2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

Event-Triggered Reinforcement Learning Approach for Unknown Nonlinear Continuous-Time System

Xiangnan Zhong, Zhen Ni, Haibo He, Xin Xu, and Dongbin Zhao

Abstract— **This paper provides an adaptive event-triggered method using adaptive dynamic programming (ADP) for the nonlinear continuous-time system. Comparing to the traditional method with fixed sampling period, the event-triggered method samples the state only when an event is triggered and therefore the computational cost is reduced. We demonstrate the theoretical analysis on the stability of the event-triggered method, and integrate it with the ADP approach. The system dynamics are assumed unknown. The corresponding ADP algorithm is given and the neural network techniques are applied to implement this method. The simulation results verify the theoretical analysis and justify the efficiency of the proposed event-triggered technique using the ADP approach.**

I. Introduction

In literature, digital control are relying on the periodic transmitted data using the fixed sampling period. However, huge number of the transmitted data may cause subsequent tremendous computation, especially when the computation bandwidth or sensor power sources are constrained. In recent years, the event-triggered control method has been studied for its capability of computation efficiency [1], [2]. In the event-triggered control algorithm, the controller is only updated when an event is triggered, and thus the computation is significantly saved [3], [4], [5]. Currently, the event-triggered control methods are based on the accurate system function or model [6], [7]. In many cases, the complete knowledge of the system function is either infeasible or very difficult to obtain. Recently, neural-network-based event-triggered optimal control approaches were proposed and demonstrated with the promising performance in [8], [9], [10].

Adaptive dynamic programming (ADP) have been studied and adopted for the solution seeking for the Hamilton-Jacobi-Bellman (HJB) equation in recent years [11], [12], [13]. Extensive efforts and promising results have been achieved over the past decades, such as the special issue on feedback control provided well-known feedback control problems with new techniques of ADP [14]. Higher level exploration, like [15], [16], showed the deeper thinking for the future development on ADP community. In addition, ADP methods demonstrated the control capabilities in many real applications, like the power system stability/transient control in [17], [18], the looper system control in the iron and steel company in [19], [20], the engine torque and air-fuel control in [21] and among others [22], [23], [24]. Stability analysis of the ADP control on dynamic systems were provided under certain conditions in [25], [26], [27]. The performance index function and the control law were studied and demonstrated in [28], [29]. The robust controller with the ADP technique was also presented in [30]. More recently, a series of new ADP frameworks, namely the three-network ADP/goal representation adaptive dynamic programming (GrADP) were proposed and demonstrated in [31], [32], [33]. A novel tracking scheme was studied and demonstrated with stability analysis with this GrADP approach in [34], [35]. The maze navigation example was also tested and compared with this GrADP approach and many other reinforcement learning approaches in [36], [37].

In this paper, we integrate the event-triggered control technique into the ADP approach for the unknown nonlinear continuous-time system. We first study the stability analysis for the event-triggered method. The event-triggered controller is then implemented with the neural network techniques. That is, we use an action network to approximate the control law based on the event-triggered sample data (with event-triggered techniques), and use a critic network to evaluate the control performance with the value function. The pseudo-code for the event-triggered algorithm is provided and the weights updating rules are subsequently derived. The weights evolution in the learning process are provided to show the achieved learned/optimal policy. For comparative studies, we also provide the performance of both the traditional ADP approach and the proposed event-triggered ADP approach in the simulation studies. The theorem is verified with simulation results. During the simulation, the system function is assumed to be unknown and only the input/output data are measured.

The rest of this paper is organized as follows. In Section II, we provide the problem formulation of the event-triggered ADP approach on the continuous-time system. The stability analysis of the event-triggered control law is provided in Section III. Section IV first presents the integration of the event-triggered technique into the ADP approach. Then the neural networks, namely the adaptive critic networks, are introduced to implement this ADP scheme. In Section V,

X. Zhong, Z. Ni and H. He are with the Department of Electrical, Computer and Biomedical Engineering, University of Rhode Island, Kingston, RI 02881, USA. (email: xzhong@ele.uri.edu, ni@ele.uri.edu, he@ele.uri.edu).

X. Xu is with the College of Mechatronics and Automation, National University of Defense Technology, Changsha 410073, China (e-mail:xinxu@nudt.edu.cn)

D. Zhao is with the State Key Lab of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, P. R. China (e-mail: dongbin.zhao@ia.ac.cn)

This work was supported in part by the National Science Foundation (NSF) under grant CAREER ECCS 1053717, Army Research Office (ARO) under grant W911NF-12-1-0378, NSF-DFG Collaborative Research on "Autonomous Learning" (a supplement grant to CNS 1117314), and the Program for New Century Excellent Talents in Chinese University (NCET-10-0901).

978-1-4799-1484-5/14/$31.00 ©2014 IEEE

a single link robot arm is studied with two settings to demonstrate the control performance of the proposed method. The conclusion is provided in the section VI.

II. Problem Statement

Consider a nonlinear continuous-time system with the form

$$\dot{x}(t) = f(x(t)) + g(x(t))u(t) \tag{1}$$

where $x(t) \in R^n$ denotes the system state variable with the initial state $x(0) = x_0$ and $u(t) \in R^m$ is the control input. $f(x(t))$ and $g(x(t))$ are the unknown system functions. Assume that $f(x(t)) + g(x(t))u(t)$ is Lipschitz continuous on a set $\Omega \subseteq R^n$, and $f(0) = 0$, $g(0) = 0$. In order to save resources, this paper introduces a sampled-data system that is characterized by a monotonically increasing sequence of sampling instants $\{\delta_j\}_{j=0}^{\infty}$, where $\delta_j < \delta_{j+1}$ for $j = 0, 1, 2, \cdots, \infty$. The time δ_j denotes the jth consecutive sampling instant. The output of the sampled-data system is a sequence of the sampled states which can be described by

$$\hat{x}_j = x(\delta_j). \tag{2}$$

For simplicity, we assume that the sampled-data system has zero task delay. Define the gap function for $\forall t \in [\delta_j, \delta_{j+1})$ as

$$e_j(t) = \hat{x}_j - x(t) \tag{3}$$

which is the difference between the sampled state and the current state. It is obvious that at the beginning of the interval $[\delta_j, \delta_{j+1})$, the gap in (3) is equal to zero. After that, one expects the norm of the gap to increase. When the gap is larger than a threshold e_T, then the system state is again sampled by setting $\hat{x}_j = x(t)$, thereby forcing the gap to zero again.

We are interested in the state-feedback controller $\gamma(\hat{x}_j)$, which maps the sampled state onto a control vector. Assume that $\gamma(\hat{x}_j)$ is a Lipschitz continuous function. The obtained control sequence $\{\gamma(\hat{x}_j)\}_{j=0}^{\infty}$ becomes a continuous-time signal through a zero-order hold (ZOH). In particular, this control signal can be seen as a piecewise constant function and within any time interval $[\delta_j, \delta_{j+1})$, the controller is $u(t) = \gamma(\hat{x}_j)$, $j = 0, 1, 2, \cdots, \infty$.

Rewrite equation (3) as $\hat{x}_j = x(t) + e_j(t)$, so that the closed loop dynamics can be described as

$$\dot{x}(t) = f(x(t)) + g(x(t))\gamma(x(t) + e_j(t)), \ \ \forall t \in [\delta_j, \delta_{j+1}). \tag{4}$$

Similar to the traditional ADP problem, it is desired to find a controller $u(t)$ that minimizes the performance index given as

$$\begin{aligned} V(x_0) &= \int_0^{\infty} U(x(\tau), u(\tau))d\tau \\ &= \sum_{\bigcup_j [\delta_j, \delta_{j+1}) = [0, \infty)} \int_{\delta_j}^{\delta_{j+1}} U(x(\tau), \gamma(\hat{x}_j))d\tau \end{aligned} \tag{5}$$

where $U(x(\tau), \gamma(\hat{x}_j))$ is the utility function with $U(0, 0) = 0$. In this paper, the utility function is given by

$$U(x(t), \gamma(\hat{x}_j)) = x^T(t)Qx(t) + \gamma^T(\hat{x}_j)R\gamma(\hat{x}_j) \tag{6}$$

in which Q and R are symmetric and positive definite matrices with appropriate dimensions. Moreover, they can be described by

$$\begin{aligned} Q &= q \cdot q^T \\ R &= r \cdot r^T \end{aligned} \tag{7}$$

Definition 1: A law $u(t)$ is said to be an admissible control with respect to (5) on Ω, if $u(t)$ is continuous on Ω and can stabilize system (1) for all $x_0 \in \Omega$, $u(t) = 0$ if $x(t) = 0$, and $V(x_0)$ is finite, $\forall x(t) \in \Omega$.

Equation (5) can be expanded as follows

$$\begin{aligned} V(x_0) = &\sum_{\bigcup_j [\delta_j, \delta_{j+1}) = [0, \delta_1)} \int_{\delta_j}^{\delta_{j+1}} U(x(\tau), \gamma(\hat{x}_j))d\tau \\ &+ \sum_{\bigcup_j [\delta_j, \delta_{j+1}) = [\delta_1, \infty)} \int_{\delta_j}^{\delta_{j+1}} U(x(\tau), \gamma(\hat{x}_j))d\tau \\ = &\int_0^{\delta_1} U(x(\tau), \gamma(\hat{x}_j))d\tau + V(x(\delta_1)). \end{aligned} \tag{8}$$

After transformation, equation (8) becomes

$$\begin{aligned} &\lim_{\delta_1 \to 0} \left[\frac{V(x(\delta_1)) - V(x_0)}{\delta_1} \right] \\ &= -\lim_{\delta_1 \to 0} \frac{1}{\delta_1} \int_0^{\delta_1} [x^T(\tau)Qx(\tau) + \gamma^T(\hat{x}_j)R\gamma(\hat{x}_j)]d\tau. \end{aligned} \tag{9}$$

Then, we obtain the infinitesimal version of (5) as

$$\begin{aligned} V_x^T(f(x(t)) + g(x(t))\gamma(\hat{x}_j, t)) + x^T(t)Qx(t) \\ +\gamma^T(\hat{x}_j, t)R\gamma(\hat{x}_j, t) = 0 \end{aligned} \tag{10}$$

where $V_x = \frac{\partial V(x(t))}{\partial x(t)}$ is the partial derivative of the performance index with respect to the state and $\gamma(\hat{x}_j, t)$ is the continuous-time signal of the event-triggered control law $\gamma(\hat{x}_j)$. Given that $u(t) = \gamma(\hat{x}_j, t)$ is an admissible control law, if $V(x(t))$ satisfies (10) and $Q \geq 0$, $R \geq 0$, then $V(x(t))$ is a Lyapunov function for the system (1) with the control law $u(t) = \gamma(\hat{x}_j, t)$. Note that, in order to simplify the expression, we use $\gamma(\hat{x}_j)$ to represent $\gamma(\hat{x}_j, t)$ in the following presentation.

According to Bellman's optimality equation, the optimal performance index $V^*(x(t))$ satisfies

$$\begin{aligned} \min_{\gamma(\hat{x}_j)} [V_x^{*T}(f(x(t)) + g(x(t))\gamma(\hat{x}_j)) + x^T(t)Qx(t) \\ +\gamma^T(\hat{x}_j)R\gamma(\hat{x}_j)] = 0. \end{aligned} \tag{11}$$

Assume that the minimum on the left-hand side of the equation (11) exists and is unique. Therefore, the optimal control $\gamma^*(\hat{x}_j)$ satisfies the first-order necessary condition, which is given by the gradient of (10) with respect to $\gamma(\hat{x}_j)$. Note that, in the event-triggered method, the controller is only updated when an event is triggered. In other words, the controller is designed based on the event-triggered sampling state $\hat{x}_j$ rather than the real state $x(t)$. Hence, we have $g(x(t)) = g(\hat{x}_j)$ and $V_x = V_{\hat{x}_j}$, where $V_{\hat{x}_j} = \frac{\partial V(\hat{x}_j)}{\partial x(t)}$ is the partial derivative of the event-triggered performance index

with respect to the state. Therefore, we obtain the event-triggered optimal control as

$$u^*(t) = \gamma^*(\hat{x}_j) = -\frac{1}{2}R^{-1}g^T(\hat{x}_j)V^*_{\hat{x}_j}. \tag{12}$$

By substituting (12) into (10), we obtain the HJB equation under event-triggered method as follows

$$\begin{aligned} &V_x^{*T}f(x(t)) - \frac{1}{2}V_x^{*T}g(x(t))R^{-1}g^T(\hat{x}_j)V^*_{\hat{x}_j} \\ &+\frac{1}{4}V^{*T}_{\hat{x}_j}g(\hat{x}_j)R^{-1}g^T(\hat{x}_j)V^*_{\hat{x}_j} + x^T(t)Qx(t) = 0 \end{aligned} \tag{13}$$

with $V^*(0) = 0$.

In the next section, we will show the event-triggered control (12) is admissible and can stabilize the nonlinear continuous-time system (1).

III. Stability Analysis of the Event-Triggered Method

Assumption: The controller $\gamma(x)$ is Lipschitz continuous with respect to the gap,

$$\|\gamma(x(t)) - \gamma(\hat{x}_j)\| = \|\gamma(x(t)) - \gamma(x(t) + e_j(t))\| \le L\|e_j(t)\| \tag{14}$$

where L is a positive real constant.

The stability analysis of the event-triggered controller is provided as follows.

Theorem 1: Consider the nonlinear continuous-time system (1). For $\forall t \in [\delta_j, \delta_{j+1})$, the control law is given by (12) and assume $V^*(x(t))$ is the solution of the event-triggered HJB equation (13). If the triggered condition is defined as follows

$$\|e_j(t)\|^2 \le \|e_T\|^2 = \frac{(1-\alpha^2)}{L^2\|r\|^2}\underline{\lambda}(Q)\|x(t)\|^2 + \frac{1}{L^2}\|\gamma^*(\hat{x}_j)\|^2 \tag{15}$$

where $\underline{\lambda}(Q)$ is the minimal eigenvalue of Q, $\alpha \in (0,1)$ is the designed parameter, and e_T is the threshold of the gap between the sampled and the real state, then the following conditions hold.

(1) The event-triggered control law (12) is an admissible control.

(2) The event-triggered control law (12) can asymptotically stabilize the nonlinear system (1).

Proof: Let us start with the admissibility part. From equation (12), we know when the state $\hat{x}_j = 0$, then $g(\hat{x}_j) = 0$ and hence $\gamma^*(\hat{x}_j) = 0$. The continuity assumption on $f(x(t)) + g(x(t))u(t)$ and $\gamma^*(\hat{x}_j)$ implies that $\gamma^*(\hat{x}_j)$ is continuous and the system (1) cannot jump to infinity by any one step of finite control. Moreover because $f(0) = 0$, $g(0) = 0$, when the system state $x(t)$ reaches the equilibrium state, $\gamma^*(\hat{x}_j)$ becomes zero and the state is kept at zero. Therefore, according to Definition (1), we obtain event-triggered control law $\gamma^*(\hat{x}_j)$ is an admissible control which proves the part (1).

Now we will show that $\gamma^*(\hat{x}_j)$ can asymptotically stabilize the nonlinear continuous-time system (1). Let $\gamma^*(\hat{x}_j(t))$ and $V^*(x(t))$ be the optimal event-triggered control law and the optimal performance index obtained in equation (12) and (13), respectively. From equation (5), we know $V^*(x(t))$ is a positive definite function, namely, $V^*(x(t)) > 0$ for any $x(t) \neq 0$ and $V^*(x(t)) = 0$ when $x(t) = 0$. Hence, $V^*(x(t))$ can be seen as a Lyapunov function.

With the event-triggered controller, the derivative of $V^*(x(t))$ along the system trajectory can be obtained as,

$$\begin{aligned} \dot{V}^*(x(t)) &= \left(\frac{\partial V^*(x(t))}{\partial x(t)}\right)^T \cdot \dot{x} \\ &= V_x^{*T}f(x(t)) + V_x^{*T}g(x(t))\gamma^*(\hat{x}_j) \end{aligned} \tag{16}$$

Here, we recall the control law and the HJB equation in the traditional ADP method as

$$u^*(t) = -\frac{1}{2}R^{-1}g^T(x)V^*_{x(t)} \equiv \gamma^*(x(t)) \tag{17}$$

and

$$\begin{aligned} V_x^{*T}f(x(t)) - \frac{1}{4}V_x^{*T}g(x(t))R^{-1}g^T(x(t))V_x^* \\ +x^T(t)Qx(t) = 0. \end{aligned} \tag{18}$$

Therefore,

$$g^T(x(t))V_x^* = -2R\gamma^*(x(t)) \tag{19}$$

$$\begin{aligned} V_x^{*T}f(x(t)) =& \frac{1}{4}V_x^{*T}g(x(t))R^{-1}g^T(x(t))V_x^* \\ &+ x^T(t)Qx(t) \end{aligned} \tag{20}$$

Substitute (19) and (20) into (16), we have

$$\begin{aligned} \dot{V}^*(x(t)) =& \frac{1}{4}V_x^{*T}g(x(t))R^{-1}g^T(x(t))V_x^* + x^T(t)Qx(t) \\ &+ V_x^{*T}g(x(t))\gamma^*(\hat{x}_j) \\ =& \gamma^{*T}(x(t))R\gamma^*(x(t)) - x^T(t)Qx(t) \\ &- 2\gamma^{*T}(x(t))R\gamma^*(\hat{x}_j) \end{aligned} \tag{21}$$

Because $R = r \cdot r^T$, we obtain

$$\begin{aligned} &\gamma^{*T}(x(t))R\gamma^*(x(t)) - 2\gamma^{*T}(x(t))R\gamma^*(\hat{x}_j) \\ =&\|r^T\gamma^*(x(t)) - r^T\gamma^*(\hat{x}_j)\|^2 - \|r^T\gamma^*(\hat{x}_j)\|^2. \end{aligned} \tag{22}$$

By substituting (22) into (21) and using the Lipschitz condition from Assumption 1, we have

$$\begin{aligned} \dot{V}^*(x(t)) =& \|r^T\gamma^*(x(t)) - r^T\gamma^*(\hat{x}_j)\|^2 - \|r^T\gamma^*(\hat{x}_j)\|^2 \\ &- x^T(t)Qx(t) \\ \le& L^2\|r\|^2\|e_j(t)\|^2 - \|r^T\gamma^*(\hat{x}_j)\|^2 - x^T(t)Qx(t) \\ \le& L^2\|r\|^2\|e_j(t)\|^2 - \|r^T\gamma^*(\hat{x}_j)\|^2 - \underline{\lambda}(Q)\|x(t)\|^2 \\ =& -\alpha^2\underline{\lambda}(Q)\|x(t)\|^2 + \Big[-(1-\alpha^2)\underline{\lambda}(Q)\|x(t)\|^2 \\ &+ L^2\|r\|^2\|e_j(t)\|^2 - \|r^T\gamma^*(\hat{x}_j)\|^2\Big] \end{aligned} \tag{23}$$

since $-x^T(t)Qx(t) \le -\underline{\lambda}(Q)\|x(t)\|^2$, where $\underline{\lambda}(Q)$ is the minimal eigenvalue of Q.

Based on the condition (15), we know that the last three terms in (23) is guaranteed negative. Therefore, (23) can be modified as follows

$$\begin{aligned}\dot{V}^*(x(t)) \leq & (1-\alpha^2)\underline{\lambda}(Q)\|x(t)\|^2 + \|r\|^2\|\gamma^*(\hat{x}_j)\|^2 \\ & - \|r^T\gamma^*(\hat{x}_j)\|^2 - \underline{\lambda}(Q)\|x(t)\|^2 \\ = & -\alpha^2\underline{\lambda}(Q)\|x(t)\|^2 \\ < & 0\end{aligned} \tag{24}$$

for any $x(t) \neq 0$. Thus, $u^*(t) = \gamma^*(\hat{x}_j)$ can asymptotically stabilize the nonlinear continuous-time system (1). The conclusion holds. ■

It can be seen that the controller is guaranteed stable (under certain conditions) with the event-triggered sample data. In the next section, we are applying the neural network methods to implement the event-triggered ADP approach.

IV. Event-Triggered Controller Design by Neural Network Techniques

In this section, an ADP approach is provided to solve the event-triggered HJB equation (13) and approximate the optimal event-triggered control law (12). The neural network techniques are employed to implement this approach. Two subsections are included. The first one shows the even-triggered online learning ADP algorithm for nonlinear continuous-time system. The neural network implementation is presented in the second subsection.

A. ADP Approach to Approximate the Event-Triggered Control Law

Set the initial triggered state as $\hat{x}_0 = x_0$. Note that if we use equation (12) to calculate the event-triggered control law, the system function $g(\hat{x}_j)$ is required which is unknown in this paper. Hence, we provide a method to approximate the control updating equation (12). The algorithm can be described as Algorithm 1.

Algorithm 1 Event-triggered Algorithm with unknown system dynamics.

Set $i = 0$, $j = 0$, $\hat{x}_0 = x_0$, $x(t) = x_0$
Approximate $\gamma(\hat{x}_j) = \arg\min_{\gamma(\hat{x}_j)}\{V(x_0)\}$
for all $i < N_{run}$ **do**
  Policy evaluation:
  $V(x(t)) = \min_{\gamma(\hat{x}_j)} \int_0^\infty U(x(\tau), \gamma(\hat{x}_j))d\tau$
  if $\hat{x}_j - x(t) = e_j(t) > e_T$ **then**
    Set $j = j + 1$, $\hat{x}_j = x(t)$
    Update $\gamma(\hat{x}_j) = \arg\min_{\gamma(\hat{x}_j)}\{V(\hat{x}_j)\}$
  end if
  Update state $\dot{x}(t) = f(x(t)) + g(x(t))\gamma(\hat{x}_j)$
  Set $i = i + 1$
end for

From Algorithm 1, we know that by estimation of the control law, no system information is required during the learning process. In the next subsection, we will provide the approximation process by neural network techniques in detail.

B. Neural-Network-based Implementation

The neural networks are employed in this subsection to approximate the event-triggered control law. The architecture of this event-triggered method is shown in Fig.1. A critic network and an action network are built to approximate the performance index and the control law of the event-triggered method, respectively. We can observe that a sampled-data system is used during this process with the sampling instants $\{\delta_j\}_{j=0}^\infty$. As we provided above, $\{\delta_j\}_{j=0}^\infty$ are based on the gap $(e_j(t))$ which is the difference between the current and the sampled state. When $e_j(t)$ is larger than the threshold e_T, the system state is sampled by $\hat{x}_j = x(\delta_j)$, and the action network is updated based on the event-triggered sample state. Then through the ZOH, the control law sequence is transformed into a continuous-time control signal. Assume that the sampling period for the discretization is $\triangle t$. We set both the critic and the action network used in this paper be the three-layer networks. In the following part, we will provide the online learning rules for both networks.

1) Critic Network: The critic network is used to approximate the performance index $V(x(t))$ which can be formulated as

$$V(x(t)) = \omega_{c2}^T(t)\Phi(h(t)) \tag{25}$$

where $\omega_{c2}^T(t)$ is the weight matrix between the hidden and the output layer of the critic network and $h(t) = \omega_{c1}^T[x^T(t), \gamma^T(\hat{x}_j)]$, to which ω_{c1} denotes the weight matrix between the hidden and the input layer. Note that ω_{c1} is randomly chosen as initial and is kept constantly during the implementation process in this paper.

$\Phi(x)$ is a sigmoid function that can be described as

$$\Phi(x) = \frac{1 - e^{-x}}{1 + e^{-x}}. \tag{26}$$

The purpose of the sigmoid function is to constrain the output into $[-1, 1]$. Here the sigmoid function is applied on the hidden to output nodes.

Define the error function for the critic network by

$$e_c(t) = V(x(t)) - [V(x(t - \triangle t)) - U(x(t), \gamma^T(\hat{x}_j)] \tag{27}$$

where $\triangle t$ is the sampling period during discretization.

Therefore, to update the weight matrix is to minimize the following objective function

$$E_c(t) = \frac{1}{2}e_c^2(t). \tag{28}$$

Hence, we obtain the critic network weights adjustments for the hidden to the output layer

$$\omega_{c2}(t + \triangle t) = \omega_{c2}(t) - \beta_c\left(\frac{\partial E_c(t)}{\partial \omega_{c2}(t)}\right) \tag{29}$$

where $\beta_c > 0$ is the learning rate of the critic network. According to the chain-backpropagation rules, we derive the tuning formula as

$$\frac{\partial E_c(t)}{\partial \omega_{c2}(t)} = \frac{\partial E_c(t)}{\partial V(x(t))}\frac{\partial V(x(t))}{\partial \omega_{c2}(t)} \tag{30}$$

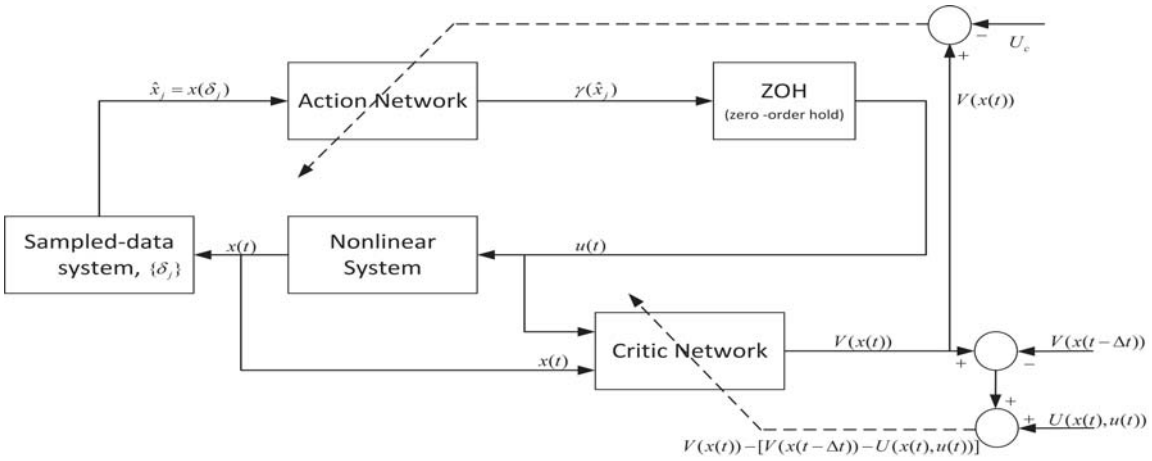

Fig. 1. Architecture of the event-triggered method based on the ADP approach.

2) Action Network: The purpose of the action network is to estimate the optimal event-triggered control law. As we discussed, the action network is only updated when an event is triggered. Therefore, the estimated control law can be formulated as

$$\gamma^T(\hat{x}_j) = \Phi(\omega_{a2}^T(\delta_j)g(\delta_j)) \tag{31}$$

$$g(\delta_j) = \Phi(\omega_{a1}^T(\delta_j)\hat{x}_j) \tag{32}$$

where $\omega_{a1}(\delta_j)$ and $\omega_{a2}(\delta_j)$ are the weight matrices of the input-to-hidden and the hidden-to-output layer at the sampled time δ_j, respectively. Sigmoid function is applied on both hidden and the output side. $\hat{x}_j$ is the sampled state and is also the input of the action network. The same as above, we fix the input-to-hidden layer weight matrix $\omega_{a1}(\delta_j)$ which is chosen initially at random. Therefore, only the weight matrix $\omega_{a2}(\delta_j)$ between the hidden and the output layer is needed to be updated.

We know the objective for the action network is to minimize the total future cost, hence we define the error function here by

$$e_a(\delta_j) = V(\hat{x}_j) - U_c \tag{33}$$

where U_c is the ultimate utility function. The value of U_c is critical in ADP design and it could be variant in different application. In this paper, we choose $U_c = 0$.

The objective function of the action, therefore, can be written as

$$E_a(\delta_j) = \frac{1}{2}e_a^2(\delta_j) \tag{34}$$

The gradient descent method is also applied to minimize the approximation error (34) as

$$\omega_{a2}(\delta_{j+1}) = \omega_{a2}(\delta_j) - \beta_a\left(\frac{\partial E_a(\delta_j)}{\partial \omega_{a2}(\delta_j)}\right) \tag{35}$$

where $\beta_a > 0$ is the learning rate of the action network. From the chain backpropagation rule, we obtain

$$\frac{\partial E_a(\delta_j)}{\partial \omega_{a2}(\delta_j)} = \frac{\partial E_a(\delta_j)}{\partial V(\hat{x}_j)}\frac{\partial V(\hat{x}_j)}{\partial \gamma^T(\hat{x}_j)}\frac{\partial \gamma^T(\hat{x}_j)}{\partial \omega_{a2}(\delta_j)} \tag{36}$$

V. Simulation Results

Consider a single link robot arm with the following dynamic function

$$\ddot{\theta}(t) = -\frac{MgH}{G}\sin(\theta(t)) - \frac{D}{G}\dot{\theta}(t) + \frac{1}{G}u(t) \tag{37}$$

where $\theta(t)$ is the angle position of robot arm, and $u(t)$ is the control input. Moreover, M is the mass of the payload, G is the moment of inertia, g is the acceleration of gravity, H is the length of the arm and D is the viscous friction, where g, H, D are the system parameters and M, G are the design parameters. Set the values of the system parameters as $g = 9.81$, $D = 2$, and $L = 0.5$, and the design parameters M and G are alterable. Assuming $x_1(t) = \theta(t)$ and $x_2(t) = \dot{\theta}(t)$, the dynamic function (37) can be rewritten by

$$\begin{cases} \dot{x}_1(t) = x_2(t) \\ \dot{x}_2(t) = -\dfrac{2}{G}x_2(t) + \dfrac{1}{G}u(t) - \dfrac{4.905M\sin(x_1(t))}{G}. \end{cases} \tag{38}$$

We use the event-triggered method proposed in this paper to solve the problem. Choose the threshold according to condition (15) with $L = 3$, $\alpha = 0.95$. Set Q, R and r are the identity matrices with appropriate dimensions. Therefore, the threshold is

$$\begin{aligned} \|e_T\|^2 &= \frac{(1-\alpha^2)}{L^2\|r\|^2}\underline{\lambda}(Q)\|x(t)\|^2 + \frac{1}{L^2}\|\gamma(\hat{x}_j)\|^2 \\ &= \frac{1-0.95^2}{9}\|x(t)\|^2 + \frac{1}{9}\|\gamma(\hat{x}_j(t))\|^2. \end{aligned} \tag{39}$$

When the gap $e_j(t) = \hat{x}_j - x(t)$ satisfies the condition $\|e_j(t)\|^2 > \|e_T\|^2$, then the system state is again sampled by setting $\hat{x}_j = x(t)$.

Two three-layer neural networks are built as the critic and the action network. The neuron structures of the critic and the action network are $3-8-1$ and $2-6-1$, respectively. Set the learning rates of both networks as $\beta_c = \beta_a = 0.01$, and the sampling period for discretization as $\triangle t = 0.03s$. The initial weights of both networks are chosen randomly within

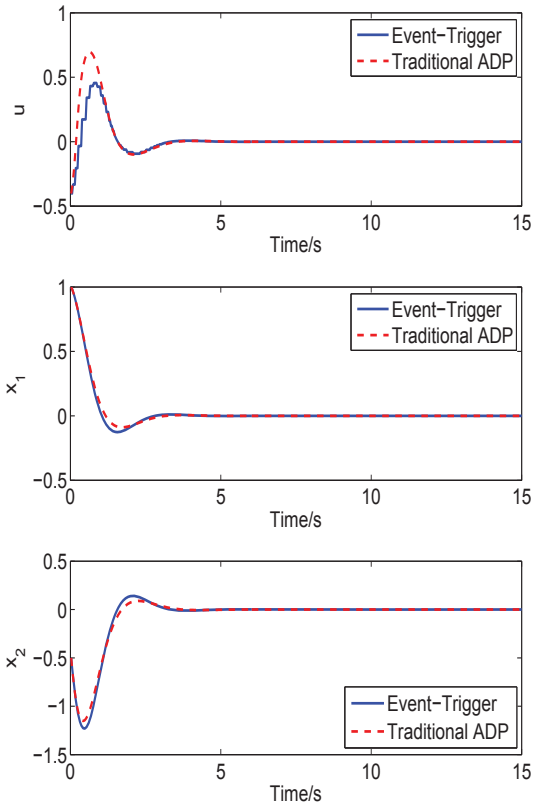

Fig. 2. Comparisons of system responses by the event-triggered and the traditional ADP method with $M = 1$ and $G = 1$.

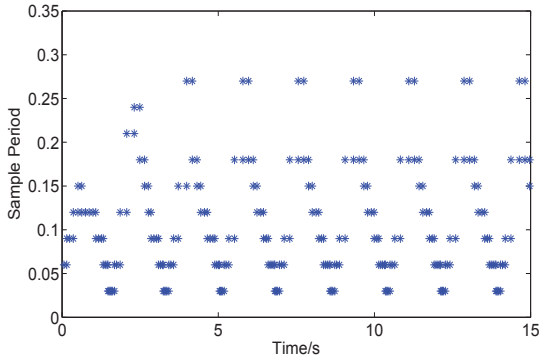

Fig. 3. Inter-event instants during the learning process with $M = 1$ and $G = 1$.

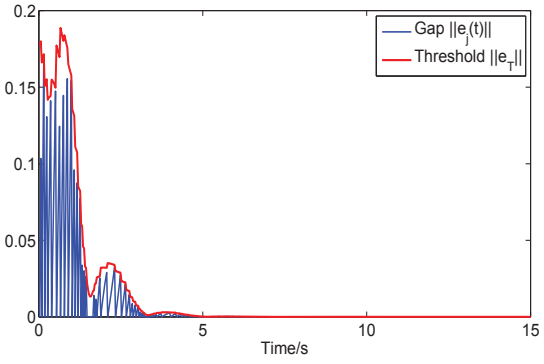

Fig. 4. Response of the gap $\|e_j(t)\|$ and the threshold $\|e_T\|$ with $M = 1$ and $G = 1$.

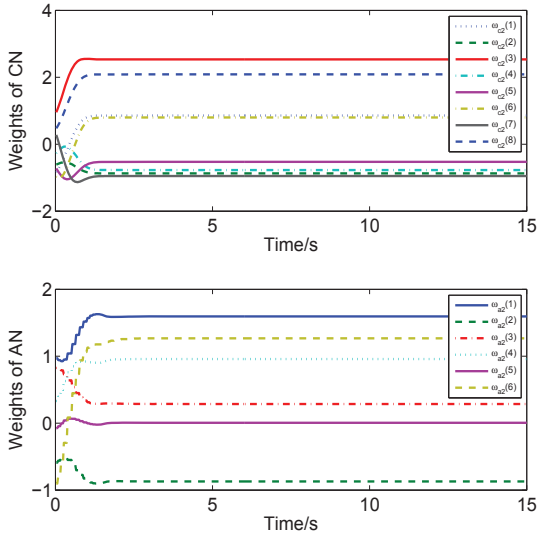

Fig. 5. Learning weights of the critic and the action network from the hidden to the output layer with $M = 1$ and $G = 1$.

$[-1, 1]$. The initial state is set to $x_0 = [1, -0.5]$. The input of the action network is the sampled state.

In the first case, we set the design parameters as $M = 1$, $G = 1$. By employing the event-triggered method proposed in this paper, we obtain the system responses in Fig.2. Note that, in order to demonstrate the performance of our method, we also conduct this example under the traditional ADP method with the same initial weights which is also presented in Fig.2. From the comparison, we know that the event-triggered control law keeps the same at period $[\delta_j, \delta_{j+1})$ and is only updated when an event is triggered. The control law evolution and the state trajectories of the event-triggered method are very close to those of the traditional ADP method. This means efficiently reducing the sampled times does not influence the system performance. The sampling period during the event-triggered learning process is provided in Fig.3 which shows that the sampling period is up to $0.27s$. The relationship between the gap $\|e_j(t)\|$ and the threshold $\|e_T\|$ is shown in Fig.4. The learning weights of the critic and the action network from the hidden to the output layer is provided in Fig.5. We know the weights converge after $3s$. In particular, comparing the event-triggered and the traditional ADP method, the event-triggered controller uses 161 samples of the state while the traditional ADP controller uses 500 samples, which means the even-triggered method improved the learning process.

In the second case, we conduct the example with the design parameters $M = 5$, $G = 5$. The comparison of the system responses by the event-triggered and the traditional ADP method with the same initial weights is presented in Fig.6. We can observe that the event-triggered method also works with the high design parameters. The sampling period during the learning process of the event-triggered method is provided in Fig.7. We know the sampling period is up to $0.39s$ in this case. The relationship between the gap $\|e_j(t)\|$

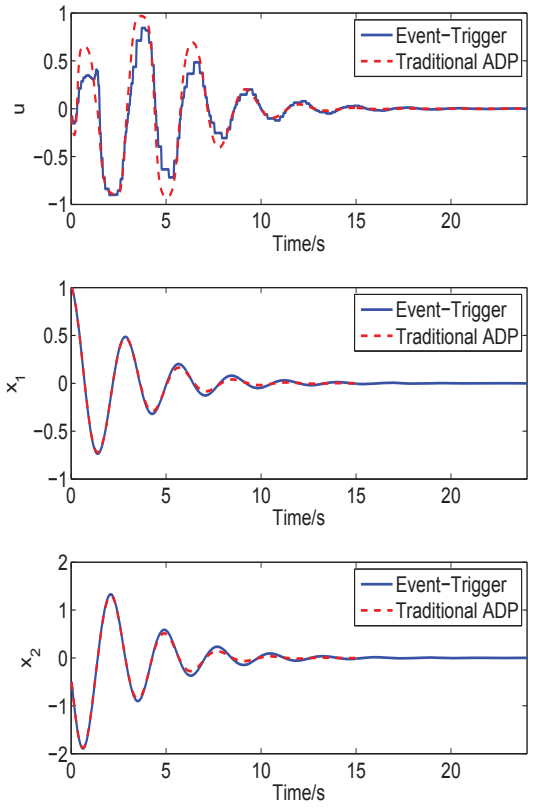

Fig. 6. Comparisons of system responses by the event-triggered and the traditional ADP method with $M = 5$ and $G = 5$.

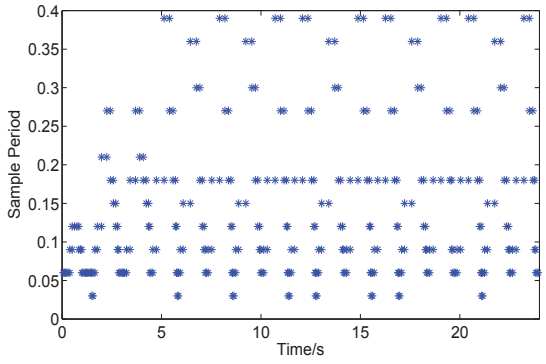

Fig. 7. Inter-event instants during the learning process with $M = 5$ and $G = 5$.

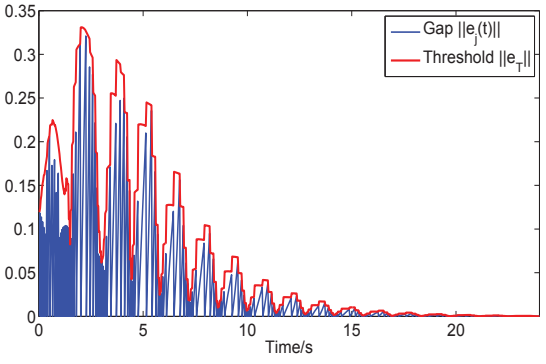

Fig. 8. Response of the gap $\|e_j(t)\|$ and the threshold $\|e_T\|$ with $M = 5$ and $G = 5$.

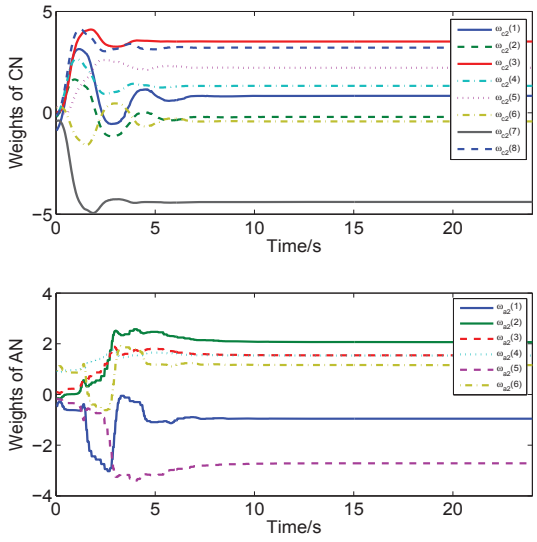

Fig. 9. Learning weights of the critic and the action network from the hidden to the output layer with $M = 5$ and $G = 5$.

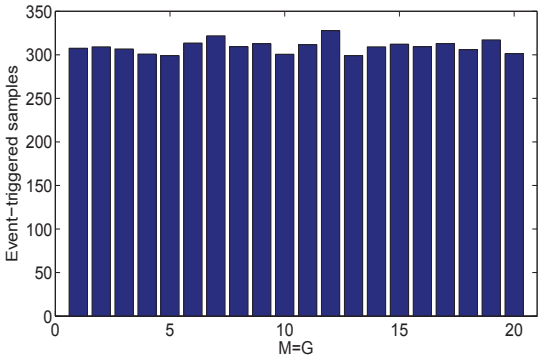

Fig. 10. The average number of the samples used by the event-triggered controller for each parameters pair within $[1, 20]$.

and the threshold $\|e_T\|$ is shown in Fig.8. Moreover, the learning weights of the critic and the action network from the hidden to the output layer is provided in Fig.9. From the simulation, we obtain that the event-triggered controller uses 291 samples of the state while the traditional ADP controller uses 800 samples. This means our method makes an great improvement in this case comparing to to the traditional ADP method, which indicates that our method is effectiveness.

Additionally, without loss of generality, we choose the values of the design parameters as $M = 1, 2, \cdots, 20$ and $G = 1, 2, \cdots, 20$. For each pair of the design parameters, we conduct the simulation based on the proposed method for 100 times. The sampling period for discretization is set as $\triangle t = 0.03s$ and each simulation lasts $25s$. This means the traditional ADP controller will use 800 samples to stabilize the system. However, from Fig.10, we know the average number of the samples used by the event-triggered controller for each parameters pair is around 300, which is significantly less than the samples used by the traditional ADP controller. This indicates that our method is effective.

VI. Conclusion

We designed an event-triggered controller for nonlinear continuous-time system using ADP approach. The system function was assumed to be unknown. The controller was updated only based on the triggered state. A zero-order hold was used to transform the control sequence into a continuous-time signal. The threshold for triggering an event was discussed and the stability of this event-triggered controller was analyzed. Neural network techniques were used to approximate the performance index and the controller in event-triggered method, respectively. The stability of the designed controller is analyzed in this paper. The simulation results demonstrate the effectiveness of the designed controller and also verify the theoretical analysis.

References

[1] W. Heemels, M. Donkers, and A. Teel, "Periodic event-triggered control for linear systems," *Automatic Control, IEEE Transactions on*, vol. 58, no. 4, pp. 847–861, 2013.

[2] P. Tallapragada and N. Chopra, "On event triggered tracking for nonlinear systems," *Automatic Control, IEEE Transactions on*, vol. 58, no. 9, pp. 2343–2348, 2013.

[3] M. Lemmon, "Event-triggered feedback in control, estimation, and optimization," in *Networked Control Systems*, pp. 293–358, Springer, 2010.

[4] E. Garcia and P. J. Antsaklis, "Model-based event-triggered control with time-varying network delays," in *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*, pp. 1650–1655, IEEE, 2011.

[5] W. Heemels, K. H. Johansson, and P. Tabuada, "An introduction to event-triggered and self-triggered control," in *Decision and Control (CDC), 2012 IEEE 51st Annual Conference on*, pp. 3270–3285, IEEE, 2012.

[6] A. Eqtami, D. V. Dimarogonas, and K. J. Kyriakopoulos, "Event-triggered control for discrete-time systems," in *American Control Conference (ACC), 2010*, pp. 4719–4724, IEEE, 2010.

[7] W. Heemels and M. Donkers, "Model-based periodic event-triggered control for linear systems," *Automatica*, 2013.

[8] A. Sahoo, H. Xu, and S. Jagannathan, "Neural network-based adaptive event-triggered control of affine nonlinear discrete time systems with unknown internal dynamics," in *American Control Conference (ACC), 2013*, pp. 6418–6423, IEEE, 2013.

[9] A. Sahoo, H. Xu, and S. Jagannathan, "Neural network-based adaptive event-triggered control of nonlinear continuous-time systems," in *Intelligent Control (ISIC), 2013 IEEE International Symposium on*, pp. 35–40, IEEE, 2013.

[10] D. Tolic, R. Fierro, and S. Ferrari, "Optimal self-triggering for nonlinear systems via approximate dynamic programming," in *Control Applications (CCA), 2012 IEEE International Conference on*, pp. 879–884, IEEE, 2012.

[11] J. Si, A. G. Barto, W. B. Powell, and D. C. Wunsch, eds., *Handbook of Learning and Approximate Dynamic Programming*. IEEE Press and John Wiley & Sons, 2004.

[12] F. Lewis and D. Liu, eds., *Reinforcement Learning and Approximate Dynamic Programming for Feedback Control*. Wiley-IEEE Press, 2013.

[13] H. Zhang, D. Liu, Y. Luo, and D. Wang, *Adaptive Dynamic Programming for Control: Algorithms and Stability (Communications and Control Engineering)*. Springer, 2013.

[14] F. L. Lewis, D. Liu, and G. G. Lendaris, "Special issue on adaptive dynamic programming and reinforcement learning in feedback control," *IEEE Transactions on System, Man and Cybernetics, Part B*, vol. 38, no. 4, pp. 896–897, 2008.

[15] P. J. Werbos, "Adp: The key direction for future research in intelligent control and understanding brain intelligence," *IEEE Transactions on Systems Man and Cybernetics Part B-Cybernetics*, vol. 38, no. 4, pp. 898–900, 2008.

[16] G. G. Lendaris, "Higher level application of adp: A next phase for the control field?," *IEEE Transactions on System, Man and Cybernetics, Part B*, vol. 38, no. 4, pp. 901–912, 2008.

[17] W. Qiao, G. Venayagamoorthy, and R. Harley, "DHP-based wide-area coordinating control of a power system with a large wind farm and multiple FACTS devices," in *Proc. IEEE Int. Conf. Neural Netw.*, pp. 2093–2098, 2007.

[18] S. Ray, G. K. Venayagamoorthy, B. Chaudhuri, and R. Majumder, "Comparison of adaptive critics and classical approaches based wide area controllers for a power system," *IEEE Trans. on Syst. Man, Cybern., Part B*, vol. 38, no. 4, pp. 1002–1007, 2008.

[19] J. Fu, H. He, and X. Zhou, "Adaptive learning and control for mimo system based on adaptive dynamic programming," *IEEE Trans. Neural Networks*, vol. 22, no. 7, pp. 1133–1148, 2011.

[20] J. Fu, H. He, and Z. Ni, "Adaptive Dynamic Programming with Balanced Weights Seeking Strategy," in *IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL), IEEE Symposium Series on Computational Intelligence (SSCI)*, (Paris, France), 2011.

[21] D. Liu, H. Javaherian, O. Kovalenko, and T. Huang, "Adaptive critic learning techniques for engine torque and air-fuel ratio control," *IEEE Transactions on Systems Man and Cybernetics Part B-Cybernetics*, vol. 38, no. 4, pp. 988–993, 2008.

[22] D. Liu, Y. Zhang, and H. G. Zhang, "A self-learning call admission control scheme for cdma cellular networks," *IEEE Transactions on Neural Networks*, vol. 16, no. 5, pp. 1219–1228, 2005.

[23] Y. Tang, H. He, Z. Ni, J. Wen, and X. Sui, "Reactive power control of grid-connected wind farm based on adaptive dynamic programming," *Neurocomputing*, vol. 125, pp. 125–133, Feb. 2014.

[24] C. Lu, J. Si, and X. Xie, "Direct heuristic dynamic programming for damping oscillations in a large power system," *IEEE Trans. Sys. Man Cyber. Part B*, vol. 38, no. 4, pp. 1008–1013, 2008.

[25] A. Al-Tamimi, F. L. Lewis, and M. Abu-Khalaf, "Discrete-time nonlinear hjb solution using approximate dynamic programming: Convergence proof," *IEEE Transactions on System, Man and Cybernetics, Part B*, vol. 38, no. 4, pp. 943–949, 2008.

[26] D. Liu and Q. Wei, "Finite-approximation-error-based optimal control approach for discrete-time nonlinear systems," *IEEE Transactions on Cybernetics*, vol. 43, no. 2, pp. 779–789, 2013.

[27] P. He and S. Jagannathan, "Reinforcement learning neural-network-based controller for nonlinear discrete-time systems with input constraints," *IEEE Transactions on Systems Man and Cybernetics Part B-Cybernetics*, vol. 37, no. 2, pp. 425–436, 2007.

[28] D. Liu and Q. Wei, "Policy iteration adaptive dynamic programming algorithm for discrete-time nonlinear systems," *IEEE Trans. on Neural Networks and Learning Systems*, in press.

[29] D. Liu, D. Wang, and H. Li, "Decentralized stabilization for a class of continuous-time nonlinear interconnected systems using online learning optimal control approach," *IEEE Trans. on Neural Networks and Learning Systems*, in press.

[30] X. Zhong, H. He, and D. V. Prokhorov, "Robust controller design of continuous-time nonlinear system using neural network," in *The 2013 International Joint Conference on Neural Networks (IJCNN)*, Aug. 2013.

[31] H. He, *Self-Adaptive Systems for Machine Intelligence*. Wiley, 2011.

[32] H. He, Z. Ni, and J. Fu, "A three-network architecture for on-line learning and optimization based on adaptive dynamic programming," *Neurocomputing*, vol. 78, no. 1, pp. 3–13, 2012.

[33] H. He, Z. Ni, and D. Zhao, *Reinforcement Learning and Approximate Dynamic Programming for Feedback Control*, ch. Learning and Optimization in Hierarchical Adaptive Critic Design, pp. 78–95. Wiley-IEEE Press, 2013.

[34] Z. Ni, H. He, and J. Wen, "Adaptive learning in tracking control based on the dual critic network design," *IEEE Trans. on Neural Networks and Learning Systems*, vol. 6, no. 24, pp. 913–928, 2013.

[35] Z. Ni, X. Fang, H. He, D. Zhao, and X. Xu, "Real-time tracking control on adaptive critic design with uniformly ultimately bounded condition," in *IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL'13), IEEE Symposium Series on Computational Intelligence (SSCI)*, Apr. 2013.

[36] Z. Ni, H. He, J. Wen, and X. Xu, "Goal representation heuristic dynamic programming on maze navigation," *IEEE Trans. on Neural Networks and Learning Systems*, vol. 24, no. 12, pp. 2038–2050, 2013.

[37] Z. Ni and H. He, "Heuristic dynamic programming with internal goal representation," *Soft Computing*, vol. 17, pp. 2101–2108, 2013.