Event-triggered constrained control using explainable global dual heuristic programming for nonlinear discrete-time systems

Sun, Bo; van Kampen, Erik Jan

DOI
10.1016/j.neucom.2021.10.046

Publication date
2022

Document Version
Final published version

Published in
Neurocomputing

Citation (APA)
Sun, B., & van Kampen, E. J. (2022). Event-triggered constrained control using explainable global dual heuristic programming for nonlinear discrete-time systems. *Neurocomputing*, *468*, 452-463. https://doi.org/10.1016/j.neucom.2021.10.046

Neurocomputing 468 (2022) 452–463

Contents lists available at ScienceDirect

Neurocomputing

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

Event-triggered constrained control using explainable global dual heuristic programming for nonlinear discrete-time systems

Bo Sun *, Erik-Jan van Kampen

Department of Control and Operations, Delft University of Technology, Delft 2629HS, The Netherlands

ARTICLE INFO

Article history:
Received 3 April 2021
Revised 2 October 2021
Accepted 19 October 2021
Available online 22 October 2021
Communicated by Zidong Wang

Keywords:
Event-triggered control
Global dual heuristic programming
Asymmetric input constraints
Explainable artificial intelligence
Adaptive dynamic programming

ABSTRACT

This paper develops an event-triggered optimal control method that can deal with asymmetric input constraints for nonlinear discrete-time systems. The implementation is based on an explainable global dual heuristic programming (XGDHP) technique. Different from traditional GDHP, the required derivatives of cost function in the proposed method are computed by explicit analytical calculations, which makes XGDHP more explainable. Besides, the challenge caused by the input constraints is overcome by the combination of a piece-wise utility function and a bounding layer of the actor network. Furthermore, an event-triggered mechanism is introduced to decrease the amount of computation, and the stability analysis is provided with fewer assumptions compared to most existing studies that investigate event-triggered discrete-time control using adaptive dynamic programming. Two simulation studies are carried out to demonstrate the applicability of the constructed approach. The results present that the developed event-triggered XGDHP algorithm can substantially save the computational load, while maintain comparable performance with the time-based approach.

1. Introduction

Optimality is one of the most significant properties of a control system. The optimal control problem can be solved using the Hamilton–Jacobi-Bellman (HJB) equation. However, until now, there is no effective way to analytically solve the HJB equation for nonlinear systems [1,2]. Nevertheless, adaptive dynamic programming (ADP) offers a promising tool to attain satisfying numerical solutions by incorporating artificial neural networks (ANNs), which has been applied to a wide range of nonlinear industrial applications [3–7]. As a branch of reinforcement learning (RL), ADP approximately addresses optimal control problems by iterations between policy improvement and policy evaluation [8,9]. When dealing with the discrete-time (DT) optimal control problem, obtaining the current control policy usually relies on the control performance at the next time step [7,9]. This bootstrapping property [9] can be addressed by the actor-critic scheme using two separate ANNs that respectively improves and evaluates the policy.

When multiple ANNs are involved, ADP is often called adaptive critic design (ACD) [7]. Based on the information utilized by the critic network, ACDs can be categorized into heuristic dynamic programming (HDP), dual HDP (DHP), and global DHP (GDHP) [4,7]. GDHP combines the information of cost function and its derivatives, and recently has attained much attention [10,11,4,12]. The most common architecture of GDHP is the straightforward form that approximates the cost function and its derivatives simultaneously [10,11]. However, as claimed in [4], in this structure two kinds of outputs of the critic network share the same input and hidden layers, making them strongly coupled. Without analytical calculations, the approximated cost function and its derivatives can suffer from inconsistent errors. With the development of artificial intelligence (AI), there is an emerging need for understanding how strategies are made by AI methods, which arouses explainable AI (XAI) [13]. Following this idea, [4] introduces explicit analytical calculations to the GDHP technique, which makes it more explainable to designers because the approximate cost derivatives are explicitly computed from the approximate cost function. This explainable GDHP (XGDHP) algorithm has shown its applicability in aerospace control systems [4,12,14]. However, matrix dimensionality transformations, a.k.a. tensor operations, are involved in these studies, making it complicated to implement.

Besides, in practical applications, due to physical limitations or safety considerations, handling input constraints is a common demand for control systems [15]. A classic approach is to design

* Corresponding author.
E-mail addresses: b.sun-1@tudelft.nl (B. Sun), E.vanKampen@tudelft.nl (E.-J. van Kampen).

https://doi.org/10.1016/j.neucom.2021.10.046
0925-2312/

a non-quadratic cost function, such that the control inputs obtained by solving the HJB equation is limited by a symmetric bounded function [16]. However, although there are many researches aiming at dealing with symmetric input constraints in nonlinear optimal control problems [15,17,12,16,18], little attention has been paid to the situation subject to asymmetric input constraints. Motivated by the industrial need, Yang et al. [19] managed to cope with the asymmetric input constraints by adjusting the cost function with the mean and range of the control input constraints. However, there are two limitations in their proposed method: 1) when the system states go to zero, the control inputs are still non-zero values, specifically, the mean values of the constraint range; 2) when the control inputs go to zero, the cost caused by the control inputs is not zero. Consequently, this approach is not applicable for the stabilization problem with an origin equilibrium point, which inspires our study.

Furthermore, in order to maintain the system stability, a significant number of iterations within a sampling interval are normally required for ACDs, which result in a high computational cost [20]. To enhance the resource utilization and reduce the computational burden, event-triggered control (ETC) has been evolved as an alternate control paradigm and acquired more attentions in recent days [2,20]. ETC is originally proposed in the networked system to deal with the limitation of communication bandwidth [21–24]. These researches target solving communication issues such as synchronization, time delays and disturbances, rather than pursuing optimality or tackling control input constraints. A cross fertilization of ETC and ACD leads to the event-triggered ACD that targets for solving optimal control problem in an event-driven manner [2,8]. Most event-triggered studies focus on continuous-time systems [25–27] and only a few articles discuss the DT system. The HDP algorithm is combined with the ETC in [20,28–30,18,2] describe the event-triggered DHP algorithm. Although [11] applies the event-triggered GDHP algorithm to a network control scenario, till now there is no related research on event-triggered XGDHP. Among them, only [18] attempts to deal with symmetric input constraints merely using the non-quadratic cost function, which however is not rigorous because the control input is directly generated by the actor network that is not bounded. Furthermore, the essence of the ETC scheme lies in that a task is executed only if a predefined triggered condition is satisfied. Therefore, defining a sound triggering condition is always the primary task for the ETC scheme. For the nonaffine system, the same triggering condition is employed in [2,28,18,11] and among them [2,28,18] provide the stability analysis regarding the triggering condition. However, in [2] an extra assumption that the state norm is bounded by the supremum of control input norm is required, whereas in [28,18] the input-to-state stability (ISS) Lyapunov function is directly assumed to exist without pointing out its specific form and additional hyperparameters are involved in [18]. These limitations prevent the proposed triggering condition from wider applications.

Motivated to tackle the limitations existing in literature, we conduct this research by concentrating on the event triggered XGDHP algorithm subject to asymmetric input constraints. The contributions are summarized as follows:

1. XGDHP is developed to solve optimal control problems online. Compared to [4], the XGDHP approach developed in this paper simplifies the calculation by eliminating matrix dimensionality transformations.
2. To the best of our knowledge, it is the first time that the asymmetric control input constraints are overcome for zero-equilibrium-point stabilization problems. The combination of a novel segmented utility function and the bounding layer of the actor network guarantees strictly bounded inputs without affecting stability.
3. An event-triggered mechanism is introduced to save computational and communication's load. It is the first time that ETC is combined with XGDHP for DT systems. Compared to existing literature, fewer assumptions are required to guarantee the stability of the triggering condition and a more specific proof is provided, which demonstrates the advantage for wider applications.

The remainder of this paper is organized as follows: Section 2 states the event-triggered optimal control problem with asymmetric input constraints for the general nonlinear DT system. The triggering condition and the stability analysis of the system are provided in Section 3. Section 4 introduces the iterative XGDHP algorithm with the facilitation of three ANNs. The simulation verification is presented in Section 5 by applying the proposed approach to two nonlinear DT systems and Section 6 summarizes this paper and discusses further research.

2. Problem description

Consider a general nonlinear DT system described by:

$$x_{t+1} = f(x_t, u_t), t \in \mathbb{N}, \tag{1}$$

where t denotes the time instant, $x_t \in \Omega \subset \mathbb{R}^n$ is the state vector, and $u_t \in \Omega_u$ is the control input vector. $\Omega_u = \{u | u \in \mathbb{R}^m, u_{\min} < u_i < u_{\max}, i = 1, \ldots, m\}$, with $u_{\min} < 0$ and $u_{\max} > 0$ denoting the minimum and maximum constraint of u_i, respectively. $|u_{\min}| \neq |u_{\max}|$, i.e., the input constraints are asymmetric.

Assumption 1. System (1) is controllable and observable. $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is a Lipschitz continuous function and assumed unknown. The origin $x_t = 0$ is the unique equilibrium point of the system (1) under u_t, i.e., $f(0, 0) = 0$.

Assumption 1 implies that there exists a continuous state feedback control policy $u_t = \mu(x_t), \mu : \Omega \to \Omega_u$ that can stabilize system (1) to the equilibrium point.

Considering the event-triggered scheme, we define a sequence of triggering instants $\{s_k\}_{k=0}^{\infty}$, with s_k satisfying $s_k < s_{k+1}, k \in \mathbb{N}$. The control input is only updated at the triggering instant when a certain triggering condition is satisfied, and remains constant during the time interval $[s_k, s_{k+1})$ by involving a zero-order hold (ZOH) [28,29]. Therefore, a gap function can be defined using the event error:

$$e_t = x_{s_k} - x_t, \forall t \in [s_k, s_{k+1}), \tag{2}$$

where x_t is the current state and x_{s_k} is the triggering state held by the ZOH. Subsequently, the feedback control policy can be represented as:

$$u_t = \mu(x_{s_k}) = \mu(e_t + x_t). \tag{3}$$

Accordingly, system (1) takes the form:

$$x_{t+1} = f(x_t, \mu(e_t + x_t)). \tag{4}$$

Considering the characteristics of system (4), we introduce a discounted cost formulated as:

$$J(x_t) = \sum_{l=t}^{\infty} \gamma^{l-t} U(x_l, \mu(x_{s_k})), \tag{5}$$

where $\gamma \in (0, 1]$ is the discount factor, and $U(x_t, \mu(x_{s_k}))$ is the utility function. For the regulation task, $U(x_t, \mu(x_{s_k}))$ is supposed to satisfy $U(x, \mu) \geqslant 0$ and $U(0, 0) = 0$. Therefore, we define $U(x_t, \mu(x_{s_k}))$ as followings:

$$U(x_t, \mu(x_{s_k})) = x_t^{\mathrm{T}} Q x_t + Y(\mu(x_{s_k})), \tag{6}$$

where $Q \in \mathbb{R}^{n\times n}$ is a symmetrical positive definite matrix, and $Y(\mu(x_{s_k}))$ is a positive semi-definite function that satisfies $Y(\mu(x_{s_k})) \geqslant 0$.

Remark 1. The discount factor γ indicates the extent to which the short-term cost or long-term cost is concerned [4,12]. For the regulation task, given Assumption 1, $\gamma \leqslant 1$ can hold because of the origin equilibrium point, whereas for tasks where the equilibrium point is not the origin, $\gamma < 1$ must be satisfied to guarantee the cost function is finite [31,32].

The input constraints are asymmetric, which cannot be handled by the integrand function utilized in [15,33,16], and for the regulation task, the modified function proposed in [19] is not applicable. Inspired by these studies, we design $Y(\mu(x_{s_k}))$ as the following novel piece-wise integrand function:

$$Y = \begin{cases} 2u_{\max}\int_0^{\mu(x_{s_k})} \tanh^{-T}(\upsilon/u_{\max}) R d\upsilon, & \mu(x_{s_k}) > 0, \\ 2|u_{\min}|\int_0^{\mu(x_{s_k})} \tanh^{-T}(\upsilon/|u_{\min}|) R d\upsilon, & \mu(x_{s_k}) \leqslant 0, \end{cases} \tag{7}$$

where $\tanh^{-\mathrm{T}}(\cdot)$ stands for $\left(\tanh^{-1}(\cdot)\right)^T$, and $\tanh^{-1}(\cdot)$ is the inverse function of the hyperbolic tangent function $\tanh(\cdot)$, both of which are monotonic odd. $R = \mathrm{diag}([r_1, \cdots, r_m]) \in \mathbb{R}^{m\times m}$ is a positive definite weight matrix, where $\mathrm{diag}(\cdot)$ reshapes the vector to a diagonal matrix. It is worth mentioning that although $Y(\mu(x_{s_k}))$ is piecewise, it is at least second-order continuous with respect to $\mu(x_{s_k})$, and that only when $\mu(x_{s_k}) = 0, Y(\mu(x_{s_k})) = 0$.

Our target is to search for a feedback control law μ to minimize the designed discounted cost function (5). On the basis of Bellman's principle of opitmality [34], the optimal cost function $J^*(x_t)$ conforms to the DT HJB equation:

$$J^*(x_t) = \min_{\mu(x_{s_k})} \left\{ U(x_t, \mu(x_{s_k})) + \gamma J^*(x_{t+1}) \right\}. \tag{8}$$

The optimal control law $\mu^*(x_{s_k})$ at time instant t is accordingly defined as:

$$\mu^*(x_{s_k}) = \arg\min_{\mu(x_{s_k})} \left\{ U(x_t, \mu(x_{s_k})) + \gamma J^*(x_{t+1}) \right\}. \tag{9}$$

It is worthy mentioning that $\mu^*(x_{s_k})$ is the optimal feedback control law for the sampled state x_{s_k} at the triggering instant s_k, instead of the current state x_t. To obtain appropriate triggering instants for system (4), we define a triggering condition as follows:

$$||e_t|| > e_{\mathrm{Thr}}, \tag{10}$$

where e_{Thr} is the threshold to be determined. Therefore, it is a primary task for event-triggered control to design a sound threshold, which will be discussed in the next section.

3. Event-triggered system analysis

In this section, the triggering condition for the DT system is developed and the ISS analysis is carried out. First of all, the following assumption is necessary [28,2]:

Assumption 2. For system (4), there exists a positive constant $C \in (0, 0.5)$ guaranteeing the following equation:

$$||f(x_t, \mu(e_t + x_t))|| \leqslant C||x_t|| + C||e_t||, \tag{11}$$

and $||e_t||$ satisfies $||e_t|| \leqslant ||x_t||$.

Lemma 1. *If Assumption 2 holds, the triggering condition can be defined as follows:*

$$||e_t|| > e_{\mathrm{Thr}} = C\frac{1-(2C)^{t-s_k}}{1-2C}||x_{s_k}||. \tag{12}$$

Proof. Regard s_k as the last triggered instant. According to Assumption 2, for each $t \in [s_k, s_{k+1})$, we have:

$$||e_{t+1}|| = ||x_{s_k} - x_{t+1}|| \leqslant ||x_{t+1}||. \tag{13}$$

Substituting (11) into (13) yields:

$$||e_{t+1}|| \leqslant C||x_t|| + C||e_t||. \tag{14}$$

With (2), (14) can be rewritten as:

$$||e_{t+1}|| \leqslant 2C||e_t|| + C||x_{s_k}||. \tag{15}$$

Therefore, by conducting back-forward recursion, we obtain the following inequality:

$$\begin{aligned} ||e_t|| \leqslant\ & 2C||e_{t-1}|| + C||x_{s_k}|| \\ & \cdots \\ & \leqslant (2C)^{t-s_k}||e_{s_k}|| + (2C)^{t-s_k-1}C||x_{s_k}|| \\ & + \cdots + (2C)C||x_{s_k}|| + C||x_{s_k}||. \end{aligned} \tag{16}$$

By solving (16) with initial condition $e_{s_k} = 0$, we attain:

$$||e_t|| \leqslant C\frac{1-(2C)^{t-s_k}}{1-2C}||x_{s_k}||. \tag{17}$$

If (17) is violated, i.e., (12) is satisfied, the event is triggered. This completes the proof.

It is noted that the threshold value e_{Thr} is not unique since it is influenced by the triggered state x_{s_k} and the designed constant C that is usually chosen experimentally. Subsequently, inspired by [30], we proceed to prove the system (4) is asymptotically stable under the triggering condition (12).

Definition 1. [2] A continuous function $V : \mathbb{R}^n \to \mathbb{R}_+$ is called an ISS Lyapunov function for the system (4), if there exist $\mathcal{K}_\infty$ functions α_1, α_2, and α_3, and a $\mathcal{K}$ function ρ, such that:

$$\alpha_1(||x_t||) \leqslant V(x_t) \leqslant \alpha_2(||x_t||), \tag{18}$$

$$V(f(x_t, \mu(e_t + x_t))) - V(x_t) \leqslant -\alpha_3(||x_t||) + \rho(||e_t||), \tag{19}$$

hold for all $x_t \in \mathbb{R}^n$ and $e_t \in \mathbb{R}^n$.

Theorem 1. With Assumption 2 and the triggering condition (12), the event-triggered system is input-to-state stable and is asymptotically stable.

Proof. The following proof only takes the situation that the event is not triggered at the time instant $t+1$ into consideration, because when the event is triggered, the control input will be updated, and it will be equivalent to the time-based control at $t+1$. According to the optimal control theory, the stability can be guaranteed at this single instant.

We firstly define a Lyapunov function as followings:

$$V(x_t) = x_t^{\mathrm{T}} Q x_t + Y(\mu(x_t)). \tag{20}$$

Then, we define a series of functions as follows:

$$\alpha_1(||x_t||) = x_t^{\mathrm{T}} Q_1 x_t + Y(\mu(x_t)), \tag{21}$$

$$\alpha_2(||x_t||) = x_t^{\mathrm{T}} Q_2 x_t + Y(\mu(x_t)), \tag{22}$$

$$\alpha_3(||x_t||) = ||q||^2\left(1-2C^2\right)||x_t||^2, \tag{23}$$

$$\rho(||e_t||) = 2C^2||q||^2||e_t||^2, \tag{24}$$

where Q_1 and Q_2 can be selected to satisfy (18), and the vector q in (23) and (24) can be determined from (6) as $Q = qq^{\mathrm{T}}$. Therefore, $x_t^T Q x_t = x_t^T qq^T x = ||x^T q||^2$.

Subsequently, the proof is conducted by presenting that (20) is an ISS Lyapunov function and it is non-increasing, i.e., $\Delta V = V(x_{t+1}) - V(x_t) \leqslant 0$.

For all $t \in [s_k, s_{k+1})$, according to the ETC mechanism, $\mu(x_{t+1}) = \mu(x_t) = \mu(x_{s_k})$, and therefore, we have

$$\begin{aligned}\Delta V = \; & x_{t+1}^{\mathrm{T}} Q x_{t+1} + Y(\mu(x_{t+1})) \\ & -\left(x_t^{\mathrm{T}} Q x_t + Y(\mu(x_t))\right) \\ & = x_{t+1}^{\mathrm{T}} Q x_{t+1} - x_t^T Q x_t.\end{aligned} \tag{25}$$

Substituting (11) into (25) yields:

$$\Delta V \leqslant ||q||^2\left((C||x_t|| + C||e_t||)^2 - ||x_t||^2\right). \tag{26}$$

According to the Cauchy–Schwarz inequality, (26) becomes:

$$\begin{aligned}\Delta V \; & \leqslant ||q||^2\left(2C^2||x_t||^2 + 2C^2||e_t||^2 - ||x_t||^2\right) \\ & = \left(2C^2-1\right)||q||^2||x_t||^2 + 2C^2||q||^2||e_t||^2 \\ & = -\alpha_3(||x_t||) + \rho(||e_t||).\end{aligned} \tag{27}$$

Consequently, referring to Definition 1, (20) is an ISS Lyapunov function. According to [18,2], a system is input-to-state stable if it admits a smooth ISS Lyapunov function.

Then, considering (2) and (17), (27) continues as:

$$\begin{aligned}\Delta V \; & \leqslant \left(4C^2-1\right)||q||^2||e_t||^2 + \left(2C^2-1\right)||q||^2||x_{s_k}||^2 \\ & \leqslant \left[\left(4C^2-1\right)C\tfrac{1-(2C)^{t-s_k}}{1-2C} + \left(2C^2-1\right)\right]||q||^2||x_{s_k}||^2,\end{aligned} \tag{28}$$

Since $C < 0.5$ and $4C^2 - 1 = (2C-1)(2C+1)$, the last inequality in (28) can be rewritten as:

$$\begin{aligned}\Delta V = \; & -\left[(2C+1)C(1-(2C)^{t-s_k}) + \left(1-2C^2\right)\right]||q||^2||x_{s_k}||^2 \\ \leqslant \; & 0,\end{aligned} \tag{29}$$

where $\Delta V = 0$ if and only if $||x_{s_k}|| = 0$, which implies that the system has been stabilized since the time instant s_k.

Overall, we can conclude that the event-triggered system (4) is input-to-state stable and is asymptotically stable with the triggering condition (12), which completes the proof.

Remark 2. The triggering condition (12) has the same form of that in some existing literature [28,18,2,11]. Nevertheless, different from them, fewer assumptions are required to guarantee the asymptotic stability. Furthermore, [28,18] assume the existence of an ISS Lyapunov function without providing its specific formula, whereas in this paper the ISS Lyapunov function is specifically defined by (20)–(24).

The simple diagram of the ETC scheme is illustrated in Fig. 1. Only when an event is triggered, will the XGDHP algorithm be activated and the control input be updated. In the next section, the detailed implementation of the XGDHP algorithm will be presented.

4. Event-triggered iterative ACD with The XGDHP technique

In this section, according to the universal approximation property of ANNs [20], we first construct a model network, represented

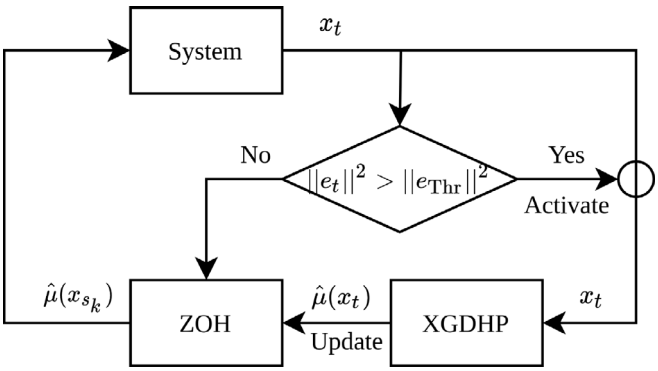

Fig. 1. Simple diagram of the ETC scheme incorporating the XGDHP algorithm.

by subscript m, to identify the system dynamics. Then, the event-triggered iterative adaptive critic algorithm is introduced, and the actor and critic networks, respectively represented by subscript a and c, are built to facilitate the implementation. The XGDHP technique is developed based on explicit analytical computations in the critic network, and the asymmetric input constraints are addressed by modifying the output layer of the actor network.

All ANNs are constructed with the full-connected feed-forward architecture, and their hidden layers respectively has l_m, l_a, and l_c neurons, all of which adopt a sigmoid function as the activation function:

$$\sigma(\tau) = \frac{1-e^{-\tau}}{1+e^{-\tau}}, \tag{30}$$

whose derivative is $\sigma\prime(\tau) = 0.5\left(1-(\sigma(\tau))^2\right)$.

4.1. The model network

Since the system dynamics is unknown, a model network is built and trained in advance before implementing the XGDHP technique. The model network is constructed offline to identify the dynamics and predict the next state as follows:

$$x_{t+1} = w_{m2}^{\mathrm{T}}\sigma\left(\begin{bmatrix} w_{m1,x} \\ w_{m1,u}\end{bmatrix}^{\mathrm{T}}\begin{bmatrix} x_t \\ u_t\end{bmatrix}\right) + \varepsilon_{m,t}, \tag{31}$$

in which $w_{m2} \in \mathbb{R}^{l_m \times n}, w_{m1,x} \in \mathbb{R}^{n \times l_m}$, and $w_{m1,u} \in \mathbb{R}^{m \times l_m}$ are ideal weight matrices of the model network, and $\varepsilon_{m,t} \in \mathbb{R}^n$ is the reconstruction error. Subsequently, by defining $w_{m1} = \left[w_{m1,x}^{\mathrm{T}}, w_{m1,u}^T\right]^T$, the identification scheme is described as:

$$\hat{x}_{t+1} = \hat{w}_{m2}^{\mathrm{T}}\sigma\left(\hat{w}_{m1}^{\mathrm{T}}\left[x_t^T, u_t^T\right]^T\right), \tag{32}$$

where $\hat{x}_{t+1}, \hat{w}_{m1}$, and $\hat{w}_{m2}$ are the estimations of x_{t+1}, w_{m1}, and w_{m2}, respectively.

The model network is supposed to minimize the identification error $\tilde{x}_{m,t+1} = \hat{x}_{t+1} - x_{t+1}$, and therefore the target performance measure is defined as:

$$E_{m,t} = \frac{1}{2}\tilde{x}_{m,t+1}^{\mathrm{T}}\tilde{x}_{m,t+1}. \tag{33}$$

The weight tuning law is designed obeying a gradient-descent algorithm:

$$\begin{aligned}\Delta\hat{w}_{m2} \; & = -\eta_m \tfrac{\partial\hat{x}_{t+1}}{\partial\hat{w}_{m2,t}}\tfrac{\partial E_{m,t}}{\partial\hat{x}_{t+1}}, \\ \Delta\hat{w}_{m1} \; & = -\eta_m \tfrac{\partial\hat{x}_{t+1}}{\partial\hat{w}_{m1,t}}\tfrac{\partial E_{m,t}}{\partial\hat{x}_{t+1}},\end{aligned} \tag{34}$$

where $\eta_m > 0$ is the learning rate, and $\Delta\hat{w}_{m1}$ and $\Delta\hat{w}_{m2}$ are the differences of two subsequent updating steps.

After a sufficient training session, the model network can achieve a satisfying precision, with the weight matrix converging to a constant value. It is important to note that after training, the model weight matrix is kept unchanged for controller design. With the model network, the necessary partial derivative information can be obtained for training critic and actor networks.

Considering the event-triggered framework, by replacing u_t with $\mu(x_{s_k})$ in (32) and taking the partial derivative with respect to x_t and $\mu(x_{s_k})$, respectively, we get:

$$\frac{\partial \hat{x}_{t+1}}{\partial x_t} = \hat{w}_{m1,x}\left(\sigma\prime\left(\hat{w}_{m1}^{\mathrm{T}}\left[x_t^T, \mu^T(x_{s_k})\right]^T\right) \odot \hat{w}_{m2}\right), \tag{35}$$

$$\frac{\partial \hat{x}_{t+1}}{\partial \mu(x_{s_k})} = \hat{w}_{m1,u}\left(\sigma\prime\left(\hat{w}_{m1}^{\mathrm{T}}\left[x_t^T, \mu^T(x_{s_k})\right]^T\right) \odot \hat{w}_{m2}\right). \tag{36}$$

By denoting $F_t = \partial \hat{x}_{t+1}^{\mathrm{T}}/\partial x_t$ and $G_t = \partial \hat{x}_{t+1}^{T}/\partial \mu(x_{s_k})$, we can approximate (4) as a new affine system:

$$\hat{x}_{t+1} = F_t x_t + G_t \mu(x_{s_k}). \tag{37}$$

With (37), the optimal event-triggered control law $\mu^*(x_{s_k})$ can accordingly be approximated as:

$$\hat{\mu}^*(x_{s_k}) = \phi\left(\hat{D}^*(x_{s_k})\right), \tag{38}$$

where $\phi(\cdot)$ is a one-to-one piece-wise function defined as:

$$\phi(\tau) = \begin{cases} u_{\max}\tanh(\tau/u_{\max}) & , \quad \tau > 0, \\ |u_{\min}|\tanh(\tau/u_{\min}) & , \quad \tau \leqslant 0, \end{cases} \tag{39}$$

and $\hat{D}^*(x_{s_k})$ is described by:

$$\hat{D}^*(x_{s_k}) = -\frac{\gamma}{2} R^{-1} G_{s_k}^T \hat{\lambda}^*(\hat{x}_{s_k+1}), \tag{40}$$

in which $\hat{\lambda}^*(\hat{x}_{s_k+1}) = \partial \widehat{J}^*(\hat{x}_{s_k+1})/\partial \hat{x}_{s_k+1}$ is the costate function. It can be found that $\phi(\cdot)$ is at least second-order continuous. Accordingly, the approximate DTHJB equation takes the form:

$$\widehat{J}^*(x_t) = U(x_t, \hat{\mu}^*(x_{s_k})) + \gamma \widehat{J}^*(F_t x_t + G_t \hat{\mu}^*(x_{s_k})). \tag{41}$$

4.2. Iterative adaptive critic algorithm

Through (38) and (41), it can be found that the computation of $\widehat{J}^*(x_t)$ and $\hat{\mu}^*(x_{s_k})$ requires the future information. Clearly, although the system dynamics has been identified, this bootstrapping phenomenon [9] makes it intractable or impossible to obtain the analytical solution of the DTHJB equation for nonlinear systems. Consequently, we introduce an iterative adaptive critic algorithm with the XGDHP technique to iteratively solve it.

The procedure of the DT iterative adaptive critic algorithm is briefly depicted in Algorithm 1, where $b_{\Delta J} > 0$ is a designed threshold and $i \in \mathbb{N}$ denotes the iteration index. It is worth mentioning that only the situation that $||e_t|| > ||e_{\mathrm{Thr}}||$, i.e., $t = s_k$, is considered, because the control input is updated only at the triggered instant. The main idea is to construct two iterative sequences $\left\{J^{(i)}(x_{s_k})\right\}$ and $\{\mu^{(i)}(x_{s_k})\}$ to perform the value iteration process so as to achieve approximately optimal values [2,9].

Algorithm 1: Iterative Adaptive Critic Algorithm

Initialization: Choose $b_{\Delta J}$, and set $i = 0$ and $J^{(0)}(\cdot) = 0$;

while $|J^{(i+1)}(x_{s_k}) - J^{(i+1)}(x_{s_k})| > b_{\Delta J}$ **do**

compute the iterative control input as

$$\begin{aligned}\mu^{(i)}(x_{s_k}) &= \arg\min_{\mu(x_{s_k})}\{U(x_{s_k}, \mu(x_{s_k})) \\ &\quad + \gamma J^{(i)}(\hat{x}_{s_k+1})\} \\ &= \phi(\hat{D}_{s_k}^{(i)}(x_{s_k}))\end{aligned}$$

update the iterative cost function as

$$\begin{aligned}J^{(i+1)}(x_{s_k}) &= \min_{\mu(x_{s_k})}\{U(x_{s_k}, \mu^{(i)}(x_{s_k})) \\ &\quad + \gamma J^{(i)}(\hat{x}_{s_k+1})\} \\ &= U(x_{s_k}, \mu^{(i)}(x_{s_k})) \\ &\quad + \gamma J^{(i)}(F_{s_k} x_{s_k} + G_{s_k}\mu(x_{s_k}));\end{aligned}$$

i=i+1;

end

Results: Obtain the near optimal control law $\hat{\mu}^*(x_{s_k})$.

The convergence analysis of the DT iterative adaptive critic algorithm has been carried out in [34,17,35] and thus is omitted here. The core procedure is to prove that $\left\{J^{(i)}(x_{s_k})\right\}$ is a non-decreasing sequence with an upper bound b_J, i.e.,

$$J^{(0)}(\cdot) \leqslant J^{(1)}(\cdot) \leqslant \cdots \leqslant J^{(\infty)}(\cdot) \leqslant b_J. \tag{44}$$

Accordingly, we can further derive that the iteration between the sequences (42) and (43) guarantees the convergence to the optimal values for both sequences, i.e., $J^{(i)}(x_{s_k}) \to J^{(\infty)}(x_{s_k}) = \widehat{J}^*(x_{s_k})$ and $\mu^{(i)}(x_{s_k}) \to \hat{\mu}^*(x_{s_k})$ as $i \to \infty$ [36,2].

Remark 3. Since $J^{(i)}(x_{s_k}) \to \widehat{J}^*(x_{s_k})$ as $i \to \infty$, by denoting $\lambda^{(i)}(x_{s_k}) = \partial J^{(i)}(x_{s_k})/\partial x_{s_k}$, we can conclude that the costate function sequence $\left\{\lambda^{(i)}(x_{s_k})\right\}$ is also convergent with $\lambda^{(i)}(x_{s_k}) \to \hat{\lambda}^*(x_{s_k})$ as $i \to \infty$. Nevertheless, in practical implementation, the satisfying convergent results can already be observed when the iteration index i is sufficient large, rather than infinite.

Subsequently, for carrying out the iterative adaptive critic algorithm, the actor and critic networks are constructed to respectively approximate the control law and the cost function in the following subsections. The derivation presents the calculations in one iteration step and therefore the superscript is omitted for simplicity.

4.3. The actor network

For building a direct differentiable mapping from the state to the control input, the actor network is constructed, whose output

is directly introduced to the model network and the real system. Inspired by [31,12,14], to guarantee the asymmetric input constraints, a bounding layer is connected to the original output layer of the three-layer network. In this bounding layer, the aforementioned function $\phi(\cdot)$ that is defined in (39) is adopted as the activation function. Fig. 2 illustrates the architecture of the actor network, and its output is presented as:

$$\hat{\mu}(x_{s_k}) = \phi\left(\hat{w}_{a2}^{\mathrm{T}}\sigma\left(\hat{w}_{a1}^{\mathrm{T}}x_{s_k}\right)\right), \tag{45}$$

where $\hat{w}_{a1} \in \mathbb{R}^{n\times l_a}$ and $\hat{w}_{a2} \in \mathbb{R}^{l_a\times m}$ are the estimations of the ideal weight matrices $w_{a1} \in \mathbb{R}^{n\times l_a}$ and $w_{a2} \in \mathbb{R}^{l_a\times m}$, respectively.

Based on (42) and (45), the performance to be minimized for the actor network can be defined as:

$$E_{a,s_k} = \frac{1}{2}\left[\hat{\mu}(x_{s_k}) - \mu(x_{s_k})\right]^{\mathrm{T}}\left[\hat{\mu}(x_{s_k}) - \mu(x_{s_k})\right]. \tag{46}$$

Similarly, with a learning rate $\eta_a > 0$, the weight matrices are updated by:

$$\begin{aligned}\Delta\hat{w}_{a2} &= -\eta_a \frac{\partial\hat{\mu}(x_{s_k})}{\partial\hat{w}_{a2}}\frac{\partial E_{a,s_k}}{\partial\hat{\mu}(x_{s_k})},\\ \Delta\hat{w}_{a1} &= -\eta_a \frac{\partial\hat{\mu}(x_{s_k})}{\partial\hat{w}_{a1}}\frac{\partial E_{a,s_k}}{\partial\hat{\mu}(x_{s_k})}.\end{aligned} \tag{47}$$

Remark 4. The combination of the segmented utility function and the bounding layer of the actor network is one of the highlights of this paper. With the segmented utility function, a target policy within the designed asymmetric range is provided to the actor network to learn. Besides, the bounding layer is necessary because the signal $\hat{\mu}(x_{s_k})$, which is an output of the actor network, is directly utilized to control the system.

4.4. The critic network

For the conventional GDHP technique, the critic network outputs the approximation of cost function and its derivatives simultaneously [11,10], whose description is as follows:

$$\begin{bmatrix}\widehat{J}(x_{s_k})\\ \hat{\lambda}(x_{s_k})\end{bmatrix} = \begin{bmatrix}\hat{w}_{c2,J}\\ \hat{w}_{c2,\lambda}\end{bmatrix}^{\mathrm{T}}\sigma\left(\hat{w}_{c1}^{\mathrm{T}}x_{s_k}\right), \tag{48}$$

where $\hat{w}_{c1} \in \mathbb{R}^{n\times l_c}, \hat{w}_{c2,J} \in \mathbb{R}^{l_c}$, and $\hat{w}_{c2,\lambda} \in \mathbb{R}^{l_c\times n}$ respectively denotes the estimation of the ideal weights $w_{c1} \in \mathbb{R}^{n\times l_c}$ $w_{c2,J} \in \mathbb{R}^{l_c}$, and $w_{c2,\lambda} \in \mathbb{R}^{l_c\times n}$. However, due to the inevitable approximation error, $\widehat{J}(x_{s_k})$ and $\hat{\lambda}(x_{s_k})$ approximated in this way cannot exactly provide the derivative relationship, which is called suffering from the inconsistency error [4].

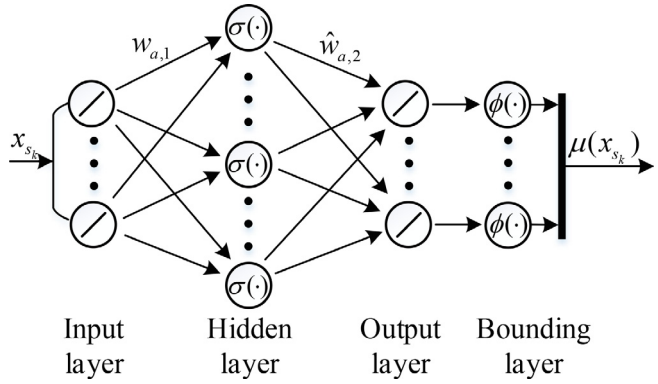

Fig. 2. The structure of the actor network, where the input layer and the output layer employ a unit-proportion linear activation function, while the hidden layer and the bounding layer exploit aforementioned $\sigma(\cdot)$ and $\phi(\cdot)$ as their activation functions, respectively.

Therefore, inspired by [4,12], a novel XGDHP technique that takes the advantage of explicit analytical calculations is developed, with the critic network only approximating the cost function as follows:

$$\widehat{J}(x_{s_k}) = \hat{w}_{c2}^{\mathrm{T}}\sigma\left(\hat{w}_{c1}^{\mathrm{T}}x_{s_k}\right), \tag{49}$$

where $\hat{w}_{c2} \in \mathbb{R}^{l_c}$ is the estimation of the ideal weight matrix $w_{c2} \in \mathbb{R}^{l_c}$. By taking the explicit analytical calculations, we obtain $\hat{\lambda}(x_{s_k})$ as:

$$\hat{\lambda}(x_{s_k}) = \frac{\partial\widehat{J}(x_{s_k})}{\partial x_{s_k}} = w_{c1}\left(\hat{w}_{c2}\odot\sigma\prime\left(w_{c1}^{\mathrm{T}}x_{s_k}\right)\right), \tag{50}$$

where $\odot$ is the Hadamard product.

XGDHP makes use of the cost function and its derivative information, so recalling (43), the critic network is expected to minimize the following performance measure:

$$e_{c1,s_k} = \widehat{J}(x_{s_k}) - U\left(x_{s_k}, \hat{\mu}(x_{s_k})\right) - \gamma\widehat{J}(\hat{x}_{s_k+1}), \tag{51}$$

$$e_{c2,s_k} = \frac{\partial\left[\widehat{J}(x_{s_k}) - U\left(x_{s_k}, \hat{\mu}(x_{s_k})\right) - \gamma\widehat{J}(\hat{x}_{s_k+1})\right]}{\partial x_{s_k}}, \tag{52}$$

$$E_{c,s_k} = \beta\tfrac{1}{2}e_{c1,s_k}^2 + (1-\beta)\tfrac{1}{2}e_{c2,s_k}^{\mathrm{T}}e_{c2,s_k}, \tag{53}$$

where β is a scalar within a range of $[0, 1]$. If $\beta = 1$, it becomes pure HDP, whereas if $\beta = 0$, then the weight matrix is tuned merely based on the computed derivatives $\hat{\lambda}(x_{s_k})$, and consequently it is equivalent to DHP [12].

Different from [2,11], we also take the partial derivative of $\hat{\mu}(x_{s_k})$ with respect to x_{s_k} into consideration in the critic network updating procedure for more precise calculations. According to the chain rule, (52) can further be derived as:

$$\begin{aligned}e_{c2,s_k} = {} & \hat{\lambda}(x_{s_k}) - 2Qx_{s_k} - \frac{\partial\hat{\mu}(x_{s_k})}{\partial x_{s_k}}\frac{\partial Y\left(\hat{\mu}(x_{s_k})\right)}{\partial\hat{\mu}(x_{s_k})}\\ & -\gamma\left(\frac{\partial\hat{x}_{s_k+1}}{\partial x_{s_k}} + \frac{\partial\hat{\mu}(x_{s_k})}{\partial x_{s_k}}\frac{\partial\hat{x}_{s_k+1}}{\partial\hat{\mu}(x_{s_k})}\right)\hat{\lambda}(x_{s_k+1}),\end{aligned} \tag{54}$$

where $\partial\hat{\mu}(x_{s_k})/\partial x_{s_k}$ is computed with the facilitation of the actor network, while $\partial\hat{x}_{s_k+1}/\partial x_{s_k}$ and $\partial\hat{x}_{s_k+1}/\partial\hat{\mu}(x_{s_k})$ are computed through the model network.

Given a learning rate $\eta_c > 0$, the weight updating algorithm is conducted by:

$$\Delta\hat{w}_{c2} = -\eta_c\frac{\partial E_{c,s_k}}{\partial\hat{w}_{c2}}, \Delta\hat{w}_{c1} = -\eta_c\frac{\partial E_{c,s_k}}{\partial\hat{w}_{c1}}, \tag{55}$$

and

$$\begin{aligned}\frac{\partial E_{c,s_k}}{\partial\hat{w}_{c2}} &= \beta\frac{\partial\widehat{J}(x_{s_k})}{\partial\hat{w}_{c2}}e_{c1,s_k} + (1-\beta)\frac{\partial\hat{\lambda}(x_{s_k})}{\partial\hat{w}_{c2}}e_{c2,s_k},\\ \frac{\partial E_{c,s_k}}{\partial\hat{w}_{c1}} &= \beta\frac{\partial\widehat{J}(x_{s_k})}{\partial\hat{w}_{c1}}e_{c1,s_k} + (1-\beta)\frac{\partial\hat{\lambda}(x_{s_k})}{\partial\hat{w}_{c1}}e_{c2,s_k},\end{aligned} \tag{56}$$

where $\partial\hat{\lambda}(x_{s_k})/\partial\hat{w}_{c2}$ and $\partial\hat{\lambda}(x_{s_k})/\partial\hat{w}_{c1}$ are the second-order mixed gradients of the cost function $\widehat{J}(x_{s_k})$. To compute $\partial\hat{\lambda}(x_{s_k})/\partial\hat{w}_{c2}$ and $\partial\hat{\lambda}(x_{s_k})/\partial\hat{w}_{c1}$, Kronecker product and thus tensor operations are involved in [4,12,14], which result in the need for matrix dimensionality transformation. In this paper, we develop a simpler computation method as follows:

$$\begin{aligned}&\frac{\partial\hat{\lambda}(x_{s_k})}{\partial w_{c2}}e_{c2,s_k}=(\hat{w}_{c1}^{\mathrm{T}}e_{c2,s_k})\odot\sigma\prime(\hat{w}_{c1}^{\mathrm{T}}x_{s_k}),\\&\frac{\partial\hat{\lambda}(x_{s_k})}{\partial w_{c1}}e_{c2,s_k}=e_{c2,s_k}(\hat{w}_{c2}\odot\sigma\prime(\hat{w}_{c1}^{\mathrm{T}}x_{s_k}))^{\mathrm{T}}\\&-x_{s_k}(\hat{w}_{c1}^{\mathrm{T}}x_{s_k}\odot\hat{w}_{c2}\odot\sigma(\hat{w}_{c1}^{\mathrm{T}}x_{s_k})\odot\sigma\prime(\hat{w}_{c1}^{T}x_{s_k}))^{T}.\end{aligned}\tag{57}$$

Through mathematical derivation, it can be found that (57) is equivalent to the method proposed in [4].

The closed-loop stability and the convergence of weights of ANNs can be found in [28]. Note that the weights between the input layer and the hidden layer of these networks are also updated, which is the same as in [2,18,10,11,31] but different from [29,28] where they are fixed after the initialization. Nevertheless, the update behaviours in these methods obey the same gradient descent logic and similar rules. The update is proved successful through the simulation studies in this paper and others [2,18,10,11,31]. In practice, one can manually set a constraint for the weights to guarantee the boundedness and safety.

Overall, the structural diagram of the present XGDHP implementation is depicted in Fig. 3 to clarify the design procedure, where DER is given by:

$$\mathrm{DER}=\frac{\partial\hat{x}_{s_k+1}}{\partial x_{s_k}}+\frac{\partial\hat{\mu}(x_{s_k})}{\partial x_{s_k}}\frac{\partial\hat{x}_{s_k+1}}{\partial\hat{\mu}(x_{s_k})}.\tag{58}$$

5. Simulation studies

In this section, two simulation studies are carried out to illustrate the feasibility of the developed approach and compare the performance of the event-triggered XGDHP with the time-based approach.

5.1. Example 1

Consider the following nonlinear affine mass-spring system [17]:

$$\begin{cases}x_{1,t+1}&=x_{1,t}+0.05x_{2,t},\\x_{2,t+1}&=-0.0005x_{1,t}-0.0335x_{1,t}^3+x_{2,t}+0.05u_t,\end{cases}\tag{59}$$

where $x_t=[x_{1,t},x_{2,t}]^{\mathrm{T}}\in\mathbb{R}^2$ and $u_t\in\Omega_u=\{u_t|u_t\in\mathbb{R},-0.5<u_t<0.2\}$. The parameters in the utility function are selected as $Q=I_2$ and $R=1$, and the forgetting factor is chosen as $\gamma=0.995$.

In what follows, we perform the proposed event-triggered XGDHP algorithm with the facilitation of ANNs. All ANNs are constructed with 8 hidden neurons, i.e., $l_m=l_c=l_a=8$. Their weight matrices between the input layer and the hidden layer are initialized within $[-1,1]$, and the weights between the hidden layer and the output layer are randomly initialized with the uniform distribution within $[-0.01,0.01]$. The initial weights of the actor and the critic networks utilized to present results are provided in AppendixA. The learning rates are experimentally set as $\eta_m=\eta_c=\eta_a=0.01$.

First of all, we employ 500 data samples to train the model network for 500 times, and then utilize another 500 data samples for testing. The identification errors of the model network of testing samples are illustrated in Fig. 4, from which, we can see that the mean sum of squares of the identification errors is below 1.4×10^{-3}. Therefore, we can say that the model network with high accuracy has been obtained. After training, the weights of the model network are kept unchanged for controller design.

Next, we start the controller design procedure. It is noted that the simulation of the control algorithm is conducted in an online manner, which means that the control policy improves as it is applied to the real system. Through setting $C=0.12$, we can accordingly obtain the triggering threshold e_{Thr} as:

$$e_{\mathrm{Thr}}=0.12\cdot\frac{1-(0.24)^{t-s_k}}{1-0.24}||x_{s_k}||.\tag{60}$$

If the condition $||e_t||^2>||e_{\mathrm{Thr}}||^2$ the controller will be updated, and the triggering state x_{s_k} is reset with current state. Before the occurring the next triggering event, the control input u_t is remained by ZOH as u_{s_k}. Different from all other works that applies ETC to DT systems using ADP algorithms [29,18,2,28,30,11], the actor and critic networks are not updated until an event is triggered in this paper so as to further reduce computational burden. For the XGDHP technique, we set $\beta=0.5$ to combine the information of the cost function and its derivatives. For ensuring sufficient learning, the prespecified accuracy $b_{\Delta J}$ is set to be 10^{-4}, and during each time step that is triggered, at most 1000 internal cycles for training the critic and actor networks are included to achieve satisfying performance.

With the initial state chosen as $x_0=[1,-1]^{\mathrm{T}}$, we conduct the proposed event-triggered XGDHP algorithm in comparison to time-based XGDHP algorithm. Both control algorithms share same settings and parameters except for the triggering mechanism. The simulation results corresponding to the systems state and the

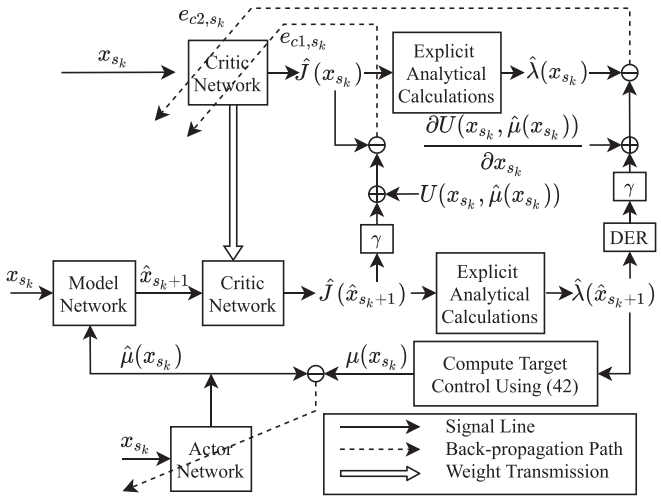

Fig. 3. Structural diagram of the developed XGDHP algorithm.

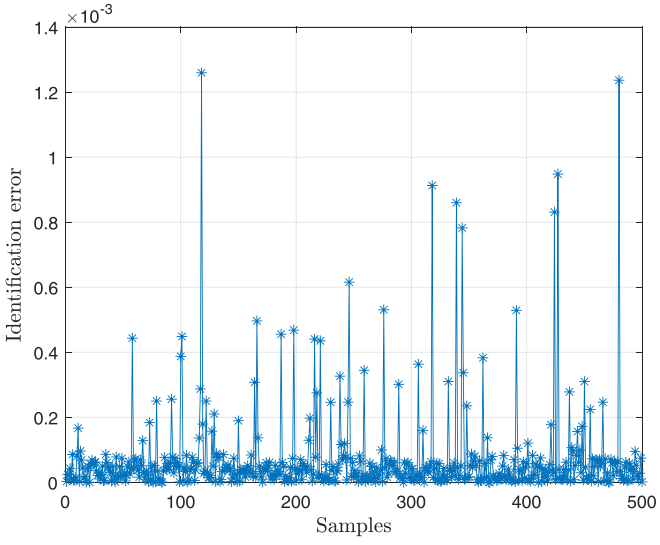

Fig. 4. The mean sum of squares of the identification errors for Example 1.

control input are depicted in Figs. 5 and 6, respectively. Due to the event-triggered mechanism, the event-triggered XGDHP algorithm presents a stair-steeping control input signal. With the piece-wise integral function (7) and the bounding layer in the actor network, it can be observed that the control input is bounded within the range of $(-0.5, 0.2)$. Therefore, we can say that the asymmetric control input constraints have been addressed. The evolution of the weights of ANNs is depicted in Fig. 7, where solid lines denote the weights between the input layer and the hidden layer while the weights between the hidden layer and the output layer are represented by dashed lines. It can be observed that all weights eventually converge to constant values during the online learning process. The evolution of the one-step cost and the accumulative cost in the learning process is demonstrated in Fig. 8. Utilizing fewer data samples, the event-triggered XGDHP requires 3 more steps to control the system achieving the stage where the one-step cost is kept below 0.05, and 7 more steps below 0.01. Because of the delayed control, the event-triggered XGDHP shows a greater overshot, which results in larger accumulative cost. Nevertheless, in many practical scenarios where saving computational resource is preferred, this depletion of the control effectiveness is acceptable.

In addition, the evolution curve of triggering threshold is depicted in Fig. 9, which converges to around zero along with the event error. The inter-execution time is illustrated in Fig. 10, which presents the time interval between two triggered instants. It is worth mentioning that the time-based controller requires every samples in this 100-step task, whereas the proposed event-triggered approach only utilizes 60 samples. Since at each triggered instant, the critic and actor networks are trained for 1000 steps, the event-triggered approach greatly reduces the computational burden up to 40%.

Remark 5. The simulation results show that the system is gradually stabilized. Nevertheless, it is noted that due to the asymptotic stability property, the system states may not exactly converge to zero, which makes the ETC scheme keep working during the whole presented time range, as depicted in Figs. 9 and 10. This phenomenon is because the triggering condition described by (12) is dependent on the system states. As the stabilization continues, the triggering threshold will accordingly adapt to a stricter value to guarantee the precision. In practice, a threshold can be set for the controller, such that, when the system states reach a certain range, the controller can be deactivated to further save resources.

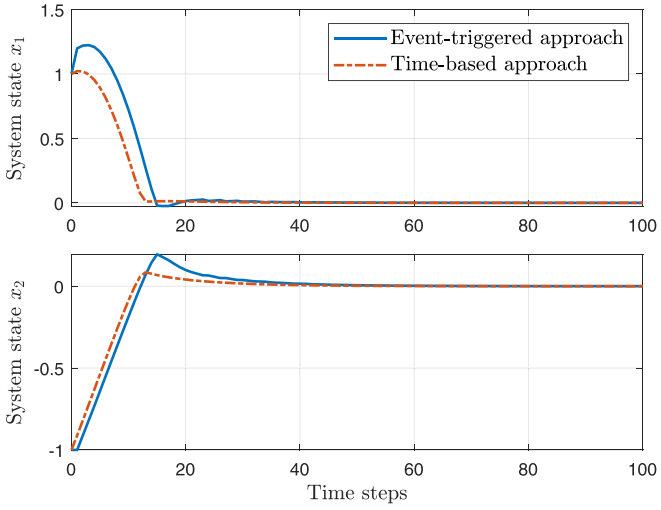

Fig. 5. Evolution of the system state in the online learning process for Example 1. Controllers aim at stabilizing system states initially from $x_0 = [1, -1]^\mathrm{T}$ to 0.

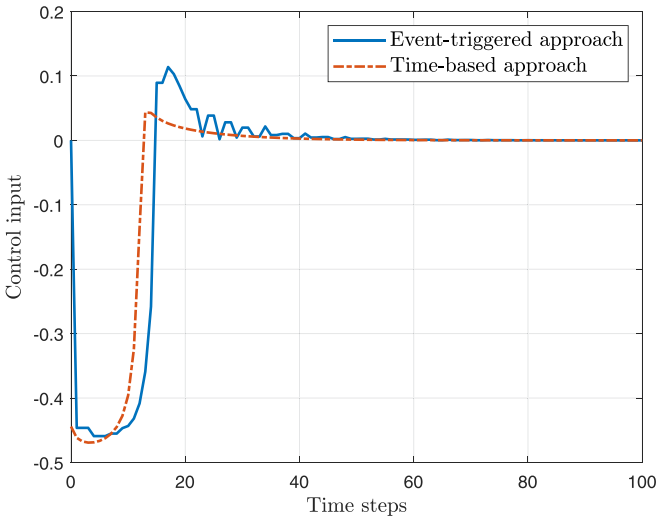

Fig. 6. Evolution of the control input in the online learning process for Example 1.

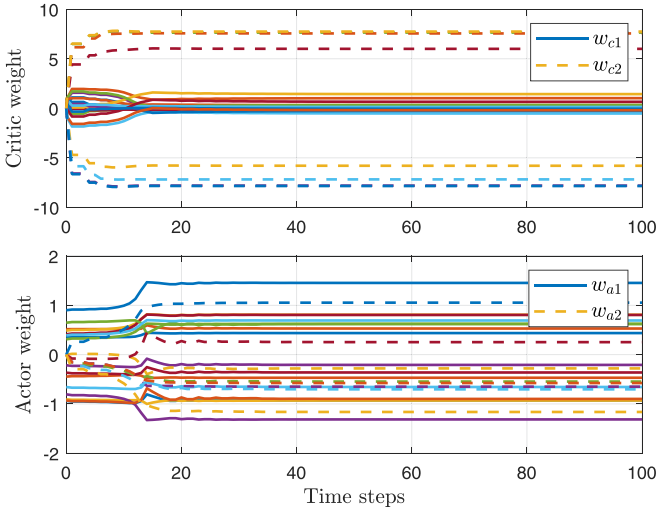

Fig. 7. Evolution of the weights of ANNs in the online earning process for Example 1. Subscripts c and a denote the critic and the actor networks, respectively. Subscripts 1 and 2 denote the weights between the input and the hidden layers and the weights between the hidden and output layers, respectively.

5.2. Example 2

The second numerical example considered is a nonlinear multiple-input-multiple-output nonaffine system [2] described by:

$$\begin{cases} x_{1,t+1} &= x_{1,t} + 0.1x_{2,t}, \\ x_{2,t+1} &= -0.17\sin(x_{1,t}) + 0.98x_{2,t} + 0.1u_{1,t}, \\ x_{3,t+1} &= 0.1x_{1,t} + 0.2x_{2,t} + x_{3,t}\cos(u_{2,t}), \end{cases} \tag{61}$$

where $x_t = [x_{1,t}, x_{2,t}, x_{3,t}]^\mathrm{T} \in \mathbb{R}^3$ and $u_t \in \Omega_u = \{u_t | u_t \in \mathbb{R}^2, -4 < u_{i,t} < 2, i = 1, 2\}$.

The settings for the second system are similar to those in Example 1. The parameters in the utility function are chosen as $Q = I_3$ and $R = 0.01I_2$, and the forgetting factor is set as $\gamma = 0.95$. Accord-

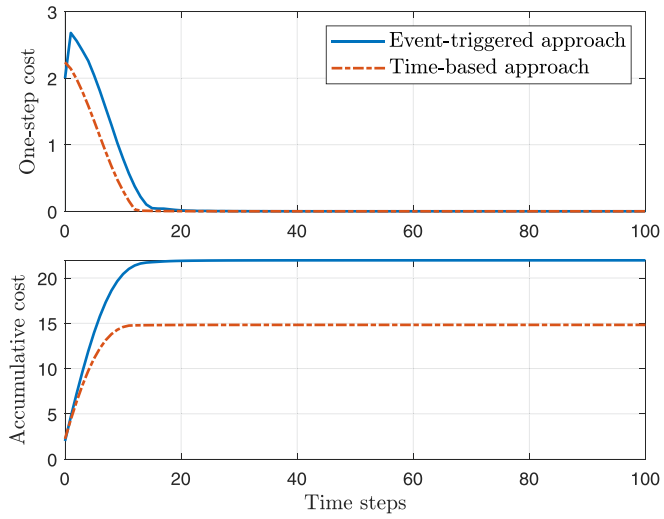

Fig. 8. Evolution of the one-step cost and the accumulative cost in the online learning process for Example 1.

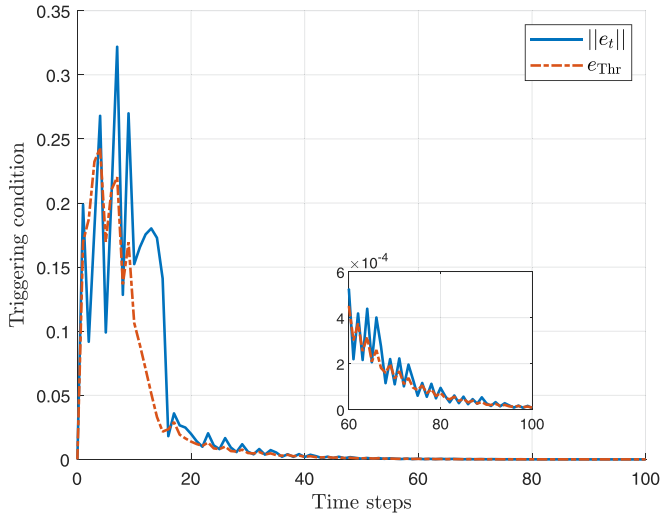

Fig. 9. Evolution of the triggering condition in the online learning process for Example 1.

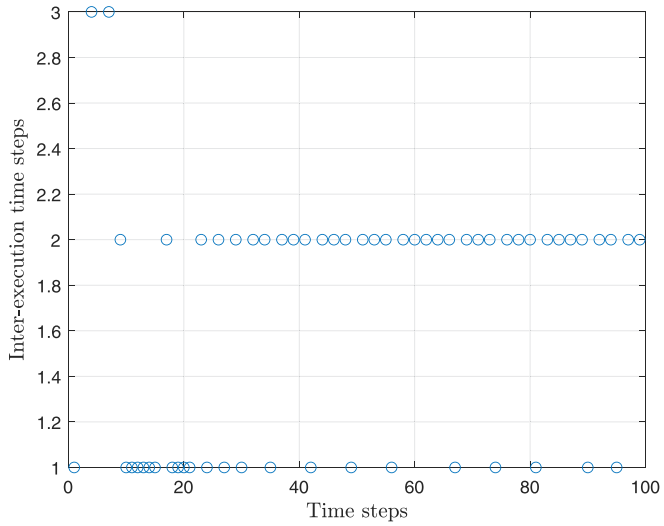

Fig. 10. Evolution of the inter-execution time in the online learning process for Example 1.

ing to the dimensions of x_t and u_t, the model network is established with the structure of 5-10-3 while both of the critic and actor networks are built as 3-10-2, i.e., $l_m = l_c = l_a = 10$. The weights of all the three networks are initialized within $[-0.1, 0.1]$. The initial weights of the actor and the critic networks utilized to present results are provided in AppendixA. Letting $\eta_m = 0.01$, we train the model network for 1000 times using 1000 data samples and examine its performance on a testing data set of another 500 samples. From Fig. 11, it is evidently observed that the mean sum of squares of the identification errors has been decreased to less than 6×10^{-4}, which indicates the high accuracy of identification.

For the implementation of the XGDHP technique, we set $\beta = 0.5$ and $b_{\Delta J} = 5 \times 10^{-6}$, and train the critic and actor networks for at most 1000 steps with the learning rates of $\eta_c = \eta_a = 0.001$ at the triggered instant. The triggering threshold is obtained by setting $C = 0.15$ as:

$$e_{\text{Thr}} = 0.15 \cdot \frac{1 - (0.3)^{t - s_k}}{1 - 0.3} ||x_{s_k}||. \tag{62}$$

By initializing the system state as $x_0 = [0.5, 0.5, 0.5]^{\mathrm{T}}$, we carry out the online control simulation to verify the performance of the proposed event-triggered XGDHP algorithm. The state trajectories of the event-triggered approach and the time-based approach are displayed in Fig. 12. Comparing the event-triggered and time-based approaches, we can observe that, although the event-triggered XGDHP algorithm involves fewer calculations, the state eventually converges to the equilibrium point without obviously deteriorating the converge rate. The control inputs are bounded within $[-4, 2]$, whose curves are depicted in Fig. 13. As depicted in Fig. 14, the weights of both critic and actor networks are initialized randomly and updated as the controller works, and all weights eventually converge to constant values.

As to the optimal control performance, the event-triggered XGDHP takes 10 more steps to keep the one-step cost below 0.1 and 7 more steps below 0.01. Different from that in Example 1, although the time based approach converges faster, the accumulative cost of the event-triggered XGDHP is less than the time-based approach. This phenomenon is because the second example requires more oscillations before be stabilized. Since the time-based approach exerts control in each time step, due to the near-optimal property, it shows more aggressive strategies and leads to larger accumulative cost, as illustrated in Fig. 15. Remarkably,

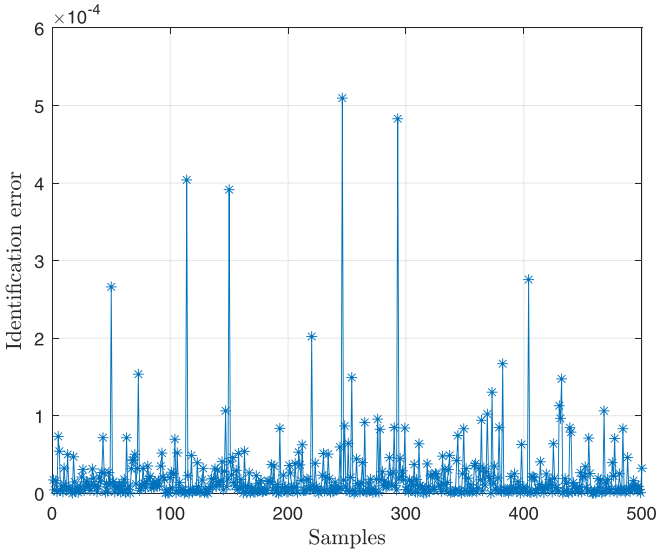

Fig. 11. The mean sum of squares of the identification errors for Example 2.

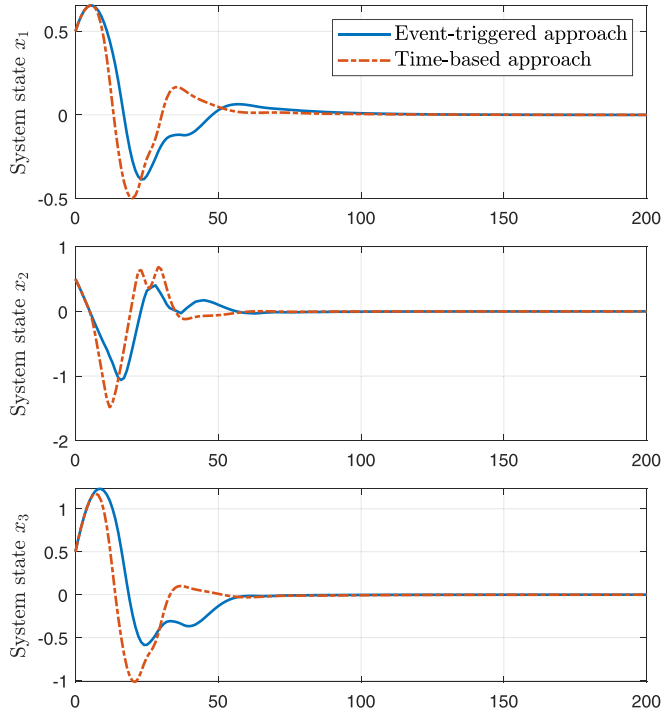

Fig. 12. Evolution of the system state in the online learning process for Example 2. Controllers aim at stabilizing system states initially from $x_0 = [0.5, 0.5, 0.5]^T$ to 0.

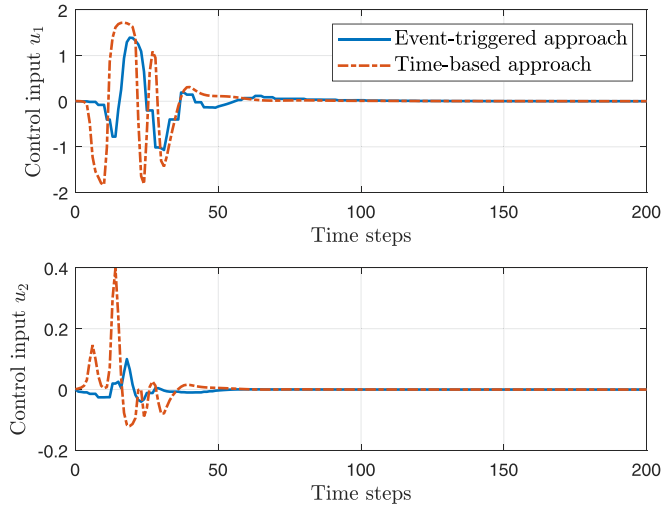

Fig. 13. Evolution of the control input in the online learning process for Example 2.

the control input is only updated 64 times in a total of 200 simulation steps with the event-triggered approach, saving up to 68% of computational load, which improves the resource utilization.

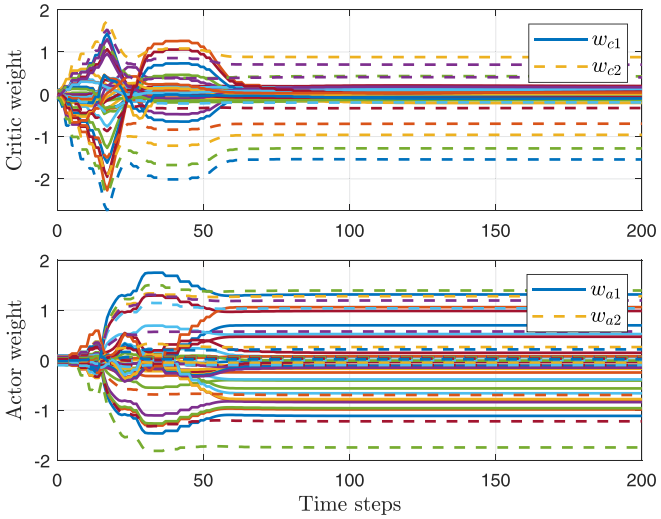

Fig. 14. Evolution of the weights of ANNs in the online earning process for Example 2. Subscripts c and a denote the critic and the actor networks, respectively. Subscripts 1 and 2 denote the weights between the input and the hidden layers and the weights between the hidden and output layers, respectively.

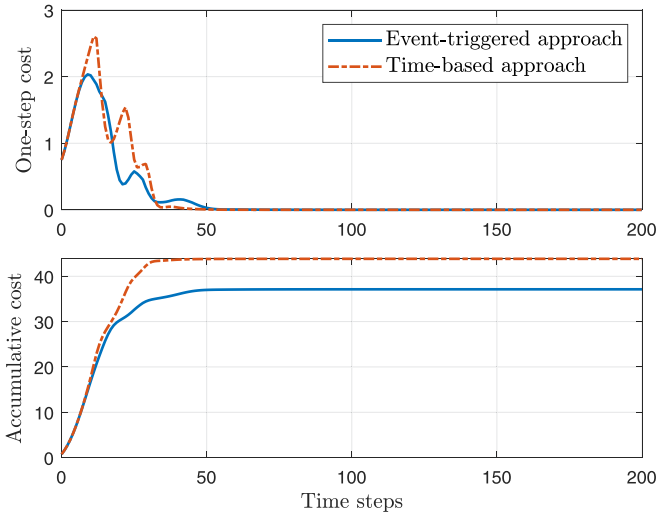

Fig. 15. Evolution of the one-step cost and the accumulative cost in the online learning process for Example 2.

The evolution curves of triggering threshold and the inter-execution time are illustrated in Figs. 16 and 17, respectively. All the simulation results uniformly verify the effectiveness of the event-triggered XGDHP control algorithm proposed in this paper.

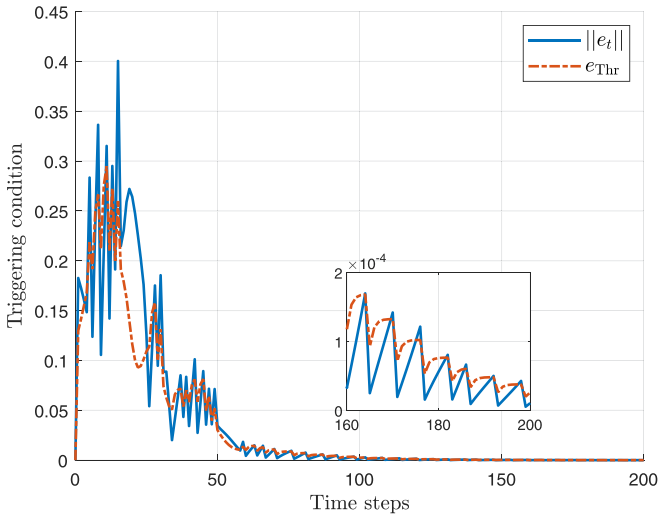

Fig. 16. Evolution of the triggering condition in the online learning process for Example 2.

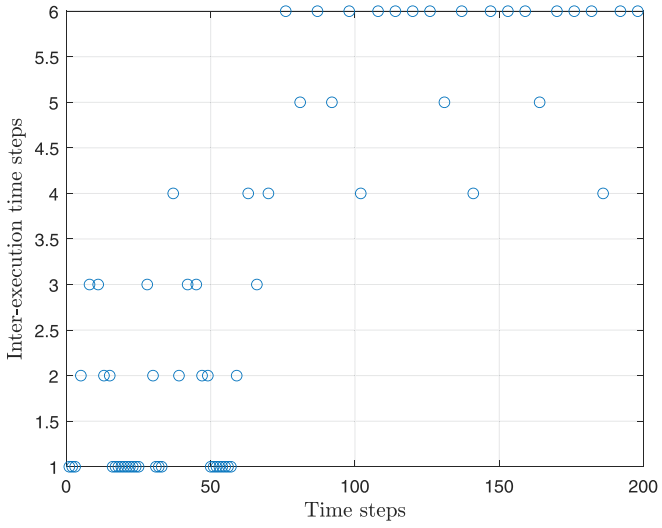

Fig. 17. Evolution of the inter-execution time in the online learning process for Example 2.

6. Conclusion

In this paper, we develop an event-triggered optimal control algorithm that can deal with asymmetric input constraints for unknown nonlinear discrete-time systems. The stability of the event-triggered control system is analyzed based on the triggering condition with fewer assumptions than existing literature. Besides, the asymmetric input constraints are coped with by the combination of a piece-wise integral function and a bounding layer of the actor network. In addition, with the facilitation of artificial neural networks, the explainable global dual heuristic programming (XGDHP) algorithm is developed to online solve the nonlinear optimal control problem, and the calculations for the derivative of the cost function are simplified without matrix dimensionality transformations.

Two numerical studies are included to illustrate the feasibility and effectiveness of the proposed method. The experimental results present that the nonlinear system can successfully be stabilized with the asymmetric input constraints handled. Furthermore, compared to the conventional time-based approach, the developed event-triggered approach can stabilize these nonlinear systems with at most 10 time steps delayed, while significantly reducing computational burden up to 40% and 68%, respectively. The communication's load between the controller and the plant can also be saved. The results collectively demonstrate the applicability of the proposed approach.

This paper utilizes a triggering condition that is derived based on the state feedback scheme. However, in many practical systems, full-state feedback is infeasible. Therefore, further investigation into output-feedback control approaches is highly recommended.

CRediT authorship contribution statement

Bo Sun: Conceptualization, Methodology, Software, Validation, Investigation, Writing - original draft, Visualization, Funding acquisition. **Erik-Jan van Kampen:** Investigation, Resources, Writing - review & editing, Supervision, Project administration.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Initial Weights

The initial weights for Example 1 are as follows:

$$w_{a1} = \begin{bmatrix} 0.3575, 0.4863, 0.3110, 0.4121, -0.4462, \\ -0.8057, 0.3897, 0.9004 \\ 0.5155, -0.2155, -0.6576, -0.9363, -0.9077, \\ 0.6469, -0.3658, -0.9311 \end{bmatrix},$$

$$w_{a2} = [-0.0012, -0.0024, 0.0053, 0.0059, -0.0063, -0.0002, -0.0011, 0.0029]^{\mathsf{T}},$$

$$w_{c1} = \begin{bmatrix} 0.6294, -0.7460, 0.2647, -0.4430, 0.9150, -0.6848, 0.9143, 0.6006 \\ 0.8116, 0.8268, -0.8049, 0.0938, 0.9298, 0.9412, -0.0292, -0.7162 \end{bmatrix},$$

$$w_{c2} = [-0.0016, -0.0083, 0.0058, 0.0092, 0.0031, -0.0093, -0.0070, 0.0087]^{\mathsf{T}}.$$

The initial weights for Example 2 are as follows:

$$w_{a1} = \begin{bmatrix} -0.0154, -0.0058, 0.0277, -0.0361, -0.0185, 0.0937, \\ -0.0789, -0.0153, -0.0693, 0.0054 \\ -0.0812, 0.0392, -0.0933, 0.0062, 0.0640, 0.0063, 0.0222, \\ -0.0818, -0.0438, -0.0085 \\ 0.0197, 0.0400, -0.0862, 0.0309, 0.0437, -0.0350, 0.0058, \\ -0.0467, -0.0120, 0.0751 \end{bmatrix},$$

$$w_{a2} = \begin{bmatrix} 0.0036, 0.0887, 0.0275, 0.0915, -0.0519, 0.0352, \\ -0.0422, 0.0344, 0.0390, -0.0864 \\ -0.0490, -0.0552, 0.0336, 0.0689, -0.0311, 0.0561, 0.0351, \\ -0.0987, 0.0204, -0.0226 \end{bmatrix}^{\mathsf{T}},$$

$$w_{c1} = \begin{bmatrix} -0.0935, 0.0338, -0.0079, 0.0711, -0.0618, -0.0759, \\ -0.0231, -0.0419, 0.0649, -0.0312 \\ 0.0122, -0.0619, 0.0963, 0.0290, -0.0143, 0.0179, \\ 0.0166, 0.0234, 0.0965, 0.0168 \\ 0.0764, -0.0262, -0.0687, -0.0247, -0.0036, \\ -0.0548, -0.0496, -0.0469, 0.0460, -0.0784 \end{bmatrix},$$

$$w_{c2} = [0.0813, 0.0759, 0.0636, -0.0479, 0.0189, -0.0955, -0.0149, -0.0375, -0.0677, -0.0642]^{\mathsf{T}}.$$

References

[1] J. Zhao, J. Na, G. Gao, Adaptive dynamic programming based robust control of nonlinear systems with unmatched uncertainties, Neurocomputing 395 (2020) 56–65.

[2] D. Wang, M. Ha, J. Qiao, Self-learning optimal regulation for discrete-time nonlinear systems under event-driven formulation, IEEE Trans. Autom. Control 65 (3) (2020) 1272–1279.

[3] Q. Liu, T. Li, Q. Shan, R. Yu, X. Gao, Virtual guide automatic berthing control of marine ships based on heuristic dynamic programming iteration method, Neurocomputing 437 (2021) 289–299.
[4] B. Sun, E.-J. van Kampen, Incremental model-based global dual heuristic programming with explicit analytical calculations applied to flight control, Eng. Appl. Artif. Intell. 89 (2020) 103425.
[5] L. Kong, S. Zhang, X. Yu, Approximate optimal control for an uncertain robot based on adaptive dynamic programming, Neurocomputing 423 (2021) 308–317.
[6] G. Che, Z. Yu, Neural-network estimators based fault-tolerant tracking control for AUV via ADP with rudders faults and ocean current disturbance, Neurocomputing 411 (2020) 442–454.
[7] Y. Zhou, E.-J. van Kampen, Q.P. Chu, Incremental model based online dual heuristic programming for nonlinear adaptive control, Control Eng. Practice 73 (2018) 13–25.
[8] B. Kiumarsi, K.G. Vamvoudakis, H. Modares, F.L. Lewis, Optimal and autonomous control using reinforcement learning: a survey, IEEE Trans. Neural Networks Learn. Syst. 29 (6) (2018) 2042–2062.
[9] R.S. Sutton, A.G. Barto, Reinforcement Learning: An Introduction, 2nd., MIT Press, 2018.
[10] D. Liu, D. Wang, D. Zhao, Q. Wei, N. Jin, Neural-network-based optimal control for a class of unknown discrete-time nonlinear systems using globalized dual heuristic programming, IEEE Trans. Autom. Sci. Eng. 9 (3) (2012) 628–634.
[11] J. Yi, S. Chen, X. Zhong, W. Zhou, H. He, Event-triggered globalized dual heuristic programming and its application to networked control systems, IEEE Trans. Industr. Inf. 15 (3) (2019) 1383–1392.
[12] B. Sun, E.-J. van Kampen, Intelligent adaptive optimal control using incrementalmodel-based global dual heuristic programming subject topartial observability, Appl. Soft Comput. 103 (2021) 107153.
[13] A.B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins, et al., Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible ai, Inform. Fusion 58 (2020) 82–115.
[14] B. Sun, E.-J. van Kampen, Reinforcement-learning-based adaptive optimal flight control with output feedback and input constraints, J. Guidance Control Dyn. 44 (9) (2021) 1685–1691.
[15] M. Abu-Khalaf, F.L. Lewis, Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network HJB approach, Automatica 41 (5) (2005) 779–791.
[16] D. Liu, X. Yang, D. Wang, Q. Wei, Reinforcement-learning-based robust controller design for continuous-time uncertain nonlinear systems subject to input constraints, IEEE Trans. Cybern. 45 (7) (2015) 1372–1385.
[17] H. Zhang, Y. Luo, D. Liu, Neural-network-based near-optimal control for a class of discrete-time affine nonlinear systems with control constraints, IEEE Trans. Neural Networks 20 (9) (2009) 1490–1503.
[18] M. Ha, D. Wang, D. Liu, Event-triggered constrained control with DHP implementation for nonaffine discrete-time systems, Inf. Sci. 519 (2020) 110–123.
[19] X. Yang, Q. Wei, Adaptive critic learning for constrained optimal event-triggered control with discounted cost, IEEE Trans. Neural Networks Learn. Syst. 32 (1) (2021) 91–104.
[20] A. Sahoo, H. Xu, S. Jagannathan, Near optimal event-triggered control of nonlinear discrete-time systems using neurodynamic programming, IEEE Trans. Neural Networks Learn. Syst. 27 (9) (2016) 1801–1815.
[21] M. Dai, J. Xia, H. Xia, H. Shen, Event-triggered passive synchronization for markov jump neural networks subject to randomly occurring gain variations, Neurocomputing 331 (2019) 403–411.
[22] H. Zhang, Z. Qiu, J. Cao, M. Abdel-Aty, L. Xiong, Event-triggered synchronization for neutral-type semi-markovian neural networks with partial mode-dependent time-varying delays, IEEE Trans. Neural Networks Learn. Syst. 31 (11) (2020) 4437–4450.
[23] S. Wang, Y. Cao, T. Huang, Y. Chen, S. Wen, Event-triggered distributed control for synchronization of multiple memristive neural networks under cyber-physical attacks, Inf. Sci. 518 (2020) 361–375.
[24] J. Wang, X.-M. Zhang, Q.-L. Han, Event-triggered generalized dissipativity filtering for neural networks with time-varying delays, IEEE Trans. Neural Networks Learn. Syst. 27 (1) (2016) 77–88.
[25] S. Zhang, B. Zhao, Y. Zhang, Event-triggered control for input constrained non-affine nonlinear systems based on neuro-dynamic programming, Neurocomputing 440 (2021) 175–184.
[26] S. Xue, B. Luo, D. Liu, Y. Li, Adaptive dynamic programming based event-triggered control for unknown continuous-time nonlinear systems with input constraints, Neurocomputing 396 (2020) 191–200.
[27] R. Song, L. Liu, Event-triggered constrained robust control for partly-unknown nonlinear systems via adp, Neurocomputing 404 (2020) 294–303.
[28] L. Dong, X. Zhong, C. Sun, H. He, Adaptive event-triggered control based on heuristic dynamic programming for nonlinear discrete-time systems, IEEE Trans. Neural Networks Learn. Syst. 28 (7) (2017) 1594–1605.
[29] M. Ha, D. Wang, D. Liu, Event-triggered adaptive critic control design for discrete-time constrained nonlinear systems, IEEE Trans. Syst., Man, Cybern.: Syst. 50 (9) (2020) 3158–3168.
[30] Z. Wang, Q. Wei, D. Liu, A novel triggering condition of event-triggered control based on heuristic dynamic programming for discrete-time systems, Optim. Control Appl. Methods 39 (4) (2018) 1467–1478.
[31] B. Kiumarsi, F.L. Lewis, Actor–critic-based optimal tracking for partially unknown nonlinear discrete-time systems, IEEE Trans. Neural Networks Learn. Syst. 26 (1) (2015) 140–151.
[32] L. Liu, Z. Wang, H. Zhang, Neural-network-based robust optimal tracking control for MIMO discrete-time systems with unknown uncertainty using adaptive critic design, IEEE Trans. Neural Networks Learn. Syst. 29 (4) (2018) 1239–1251.
[33] H. Modares, F.L. Lewis, M.-B. Naghibi-Sistani, Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems, Automatica 50 (1) (2014) 193–202.
[34] A. Al-Tamimi, F.L. Lewis, M. Abu-Khalaf, Discrete-time nonlinear HJB solution using approximate dynamic programming: Convergence proof, IEEE Trans. Syst., Man, Cybern. Part B (Cybernetics) 38 (4) (2008) 943–949.
[35] Q. Wei, D. Liu, H. Lin, Value iteration adaptive dynamic programming for optimal control of discrete-time nonlinear systems, IEEE Trans. Cybern. 46 (3) (2016) 840–853.
[36] T. Dierks, B.T. Thumati, S. Jagannathan, Optimal control of unknown affine nonlinear discrete-time systems using offline-trained neural networks with proof of convergence, Neural Networks 22 (5–6) (2009) 851–860.

Bo Sun received the B.S. degree and the M.S. degree in aerospace engineering from Northwestern Polytechnical University, Xi'an, China, 2016 and 2019, respectively. He is currently pursuing the Ph.D. degree with the Department of Control and Operations, Delft University of Technology, Delft, Netherlands. His current research interests include adaptive dynamic programming, reinforcement learning, and aerospace engineering.

Erik-Jan van Kampen received the B.Sc. degree in aerospace engineering, the M.Sc. degree in control and simulation, and Ph.D. degree in aerospace engineering from the Delft University of Technology, Delft, the Netherlands, in 2004, 2006, and 2010, respectively. He is currently an Assistant Professor with the Control and Simulation Division, Delft University of Technology. His current research interests include intelligent flight control, adaptive control, and interval optimization.