψ could result in chattering effects. Hence, we suggest the minimum value of member (u_{1L}) should be very small (e.g., 0), and the trials of selecting the maximum value of member (u_{pL}) should start at a big value and then gradually decrease the value until the chattering effect is insignificant. Moreover, the rest members $(u_{2L}, \ldots, u_{(p-1)L})$ in the action group are suggested to be evenly distributed between the minimum value u_{1L} and the maximum value u_{pL} in order to calculating smooth control gains.

4. Simulation Result

Simulation was run to verify the effectiveness of the used controller. Our control target was to accurately control the position of the load (m_2) with as little the sway angle of the rope as possible. In other words, the load (m_2) is supposed to reach the designated position, and the angle of the rope is supposed to be stabilized around 0 by the used control scheme.

After carefully selecting parameters based on the aforementioned rules of selection shown in Remarks 1–5, the detailed parameters of the controller are determined. Various parameters of the robot system and the controller during the simulation are shown in Table 1.

The numbers of linguistical variables (fuzzy sets shown in equations (16) and (17)) to describe Γ_1 and Γ_2 are set as 10. The number of action candidates in the action group for calculating the control gain ψ is set as 50 on each fuzzy rule. As a result, after carefully selecting the parameters based on Remark 2, the membership function of Γ_1 has $\text{lin}(\Gamma_1) = \{-0.0015, -0.0012, -0.0008, -0.0005, -0.0002, 0.0002, 0.0005, 0.0008, 0.0012, 0.0015\}$, and the membership function of Γ_2 has $\text{lin}(\Gamma_2) = \{-0.02, -0.0156, -0.0111, -0.0067, -0.0022, 0.0022, 0.0067, 0.0111, 0.0156, 0.02\}$. After carefully selecting the parameters based on Remark 6, the action group of control gain on each rule is $u \in (0, (0.04/50), (0.04/49), \ldots, 0.04)$.

The desirable of trolley's position is set as a constant so that $x_d = 0.1$ m, while the sway angle is supposed to be minimized so that $\theta_d = 0$ rad. The length of the rope is the crucial element influencing the stability of the crane system [19], and we set an uncertainty of $\pm$ 20% to the rope length to show the ability of the applied control scheme to handle the uncertainty of rope length. The probability to explore potentially optimal fuzzy rules is set as $\varepsilon = 0.5$ during initial 60 s, $\varepsilon = 0.3$ from 60 s to 100 s, and $\varepsilon = 0$ after 100 s.

In the simulation, the proposed controller is compared with the conventional sliding mode controller (SMC) mentioned in [21].

The performance of the proposed reinforcement learning-based BSC and the conventional SMC on tracking a constant trolley's position are shown in Figure 5. Clearly, in the proposed reinforcement learning-based BSC, the significant overshooting is observed during the initial 20 s, and a longer time is used to drive the trolley to reach the target position compared with conventional SMC, which could be resulted from the exploration on the bad fuzzy rules during the initial stage of reinforcement learning. However, compared with the conventional SMC, the designed controller can achieve the less steady-state error (SSE) after learning the optimal rules during the later stage of control (shown as the subplot on the right part of Figure 5). In other words, the designed controller can achieve the position tracking of the trolley with more accuracy at the expense of fluctuations during the initial stage of control, which is similar to the nature of reinforcement learning that the optimal solutions are obtained after trying many bad solutions. For avoiding the observed overshooting and fluctuations, the good fuzzy rules that could be obtained from the experience of designers and the prior knowledge of the crane system could be used as the initial rules of reinforcement learning.

Table 1: Parameters of the crane and the controller.

	Parameter	Value
Container crane	m_1 (kg)	4.5
	m_2 (kg)	1
	l (m)	0.7
	β_1	1
Controller	$[\psi_{\max}, \psi_{\min}]$	$[0, 4e^{-2}]$
	$[\Gamma_{1,\min}, \Gamma_{1,\max}]$	$[-1.5e^{-3}, 1.5e^{-3}]$
	$[\Gamma_{2,\min}, \Gamma_{2,\max}]$	$[-2e^{-2}, 2e^{-2}]$
	η	0.1
	λ	0.5

The performance of the designed controller and the conventional SMC on stabilizing the sway angle at 0 degree are compared in Figure 6. Compared with the conventional SMC, although the fluctuation of sway angle under the used reinforcement learning BSC lasts longer (sway angle takes longer time to reach 0), it is clear that the sway angle can be around 0 degree with less chattering effects and less steady-state errors during the late period of control, shown as the 2 subplots in Figure 6. In other words, the applied controller on stabilizing sway angle starts at bad performance (longer settling time 0~60 s) and then achieves better performance than conventional SMC during the late stage of control (60 s~200 s). The reason of such performance of sway angle is the same to that of the trolley position, the nature of reinforcement learning that optimal solutions are obtained at the cost of trying many bad solutions. On top of that, the less overshooting of sway angle during the initial stage is also observed in the proposed controller compared with that of the conventional SMC.

Figure 7 shows the control forces generated by the reinforcement learning-based BSC and the conventional SMC. The applied reinforcement learning-based BSC provides outstanding chattering reduction and smaller control forces in comparison with conventional SMC during the entire period of control. It is also observed in Figure 7 that the control forces generated by reinforcement learning BSC in the initial stage of control (0–80 s) is relatively chattering than that in the late stage of control (80–200 s), which is in accordance with Figures 5 and 6. The reason is the same to that of the trolley position and sway angle, which have been explained in the previous discussion of results for Figures 5 and 6.

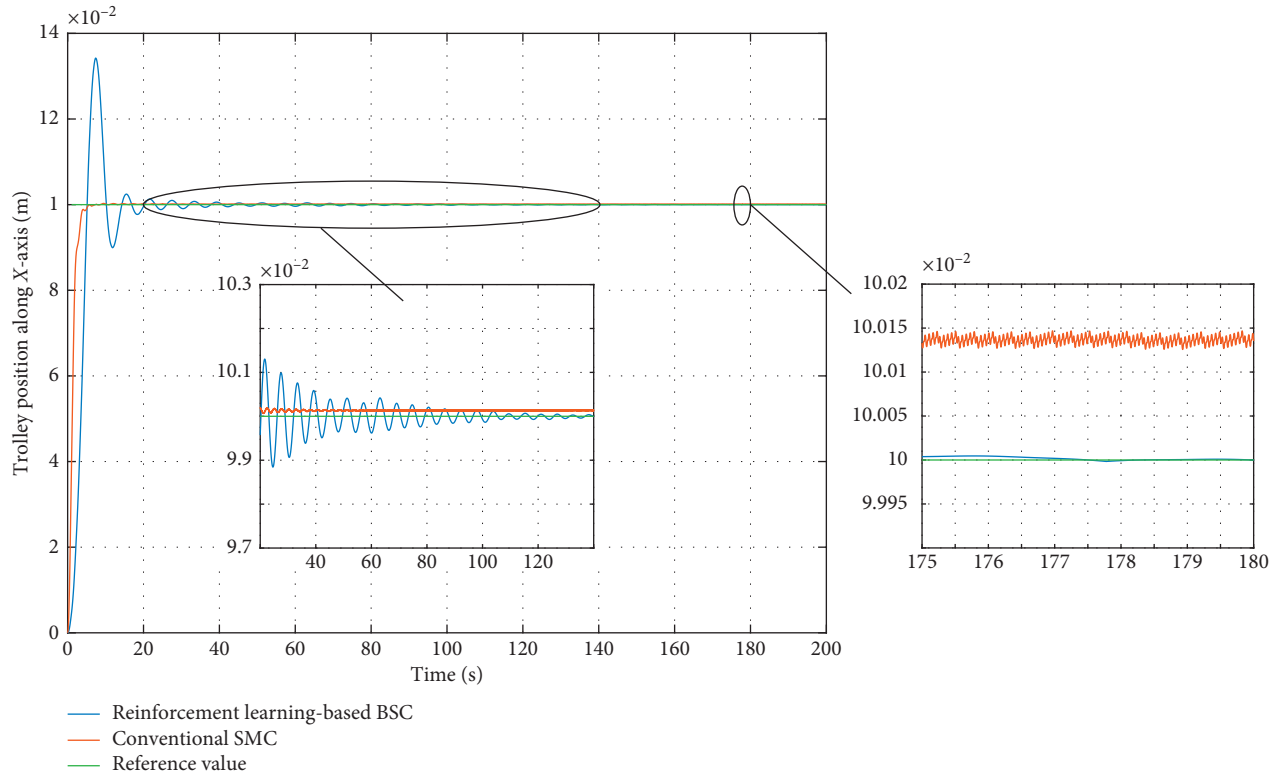


Figure 5: Performance on the signal tracking of trolley's position.

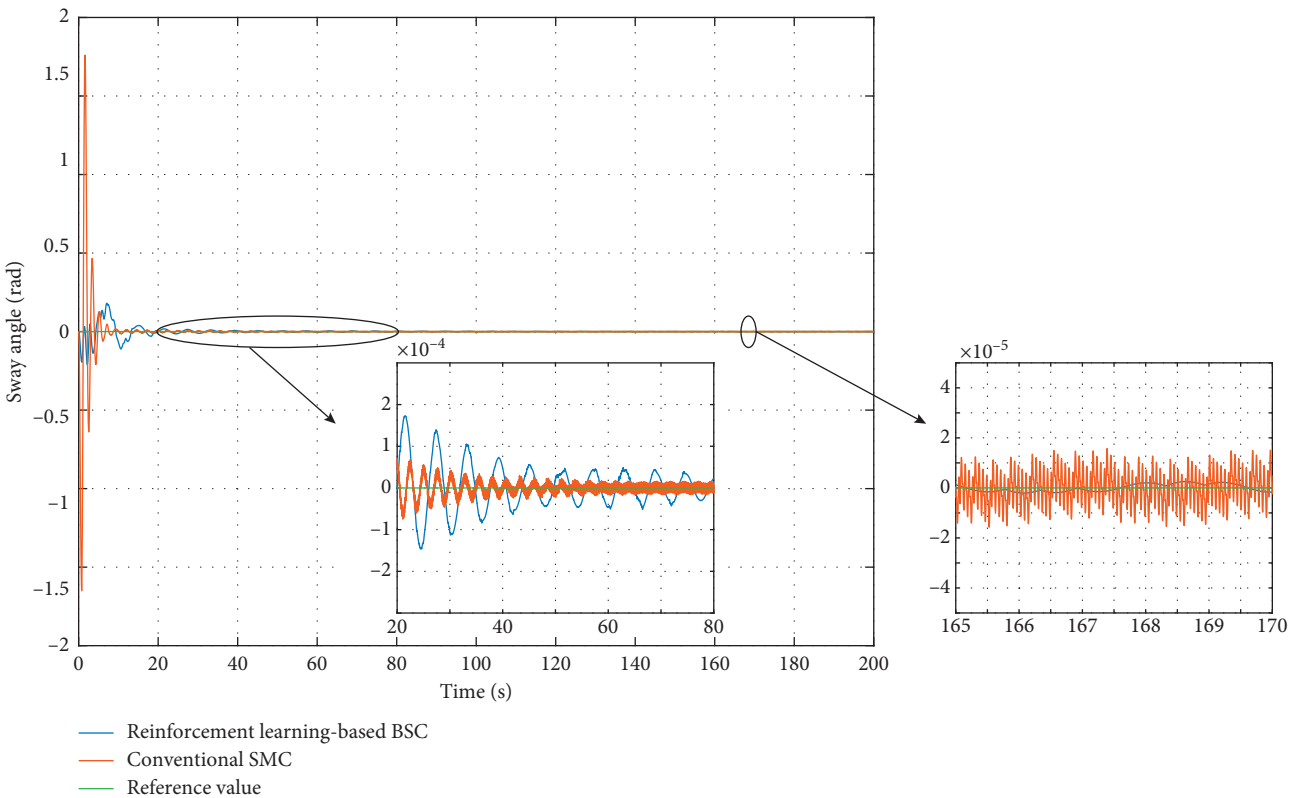


Figure 6: Performance on the signal tracking of sway angle.

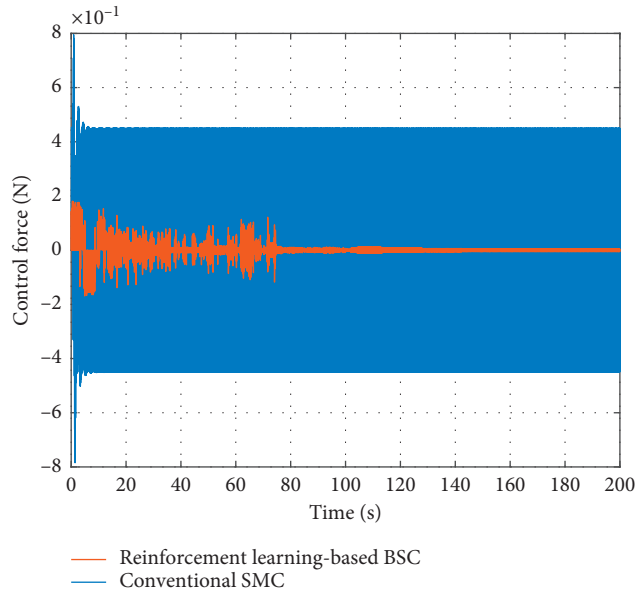


Figure 7: Control forces.

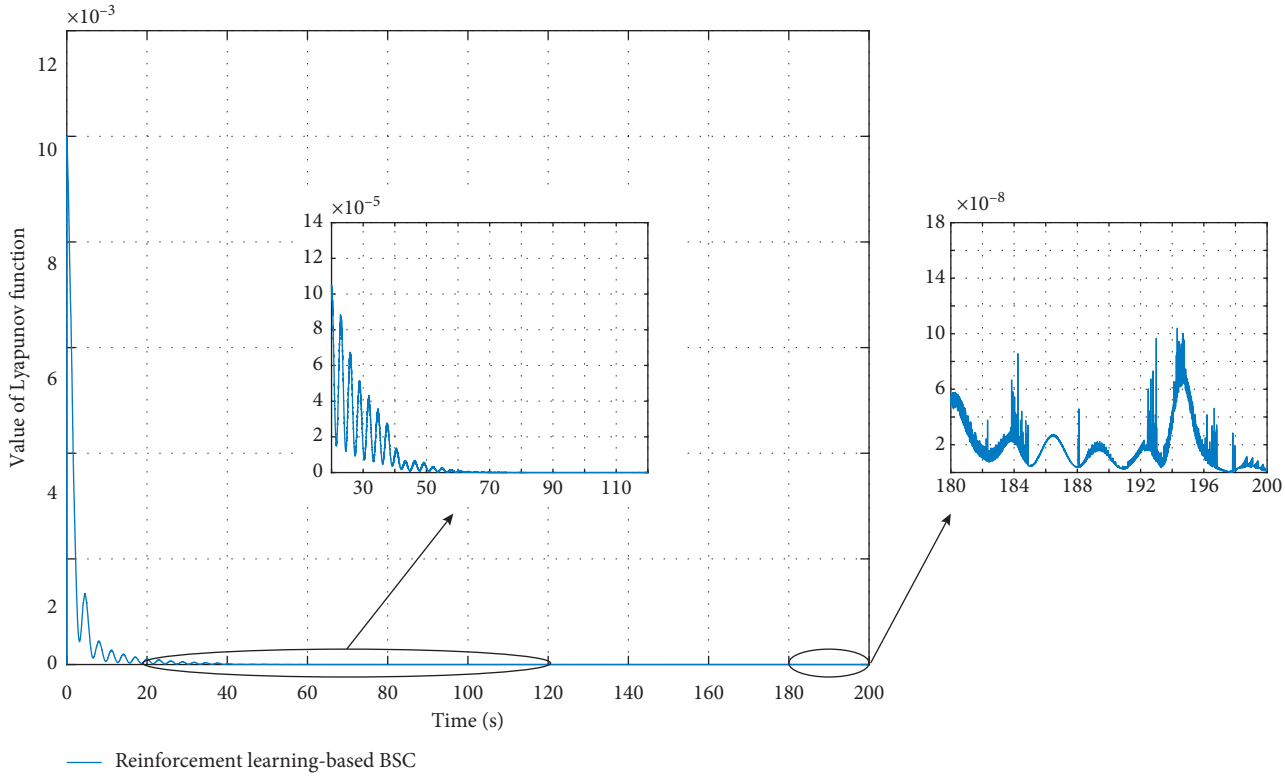


Figure 8: Value of Lyapunov function.

The dynamics of Lyapunov function are shown in Figure 8. It is clear that the Lyapunov function decreases with the fluctuation wearing off along with the time, which indicates stability is achieved by the appropriate control gain from the reinforcement learning. Fluctuating values of Lyapunov function are observed during the initial stage of control (0–50 s) because the reinforcement learning mechanism was trying bad solutions, which are in accordance with the fluctuations of the trolley position, sway angle, and control force during the initial stage of control. However, the fluctuation of Lyapunov function is still observed during the late stage (180–200 s) after learning, which is shown as the subplot in the right part of Figure 8. It is because the inputs (Γ_1 and Γ_2) used to present dynamics of

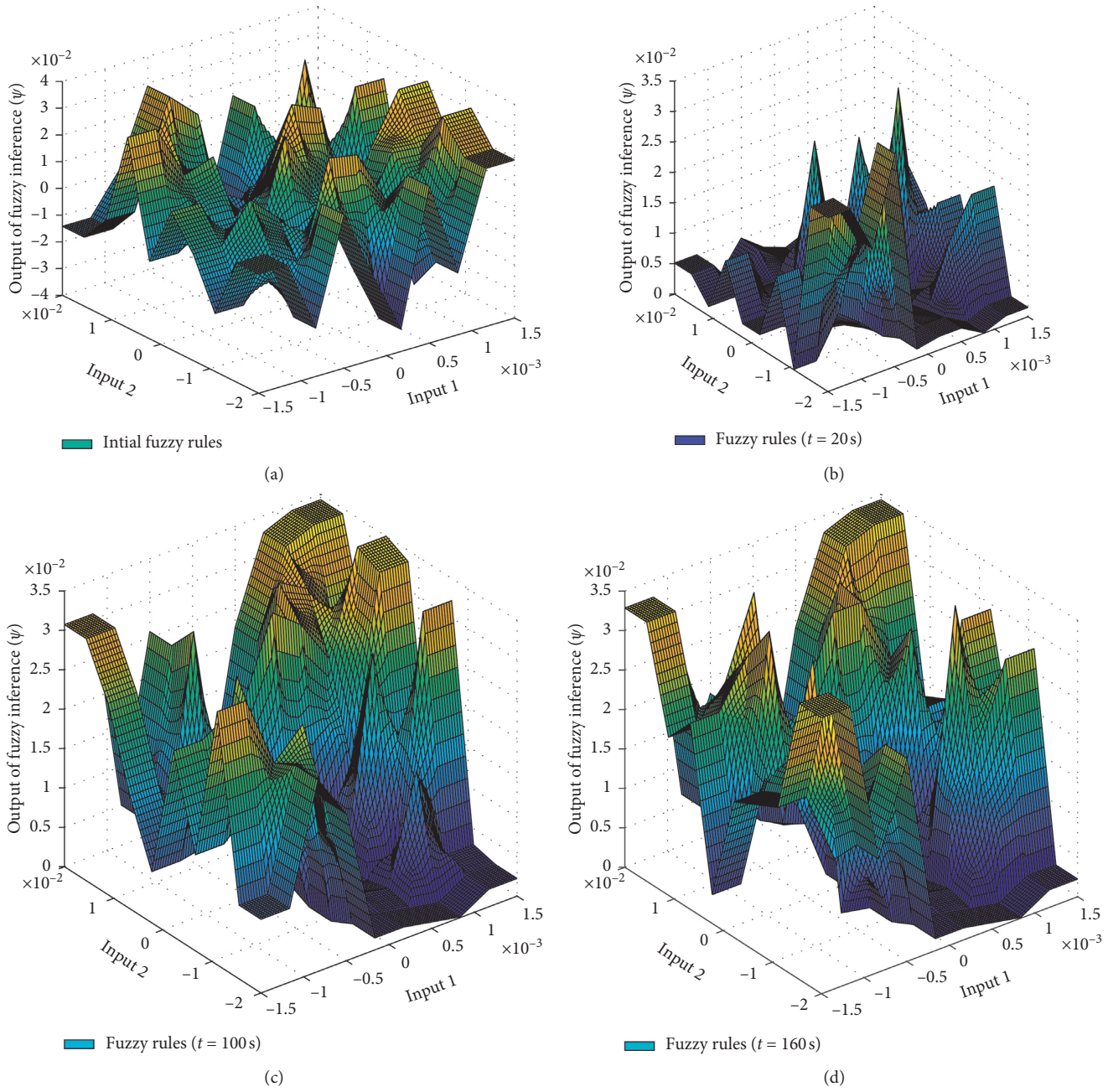
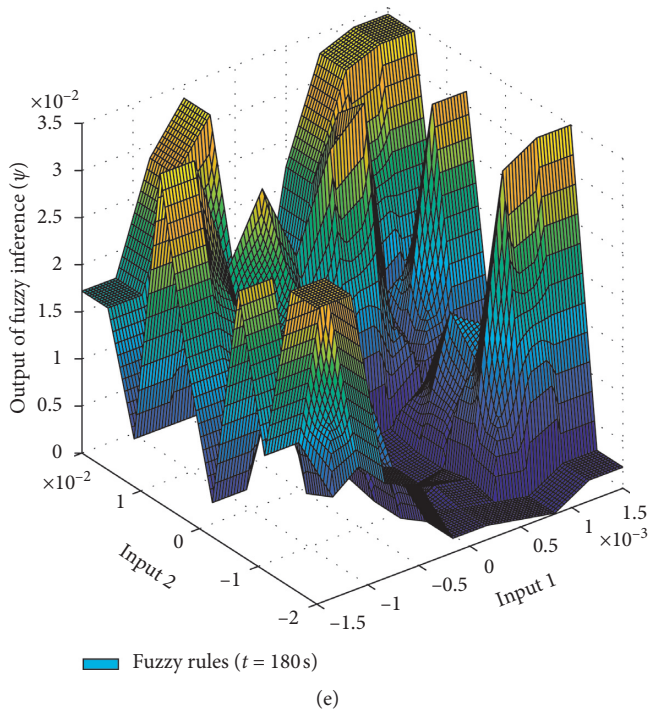


Figure 9: Continued.



(e)

Figure 9: Output from the fuzzy inference-based fuzzy rules at different times.

the second Lyapunov function are described by the limited linguistical variables (fuzzy sets) so that there are no enough fuzzy rules to correctly determine the appropriate control gain achieving the convergence when the value of Lyapunov function is close to zero. For example, the linguistic variables describing the fuzzy inputs are "zero" when $|\Gamma_1| \ll 0.0002$ and $|\Gamma_2| \ll 0.002$, which means there is only one fuzzy rule corresponding to all the states inside the range ($|\Gamma_1| \ll 0.0002, |\Gamma_2| \ll 0.0022$) though those states would need different values of control gain to achieve the convergence of Lyapunov function. A more detailed example can be used to illustrate this issue; the state of pair ($|\Gamma_1| = 0.0002, |\Gamma_2| = 0.002$) and the state of pair ($|\Gamma_1| = 0.00001, |\Gamma_2| = 0.0001$) have the same linguistic description "Γ_1 is zero and Γ_2 is zero" in the fuzzy reasoning, and consequently, the same control gain will be generated for those 2 states though they would need different control gains to achieve convergence of Lyapunov function. Therefore, the Lyapunov function will continuously fluctuate as a result of the only one fuzzy rule that corresponds to the state of "Γ_1 is zero and Γ_2 is zero" and adaptively tries different action candidates in different states inside a small range (e.g., $|\Gamma_1| \ll 0.0002$ and $|\Gamma_2| \ll 0.002$).

The outputs of fuzzy logic inference in different times are shown in Figure 9. Clearly, the output function of fuzzy logic inference is different in different times because the fuzzy rules on which the fuzzy logic inference depends are different in different instants of learning. More precisely, the initial fuzzy rules are randomly set so that the output function of fuzzy inference is random at the beginning. And then, the output function of fuzzy logic inference varies significantly during the medium period of learning ($t = 20$ s; $t = 100$ s). After that, the output function of fuzzy logic inference tends to be stable with a less variance on its shape ($t = 100$ s, $t = 160$ s, and $t = 180$ s), which means the optimal rules have been learnt.

5. Conclusions

In this paper, we proposed a reinforcement learning-based backstepping control scheme (BSC) that can handle the underactuated system of container cranes. In the control scheme, the control gain of BSC, which influences the stability, is tuned by the reinforcement fuzzy Q-learning that automatically searches the optimal fuzzy rules to generate the appropriate control gains of BSC to achieve the decrease of Lyapunov function. The simulation results show the effectiveness of the applied control scheme by the less chattering effect and less steady-state error under the uncertainty of rope length compared with the conventional SMC.

Data Availability

The data used to support the findings of this study are included within the article and can be used for other research studies.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The authors would like to thank LetPub (http://www.letpub.com) for their assistance on correcting writing issues during the

preparation of this manuscript and on improving the quality of the figures used in the manuscript. This work was financially supported by major national planning projects of China's Ministry of Industry and Information (Z135060009002-50).

References

[1] C.-Y. Chang, "The switching algorithm for the control of overhead crane," *Neural Computing and Applications*, vol. 15, no. 3-4, pp. 350–358, 2006.

[2] Y.-G. Sung and W. E. Singhose, "Robustness analysis of input shaping commands for two-mode flexible systems," *IET Control Theory & Applications*, vol. 3, no. 6, pp. 722–730, 2009.

[3] Y.-B. Kim, "A new approach to anti-sway system design problem," *KSME International Journal*, vol. 18, no. 8, pp. 1306–1311, 2004.

[4] Q. H. Ngo and K.-S. Hong, "Sliding-mode antisway control of an offshore container crane," *IEEE/ASME Transactions on Mechatronics*, vol. 17, no. 2, pp. 201–209, 2012.

[5] Q. H. Ngo and K.-S. Hong, "Adaptive sliding mode control of container cranes," *IET Control Theory & Applications*, vol. 6, no. 5, pp. 662–668, 2012.

[6] S. Frikha, M. Djemel, and N. Derbel, "Observer based adaptive neuro-sliding mode control for MIMO nonlinear systems," *International Journal of Control, Automation and Systems*, vol. 8, no. 2, pp. 257–265, 2010.

[7] D. Liu, J. Yi, D. Zhao, and W. Wang, "Swing-free transporting of two-dimensional overhead crane using sliding mode fuzzy control," in *Proceedings of the 2004 American Control Conference*, pp. 1764–1769, Boston, MA, USA, June 2004.

[8] L. A. Tuan and S.-G. Lee, "Sliding mode controls of double-pendulum crane systems," *Journal of Mechanical Science and Technology*, vol. 27, no. 6, pp. 1863–1873, 2013.

[9] F. Omar, F. Karray, O. Basir, and L. Yu, "Autonomous overhead crane system using a fuzzy logic controller," *Journal of Vibration and Control*, vol. 10, no. 9, pp. 1255–1270, 2004.

[10] A. Kaur, Priyahansha, S. Kumari, and T. Singh, "Position control of overhead cranes using fuzzy controller," *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, vol. 3, no. 5, pp. 9341–9350, 2014.

[11] L. A. Tuan, S.-C. Moon, W. G. Lee, and S.-G. Lee, "Adaptive sliding mode control of overhead cranes with varying cable length," *Journal of Mechanical Science and Technology*, vol. 27, no. 3, pp. 885–893, 2013.

[12] W. He, S. Zhang, and S. S. Ge, "Adaptive control of a flexible crane system with the boundary output constraint," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 8, pp. 4126–4133, 2014.

[13] A. K. Pal and P. K. Mudi, "An adaptive PD-type FLC and its real time implementation to overhead crane control," *International Journal of Emerging Technologies in Computational and Applied Sciences (IJETCAS)*, vol. 6, no. 2, pp. 178–183, 2013.

[14] M. Nazemizadeh, "A PID tuning method for tracking control of an underactuated gantry crane," *Universal Journal of Engineering Mechanics*, vol. 1, pp. 45–49, 2013.

[15] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, The MIT Press, Cambridge, MA, USA, 2nd edition, 2018.

[16] C. J. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[17] G. A. Rummery and M. Niranjan, *On-Line Q-Learning Using Connectionist Systems*, Vol. 37, Department of Engineering, University of Cambridge, Cambridge, UK, 1994.

[18] R. Sharma, "Lyapunov theory based stable Markov game fuzzy control for non-linear systems," *Engineering Applications of Artificial Intelligence*, vol. 55, pp. 119–127, 2016.

[19] Y. Li, L. Chen, K. P. Tee, and Q. Li, "Reinforcement learning control for coordinated manipulation of multi-robots," *Neurocomputing*, vol. 170, pp. 168–175, 2015.

[20] A. Kumar and R. Sharma, "A stable Lyapunov constrained reinforcement learn- ing based neural controller for nonlinear systems," in *Proceedings of Inter-National Conference on Computing, Communication and Automation (ICCCA)*, pp. 185–189, Greater Noida, India, May 2015.

[21] D. Liu, J. Yi, D. Zhao, and W. Wang, "Adaptive sliding mode fuzzy control for a two-dimensional overhead crane," *Mechatronics*, vol. 15, no. 5, pp. 505–522, 2005.