entropy

MDPI

Article

Optimal Tracking Control of a Nonlinear Multiagent System Using Q-Learning via Event-Triggered Reinforcement Learning

Ziwei Wang, Xin Wang *, Yijie Tang, Ying Liu and Jun Hu

College of Electronic and Information Engineering, Southwest University, Chongqing 400700, China
* Correspondence: xinwangswu@163.com

Abstract: This article offers an optimal control tracking method using an event-triggered technique and the internal reinforcement Q-learning (IrQL) algorithm to address the tracking control issue of unknown nonlinear systems with multiple agents (MASs). Relying on the internal reinforcement reward (IRR) formula, a Q-learning function is calculated, and then the iteration IRQL method is developed. In contrast to mechanisms triggered by time, an event-triggered algorithm reduces the rate of transmission and computational load, since the controller may only be upgraded when the predetermined triggering circumstances are met. In addition, in order to implement the suggested system, a neutral reinforce-critic-actor (RCA) network structure is created that may assess the indices of performance and online learning of the event-triggering mechanism. This strategy is intended to be data-driven without having in-depth knowledge of system dynamics. We must develop the event-triggered weight tuning rule, which only modifies the parameters of the actor neutral network (ANN) in response to triggering cases. In addition, a Lyapunov-based convergence study of the reinforce-critic-actor neutral network (NN) is presented. Lastly, an example demonstrates the accessibility and efficiency of the suggested approach.

Keywords: neural networks (NNs); optimal tracking control; event-triggered mechanism; reinforcement learning (RL); systems with multiple agents

Citation: Wang, Z.; Wang, X.; Tang, Y.; Liu, Y.; Hu, J. Optimal Tracking Control of a Nonlinear Multiagent System Using Q-Learning via Event-Triggered Reinforcement Learning. *Entropy* **2023**, *25*, 299. https://doi.org/10.3390/e25020299

Academic Editor: Adam Lipowski

Received: 13 December 2022
Revised: 25 January 2023
Accepted: 27 January 2023
Published: 5 February 2023

1. Introduction

Recently, distributed coordination control of MASs has received a great deal of attention as a result of its extensive applications in power systems [1,2], multi-vehicle [3] and multi-area power systems [4], and other fields. MASs have a variety problems, such as consensus control [5–7], synchronization control [8,9], anti-synchronization control [10], and tracking control [11]. Reinforcement learning (RL) [12] and adaptive dynamic programming (ADP) methods [13,14] have been employed by researchers as a means of solving the optimal control problems. Due to its excellent ability for global approximation, neural networks are excellent for dealing with nonlinearities and uncertainties [15]. ADP has great online learning and adaptive ability when it uses neural networks. Furthermore, researchers used RL/ADP algorithms to settle optimal coordination control matters, proposed a lot of directions, tracked control [16–19], graphical games [19], consensus control [20], containment control [21] and formation control [22]. The controller is designed in the above ways were relying on traditional time-triggered methods. Event-triggered in [23,24], it was suggested that the traditional implementation be changed to an event-triggered one. Because of the increasing number of agents, MASs are required to resolve many computing costs related to the exchange of information. Traditionally, the controller or actuator is constantly updated over a fixed period while the system is in operation. In order to minimize computation and preserve resources, aperiodic sampling is employed in the method of triggering events to improve the controller's computation efficiency. There have been a number of developments in methods that are based on events for addressing discrete time

systems [24]. The traditional implementation was suggested to be replaced by one that is triggered by events.

With an increase in the number of agents, MASs must solve a large number of computing costs related to information exchange. Traditionally, the controller or actuator is constantly updated frequently using a predetermined period of sampling during system operation. To lessen the computational and save resources, aperiodic sampling is used in the event-triggering scheme to improve the associated controller's computational efficiency. Researchers have developed some event-based methods to address discrete time systems [25] as well as systems based on continuous time [26,27]. Several algorithms based on triggered events have been designed to solve discrete-time systems [25], as well as systems that operate in continuous time [26,27]. According to these results, the system dynamics are assumed to be accurate ahead of time. However, it is not always possible to understand dynamics properly in practice. According to [24], a controller that was triggered by events was proposed which was designed with inaccurate or unknown dynamics for the system.

The application of Q-learning to process control [28], chemical process control, industrial process automatic control, and other areas was an early application of reinforcement learning (RL). The Q-learning algorithm provides a modeless data-driven method for solving control problems. A key point to keep in mind is all potential actions in the present state. Q-learning is currently used primarily for routing optimization and reception processing in network communication within the context of network management. The Q-learning algorithm supports a modeless data-driven method for solving control problems. It is important to note that all potential actions in the present state [29] are evaluated in the Q-learning method, relying on the Q-function. At present, Q-learning is used primarily for routing optimization and reception processing in network communication in the domain of network management [30]. As a result of AlphaGo's emergence, dynamic research has been conducted in the field of game theory, and tracking control research has been conducted on issues associated with nonlinear MAS tracking control based on Q-learning, such as in [31]. At present, there is some research for tracking control issues for nonlinear MASs based on Q-learning, such as in [32].

The MAS's issue of optimal control was solved using the RL/ADP method, as mentioned above. The majority of the above results share two common features. First, the direct use of the immediate or immediate reward (IR) signal to define each agent's performance index function results in limited learning opportunities. As a second step, a state's value function is used to determine the Hamilton–Jacobi–Bellman (HJB) equation. The corresponding controller is designed using RL/ADP, which results in efficient learning of the MAS equation. It is beneficial to provide each agent with more information signals in a wide range of realistic applications in order to enhance their learning capabilities. In addition to merely considering performance in terms of status, performance can also be viewed from a broader perspective. The purpose of our research is to avoid the limitations described above.

Taking into consideration the aforementioned findings, this work investigates an ideal solution to the optimum control issue for MASs with unknown nonlinearity to enhance the process of learning as well as the effectiveness of control systems. Utilizing the graph theory, a coordination control problem is first identified. According to the gathered information of the IR, increased reinforcement reward (IRR) signals are provided for a longer-term reward period. Based on the IRR function, a Q-function is then developed to assess the efficacy of each agent's control system. In addition, a tracking control technique is developed using iterative IrQL to derive the HJB equation for each agent. Then, based on the IrQL technique, triggering mechanisms are employed to establish a tracking control system. Finally, an optimum event-triggered controller based on a network topology of reinforce-actor-critic is created. The event triggering mechanism in a closed-loop approach guarantees that the network weights converge and the system remains stable. In light of the findings of this study, an additional contribution has been made to the literature:

(1) With respect to nonlinear MAS tracking control, the authors of [32] proposed an IrQL framework, which differs from [18,33,34], and the design of a new long-term IRR signal is completed. This product was designed on the basis of the data of neighbors to provide more information to the agent. The IRR function is used to define a Q-function, and an iterative IrQL method is proposed for obtaining control schemes that are optimally distributed.

(2) It is designed to trigger a new condition and cite in an asynchronous and distributed manner [24]. As a result, each agent triggers at its own time. Consequently, there is no need to update the controller on a regular basis. For the purpose of achieving online learning, a reinforce-actor-critic neural network based on triggered events is established to determine the optimal control scheme for triggered events. When compared with other papers [18,33,35,36], this paper adjusts the weights non-periodically, and the ANN is only adjusted when a trigger is encountered.

(3) In this paper, the objective is to develop the most effective tracking control method using a new triggering mechanism developed using the IrQL method. As far as event-triggered optimal control mechanisms are concerned, the Lyapunov approach is used to determine the rigorous stability assurance of closed-loop multi-agent networks. The designed RCA-NN framework [32] offers an effective means of executing the proposed method online without requiring any knowledge of the dynamics of the system. We made a comparison between the traditional activation method and the IrQL method. According to the simulation results, the designed algorithm is capable of detecting control problems with good tracking performance.

This article is organized as follows. The graph theory and problems of Section 2 provide an overview of some foundations. In Section 3, IrQL-based HJB equations are obtained. As described in Section 4, the most appropriate controller design should be triggered by an event to build the proposed algorithm. Section 5 develops the RCA-NN. The use of Lyapunov technology leads to convergence of weights in the neural networks. Through analogy examples and comparisons, its effectiveness and correctness of the method are demonstrated in Section 6. The last part includes our final thoughts.

2. Preliminary Findings

2.1. Theoretical Basis of Graphs

It would be possible to model the exchange of information using a directed graph between agents $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A})$, in which $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ represents N nonempty notes and $\mathcal{E} = \{(v_i, v_j)|v_i, v_j \in \mathcal{V}\} \in \mathcal{V} \times \mathcal{V}$ represents an edge set, indicating agent i could derive the data from agent j. We define $\mathcal{A} = [a_{ij}]$, which is a matrix that is adjacency relevant and does not contain negative elements a_{ij}, where $a_{ij} > 0$ is satisfied if $(i, j) \in \mathcal{E}$. Otherwise, $a_{ij} = 0$. $\mathcal{N}_i = \{j|(i, j) \in \mathcal{E}\}$ is defined as the set of nodes that are neighbors with node i, and $a_{ij} > 0$ is satisfied for each $j \in \mathcal{N}_i$. We denote the input matrix $\mathcal{D} = diag\{d_i\}$, where $d_i = \sum_{j \in \mathcal{N}} \mathcal{A}_{ij}$. The Laplacian matrix is then defined as $L = \mathcal{D} - \mathcal{A} \in \mathbb{R}^{N \times N}$.

A leader's relationship with its followers is the subject of this article. In order to describe follower-leader interactions, we propose an enhanced directed graph model, (i.e., $\hat{\mathcal{G}} = (\hat{\mathcal{V}}, \hat{\mathcal{E}})$, in which $\hat{\mathcal{V}} = \{0, 1, 2, \ldots, N\}$ and $\hat{\mathcal{E}} \in \hat{\mathcal{V}} \times \hat{\mathcal{V}}$). A leader's communication with his or her followers is determined by b_i. If $b_i > 0$, then there is an assumption that the leader and followers are in communication. Otherwise, $b_i = 0$. $\mathcal{B} = diag\{b_1, \ldots, b_n\} \in \mathbb{R}^{N \times N}$ is defined as the matrix of related connections.

2.2. Problem Formulation

If a nonlinear MAS has one leader as well as N followers, then the dynamics for the ith follower would be as follows:

$$x_i(k+1) = Ax_i(k) + B_i u_i(k) \tag{1}$$

In this case, $x_i \in \mathbb{R}^N$ represents the system state, $u_i \in \mathbb{R}^{p_i}$ represents the control input, and $A \in \mathbb{R}^{n \times n}, B_i \in \mathbb{R}^{n \times n}$ represent unknown matrices for the plants and inputs.

The leader is written as follows:

$$x_0(k+1) = Ax_0(k) \tag{2}$$

It is assumed that $x_0 \in \mathbb{R}^n$ represents the leader state.

Assumption 1. *If there is a spanning tree with a leader, then $\hat{\mathcal{G}}$ has a network of communication interactions, and $\hat{\mathcal{G}}$ does not contain repeated edges.*

Definition 1. *As a result of our design, we are able to develop a control scheme $u_i(k)$ that only requires agent information. Therefore, the followers can keep track of the leader. In the event that the funder's conditions are met, we will be able to implement a perfect control scheme [32]:*

$$\lim_{k\to\infty} \parallel x_i(k) - x_0(k) \parallel = 0, i = 1, 2, \ldots, n \tag{3}$$

The MAS's local consensus error is expressed as follows:

$$e_i(k) = \sum_{j\in\mathcal{N}_i} a_{ij}(x_i(k) - x_j(k)) + b_i(x_i(k) - x_0(k)) \tag{4}$$

Then, an overview of the error vector is presented as follows:

$$e(k) = ((L+B)\bigotimes I_n)(x(k) - \hat{x_0}(k)) \tag{5}$$

$e(k) = ({e_1}^T(k), {e_2}^T(k), \ldots, {e_n}^T(k))^T \in \mathbb{R}^{nN}$,$x(k) = ({x_1}^T(k), {x_2}^T(k), \ldots, {x_n}^T(k))^T \in \mathbb{R}^{nN}$, $\hat{x_0}(k) = I_n \otimes x_0 \in \mathbb{R}^{nN}$, as well as vector I_n having n dimensions.

The tracking error is written as $\zeta_i(k) = x_i(k) - x_0(k)$, which has the vector form

$$\zeta(k) = x(k) - \hat{x_0}(k) \tag{6}$$

In this equation, $\zeta(k) = ({\zeta_1}^T(k), {\zeta_2}^T(k), \ldots, {\zeta_n}^T(k))^T \in \mathbb{R}^{nN}$, $\hat{x_0}(k) = ({x_0}^T(k), {x_0}^T(k), \ldots, {x_0}^T(k))^T$.

Consequently, the localized neighbor error $e_i(k)$ is represented in the following manner, in agreement with Equations (1) and (4):

$$\begin{aligned} e_i(k+1) =& Ae_i(k) + (d_i + b_i)B_iu_i(k) \\ & - \sum_{j\in\mathcal{N}_i} a_{ij}B_ju_j(k) \\ =& \mathcal{F}_i(e_i(k), u_i(k)) \end{aligned} \tag{7}$$

Given Equations (5) and (6), it is evident that $e(k)$ and $\zeta(k)$ are related as follows: $\lim_{k\to\infty} \parallel e(k) \parallel = 0$ as $\lim_{k\to\infty} \parallel \zeta(k) \parallel = 0$. Consequently, when the localized neighboring error is close to zero, the control problem is resolved.

3. Design of the IrQL Method

To resolve the issue of tracking control in systems with multiple agents, the authors of [32] developed the IrQL method. What is important is that in order to provide agents with a greater level of local information from other agents or environments, it is necessary to introduce IRR information, thereby improving control and learning efficiency. In addition, agents have been defined according to the Q-function, and the relevant HJB equation is acquired using the IrQL method.

As an example, consider the following IR function for the ith agent:

$$\begin{aligned} j_i(e_i(k), u_i(k), u_{-i}(k)) =& e_i(k)^T R_{ii} e_i(k) + u_i(k)^T Q_{ii} u_i(k) \\ & + \sum_{j\in N_i} u_j(k)^T Q_{ij} u_j(k) \end{aligned} \tag{8}$$

In this case, we can represent the agent's neighbors' input with $u_{-i} = \{u_j | j \in \mathcal{N}_i\}$. The weight matrices $R_{ii} > 0$, $Q_{ii} > 0$, and $Q_{ij} > 0$ are positive.

According to the IR function, as a function of IRR, the following is expressed:

$$R_i(e_i(k), u_i(k), u_{-i}(k)) = \sum_{s=k}^{\infty} r^{s-k} j_i(e_i(s), u_i(s), u_{-i}(s)) \tag{9}$$

where the IRR function is defined as $r \in (0, 1]$ and r is its discount factor.

The following performance indices must be minimized for every agent to find a solution to the issue of controlling tracking optimally:

$$J_i(e_i(0), u_i(0), u_{-i}(0)) = \sum_{t=0}^{\infty} \beta^t R_i(e_i(t), u_i(t), u_{-i}(t)) \tag{10}$$

In this case, its performance index discount factor is $\beta \in (0, 1]$.

Remark 1. *The function of the designed IRR function incorporates accumulated prospective long-term reward data from the IR function. The performance factor is measured depending on IRR as opposed to IR, which is contrary to the majority of methods. The advantage is that we can enhance the control actions, and the learning process can be accelerated by using a great deal of data.*

Remark 2. *Intrinsic motivation (IM) provides a possible method for enhancing the faculty of abstract actions or solving the difficulties associated with exploring the environment in its reinforcement learning direction. IRR acts as a driving agent that learns skills through intrinsic motivation [32].*

Definition 2. *In order to resolve the MAS's tracking control issue, we propose a distributed tracking control scheme. As the time step k approaches infinity, $e_i(k) \longrightarrow 0$ minimizes the performance metrics* (10) *simultaneously.*

We can obtain a state value function as follows based on the control method of the agent as well as the neighbors $u_i(t)$ and $u_{-i}(t)$:

$$V_i(e_i(k)) = \sum_{t=k}^{\infty} \beta^{t-k} R_i(e_i(t), u_i(t), u_{-i}(t)) \tag{11}$$

Equation (11) can also be expressed as the following formula:

$$V_i(e_i(k) = R_i(e_i(k), u_i(k), u_{-i}(k)) + \beta V_i(e_i(k+1)) \tag{12}$$

Based on the theory, the ideal state value function meets the following conditions:

$$V_i^*(e_i(k)) = \min_{u_i(k)} \{R_i(e_i(k), u_i(k), u_{-i}(k)) + \beta V_i^*(e_i(k+1))\} \tag{13}$$

In this case, in Bellman form, the function of IRR is expressed as

$$\begin{aligned} &R_i(e_i(k), u_i(k), u_{-i}(k)) \\ &\quad = j_i(e_i(k), u_i(k), u_{-i}(k)) \\ &\qquad + \varrho R_i(e_i(k+1), u_i(k+1), u_{-i}(k+1)) \end{aligned} \tag{14}$$

On the basis of the condition of stationarity, (i.e., $\frac{\partial V_i^*(e_i(k))}{\partial u_i(k)}$), the description of the optimal distributed control method is given below:

$$\begin{aligned} u_i^*(k) &= \underset{u_i(k)}{argmin}\{R_i(e_i(k), u_i(k), u_{N_i}(k)) + \beta V_i^*(e_i(k+1))\} \\ &= -\frac{1}{2}\beta(d_i + b_i)Q_{ii}^{-1}h_i^T(x_i(k)) \bigtriangledown V_i^*(e_i(k+1)) \end{aligned} \tag{15}$$

In this equation, $\bigtriangledown V_i^*(e_i(k+1)) = \frac{\partial V_i^*(e_i(k+1))}{\partial e_i(k+1)}$.

Remark 3. *As is well known, the state value algorithm $V_i(e_i(k))$ is highly concerned with the space of states. In accordance with the state action function, the Q-learning method is designed with RL. The Q-function can be used by each agent to estimate the properties of all possible decisions in the current situation, and we can determine what is the best behavior of the agent at each step by using the Q-function.*

The Q-function is written as follows:

$$\begin{aligned} Q_i(e_i(k), u_i(k), u_{-i}(k)) = R_i(e_i(k), &u_i(k), u_{-i}(k)) \\ &+ \beta V_i(e_i(k+1)) \end{aligned} \tag{16}$$

In accordance with the optimal scheme, the optimal Q-function is given by

$$\begin{aligned} &Q_i^*(e_i(k), u_i(k), u_{-i}(k)) \\ &= R_i(e_i(k), u_i(k), u_{-i}(k)) \\ &\quad + \beta Q_i^*(e_i(k+1), u_i^*(k+1), u_{-i}^*(k+1)) \end{aligned} \tag{17}$$

Based on Equations (16) and (17), we can express the optimal solution as follows:

$$u_i^*(k) = \underset{u_i(k)}{argmin}\{Q_i^*(e_i(k), u_i(k), u_{-i}(k))\} \tag{18}$$

In comparison with the control method of Equation (15), its optimum Q-function provides the optimal solution for the control scheme here. As a result, we intend to calculate the solution to Equation (17).

4. Designs of the Event-Driven Controller

According to a previous work [18], a time-triggered controller was developed. Nevertheless, a new event-triggering mechanism is designed to minimize computing costs for this case.

$Q_i^*, (e_i(k), u_i(k), u_{-i}(k))$ is defined as the sequence of trigger times. At the triggering instant, the sampled disagreement error is expressed as $\hat{e}_i^s$.

As a result of the threshold value and error, the triggering time varies. The control scheme can only be updated when $k = kt_s^i$ and cannot be updated under any other circumstances:

$$u_i(k) = u_i(kt_s^i), k \in [kt_s^i, kt_{s+1}^i) \tag{19}$$

To design a triggering condition, we propose a function that measures the gap arising from the existing error and the previously sampled error:

$$\epsilon_i^s(k) = \hat{e}_i^s - e_i(k), k \in [kt_s^i, kt_{s+1}^i) \tag{20}$$

We have set the triggering error equal to zero at $k = kt_s^i$.

The dynamic expression of localized mistakes based on an event-triggered controlling approach can be written as

$$e_i(k+1) = \mathcal{F}_i(e_i(k), u_i(kt_s^i)) \tag{21}$$

Thus, the equation for event-triggered events is obtained:

$$\begin{aligned} &V_i^*(e_i(k)) = \\ &\quad \min_{u_i(kt_s^i)} \left\{ R_i(e_i(k), u_i(kt_s^i), u_{-i}(kt_s^i)) + \beta V_i^*(\mathcal{F}_i(e_i(k), u_i(kt_s^i))) \right\} \end{aligned} \tag{22}$$

$$\begin{aligned} &Q_i^*(e_i(k)) \\ &\quad = R_i(e_i(k), u_i(kt_s^i), u_{-i}(kt_s^i)) \\ &\qquad + \beta Q_i^*(\mathcal{F}_i(e_i(k), u_i(kt_s^i))) \end{aligned} \tag{23}$$

It is possible to express the optimal tracking control using an event-triggered approach in the following way:

$$u_i^*(k) = arg \min_{u_i(kt_s^i)} \{Q_i^*(e_i(k))\} \tag{24}$$

Assumption 2. *There is a constant $\mathcal{L}$ that explains the inequality below:*

$$\left\| \mathcal{F}_i(e_i(k), u_i(kt_s^i)) \right\| \leq \mathcal{L}\|e_i(k)\| + \mathcal{L}\|\epsilon_i^s(k)\| \tag{25}$$

Assumption 3. *There is a triggering condition which is as follows:*

$$\|\epsilon_i^s(k)\|^2 \leq (1 - 2\mathcal{L}^2)/(2\mathcal{L}^2)\|e_i(k)\|^2 = \pi_i T \tag{26}$$

where $\pi_i T$ represents the triggering threshold and $\mathcal{L} \in (0, \sqrt{2}/2)$ [24]. Once the multi-agent system dynamics have stabilized, followers are able to track their leaders.

5. Neural Network Implementation for the Event-Triggered Approach Using the IrQL Method

This section discusses the tree-NN structure, also known as RCA-NNs. Three virtual networks are included in the tree-NN structure.

5.1. Reinforce Neutral Network (RNN) Learning Model

The reinforced NN is employed to approximate the IRR signal as follows:

$$\hat{R}(Z_{ri}(k)) = \varphi_{ri}(\omega_{r2i}^T(k) \cdot \varphi_{ri}(\omega_{r1i}^T(k) \cdot Z_{ri}(k))) \tag{27}$$

where $Z_{ri}(k)$ represents the input vector, which has $e_i(k)$, $u_i(k)$, while $u_{-i}(k)$. ω_{r1i} represents the matrix of weights for input-to-hidden layering. Meanwhile, ω_{r2i} represents the matrix of weights for hidden-to-output layering, and $\varphi_{ri}(\cdot)$ represents an activation function [24].

Due to the reinforced NN, the associated error function is as follows:

$$\begin{aligned} e_{ri}(k) = j_i(e_i(k-1), &u_i(k-1), u_{-i}(k-1)) \\ &+ \varrho \hat{R}_i(Z_{ri}(k)) - \hat{R}_i(Z_{ri}(k-1)) \end{aligned} \tag{28}$$

The loss function is written as

$$E_{ri}(k) = \frac{1}{2} e_{ri}^2(k) \tag{29}$$

For convenience's sake, only the matrices ω_{r2i} are updated, and the matrices ω_{r1i} remain unchanged during the training process.

The RNN's update law is expressed as

$$\omega_{r2i}(k+1) = \omega_{r2i}(k) - \alpha_{ri} \cdot \left(\frac{\partial E_{ri}(k)}{\partial \omega_{r2i}(k)} \right) \tag{30}$$

In this equation, α_{ri} represents the rate at which the RNN learns.

The gradient descent rule (GDR) is used to obtain an updated law for the reinforced NN's weight, which yields the following results:

$$\begin{aligned} &\omega_{ri}(k+1) \\ &= \omega_{ri}(k) - \alpha_{ri} \cdot \left(\frac{\partial E_{ri}(k)}{\partial e_{ri}(k)} \cdot \frac{\partial e_{ri}(k)}{\partial \hat{R}(Z_{ri}(k))} \cdot \frac{\partial \hat{R}(Z_{ri}(k))}{\partial \omega_{r2i}(k)} \right) \\ &= \omega_{r2i}(k) - \alpha_{ri} \varrho e_{ri}(k) \left[1 - \varphi_{ri}^2 \left(\omega_{r2i}^T(k) \cdot \Delta_{ri}(k) \right) \right] \Delta_{ri}(k) \end{aligned} \tag{31}$$

In this equation , $\Delta_{ri}(k) = \varphi_{ri}(\omega_{r1i}^T(k) \cdot Z_{ri}(k))$.

5.2. Critic Neutral Network (CNN) Learning Model

In the following section, when designing the critic NN, an attempt is made to achieve a close approximation of the Q-function:

$$\hat{Q}_i(Z_{ci}(k)) = \omega_{c2i}^T(k) \cdot \varphi_{ci}(\omega_{c1i}^T(k) \cdot Z_{ci}(k)) \tag{32}$$

In this equation, $Z_{ci}(k)$ represents the relative vector of inputs that has $\hat{R}_i(k)$, $e_i(k)$, and $u_i(k)$ as well as $u\_i(k)$, while $\omega_{c1i}^T(k)$ and $\omega_{c2i}^T(k)$ represent the input layer weight matrices and output layer weight matrices.

It is possible to express the function of the error for the CNN to be

$$e_{ci}(k) = \hat{R}_i(Z_{ri}(k-1)) + \beta \hat{Q}_i(Z_{ci}(k)) - \hat{Q}_i(Z_{ci}(k-1)) \tag{33}$$

Its function of loss is written to be

$$E_{ci}(k) = \frac{1}{2} e_{ci}^2(k) \tag{34}$$

In accordance with the operation of RNNs, only ω_{c2i} is updated, and ω_{c1i} remains unchanged.

With the help of the gradient descent rule (GDR), it can be used to express the weight update law:

$$\omega_{c2i}(k+1) = \omega_{c2i}(k) - \alpha_{ci} \left(\frac{\partial E_{ci}(k)}{\partial \omega_{c2i}(k)} \right) \tag{35}$$

where α_{ci} represents the critic NN's learning rate. Furthermore, we can obtain its weight update schemes for the critic NN:

$$\begin{aligned} &\omega_{c2i}(k+1) \\ &= \omega_{c2i}(k) - \alpha_{ci} \left(\frac{\partial E_{ci}(k)}{\partial e_{ci}(k)} \cdot \frac{\partial e_{ci}(k)}{\partial \hat{Q}_i(Z_{ci}(k))} \cdot \frac{\partial \hat{Q}_i(Z_{ci}(k))}{\partial \omega_{c2i}(k)} \right) \\ &= \omega_{c2i}(k) - \alpha_{ci} \beta \Big[\hat{R}_i(Z_{ci}(k-1)) + \beta \omega_{c2i}^T(k) \cdot \Delta_{ci}(k) \\ &- \omega_{c2i}^T(k-1) \cdot \Delta_{ci}(k-1) \Big] \cdot \Delta_{ci}(k) \end{aligned} \tag{36}$$

In this equation, $\Delta_{ci}(k) = \varphi_{ci}(\omega_{ci1}^T(k) Z_{ci}(k))$.

5.3. Actor Neutral Network (ANN) Learning Model

Based on the actor NN, an approximate optimal scheme is defined as follows:

$$\hat{u}_i(k) = \omega_{a2i}^T \cdot \varphi_{ai}(\omega_{a1i}^T \cdot Z_{ai}(k)) \tag{37}$$

where the input data of the ANN is represented by $Z_{ai}(k) = e_i(k)$, ω_{a1i} represents the weight matrices of the input layer, and ω_{a2i} represents the weight matrices of the output layer.

Based on the prediction error of the actor NN, the following result is obtained:

$$e_{ai}(k) = \hat{Q}_i(Z_{ci}(k)) - \mathcal{U}_c \tag{38}$$

It is possible to express the function of loss of the ANN to be

$$E_{ai}(k) = \frac{1}{2} e_{ai}(k) \tag{39}$$

As with RNNs and CNNs, ω_{a1i} must remain unchanged throughout the learning process. The actor NN update laws are defined as follows:

$$\omega_{a2i}(k+1) = \omega_{a2i}(k) - \alpha_{ai}\left(\frac{\partial E_{ai}(k)}{\partial \omega_{a2i}(k)}\right) \tag{40}$$

where α_{ai} represents the ANN learning rate. We can design a weight-tuning scheme for an ANN as follows:

$$\begin{aligned} &\omega_{a2i}(k+1) \\ &= \omega_{a2i}(k) - \alpha_{ai} \cdot \left(\frac{\partial E_{ai}(k)}{\partial e_{ai}(k)} \cdot \frac{\partial e_{ai}(k)}{\partial \hat{Q}_i(Z_{ci}(k))}\right. \\ &\left.\times \frac{\partial \hat{Q}_i(Z_{ci}(k))}{\partial \hat{u}_i(k)} \cdot \frac{\partial \hat{u}_i(k)}{\partial \omega_{a2i}(k)}\right) \\ &= \omega_{a2i}(k) - \alpha_{ai}\Delta_{ai}(k)\omega_{a2i}^T(k) \\ &\times \bigtriangledown_{ci}^{'}(k)\omega_{c1i}^T(k) \bigtriangledown_{\hat{u}_i}(Z_{ci}(k))\left[\omega_{a2i}^T \Delta_{ci}(k)\right] \end{aligned} \tag{41}$$

where $\Delta_{ai}(k) = \varphi_{ai}(\omega_{a1i}^T(k) Z_{ai}(k))$, $\bigtriangledown_{ci}^{'}(k) = \frac{\partial \varphi_{ci}(\omega_{c1i}(k) Z_{ci}(k))}{\partial \varphi_{ci}(\omega_{c1i}^T(k) Z_{ci}(k))}$, $\bigtriangledown_{\hat{u}_i}(Z_{ci}(k)) = \frac{\partial Z_{ci}(k)}{\partial \hat{u}_i(k)}$.

Furthermore, we can obtain

$$\omega_{a2i}(k+1) = \begin{cases} \omega_{a2i}(k) - \alpha_{ai}\Delta_{ai}(k)\omega_{a2i}^T(k) \\ \times \bigtriangledown_{ci}^{'}(k)\omega_{c1i}^T(k) \bigtriangledown_{\hat{u}_i}(Z_{ci}(k))\left[\omega_{a2i}^T \Delta_{ci}(k)\right], k = kt_s^i \\ \omega_{a2i}(k), k \in [kt_s^i, kt_{s+1}^i). \end{cases} \tag{42}$$

It is described in detail in Algorithm 1 how the controller is designed using RCA-NNs and event triggering. When the trigger conditions are met, the actor NN is updated.

For analysis of stability based on the Lyapunov method, we present an analysis of stability and convergence in the following section.

Assumption 4. *The following conditions are assumed to be true:* $\|\omega_{r2i}(k)\| \leq \omega_{rim}$, $\|\omega_{c2i}(k)\| \leq \omega_{cim}$, $\|\omega_{a2i}(k)\| \leq \omega_{aim}$. *There are bounded activation functions, i.e.,* $\|\Delta_{ri}(k)\| \leq \Delta_{rim}$, $\|\Delta_{ci}(k)\| \leq \Delta_{cim}$, $\|\Delta_{ai}(k)\| \leq \Delta_{aim}$. *What's more, the functions of activation* $\varphi_{ai}(k)$ *is the function of Lipschitz that satisfies* $\left\|\varphi_{ai}(e_i(kt_s^i)) - \varphi_{ai}(k)\right\| \leq \theta_{ai}\left\|e_i(kt_s^i) - e_i(k)\right\| = \theta_{ai}\left\|\epsilon_i^s(k)\right\| \leq \theta_{ai}\pi_i T$, *where* θ_{ai}, $\pi_i T$ *are positive constants. Approximation errors of NNs' output can be defined to be:* $\delta_{ci}(k) = \omega_{c2i}(k)\Delta_{ci}(k)$, $\delta_{ai}(k) = \omega_{a2i}(k)\Delta_{ai}(k)$, $\vartheta_{ri}(k) = \omega_{r2i}(k)\Delta_{ri}(k)$.

Theorem 1. *Assume that Assumptions 1 and 2 are true. CNN and ANN weights are renewed by (36) and (42). Upon satisfying the triggering term(26), the local inconsistency error is* $e_i(k)$, *critic*

evaluated error and actor evaluated error error are consistent and ultimately bounded. Furthermore the control method u_i converges to the optimal value u_i^*.

Evidence: Set $\widetilde{\omega}_{r2i}(k) = \omega_{r2i}(k) - \omega_{r2i}^*$ as the weighting assessment error between the optimal weights for RNNs ω_{r2i}^*. Its assessment $\omega_{r2i}(k)$, $\widetilde{\omega}_{c2i}(k) = \omega_{c2i}(k) - \omega_{c2i}^*$ is the error resulting from weighting evaluation involving the ideal CNN weights ω_{c2i}^* ; its assessed $\omega_{c2i}(k)$, as well as $\widetilde{\omega}_{a2i}(k) = \omega_{a2i}(k) - \omega_{a2i}^*$ is the weighting evaluated error involving the ideal ANN weightings ω_{a2i}^* and its estimation $\omega_{a2i}(k)$.

Algorithm 1 RCA neural networks based on the IrQL method with event triggering.

Set initial value:
1: Set initial values for $\omega_{r2i}(0), \omega_{a2i}(0), \omega_{c2i}(0)$ between $(0,1)$;
2: Set a low level of degree of precision for the calculation $\mathcal{E}$.
3: Initialize the score of $x_i(0), x_0(0)$ within $(0,1)$
The iterative process: Make *kisequalto*0. Error calculation at the localized level $e_i(k)$;
4: Keep on;
5: Based on actor NN, estimate $\hat{u}_i(k)$ by (37)
6: Update the reinforce NN;
7: Via the inputting $[e_i(k), u_i(k), u_{-i}(k)]$ into the reinforce NN, and we can obtain the estimated the function of IRR $R_i(Z_{ri}(k))$ via (27)
8: Obtain $e_{ri}(k)$ by (28);
9: Renew the matrices $\omega_{r2i}(k)$ by (31);
10: Renew the critic NN:
11: Via the inputting $[\hat{R}_i(Z_{ri}(k)), e_i(k), u_i(k), and u_{-i}(k)]$ into critic NN, and we can obtain its estimated Q-function via (32);
12: Obtain $e_{ci}(k)$ by (33);
13: Renew the matrices $\omega_{c2i}(k)$ by (36);
14: Renew the actor NN:
15: Input $[e_i(k)]$ to the actor NN, and we can obtain the estimated Q-function $\hat{u}_i(k)$ via (37)
16: Calculation $e_{ai}(k)$ via (38)
17: In the event that the triggering conditions are met, renew the matrices $\omega_{a2i}(k)$ of the actor NN using (41)
18: Otherwise, do not update the weight matrices $\omega_{a2i}(k)$
19: Until $\|\omega_{c2i}(k+1) - \omega_{c2i}(k)\| \leq \mathcal{E}$; otherwise, set $k = k+1$, then go to procedure (5)
20: Keep on $\omega_{r2i}(k), \omega_{c2i}(k), \omega_{a2i}(k)$ as the optimal weights.

(1) We can obtain the following function at the time of triggering as follows:

$$L(k) = L_1(k) + L_2(k) + L_3(k) + L_4(k) + L_5(k) \tag{43}$$

In this equation,

$$\begin{aligned} L_1(k) &= \frac{1}{\alpha_{ri}} tr(\omega_{r2i}^T(k)\omega_{r2i}(k)), \\ L_2(k) &= \frac{1}{\alpha_{ci}} tr(\omega_{c2i}^T(k)\omega_{c2i}(k)), \\ L_3(k) &= \frac{1}{\alpha_{ai}} tr(\omega_{a2i}^T(k)\omega_{a2i}(k)), \\ L_4(k) &= \varrho^k \hat{R}_i(k), \\ L_5(k) &= \beta^k \hat{Q}_i(k). \end{aligned} \tag{44}$$

$\Delta L_1(k)$ is written to be

$$\Delta L_1(k) = \frac{1}{\alpha_{ri}} tr(\omega_{r2i}^T(k+1)\omega_{r2i}(k+1) - \omega_{r2i}^T(k)\omega_{r2i}(k)). \tag{45}$$

In this equation, we have

$$\begin{aligned}\widetilde{\omega}_{r2i}(k+1) &= \omega_{r2i}(k+1) - \omega^*_{r2i} \\ &= \omega_{r2i}(k) - \alpha_{ri}\varrho[j(k-1) + \varrho\hat{R}(k) - \hat{R}(k-1)] \\ &\times \delta_{ri}(k)\Delta_{ri}(k)\end{aligned} \tag{46}$$

Furthermore, we have

$$\begin{aligned}\Delta L_1(k) =& -2\varrho^2\delta_{ri}(k)[\varrho^{-1}j(k) + \hat{R}(k) - \hat{R}(k-1)] \\ &+ \alpha_{ri}\varrho^4[\varrho^{-1}j(k) + \hat{R}(k) - \hat{R}(k-1)]^2\vartheta^2_{ri}(k) \\ =&\Big\{\delta_{ri}(k) - \varrho^2[\varrho^{-1}j(k) + \hat{R}(k) - \hat{R}(k-1)]\Big\}^2 \\ &- (1 - \alpha_{ri}(k)\Delta^2_{ri}(k))\varrho^4[\varrho^{-1}j(k) + \hat{R}(k) - \hat{R}(k-1)]^2 \\ &\times \vartheta^2_{ri}(k) - \delta^2_{ri}(k)\end{aligned} \tag{47}$$

$\Delta L_2(k)$ can be written as

$$\Delta L_2(k) = \frac{1}{\alpha_{ci}} tr(\omega^T_{c2i}(k+1)\omega_{c2i}(k+1) - \omega^T_{c2i}(k)\omega_{c2i}(k)). \tag{48}$$

Within this equation, we have

$$\begin{aligned}\widetilde{\omega}_{c2i}(k+1) &= \omega_{c2i}(k+1) - \omega^*_{c2i} \\ &= \omega_{c2i}(k) - \alpha_{ci}\beta\Delta_{ci}(k)[\hat{R}_i(k-1) \\ &+ \beta(\omega_{c2i}(k) + \omega^*_{c2i})\Delta_{ci}(k) - \omega^T_{c2i}(k-1)\Delta_{ci}(k-1)]\end{aligned} \tag{49}$$

Furthermore, we have

$$\Delta L_2(k) = \frac{1}{\alpha_{ci}}\Big\{D_1 + D_2 + D_3 - \omega^T_{c2i}(k)\omega_{c2i}(k)\Big\} \tag{50}$$

where

$$\begin{aligned}D_1 &= \omega^T_{c2i}(k)(I - \alpha_{ci}\beta^2\Delta_{ci}(k)\Delta^T_{ci}(k))^2\omega_{c2i}(k \\ &= \|\omega_{c2i}(k)\|^2 - 2\alpha_{ci}\beta^2\|\delta_{ci}(k)\|^2 \\ &+ \alpha^2_{ci}\beta^4\|\Delta_{ci}(k)\|^2\|\delta_{ci}(k)\|^2 \\ D_2 &= -2\alpha_{ci}\beta^2\delta_{ci}(k)[\beta^{-1}\hat{R}_i(k-1) + (\omega^*_{c2i})^T\Delta_{ci}(k) \\ &- \beta^{-1}\omega_{c2i}(k-1)\Delta_{ci}(k-1)]\|\Delta_{ci}(k)\|^2 \\ D_3 &= \alpha^2_{ci}\beta^4[\beta^{-1}\hat{R}_i(k-1) + (\omega^*_{c2i})^T\Delta_{ci}(k) \\ &- \beta^{-1}\omega^T_{c2i}(k-1)\Delta_{ci}(k-1)]^T \\ &\times [\beta^{-1}\hat{R}_i(k-1) + (\omega^*_{c2i})^T\Delta_{ci}(k) - \beta^{-1}\omega^T_{c2i}(k-1)\Delta_{ci}(k-1)]\end{aligned}$$

The following result is obtained by computation:

$$\begin{aligned}\Delta L_2(k) =& -\beta^2\|\delta_{ci}(k)\|^2 - \beta^2(1 - \alpha_{ci}\beta^2\|\Delta_{ci}(k)\|^2) \\ &\times ||\delta_{ci}(k) + \beta^{-1}\hat{R}_i(k-1) + (\omega^*_{c2i})^T\Delta_{ci}(k) \\ &-\beta^{-1}\omega^T_{c2i}(k-1)\Delta_{ci}(k-1))||^2 \\ &+ ||\hat{R}_i(k-1) + \beta(\omega^*_{c2i})^T\Delta_{ci}(k) \\ &- \omega^T_{c2i}(k-1)\Delta_{ci}(k-1)||^2\end{aligned} \tag{51}$$

In the case of the difference of the first order of $L_3(k)$, we can obtain

$$\Delta L_3(k) = \frac{1}{\alpha_{ai}}(\omega_{a2i}^T(k+1)\omega_{a2i}(k+1) - \omega_{a2i}^T(k)\omega_{a2i}(k)) \tag{52}$$

where,

$$\begin{aligned}\widetilde{\omega}_{a2i}(k+1) &= \omega_{a2i}(k+1) - \omega_{a2i}^* \\ &= \omega_{a2i}(k) - \alpha_{ai}\Delta_{ai}(k)\omega_{c2i}^T(k)C(k) \\ &\times [\omega_{c2i}^T(k)\Delta_{ci}(k)]\end{aligned} \tag{53}$$

Therefore, we have

$$\Delta L_3(k) = \frac{1}{\alpha_{ai}}(E1 - \omega_{a2i}^T(k)\omega_{a2i}(k)) \tag{54}$$

where

$$\begin{aligned}E1 =& ||\omega_{a2i}(k)||^2 - 2\alpha_{ai}\omega_{c2i}^T(k)C(k)\delta_{ai}(k)[\omega_{c2i}^T(k)\Delta_{ci}(k)] \\ &+ \alpha_{ai}||\omega_{c2i}^T(k)\Delta_{ci}(k)||^2||\Delta_{ai}(k)||^2||\omega_{c2i}^T(k)C(k)||^2\end{aligned} \tag{55}$$

In the case of $\Delta L_3(k)$, the simplified formula is given below:

$$\begin{aligned}\Delta L_3(k) =& -(1 - \alpha_{ai}||\Delta_{ai}(k)||^2)||\omega_{c2i}^T(k)\Delta_{ci}(k)||^2 \\ &\times ||\omega_{c2i}^T(k)C(k)||^2 - ||\delta_{ai}(k)||^2 \\ &+ ||\omega_{c2i}^T(k)C(k)\Delta_{ai}(k)\omega_{c2i}^T(k) - \delta_{ai}(k)||^2\end{aligned} \tag{56}$$

By adding Equations (47), (51), and (56), we can obtain $L(k)$ as follows:

$$\begin{aligned}\Delta L(k) &= \Delta L_1(k) + \Delta L_2(k) + \Delta L_3(k) + \Delta L_4(k) + \Delta L_5(k) \\ &= -\beta^2||\delta_{ci}(k)||^2 - \beta^2(1 - \alpha_{ci}\beta^2||\Delta_{ci}(k)||^2) \\ &\times ||\delta_{ci}(k) + \beta^{-1}\hat{R}_i(k-1) + (\omega_{c2i}^*)^T\Delta_{ci}(k) \\ &- \beta^{-1}\omega_{c2i}^T(k-1)\Delta_{ci}(k-1)||^2 \\ &- (1 - \alpha_{ai}||\Delta_{ai}(k)||^2) \\ &\times ||\omega_{c2i}^T(k)\Delta_{ci}(k)||^2||\omega_{c2i}^T(k)C(k)||^2 + ||\hat{R}_i(k-1) \\ &+ \beta(\omega_{c2i}^*)^T\Delta_{ci}(k) - \omega_{c2i}^T(k-1)\Delta_{ci}(k-1)||^2 \\ &+ ||\omega_{c2i}^T(k)C(k)\Delta_{ci}^T(k)\omega_{c2i}^T(k) - \delta_{ai}(k)||^2 \\ &- (1 - \alpha_{ri}\Delta_{ri}^2(k))\varrho^4 \\ &\times ||\varrho^{-1}j(k) + \hat{R}_i(k) - \varrho^{-1}\hat{R}_i(k-1)||^2 \\ &\vartheta^2(k) + ||\delta_{ri}(k) - \varrho^2[\varrho^{-1}j(k) + \hat{R}_i(k) \\ &- \varrho^{-1}\hat{R}_i(k-1)]||^2 \\ &- ||\delta_{ai}(k)||^2 - ||\delta_{ri}(k)||^2 + \beta^{k+1}Q_i(k+1) - \beta^kQ_i(k) \\ &+ \varrho^{k+1}R_i(k+1) - \varrho^kR_i(k)\end{aligned} \tag{57}$$

Therefore, we can obtain

$$\begin{aligned}\Delta L(k) = &-\beta^2||\delta_{ci}(k)||^2 - \beta^2(1-\alpha_{ci}\beta^2||\Delta_{ci}(k)||^2)\times||\delta_{ci}(k)\\ &+\beta^{-1}V_1(k)||^2-(1-\alpha_{ai}||\Delta_{ai}(k)||^2)||X_1(k)||^2\\ &\times||W_1(k)||^2+||V_1(k)||^2+||W^1(k)X_1^T(k)-\delta_{ai}(k)||^2\\ &-(1-\alpha_{ai}(k)||\Delta_{ri}(k)||^2)\varrho^4||\varrho^{-1}Y_1(k)||^2v_{ri}^2(k)\\ &-||\delta_{ai}(k)||^2-||\delta_{ri}(k)||^2+\beta^{k+1}Q_i(k+1)-\beta^kQ_i(k)\\ &+\varrho^{k+1}R_i(k+1)-\varrho^kR_i(k)\end{aligned} \tag{58}$$

where $V_1(k)=\hat{R}_i(k-1)+\beta(\omega_{c2i}^*)^T\Delta_{ci}(k)-\omega_{c2i}^T(k-1)\Delta_{ci}(k-1), W_1(k)=\omega_{c2i}^T(k)C(k)$, $X_1(k)=\omega_{c2i}^T(k)\Delta_{ci}(k), Y_1(k)=j(k)+\varrho\hat{R}_i(k)-\hat{R}_i(k-1)$, and we can obtain $||V_1(k)||\le V_{1m}, ||W_1(k)||\le W_{1m}, ||X_1(k)||\le X_{1m}, ||Y_1(k)||\le Y_{1m}$. Next, we can obtain

$$\begin{aligned}\Delta L(k)\le &-\beta^2||\delta_{ci}(k)||^2-\beta^2(1-\alpha_{ci}\beta^2||\Delta_{ci}(k)||^2)\\ &\times||\delta_{ci}(k)+\beta^{-1}V_1(k)||^2\\ &-(1-\alpha_{ai}||\Delta_{ai}(k)||^2)||X_1(k)||^2||W_1(k)||^2\\ &+2||W_1(k)X_1^T(k)||^2+||\delta_{ai}(k)||^2\\ &-(1-\alpha_{ri}||\Delta_{ri}(k)||^2)\varrho^2||Y_1(k)||^2\vartheta_{ri}^2(k)\\ &+2||\delta_{ri}(k)||^2+2||Y_1(k)||^2\\ &-\beta^kQ_i(k)-\varrho^kR_i(k)\end{aligned} \tag{59}$$

Moreover, we can obtain

$$\begin{aligned}\Delta L(k)\le &-\beta^2||\delta_{ci}(k)||^2-\beta^2(1-\alpha_{ci}\beta^2||\Delta_{ci}(k)||^2)\\ &\times||\delta_{ci}(k)+\beta^{-1}V_1(k)||^2\\ &-(1-\alpha_{ai}||\Delta_{ai}(k)||^2)||X_1(k)||^2||W_1(k)||^2\\ &+V_{1m}^2+2W_{1m}^2X_{1m}^2+2||(\omega_{a2i}^*)^T\Delta_{ai}(k)||^2\\ &+2||\omega_{a2i}^T\Delta_{ai}(k)||^2\\ &-(1-\alpha_{ri}||\Delta_{ri}(k)||^2)\varrho^2||Y_1(k)||^2\vartheta_{ri}^2(k)\\ &+2||\delta_{ri}(k)||^2+2||Y_1(k)||^2\\ &-\beta^kQ_i(k)-\varrho^kR_i(k)\\ \le &-\beta^2||\delta_{ci}(k)||^2-\beta^2(1-\alpha_{ci}\beta^2||\Delta_{ci}(k)||^2)\\ &\times||\delta_{ci}(k)+\beta^{-1}V_1(k)||^2\\ &-(1-\alpha_{ai}||\Delta_{ai}(k)||^2)||X_1(k)||^2||W_1(k)||^2\\ &+V_{1m}^2+2W_{1m}^2X_{1m}^2+4\omega_{aim}^2\Delta_{aim}^2\\ &-(1-\alpha_{ri}||\Delta_{ri}(k)||^2)\varrho^2||Y_1(k)||^2\vartheta_{ri}^2(k)\\ &+2\delta_{rim}^2+2Y_{1m}^2\\ &-\beta^kQ_i(k)-\varrho^kR_i(k)\end{aligned} \tag{60}$$

If the conditions are met, then we can obtain

$$\alpha_{ri}\le\frac{1}{||\Delta_{ri}(k)||^2}, \alpha_{ci}\le\frac{1}{\beta^2||\Delta_{ci}(k)||^2}, \alpha_{ai}\le\frac{1}{||\Delta_{ai}(k)||^2}$$
$$||\delta_{ci}(k)||>\sqrt{(V_{1m}^2+2W_{1m}^2X_{1m}^2+4\omega_{a2im}^2\Delta_{aim}^2+2\delta_{rim}^2+2Y_{1m}^2)/\beta^2}$$

We can derive $\Delta L(k)\le 0$. The proof has been completed.

(2) In the absence of the triggering conditions, consider the following:

$$L(k) = L_1(k) + L_2(k) + L_4(k) \tag{61}$$

where

$$\begin{aligned} L_1(k) &= \frac{1}{\alpha_{ri}} tr(\omega_{r2i}^T(k)\omega_{r2i}(k)), \\ L_2(k) &= \frac{1}{\alpha_{ci}} tr(\omega_{c2i}^T(k)\omega_{c2i}(k)), \\ L_4(k) &= e_i^T(k)e_i(k) \end{aligned}$$

$$\begin{aligned} \Delta L(k) &= \Delta L_1(k) + \Delta L_2(k) + \Delta L_4(k) \\ &= -\beta^2||\delta_{ci}(k)||^2 - \beta^2(1 - \alpha_{ci}\beta^2||\Delta_{ci}(k)||^2) \\ &\times ||\delta_{ci}(k) + \beta^{-1}V_1(k)||^2 + ||V_1(k)||^2 \\ &- (1 - \alpha_{ri}||\Delta_{ri}(k)||^2)\varrho^2||Y_1(k)||^2\vartheta_{ri}^2(k) \\ &+ 2\delta_{rim}^2 + 2Y_{1m}^2 \\ &+ e_i^T(k+1)e_i(k+1) - e_i^T(k)e_i(k) \end{aligned} \tag{62}$$

$$\begin{aligned} \Delta L(k) &\leq -\beta^2||\delta_{ci}(k)||^2 - \beta^2(1 - \alpha_{ci}\beta^2||\Delta_{ci}(k)||^2) \\ &\times ||\delta_{ci}(k) + \beta^{-1}V_1(k)||^2 + ||V_1(k)||^2 \\ &- (1 - \alpha_{ri}||\Delta_{ri}(k)||^2)\varrho^2||Y_1(k)||^2\vartheta_{ri}^2(k) \\ &+ 2\delta_{rim}^2 + 2Y_{1m}^2 \\ &+ ((\iota||e_i(k) + \iota||\epsilon_i^s||)^2 - ||e_i(k)||^2) \\ &\leq -\beta^2||\delta_{ci}(k)||^2 - \beta^2(1 - \alpha_{ci}\beta^2||\Delta_{ci}(k)||^2) \\ &\times ||\delta_{ci}(k) + \beta^{-1}V_1(k)||^2 + V_{1m}^2 \\ &- (1 - \alpha_{ri}||\Delta_{ri}(k)||^2)\varrho^2||Y_1(k)||^2\vartheta_{ri}^2(k) \\ &+ 2\delta_{rim}^2 + 2Y_{1m}^2 \\ &- (1 - 2\iota^2)||e_i(k)||^2 - 2\iota^2||\epsilon_i^s||^2 \end{aligned} \tag{63}$$

In the event that it is satisfied that $\alpha_{ri} \leq \frac{1}{||\Delta_{ri}(k)||^2}, \alpha_{ci} \leq \frac{1}{\beta^2||\Delta_{ci}(k)||^2}, \alpha_{ai} \leq \frac{1}{||\Delta_{ai}(k)||^2}$, and $||\delta_{ci}(k)|| > \sqrt{(V_{1m}^2 + 2\delta_{rim}^2 + 2Y_{1m}^2)/\beta^2}$, one has $\Delta L(k) \leq 0$. Thus, we can derive $\Delta L(k) \leq 0$, and the proof is completed.

6. Statistical Data Illustration

To demonstrate the viability of the proposed method, a simulation is presented in the following section.

Nonlinear MAS Consisting of One Leader and Six Followers

There were six followers and one leader in this tangled set of MASs which were considered. Figure 1 depicts the connection graph of the studied MASs. There was a leader of 0, and there were followers of 1, 2, 3, 4, 5, and 6. It is possible to obtain the corresponding adjacency matrix $a_{14} = a_{21} = a_{32} = a_{43} = a_{52} = a_{65} = 1$. There is a weighted relationship involving the leaders and followers where $b_1 = 1, b_2 = b_3 = b_4 = b_5 = b_6 = 0$. It is possible for agent 1 to accept the information of the leader immediately. The system model parameters for MASs with one leader as well as six followers are as follows: $A = \begin{vmatrix} 0.995 & 0.09980 \\ -0.09982 & 0.995 \end{vmatrix}, B_1 = [0, 0.2]^T, B_2 = [0, 0.5]^T, B_3 = [0, 0.4]^T, B_4 = [0, 0.3]^T, B_5 = [0, 0.6]^T$, and $B_6 = [0, 0.7]^T$.

The weight matrices are as follows: $Q_{11} = Q_{22} = Q_{33} = Q_{44} = Q_{55} = Q_{66} = 1, R_{11} = R_{22} = R_{33} = R_{44} = R_{55} = R_{66} = I_{2\times2}$, and $Q_{14} = Q_{21} = Q_{32} = Q_{43} = Q_{52} = Q_{65} = I_{2\times2}$.

The learning rates are $\alpha_{ri} = 0.95, \alpha_{ai} = 0.90$, and $\alpha_{ci} = 0.07(iisequalto1, 2, 3, 4, 5, 6)$, with a discount factor of $\varrho = 0.57, \beta = 0.9$.

For the agents, the activation function of the RNNs and ANNs is as follows: $Z_{r1}(k) = [e_1^T(k), u_1^T(kt_s^1), u_4^T(kt_s^4)]^T$,$Z_{a1}(k) = e_1(kt_s^1)$,$Z_{r2}(k) = [e_2^T(k), u_2^T(kt_s^2), u_1^T(kt_s^1)]^T$,$Z_{a2}(k) = e_2(kt_s^2)$,$Z_{r3}(k) = [e_3^T(k), u_3^T(kt_s^3), u_2^T(kt_s^2)]^T$,$Z_{a3}(k) = e_3(kt_s^3)$,$Z_{r4}(k) = [e_4^T(k), u_4^T(kt_s^4), u_3^T(kt_s^3)]^T$, $Z_{a4}(k) = e_4(kt_s^4)$,$Z_{r5}(k) = [e_5^T(k), u_5^T(kt_s^5), u_2^T(kt_s^2)]^T$,$Z_{a5}(k) = e_5(kt_s^5)$,$Z_{r6}(k) = [e_6^T(k), u_6^T(kt_s^6), u_5^T(kt_s^5)]^T$,$Z_{a6}(k) = e_6(kt_s^6)$. The initial values of the leader and followers are $x_0(0) = [0.6675, 0.7940]^T$, $x_1(0) = [0.5734, 0.6000]^T$, $x_2(0) = [0.5667, 0.7348]^T$, $x_3(0) = [0.8694, 0.7140]^T$, $x_4(0) = [1.0212, 1.3842]^T$, $x_5(0) = [0.8606, 1.5565]^T$, and $x_6(0) = [0.5274, 1.3235]^T$.

According to Figure 2, all followers of the leader were able to accurately follow the leader, and the whole MAS was able to achieve synchronization. Figure 3 illustrates the six agents' cumulative amount of trigger instants. On average, the amount of trigger instants for the six agents was approximately 220. However, using the traditional RL method, the number was approximately 1000. As a result, the computational burden was reduced by 78.0% in comparison with the conventional time-triggered method. According to Figure 4, the trigger mechanism of each agent is illustrated, which indicates that the actor network weight will be updated only when the trigger mechanism is satisfied. As can be seen in Figure 5, there is a correlation involving the error of triggering $||\epsilon_i^s(k)||^2$ as well as the minimum triggering requirements $\pi_i T$. Over time, it appears that the triggering error converged. Figures 6 and 7 illustrate the evaluation of the local neighborhood errors using the proposed control method, and it is shown that they could be converged to 0 at k = 60. The local neighborhood errors of [32] are shown in Figures 8 and 9. In comparison with Figures 8 and 9, our proposed control method produced a better convergence effect. Figures 10 and 11 show the estimation of the ANN weight parameters. With the proposed control method, the actor network weights can stabilize faster than with IrQL.

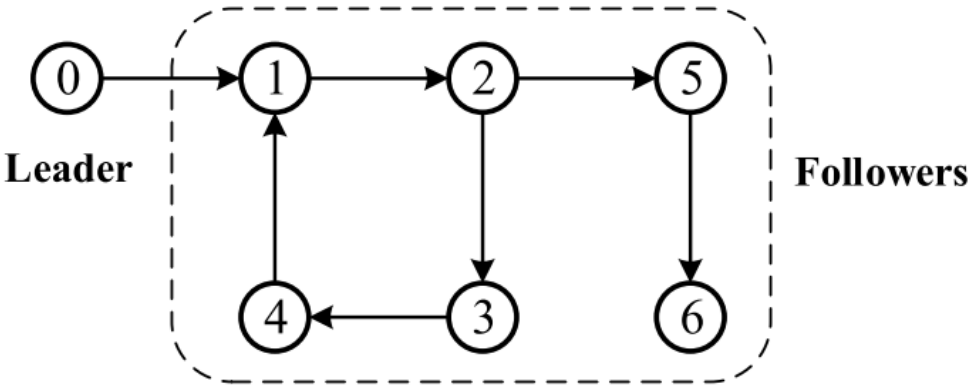

Figure 1. The topology structure for leader-follower MASs.

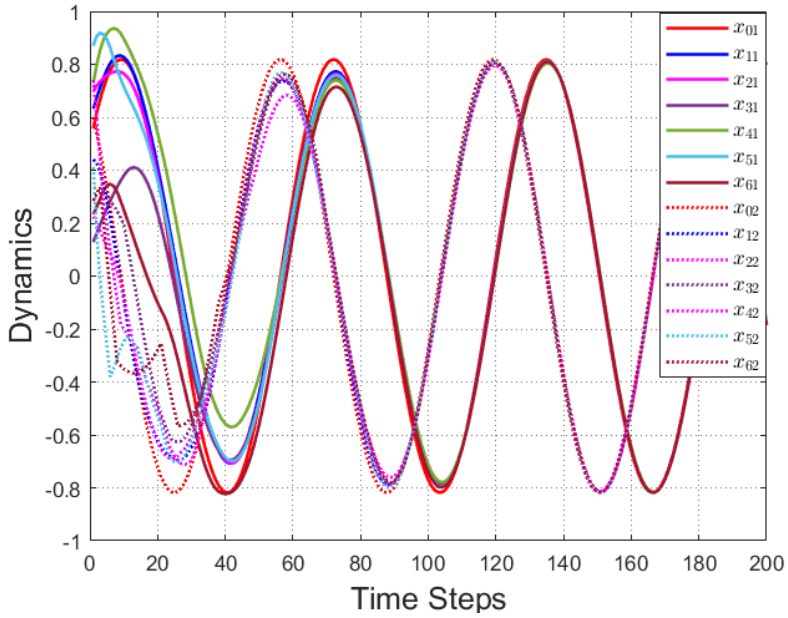

Figure 2. The tracks for the leader and followers.

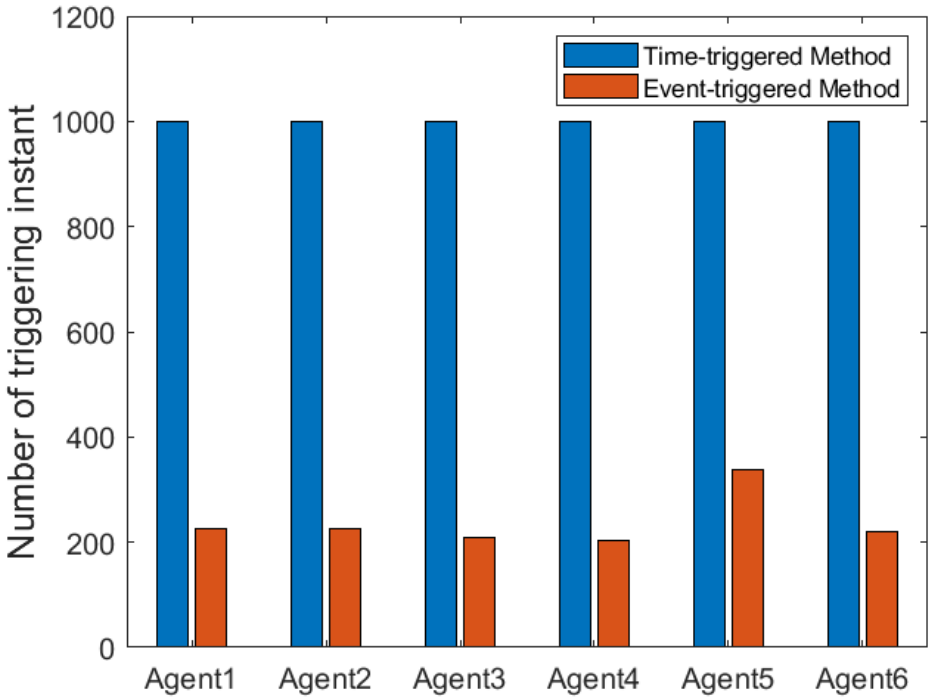

Figure 3. The comparison of the trigger time number involving the suggested method as well as the conventional approach.

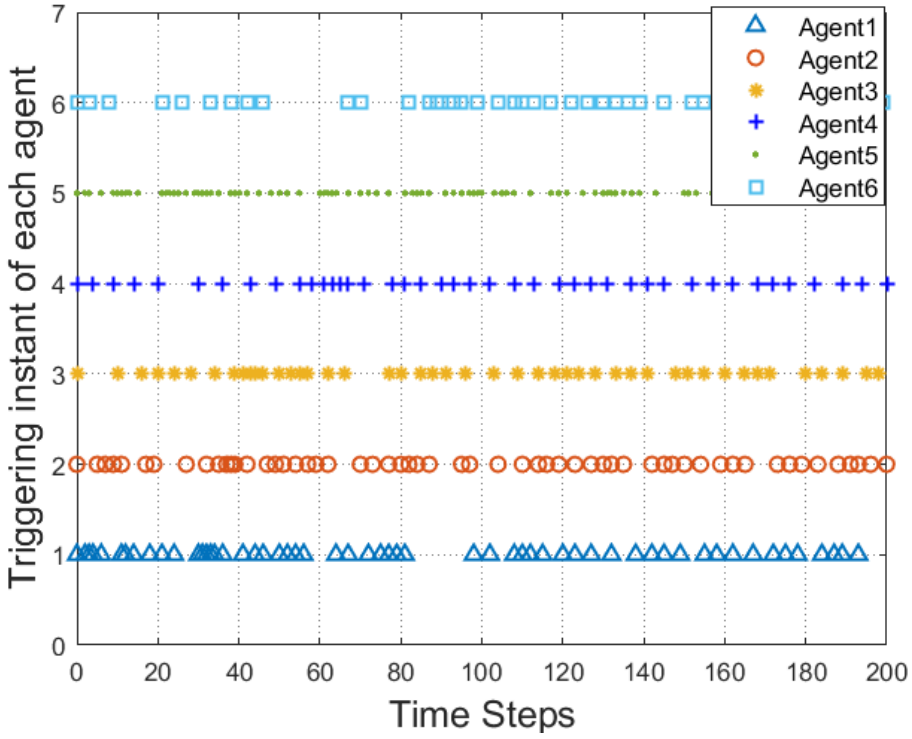

Figure 4. The triggering instant for each agent.

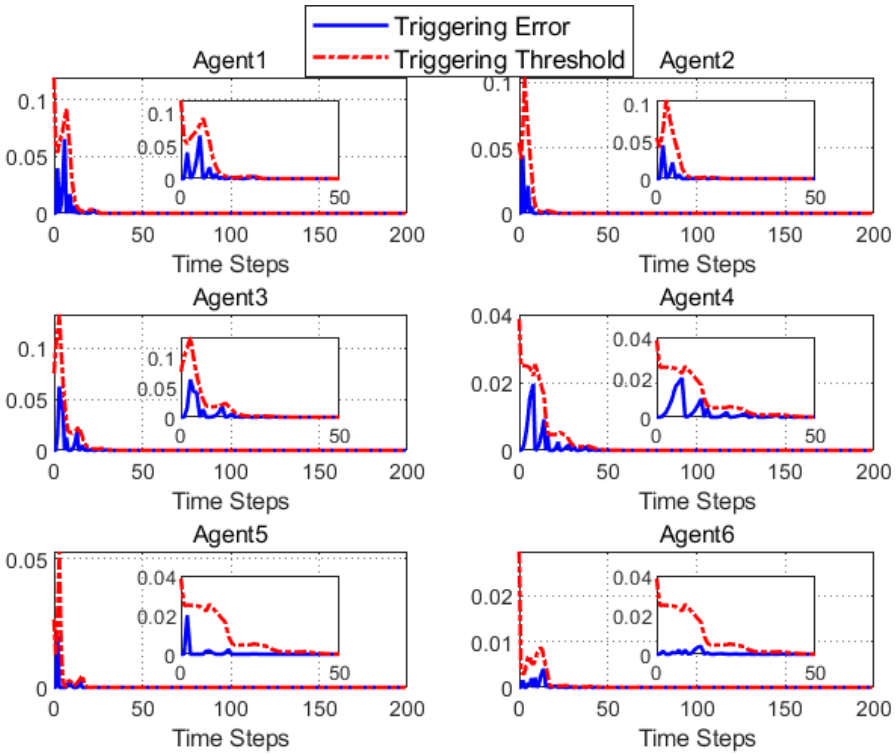

Figure 5. The triggering error trajectory $||\epsilon_i^s(k)||^2$ in addition to triggering thresholds $\pi_i T (i = 1, 2, 3, 4, 5, 6)$.

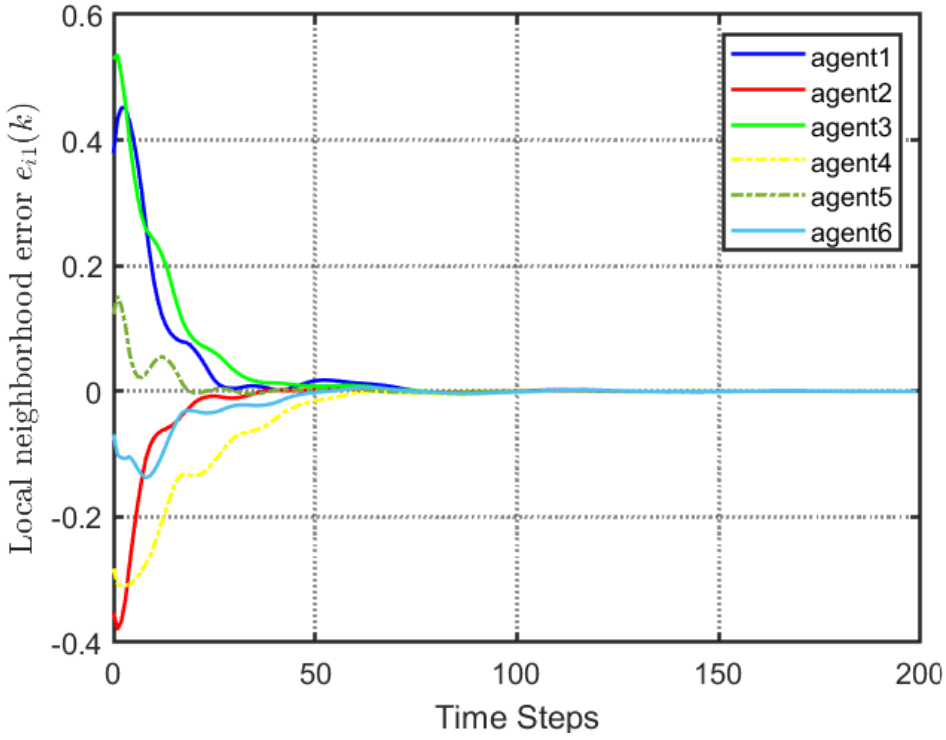

Figure 6. Local neighborhood errors $e_{i1}(k)$ with the proposed control method.

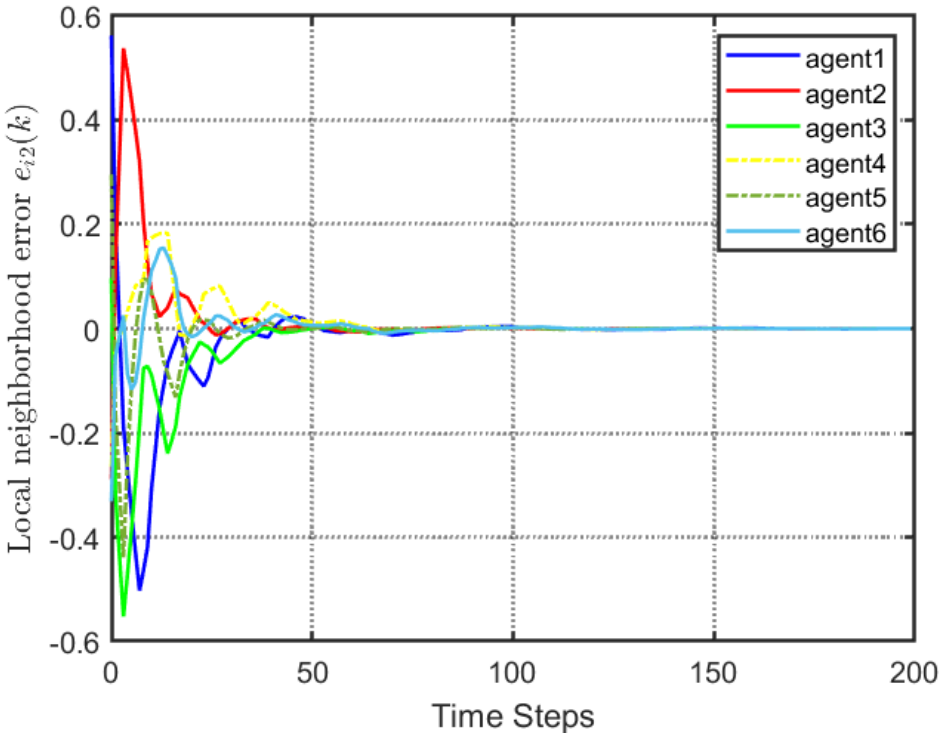

Figure 7. Local neighborhood errors $e_{i2}(k)$ with proposed control method.

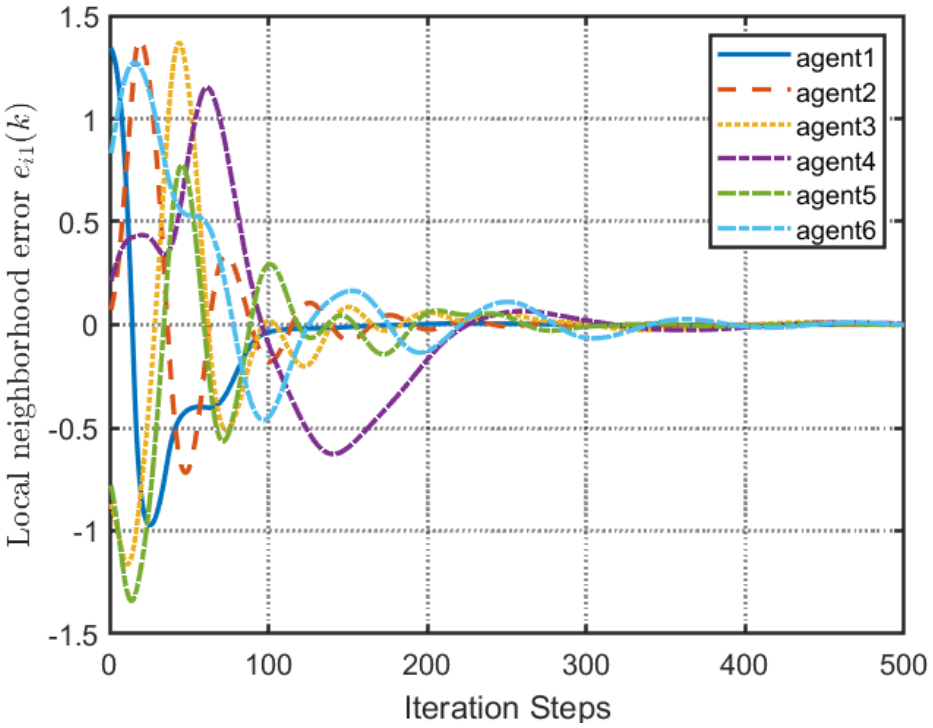

Figure 8. Local neighborhood errors $e_{i1}(k)$ of [32].

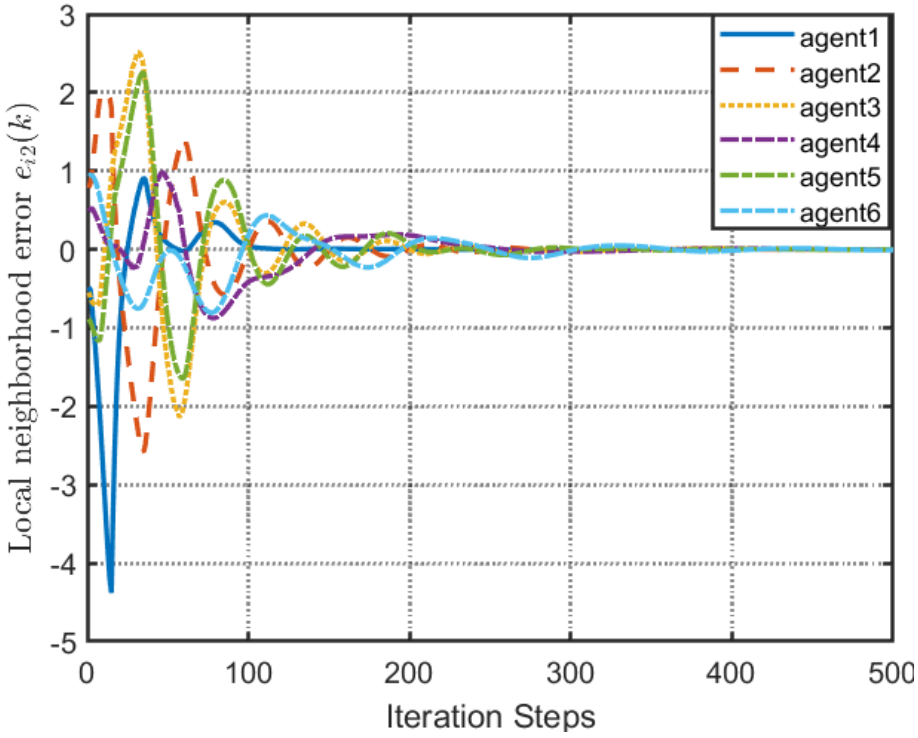

Figure 9. Local neighborhood errors $e_{i2}(k)$ of [32].

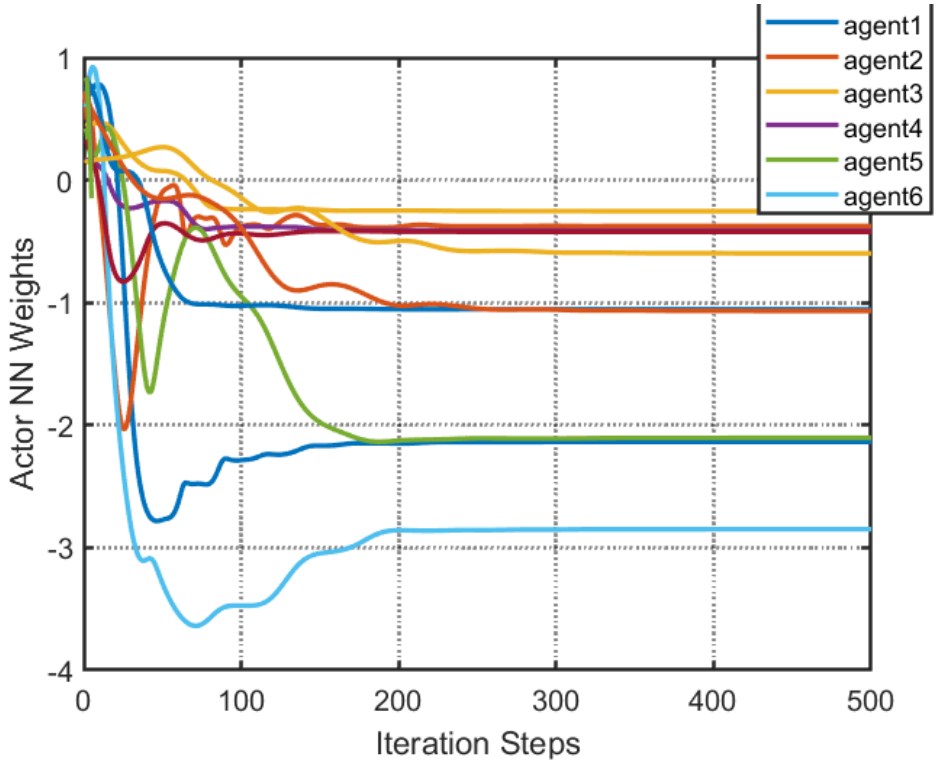

Figure 10. The estimation of weight parameters of the ANN of [32].

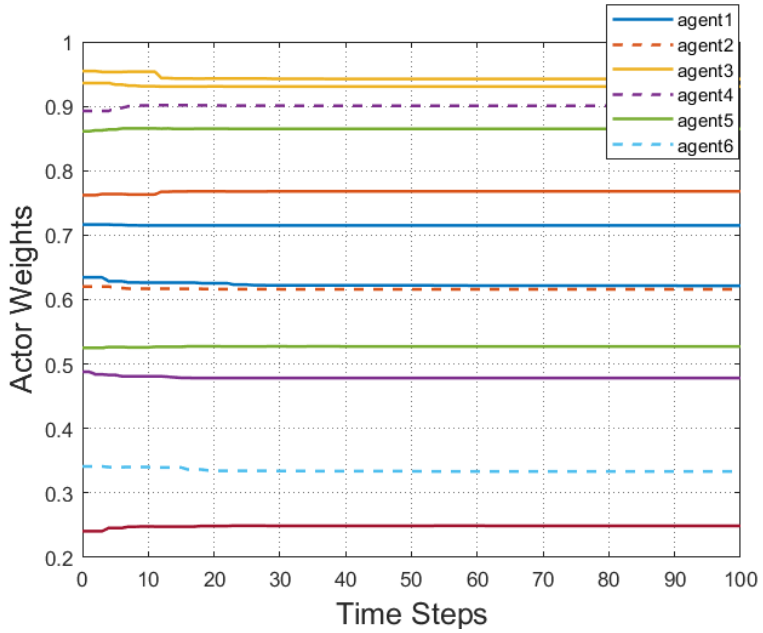

Figure 11. Estimation of the weight parameters of an ANN using the proposed control method.

7. Conclusions

According to this study, an event-triggered optimum controlling problem for model-free MASs was examined using the IrQL method based on RL. A new IrQL method was introduced by adding additional IRR functions [32], As a result, more information could be obtained by the agent. As a consequence of defining the IRR formula, we defined the Q-function and derived the corresponding HJB equation. In an iterative approach to IrQL, this method was designed to calculate the optimal control strategy. Using the IrQL algorithm, an event-triggered controller utilizing the IrQL method was presented. It was

designed to update the controller only at the time of triggering to reduce the burden on computing resources and the transmission network. An RCA-NN was used to implement the suggested approach, which eliminated the need for a model of the system. It is possible to determine the convergent weights of neural networks using the Lyapunov method. To assess the performance and control efficiency of the suggested algorithm, a simulation model was used. Further research will be conducted on the effect of the discount rates on system reliability.

Author Contributions: Software, Y.T., Y.L. and J.H.; Writing—review & editing, Z.W.; Supervision, X.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wen, G.; Yu, X.; Liu, Z.W.; Yu, W. Adaptive consensus-based robust strategy for economic dispatch of smart grids subject to communication uncertainties. *IEEE Trans. Ind. Inform.* **2018**, *14*, 2484–2496. [CrossRef]
2. Li, P.; Hu, J.; Qiu, L.; Zhao, Y.; Ghosh, B.K. A distributed economic dispatch strategy for power-water networks. *IEEE Trans. Control Netw. Syst.* **2021**, *9*, 356–366. [CrossRef]
3. Fax, J.A.; Murray, R.M. Information flow and cooperative control of vehicle formations. *IEEE Trans. Autom. Control* **2004**, *49*, 1465–1476. [CrossRef]
4. Wen, S.; Yu, X.; Zeng, Z.; Wang, J. Event-triggering load frequency control for multiarea power systems with communication delays. *IEEE Trans. Ind. Electron.* **2016**, *63*, 1308–1317. [CrossRef]
5. Wen, G.; Wang, P.; Huang, T.; Lü, J.; Zhang, F. Distributed consensus of layered multi-agent systems subject papers. *IEEE Trans. Circuits Syst.* **2020**, *67*, 3152–3162. [CrossRef]
6. Wu, Z.G.; Xu, Y.; Pan, Y.J.; Su, H.; Tang, Y. Event-triggered control for consensus problem in multi-agent systems with quantized relative state measurements and external disturbance. *IEEE Trans. Circuits Syst.* **2018**, *65*, 2232–2242. [CrossRef]
7. Liu, H.; Cheng, L.; Tan, M.; Hou, Z.G. Exponential finite-time consensus of fractional-order multiagent systems. *IEEE Trans. Syst. Man Cybern. Syst.* **2020**, *50*, 1549–1558. [CrossRef]
8. Shi, K.; Wang, J.; Zhong, S.; Zhang, X.; Liu, Y.; Cheng, J. New reliable nonuniform sampling control for uncertain chaotic neural networks under Markov switching topologies. *Appl. Math. Comput.* **2019**, *347*, 169–193. [CrossRef]
9. He, W.; Chen, G.; Han, Q.L.; Du, W.; Cao, J.; Qian, F. Multi-agent systems on multilayer networks: Synchronization analysis and network design. *IEEE Trans. Syst.* **2017**, *47*, 1655–1667.
10. Hu, J.; Wu, Y. Interventional bipartite consensus on coopetition networks with unknown dynamics. *J. Frankl. Inst.* **2017**, *354*, 4438–4456. [CrossRef]
11. Hu, J.P.; Feng, G. Distributed tracking control of leader follower multi-agent systems under noisy measurement. *Automatica* **2010**, *46*, 1382–1387. [CrossRef]
12. Wu, X.; Tang, Y.; Cao, J. Input-to-State Stability of Time-Varying Switched Systems with Time Delays. *IEEE Trans. Autom. Control* **2019**, *64*, 2537–2544. [CrossRef]
13. Chen, D.; Liu, X.; Yu, W. Finite-time fuzzy adaptive consensus for heterogeneous nonlinear multi-agent systems. *IEEE Trans. Netw. Sci. Eng.* **2021**, *7*, 3057–3066. [CrossRef]
14. Wang, J.L.; Wang, Q.; Wu, H.N.; Huang, T. Finite-time consensus and finite-time H_∞ consensus of multi-agent systems under directed topology. *IEEE Trans. Netw. Sci. Eng.* **2020**, *7*, 1619–1632. [CrossRef]
15. Ren, Y.; Zhao, Z.; Zhang, C.; Yang, Q.; Hong, K.S. Adaptive neural-network boundary control for a flexible manipulator with input constraints and model uncertainties. *IEEE Trans. Cybern.* **2021**, *51*, 4796–4807. [CrossRef]
16. Mu, C.; Zhao, Q.; Gao, Z.; Sun, C. Q-learning solution for optimal consensus control of discrete-time multiagent systems using reinforcement learning. *J. Frankl. Inst.* **2019**, *356*, 6946–6967. [CrossRef]
17. Peng, Z.; Zhao, Y.; Hu, J.; Ghosh, B.K. Data-driven optimal tracking control of discrete-time multi-agent systems with two-stage policy iteration algorithm. *Inf. Sci.* **2019**, *481*, 189–202. [CrossRef]
18. Zhang, H.; Jiang, H.; Luo, Y.; Xiao, G. Data-driven optimal consensus control for discrete-time multi-agent systems with unknown dynamics using reinforcement learning method. *IEEE Trans. Ind. Electron.* **2017**, *64*, 4091–4100. [CrossRef]
19. Abouheaf, M.I.; Lewis, F.L.; Vamvoudakis, K.G.; Haesaert, S.; Babuska, R. Multi-agent discrete-time graphical games and reinforcement learning solutions. *Automatica* **2014**, *50*, 3038–3053. [CrossRef]
20. Peng, Z.; Zhao, Y.; Hu, J.; Luo, R.; Ghosh, B.K.; Nguang, S.K. Input–output data-based output antisynchronization control of multiagent systems using reinforcement learning approach. *IEEE Trans. Ind. Inform.* **2021**, *17*, 7359–7367. [CrossRef]
21. Peng, Z.; Hu, J.; Ghosh, B.K. Data-driven containment control of discrete-time multi-agent systems via value iteration. *Sci. China Inf. Sci.* **2020**, *63*, 189205. [CrossRef]

22. Wen, G.; Chen, C.P.; Feng, J.; Zhou, N. Optimized multi-agent formation control based on an identifier-actor-critic reinforcement learning algorithm. *IEEE Trans. Fuzzy Syst.* **2018**, *26*, 2719–2731. [CrossRef]
23. Bai, W.; Li, T.; Long, Y.; Chen, C.P. Event-triggered multigradient recursive reinforcement learning tracking control for multiagent systems. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *34*, 366–379. [CrossRef]
24. Peng, Z.; Luo, R.; Hu, J.; Shi, K.; Ghosh, B.K. Distributed optimal tracking control of discrete-time multiagent systems via event-triggered reinforcement learning. *IEEE Trans. Circuits Syst.* **2022**, *69*, 3689–3700. [CrossRef] [PubMed]
25. Hu, J.; Chen, G.; Li, H.X. Distributed event-triggered tracking control of leader-follower multi-agent systems with communication delays. *Kybernetika* **2011**, *47*, 630–643. [CrossRef]
26. Eqtami, A.; Dimarogonas, D.V.; Kyriakopoulos, K.J. Event-triggered control for discrete-time systems. In Proceedings of the American Control Conference, Baltimore, MD, USA, 30 June–2 July 2010; pp. 4719–4724.
27. Chen, X.; Hao, F. Event-triggered average consensus control for discrete-time multi-agent systems. *IET Control Theory Appl.* **2012**, *6* , 2493–2498.
28. Jiang, Y.; Fan, J.; Chai, T.; Li, J.; Lewis, F.L. Data-driven flotation industrial process operational optimal control based on reinforcement learning. *IEEE Trans. Ind. Inform.* **2018**, *14*, 1974–1989. [CrossRef]
29. Watkins, C.J.C.H.; Dayan, P. Q-learning. *Mach. Learn.* **1992**, *8*, 279–292. [CrossRef]
30. Alsheikh, M.A.; Lin, S.; Niyato, D.; Tan, H.P. Machine learning in wireless sensor networks: Algorithms, strategies, and applications. *IEEE Commun. Surv. Tutor.* **2014**, *16*, 1996–2018. [CrossRef]
31. Vamvoudakis, K.G.; Modares, H.; Kiumarsi, B.; Lewis, F.L. Game theory-based control system algorithms with real-time reinforcement learning: How to solve multiplayer games online. *IEEE Control Syst.* **2017**, *37*, 33–52. [CrossRef]
32. Peng, Z.; Luo, R.; Hu, J. Optimal tracking control of nonlinear multiagent systems using internal reinforce Q-learning. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *33*, 4043–4055. https://doi.org/10.1109/TNNLS.2021.3055761.
33. Wang, D.; Liu, D.; Wei, Q.; Zhao, D.; Jin, N. Optimal control of unknown nonaffine nonlinear discrete-time systems based on adaptive dynamic programming. *Automatica* **2012**, *48*, 1825–1832. [CrossRef] [PubMed]
34. Peng, Z.; Hu, J.; Shi, K.; Luo, R.; Huang, R.; Ghosh, B.K.; Huang, J. A novel optimal bipartite consensus control scheme for unknown multi-agent systems via model-free reinforcement learning. *Appl. Math. Comput.* **2020**, *369*, 124821. [CrossRef]
35. Zhang, H.; Yue, D.; Dou, C.; Zhao, W.; Xie, X. Data-driven distributed optimal consensus control for unknown multiagent systems with input-delay. *IEEE Trans. Cybern.* **2019**, *49*, 2095–2105. [CrossRef]
36. Si, J.; Wang, Y.-T. Online learning control by association and reinforcement. *IEEE Trans. Neural Netw.* **2001**, *12*, 264–276. [CrossRef] [PubMed]