Neuro-Optimal Event-Triggered Impulsive Control for Stochastic Systems via ADP

Mingming Liang, *Member, IEEE*, and Derong Liu, *Fellow, IEEE*

***Abstract*— This article presents a novel neural-network-based optimal event-triggered impulsive control method. First, a novel general-event-based impulsive transition matrix (GITM) is constructed to represent the probability distribution evolving characteristics regarding all system states across the impulsive actions, rather than the prefixed timing sequence. On the foundation of this GITM, the event-triggered impulsive adaptive dynamic programming (ETIADP) algorithm and its high-efficiency version (HEIADP) are developed to deal with the optimization problems for stochastic systems with event-triggered impulsive controls. It is shown that the obtained controller design scheme can reduce the computational and communication burden caused by updating the controller periodically. By analyzing the admissibility, monotonicity, and optimality properties of ETIADP and HEIADP, we further establish the approximation error bound of the neural networks to address the connection between the ideal and neural-network-based realizations of the present methods. It is proven that the iterative value functions of both the ETIADP and HEIADP algorithms fall in a small neighborhood of the optimum as the iteration index increases to infinity. By adopting a novel task synchronization mechanism, the proposed HEIADP algorithm fully utilizes the computing resources of multiprocessor systems (MPSs), while significantly reducing the memory requirement compared to traditional ADP approaches. Finally, we carry out a numerical study to show that the proposed methods can fulfill the desired goals.**

***Index Terms*— Adaptive dynamic programming (ADP), event-triggering, neural networks, optimal control, value iteration.**

I. Introduction

ADAPTIVE dynamic programming (ADP) methodology along with its relevant value/policy iteration schemes are effective dealing with the optimization problems of complicated and high dimensional systems, therefore being widely investigated [1], [2], [3], [4], [5], [6], [7], [8], [9], [10]. In [11], a value iteration ADP algorithm is developed to obtain the optimum for the infinite horizon undiscounted discrete-time nonlinear systems, which permits the initializer to be an arbitrary positive semidefinite function, thus improving the universality and reliability of the ADP methods. In [12] and [13], the local value iteration ADP algorithm with a novel updating mechanism is introduced to relax the computational burden while its admissibility and global optimality properties are analyzed systematically. In [14], a novel value iteration-based off-policy ADP algorithm is proposed for the optimal control of continuous-time linear periodic systems, so that approximate optimal solutions can be obtained directly from the collected data, without the exact knowledge of system dynamics. Bhattacharya et al. [15] focus on DP problems and use policy iteration and rollout techniques to solve a class of autonomous sequential repair problems where the system states are partially observable. The stable value iteration algorithm is suggested in [16] to resolve optimization problems for nonlinear two-player zero-sum games based on ADP. It is also shown in [16] that if the iteration index reaches a given number, the generated iterative control inputs make the closed-loop system asymptotic stable. Zhu and Zhao [17] use the double Q-learning technique and provide the double-loop iterative realization methods for the ADP algorithms, to construct the optimal controllers for zero-sum games of stochastic systems. ADP also possesses great practical values. For example, it can be used to obtain the optimal controllers for permanent magnet synchronous motors [18], antilock brake systems [19], dc–dc power converters [20], etc.

Researchers in recent years have been aiming at building the high-performance impulsive controllers for the impulsively controlled systems. Haddad et al. [21] extend the dissipativity theory to nonlinear dynamical systems controlled by impulsive controllers for which the corresponding invariant set stability and Lyapunov theorems are established. Li et al. [22] design the impulsive control schemes for nonlinear systems with constant, unbounded time-varying or bounded time-varying delays. Lakshmikantham et al. [23] present the concept of impulsive evolution processes and analyze the corresponding stability properties of the impulsive controllers via methods of discontinuous Lyapunov functions. Dufour et al. [24] analyze the Hamilton–Jacobi–Bellman (HJB) equation associated with the optimal impulsive control problems of piecewise deterministic Markov processes (PDMPs). Miller et al. [25] develop the martingale representation of the stochastic systems subject to joint impulsive and gradual controls, while constructing the optimal strategy based on the DP equation. Basu and Stettner [26] provide the optimal impulsive controller design schemes for zero-sum games under several weak assumptions and weak Feller conditions. Dufour and Piunovskiy [27] study continuous-time stochastic systems on a general Borel state space governed by both impulsive and regular controllers to

Manuscript received 25 September 2022; accepted 18 December 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 62203120 and Grant 62073085 and in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2021A1515110870. *(Corresponding author: Derong Liu.)*

Mingming Liang is with the Auto Engineering Research Institute, BYD Auto Industry Company Limited, Shenzhen 518118, China (e-mail: liang.mingming@byd.com).

Derong Liu is with the Department of Mechanical and Energy Engineering, Southern University of Science and Technology, Shenzhen 518055, China, and also with the Department of Electrical and Computer Engineering, University of Illinois Chicago, Chicago, IL 60607 USA (e-mail: liudr@sustech.edu.cn; derong@uic.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2022.3232635.

Digital Object Identifier 10.1109/TNNLS.2022.3232635

minimize the infinite-time horizon discounted cost. Heydari [28] focuses on nonlinear impulsive systems and presents the controller design schemes for obtaining the impulsive instants in the optimal control problems with unlimited number of impulses. Wei et al. [29] develop a novel iterative ADP algorithm to solve the optimal impulsive control problems for infinite discrete-time nonlinear systems. Wang and Balakrishnan [30] give the optimal neuro-controller synthesis for impulse-driven systems while using neural network approximation structures to solve the optimality equations. Wang et al. [31] extend the ideas in [30] to consider the optimal impulsive control problems where the impulsive instants are calculated by a prefixed function. The optimal impulsive control technologies have also been utilized to practical applications such as the Internet congestion control problems [32], antiangiogenic tumor therapies [33], and human immunodeficiency virus treatments [34].

Event-triggered control, as a promising methodology for reducing computational and communication costs, nowadays becomes a hot topic in the community. Vamvoudakis [35] proposes the event-triggered ADP algorithm for nonlinear continuous-time systems, which reduces the controller updates by sampling the state only when an event is triggered while maintaining stability and optimality. Wang et al. [36] construct an event-triggered adaptive robust control approach for nonlinear systems through a neural DP strategy, achieving the robustness of the designed controller under a suitable triggering condition. Luo et al. [37] design the event-triggered optimal controller directly based on the solution of the Hamilton–Jacobi–Bellman equation and provide formal performance guarantees of the controller by proving a predetermined upper bound. Mu et al. [38] present an event-sampled integral reinforcement learning algorithm for partially unknown nonlinear systems using a novel dynamic event-triggering strategy, in which the actor network adopts the event-based communication to update the controller only at triggering instants. Zhao and Liu [39] develop an event-triggered decentralized tracking control approach for modular reconfigurable robots based on ADP, while solving the local HJB equation via a local critic with an asymptotically stable structure. Xue et al. [40], [41] use event-triggered ADP to design controllers for uncertain nonlinear dynamical systems.

Although the impulsive or event-triggered control technologies have made considerable progress, several problems still have to be addressed. First, the regular time-based state transition probability matrix P (its (m, n)th entry is the probability $p(\sigma_n|\sigma_m, \nu)$ where σ_m, σ_n is the discrete elements of the state space X and ν is the controller) is strictly constrained by the action time of controllers. Specifically, to reflect the probability distribution evolving characteristics from the timing node k to $k + \Delta$, from P we can derive the multistep transition matrix $P^{\Delta} = \underbrace{P * P * \cdots * P}_{\Delta}$ whose element $p_{\Delta}(\sigma_n|\sigma_m, \nu)$ establishes the relationship between the system states $x(k)$ and $x(k + \Delta)$ (we use the symbol $x \in X$ to generally refer to the system state of the stochastic systems). However, the entire transition matrix P^{Δ} requires that the action time of ν must be fixed at the time sequence $(k, k + 1, \ldots, k + \Delta - 1)$ with a prefixed time span Δ for all initial states $x(k) \in X$. This restriction of the regular time-based transition matrices (P and P^{Δ}) makes them fail to reveal the impulsive transition dynamics across the impulsive actions, since the time span between two adjoining impulsive actions (which is also called the "impulsive control cycle") is adaptively changing according to the current system state and not predefined. Consequently, the regular time-based transition matrices (P and P^{Δ}) can be used to derive the evolution of the probability distribution at the prefixed time sequence, but not the evolution of the probability distribution across the impulsive actions. In summary, these facts indicate that the restrictions of the regular time-based transition matrices and the variable impulsive control cycles of the impulsive controller show no compatibility and conflict with each other, which causes the traditional impulsive control methods [24], [25], [26], [27] complicated and highly specialized with low generality and uniformity. Noticing that the "arrival of the impulsive action" can be treated as an event, therefore to address the above issues, a novel general-event-based impulsive transition matrix is needed to represent the probability distribution evolving characteristics across the impulsive actions.

Second, the impulsive control methods proposed in [24], [25], [26], and [27] use DP to iteratively obtain the explicit solutions of the optimality equations. However, due to the increased difficulty of solving the exact solutions and the "curse of dimensionality" issues of DP (especially when the system dynamic characteristics are complex), these DP-based methods are almost impossible to implement in reality. Fortunately, the above DP-associated problems can be successfully avoided by the newly developed ADP methodology, which uses approximate structures to efficiently and numerically approach the optimum. Nevertheless, there are still no reports on utilizing ADP to solve the optimal impulsive control problems for discrete stochastic systems. Besides, to derive the optimal impulsive controllers for the stochastic systems under the ADP framework, the new general-event-based impulsive transition matrix which can reflect the impulsive transition dynamics across the impulsive actions, is crucial and required.

Furthermore, existing researches on impulsive control via ADP [28], [29], [30], [31]: 1) merely deal with the deterministic systems, and are invalid when the controlled systems are stochastic as a consequence of the issues mentioned above and 2) require the system states to be sampled periodically and the controller/actuator be updated at each time step, consuming huge computational and communication resources. As for the existing event-triggered ADP approaches [35], [36], [37], [38], [39], [40], [41], they: 1) mainly focus on deterministic cases and there are no reports on how they can be applied to impulsively controlled systems and 2) require the triggering condition to be carefully designed by the decision maker prior to the optimization of the controller, thus characterized by low portability and extendability, which also indicates that the optimality of the triggering condition needs improvement.

Finally, on the subject of algorithm execution, the traditional ADP-based approaches including [1], [2], [3], [4], [5], [6], [7], [8], [28], [29], [30], [31], [35], [36], [37], [38], [39] require the iterative items to be updated globally at each step. This updating mechanism may not be friendly to the computing devices especially with limited memory sizes. Particularly, if the complexity of the controlled systems increases, ADP needs massive amount of sampled data to globally and precisely train the utilized neural networks at each iteration, which poses a significant memory burden to the computing devices and may even cause the algorithms unfeasible. On the other hand, more and more modern computing devices come with multiple processors, which are referred to as multiprocessor

systems (MPSs), now widespread and playing critical roles in various areas of daily life and industrial production. Hence, considering the fact that traditional ADP algorithms are usually designed to run on a single processor, how to adapt ADP to fully utilize the computing resources of MPSs remains an urgent problem to be resolved.

The above issues motivate us to carry out the present research, and we summarize the main novelties in the following three aspects.

1) A novel general-event-based impulsive transition matrix (GITM) is established, which can reveal the probability distribution evolving characteristics across the predefined general events. Due to this advanced property, this novel transition matrix plays a key role in the developments of the ETIADP and HEIADP methods. Moreover, the GITM possesses strong extendability. Specifically, the "general event" of GITM is not only limited to "the arrival of the impulsive instants," but also can be further customized into "the jumping behavior of the system happens," "the switching behavior of the controller happens" or "the triggering condition is violated," etc. Hence, these unique features of GITM substantially improve the universality of the ADP-based approaches.

2) A new event-triggered impulsive controller (ETIC) design scheme is developed and for the first time ADP is applied to obtain the optimal ETICs for stochastic systems, whereas value iteration is employed to address the "curse of dimensionality" issues and effectively approximate the optimums including the optimal triggering condition.

3) The high-efficiency event-triggered impulsive ADP (HEIADP) algorithm is developed to fully utilize the computing resources of MPSs while reducing its memory requirement.

The advantages and differences of the developed methods compared to the existing methods are emphasized as follows.

1) Existing impulsive control methods use the traditional state transition probability matrices which require that the action time of the impulsive controller must have the same length for each system state. Unfortunately, for the impulsive controller, its impulsive control cycles are always adaptively changing with the system states and not fixed. This mismatch between the variable impulsive control cycles and the requirements of the regular time-triggered transition matrices causes the traditional impulsive control methods complicated and highly specialized with low generality and uniformity. In comparison, the proposed ETIADP and HEIADP algorithms are based on the GITMs which reflect the "impulsive" dynamics, possess greater flexibility and improve the extendability of the ADP approaches. (Please see Remark 1 for more details about the advantages of the GITM over the traditional one.)

2) Existing DP based impulsive control methods may cause the "curse of dimensionality" problems, making the algorithms unusable in reality. In comparison, the proposed methods are ADP based, which means neural networks are used to effectively approximate the optimums, thus successfully avoiding the DP-related issues. In addition, to the best of our knowledge, it is the first time ADP is utilized to design the optimal impulsive controllers for stochastic systems.

3) Existing impulsive control methods for deterministic systems cannot be directly extended to stochastic systems, offering limited generality. In comparison, the proposed methods can effectively solve the optimal impulsive control problems of stochastic systems, due to the established GITMs.

4) Existing impulsive control methods require the system states to be sampled periodically and the controller/actuator to be updated at each time step, consuming huge computational and communication resources. In comparison, the event-triggering mechanism is introduced to the optimal impulsive controller design scheme for the first time, further improving its efficiency. In other words, according to our literature research, there has been no reports on how to obtain the optimal event-triggered controllers for impulsive stochastic systems, which is another contribution of our article.

5) Existing ADP methods, when implemented in computing devices, consume huge memory spaces if the system complexity is high. In comparison, the proposed HEIADP algorithm can significantly reduce the memory burden.

6) Existing ADP methods, when executed in MPSs, are faced with the "task unsynchronization" problem, resulting in low utilization rate of the multiprocessor computing resources. In comparison, the proposed HEIADP algorithm can overcome the above issue by adopting a novel MPS task scheduling scheme.

II. Problem Statement

A. System Dynamics and the ETIC Design

The controlled systems are modeled as follows:

$$x(k+1) = \mathcal{F}(x(k), a(k), \omega(k)) \tag{1}$$

where x and a are the system state and control action, respectively. $\omega(k)$ is the random variable which has its own probability distribution. In turn, the global state space and control action space are denoted by X and A, respectively. If the current system state is $x(k)$ with the control action being $a(k)$, then, the value of the following state $x(k+1)$ is governed by a probability distribution $p(\cdot|x(k), a(k))$. The equilibrium of (1) is $x = \mathbf{0}$ when the control action is zero, meaning $p(\mathbf{0}|\mathbf{0}, \mathbf{0}) = 1$. The timing instants where the event-triggered impulsive controller is active are of two types: the decision-making instants and impulsive control instants. If the current time is the decision-making instant, then the ETIC determines whether the current "controller updating event (CUE)" is triggered or not, and specifies when the succeeding impulsive action takes place. If the current time is the impulsive control instant, the ETIC applies the impulsive action to the system. In particular, we use $\{\theta_l\}, l = 0, 1, \ldots,$ to represent the sequence of the decision-making instants. When $l = 0$, we have $\theta_0 = 0$. The symbol $\mathcal{G}$ is used to denote the collection of all possible impulsive intervals (the time length from the current decision-making (impulsive control) instant to the following one), i.e., $\mathcal{G} = \{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_{\max}\}$. Obviously, we have $\theta_{l+1} - \theta_l \in \mathcal{G}, l = 0, 1, \ldots$

In practical engineering, the frequency of the impulsive actions of the ETIC should be regulated. Specifically, if the frequency is too high, it may cause wear and tear of the controller and affect system reliability; If the frequency is too low, it may not achieve the desired controller performance. Hence, to acquire a proper balance between controller performance and reliability, a delay function $\psi(x, l)\colon X \times \mathbb{N}_{\geq 0} \to \mathcal{G}$ which is used to adjust the frequency of the impulsive actions, is designed. If the current time satisfies $k = \theta_l$ which is exactly the decision-making instant and the current CUE is triggered, then based on the current system state $x(\theta_l)$, ψ outputs the current "impulsive" control cycle (i.e., the time span bounded by the current decision-making instant θ_l and the next one θ_{l+1}) of the ETIC. In other words, we have $\theta_{l+1} - \theta_l = \psi(x(\theta_l), l)$. Meanwhile, under the above conditions, the control law cluster $\mathbf{u}(x, l) = [u(x, \mathcal{T}_1, l), u(x, \mathcal{T}_2, l), \ldots, u(x, \mathcal{T}_{\max}, l)]$ assigns $u(x(\theta_l), \psi(x(\theta_l), l), l)$ as the impulsive magnitude of the succeeding impulsive action which is applied at the impulsive

TABLE I
REGISTER DYNAMICS

case \ value	$r(k+1)$
$r(k)\neq 0$	$r(k)-1$
$r(k)=0, \varkappa(\varsigma(k),c(k))=1$	$\psi\big(x(k),c(k)\big)-1$
$r(k)=0, \varkappa(\varsigma(k),c(k))=0$	$\hat{r}'(k)-1$

TABLE II
RECORDER DYNAMICS

case \ value	$\hat{r}'(k+1)$	$\hat{r}''(k+1)$
$r(k)=0,$ $\varkappa(\varsigma(k),c(k))=1$	$\psi\big(x(k),c(k)\big)$	$u\big(x(k),\psi(x(k),c(k)),c(k)\big)$
otherwise	$\hat{r}'(k)$	$\hat{r}''(k)$

TABLE III
COUNTER DYNAMICS

case \ value	$c(k+1)$
$r(k)=0, \varkappa(\varsigma(k),c(k))=1, \psi\big(x(k),c(k)\big)=1$	$c(k)+1$
$r(k)=0, \varkappa(\varsigma(k),c(k))=0, \hat{r}'(k)=1$	
$r(k)=1$	
otherwise	$c(k)$

control instant, i.e., $\theta_{l+1}-1$, of the current impulsive control cycle (as its name suggests, the control law cluster $\mathbf{u}(x,l)$ is composed of a series of control laws which are defined by $u(x,\rho,l)\colon X\times\mathcal{G}\times\mathbb{N}_{\geq 0}\to A$). On the other hand, if at the current time $k=\theta_l$, the CUE is not triggered, then, the time span and the impulsive action of the current impulsive control cycle are kept the same as that of the last "triggered" one. In addition, if the current time index is not the impulsive control instant, then, the ETIC is in idle state and there is no control input, whether the current CUE is triggered or not.

We set up a register $r(k)\in\mathfrak{R}$ along with the counter $c(k)$ in the ETIC. Specifically, these two components $r(k)$ and $c(k)$ are used to signal the arrival of the decision-making or impulsive control instant to the controller. Moreover, construct the recorder $\hat{r}=[\hat{r}',\hat{r}'']^{\mathsf{T}}\in\hat{\mathfrak{R}}$, where $\hat{r}'$ and $\hat{r}''$ record the outputs of $\psi(x,l)$ and $\mathbf{u}(x,l)$ at the triggered impulsive control cycle, respectively. Then, we can define the "impulsive" state ς as

$$\varsigma=\begin{bmatrix}x\\ \hat{r}\end{bmatrix}\in\hat{X}=X\times\hat{\mathfrak{R}}.$$

Let the function $\varkappa(\varsigma,l)\colon\hat{X}\times\mathbb{N}_{\geq 0}\to\{0,1\}$ denote the CUE scheduling/triggering strategy, where $\varkappa(\varsigma(\theta_l),l)=1$ means the CUE is triggered or equivalently, the time span and the impulsive action of the current impulsive control cycle are both updated, and $\varkappa(\varsigma(\theta_l),l)=0$ means the previously calculated cycle length and impulsive action are applied to the current "impulsive control cycle" (CUE is not triggered, and the current control cycle inherits the attributes of the last triggered one). Through monitoring the system state trajectory $x(k)$, the components r, $\hat{r}$ and c renew their values accordingly. Suppose the current system state is $x(k)$, the updating rules of these items are demonstrated in Tables I–III, with the corresponding initial values configured as $r(0)=0,\hat{r}(0)=[\mathcal{T}_1,0]^{\mathsf{T}}$ and $c(0)=0$.

From Table I, if $r(k)=0$, then, ETIC treats k as the current decision-making instant, i.e., $k=\theta_l$. When $r(k)=1$, ETIC then recognizes k as the current impulsive control instant. As for the recorder $\hat{r}(k)$, the corresponding dynamics is shown in Table II, where $\hat{r}$ memorizes the cycle length of the current "triggered" impulsive control cycle, along with the impulsive action applied in the above period. In turn, if the current CUE is not triggered, $\hat{r}$ remains unchanged. Table III demonstrates the dynamics of $c(k)$, which records the total number of impulsive actions applied to the system before the current time k. Therefore, we can derive $\theta_{c(k)}\leq k\leq\theta_{c(k)+1}-1$, meaning that $\varkappa(\varsigma,c(k))$, $\psi(x,c(k))$ and $\mathbf{u}(x,c(k))$ are functioning in the ongoing impulsive control cycle $[\theta_{c(k)},\theta_{c(k)+1})$.

The collection of all accessible values of the register r is denoted by $\mathfrak{R}$ which consists of $r_0,r_1,\ldots,r_{|\mathfrak{R}|}$, with $r_0=0$. We use

$$e=e_{r,\varsigma}=\begin{bmatrix}r\\ \varsigma\end{bmatrix}\in\mathcal{E}=\mathfrak{R}\times\hat{X}$$

to represent the "expanded" system state. As for the global space $\hat{X}$ of all the "impulsive" states, it is discrete and countable and we list its elements as $\sigma_1,\sigma_2,\ldots,\sigma_{|\hat{X}|}$. Then, we use $\varpi_l(e)=[\varpi_l^{\langle 1\rangle}(e),\varpi_l^{\langle 2\rangle}(e)]^{\mathsf{T}}$ as the abstract mathematical description of the ETIC where $\varpi_l^{\langle 1\rangle}$ returns the output of the triggering strategy at the decision-making instants

$$\varpi_l^{\langle 1\rangle}(e)=\begin{cases}\varkappa(\varsigma,l), & r=0\\ 0, & \text{otherwise}\end{cases}\tag{2}$$

while $\varpi_l^{\langle 2\rangle}$ returns the control action of ETIC at the impulsive control cycle $[\theta_l,\theta_{l+1})$

$$\varpi_l^{\langle 2\rangle}(e)=\begin{cases}u(x,1,l), & r=0,\varkappa(\varsigma,l)=1,\psi(x,l)=1\\ \hat{r}'', & r=0,\varkappa(\varsigma,l)=0,\hat{r}'=1\\ \hat{r}'', & r=1\\ 0, & \text{otherwise.}\end{cases}\tag{3}$$

Equation (3) shows that the decision procedure of the ETIC is subject to the following five factors: 1) the current expanded state; 2) the total count of impulsive actions applied before the present time; 3) the delay function ψ; 4) the triggering strategy $\varkappa$; and 5) the control law cluster $\mathbf{u}$. Therefore, more accurately the ETIC ought to be marked with the symbol $\varpi_{l,\varkappa,\psi,\mathbf{u}}(e)$ instead of $\varpi_l(e)$. Nonetheless, to conveniently analyze the properties of ETIC, we still adopt ϖ_l or $\varpi_l(e)$ to represent $\varpi_{l,\varkappa,\psi,\mathbf{u}}(e)$ if no confusion occurs. Moreover, the operating mechanism of the ETIC is demonstrated intuitively in Fig. 1.

Based on the preceding analysis, we now define the event-triggered impulsive policy employed across all impulsive control cycles as $\tilde{h}=(\varkappa(\varsigma,0),\psi(x,0),\mathbf{u}(x,0),\ldots,\varkappa(\varsigma,l),\psi(x,l),\mathbf{u}(x,l),\ldots)$. In addition, we use $\mathcal{K}_l$, $\mathcal{M}_l$ and $\mathbf{C}_l$ to represent the collections of all possible triggering strategies, delay functions, and control law clusters applied at the lth impulsive control cycle, respectively. Then, the event-triggered impulsive policy space is expressed as

$$\Xi=\mathcal{K}_0\times\mathcal{M}_0\times\mathbf{C}_0\times\mathcal{K}_1\times\mathcal{M}_1\times\mathbf{C}_1\times\cdots\times\mathcal{K}_l\times\mathcal{M}_l\times\mathbf{C}_l\times\cdots$$

in which $\times$ means the Cartesian product. For the space $\mathbf{C}_l$, we have $\mathbf{C}_l=C_{\mathcal{T}_1,l}\times C_{\mathcal{T}_2,l}\times\cdots\times C_{\mathcal{T}_{\max},l}$ where $C_{\rho,l}$, $\rho\in\mathcal{G}$, represents the control law space $\{C_{\rho,l}\big|u(\cdot,\rho,l)\in C_{\rho,l}\}$.

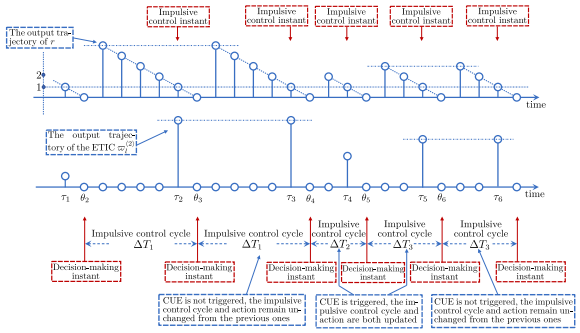


Fig. 1. Output trajectories of the ETIC and corresponding register (for illustration purpose only).

B. Development of the GITM

Suppose at the time k, the expanded state of the system is $e(k) = e_{r_i,\sigma_m}$. Under that condition, $p_\Delta(e_{r_j,\sigma_n}|e_{r_i,\sigma_m}, \varpi_l)$, $\Delta = 1, 2, \ldots,$ denotes the probability that the stochastic system goes to the state e_{r_j,σ_n} at the time index $k + \Delta$ when the ETIC ϖ_l is utilized at the time indexes k to $k + \Delta - 1$. According to the operating mechanism of ETIC, at the lth impulsive control cycle, $\varkappa(\varsigma, l)$, $\psi(x, l)$, and $\mathbf{u}(x, l)$ are employed to determine the controller outputs. Thereupon, we define the corresponding GITM as $\mathscr{P}_{\varkappa(\varsigma,l),\psi(x,l),\mathbf{u}(x,l)}$ which is as in (4), shown at the bottom of the page. Specifically, the (m, n)th component of the GITM is $\mathrm{p}(\sigma_n|\sigma_m, \varkappa(\varsigma, l), \psi(x, l), \mathbf{u}(x, l))$ which represents the probability that the impulsive state ς evolves to $\varsigma(\theta_{l+1}) = \sigma_n$ at the $(l + 1)$th decision-making instant starting from $\varsigma(\theta_l) = \sigma_m$ at the lth decision-making instant. The elements of $\mathscr{P}$ satisfy

$$\begin{aligned}&\mathrm{p}(\sigma_n|\sigma_m, \varkappa(\varsigma, l), \psi(x, l), \mathbf{u}(x, l))\\ &\quad= \sum_{j=0}^{|\Re|} p_{\varphi(\sigma_m,l)}\left(e_{r_j,\sigma_n}|e_{r_0,\sigma_m}, \varpi_l\right)\end{aligned} \tag{5}$$

where

$$\varphi(\varsigma, l) = \begin{cases} \psi(x, l), & \varkappa(\varsigma, l) = 1\\ \hat{r}', & \varkappa(\varsigma, l) = 0. \end{cases} \tag{6}$$

Remark 1: From (5) and (6), it is worth mentioning that GITM is only governed by the triggering strategy $\varkappa(\varsigma, l)$, control law cluster $\mathbf{u}(x, l)$ and delay function $\psi(x, l)$, and is not affected by the adaptively changing impulsive control cycles of the ETIC. This advantage of the GITM over the traditional one is also demonstrated intuitively in Figs. 2 and 3. In these figures, the area of the rectangle whose label and corresponding time index are σ_i and k, respectively, represents the probability of the system state being σ_i at k, while the shaded rectangles connected by the dashed lines represent the

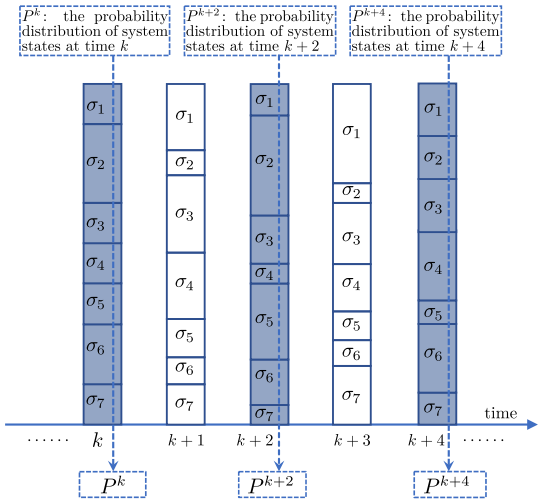


Fig. 2. Changes of the probability distribution of system states over time (for illustration purpose only).

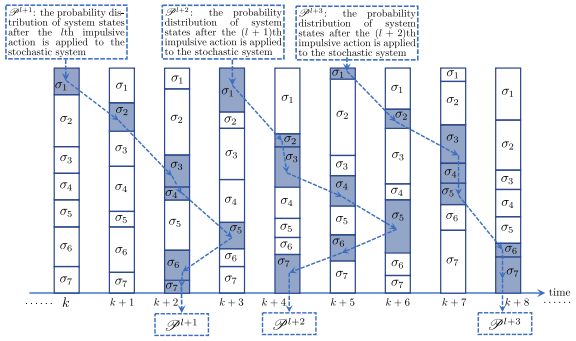


Fig. 3. Changes of the probability distribution of system states across the predefined general events (for illustration purpose only).

probability distribution of system states at a certain time point or upon the general event occurs. Specifically, Fig. 2 shows that the traditional transition matrices require the action time of the controller during system transition must be the same (e.g., 1 or 2 time units) for all current states with the same time index, in order for it to describe the changes of the probability distribution over time. Fig. 3 indicates that when a general event happens, the corresponding system states may not be at the same time (for example, after the lth impulsive action is applied, the probability of state σ_2 being at time $k + 1$ is 20%, while there is a 30% chance that the system occupies σ_3 at time $k + 2$). Fig. 3 also demonstrates the variable transition time from one general event to another, which causes the traditional matrices fail to represent the general-event-based

$$\mathscr{P}_{\varkappa(\varsigma,l),\psi(x,l),\mathbf{u}(x,l)} = \begin{bmatrix} \mathrm{p}(\sigma_1|\sigma_1, \varkappa(\varsigma, l), \psi(x, l), \mathbf{u}(x, l)) & \mathrm{p}(\sigma_2|\sigma_1, \varkappa(\varsigma, l), \psi(x, l), \mathbf{u}(x, l)) & \cdots & \mathrm{p}\left(\sigma_{|\hat{X}|}\middle|\sigma_1, \varkappa(\varsigma, l), \psi(x, l), \mathbf{u}(x, l)\right)\\ \mathrm{p}(\sigma_1|\sigma_2, \varkappa(\varsigma, l), \psi(x, l), \mathbf{u}(x, l)) & \mathrm{p}(\sigma_2|\sigma_2, \varkappa(\varsigma, l), \psi(x, l), \mathbf{u}(x, l)) & \cdots & \mathrm{p}\left(\sigma_{|\hat{X}|}\middle|\sigma_2, \varkappa(\varsigma, l), \psi(x, l), \mathbf{u}(x, l)\right)\\ \vdots & \vdots & \vdots & \vdots\\ \mathrm{p}\left(\sigma_1\middle|\sigma_{|\hat{X}|}, \varkappa(\varsigma, l), \psi(x, l), \mathbf{u}(x, l)\right) & \mathrm{p}\left(\sigma_2\middle|\sigma_{|\hat{X}|}, \varkappa(\varsigma, l), \psi(x, l), \mathbf{u}(x, l)\right) & \cdots & \mathrm{p}\left(\sigma_{|\hat{X}|}\middle|\sigma_{|\hat{X}|}, \varkappa(\varsigma, l), \psi(x, l), \mathbf{u}(x, l)\right) \end{bmatrix} \tag{4}$$

impulsive dynamics. By comparing Figs. 2 and 3, it shows that the established GITM can describe the probability distribution evolving characteristics across the "general events" while the traditional matrices cannot (from now on, the "general event" of GITM specifically refers to "the arrival of the decision-making instant"). With the above superiority of the GITM compared to the traditional transition matrices, ADP is then empowered such that it can be used in solving the optimal impulsive control problems of stochastic systems. GITM also plays a key role in analyzing the convergence, admissibility, and error boundedness properties of the ETIADP and HEIADP methods.

Choose the event-triggered impulsive policy as $\tilde{h} = (\varkappa(\varsigma, 0), \psi(x, 0), \mathbf{u}(x, 0), \ldots, \varkappa(\varsigma, l), \psi(x, l), \mathbf{u}(x, l), \ldots)$ and let $e(\theta_l) = e_{r_0,\varsigma}$. Under the above situation, the expected infinite horizon cost subjected to $\tilde{h}$ is

$$\mathcal{J}^{\tilde{h}}\left(e_{r_0,\varsigma}, l\right) = E\left\{\sum_{k=\theta_l}^{\infty} U\left(e(k), \varpi_{c(k)}(e(k))\right)\right\}. \tag{7}$$

The content within the bracket of (7) is a random variable whose expectation is calculated by the operator $E\{\cdot\}$. The utility function $U(\cdot, \cdot)$ in (7) is expressed as $U(e, \varpi_l) = Q(e, \varpi_l^{\langle 2\rangle}) + \pi(\varpi_l^{\langle 1\rangle})\varrho(e, \varpi_l^{\langle 2\rangle})$. Specifically, at each time index k, the function Q gives the immediate cost regarding the current expanded state $e(k)$ and the corresponding control action $\varpi_l^{\langle 2\rangle}$. ϱ represents the costs of resampling the current state, recalculating the outputs of the ETIC (or utilizing the computational resources) and utilizing the communication resources, when the controller updating event is triggered. The function π is expressed as

$$\pi\left(\varpi_l^{\langle 1\rangle}\right) = \begin{cases} 1, & \varpi_l^{\langle 1\rangle} = 1 \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

Designers should carefully make the choices of the functions Q and ϱ in order for the impulsive controllers and systems functioning properly and healthily. That is, the degree of the penalty (i.e., the utility function U) should be stepped up if the impulsive action frequency increases. The reason for that is that the impulsive controller/actuator can be abused or severely damaged and their lifespans may be shortened if the impulsive action frequency is too high, which therefore ought to be suppressed.

Definition 1: If $\tilde{h} = (\varkappa(\varsigma, 0), \psi(x, 0), \mathbf{u}(x, 0), \ldots, \varkappa(\varsigma, l), \psi(x, l), \mathbf{u}(x, l), \ldots)$ makes $\mathcal{J}^{\tilde{h}}(e_{r_0,\varsigma}, l) < \infty$ hold for any $\varsigma \in \hat{X}, l \in \mathbb{N}_{\geq 0}$, then we say the event-triggered impulsive policy $\tilde{h}$ or the triplet $(\varkappa(\varsigma, l), \psi(x, l), \mathbf{u}(x, l))$ is admissible.

Assumption 1: As for the control laws, we have $u(0, \rho, l) = 0, \forall \rho \in \mathcal{G}, \forall l \in \mathbb{N}_{\geq 0}$. $Q(\cdot, \cdot) \geq 0$ and $\varrho(\cdot, \cdot) \geq 0$ are both positive semi-definite functions, where

$$Q\left(e, \varpi_l^{\langle 2\rangle}\right)\begin{cases} = 0, & x = 0, \varpi_l^{\langle 2\rangle} = 0 \\ > 0, & \text{otherwise} \end{cases} \tag{9}$$

$$\varrho\left(e, \varpi_l^{\langle 2\rangle}\right)\begin{cases} = 0, & x = 0, \varpi_l^{\langle 2\rangle} = 0 \\ > 0, & \text{otherwise.} \end{cases} \tag{10}$$

The objective of this article is to obtain $\tilde{h}^* = (\varkappa^*(\varsigma, 0), \psi^*(x, 0), \mathbf{u}^*(x, 0), \ldots, \varkappa^*(\varsigma, l), \psi^*(x, l), \mathbf{u}^*(x, l), \ldots)$ so that the performance index function (7) is minimized by the optimal policy $\tilde{h}^*$. To simplify things, hereafter we concentrate attention on searching in the "stationary" policy space Ξ', whose elements, i.e., the stationary event-triggered impulsive policy $h = (\varkappa(\varsigma, 0), \psi(x, 0), \mathbf{u}(x, 0), \ldots, \varkappa(\varsigma, l), \psi(x, l), \mathbf{u}(x, l), \ldots)$, satisfy $\varkappa(\varsigma) = \varkappa(\varsigma, 0) = \cdots = \varkappa(\varsigma, l) = \cdots, \psi(x) = \psi(x, 0) = \cdots = \psi(x, l) = \cdots$, and $\mathbf{u}(x) = \mathbf{u}(x, 0) = \cdots = \mathbf{u}(x, l) = \cdots$. Therefore, the "stationary" triggering strategy, performance index function, control law cluster, delay function, ETIC and GITM are not subjected by l, and are denoted by $\varkappa(\varsigma), \mathcal{J}^h(e_{r_0,\varsigma}), \mathbf{u}(x) = [u(x, \mathcal{T}_1), \ldots, u(x, \mathcal{T}_{\max})], \psi(x), \varpi(e)$ and $\mathscr{P}_{\varkappa,\psi,\mathbf{u}}$, respectively [as a matter of fact, it can be proved that the optimal $\tilde{h}^*$ is stationary, i.e., $\tilde{h}^* = h^* = (\varkappa^*, \psi^*, \mathbf{u}^*, \varkappa^*, \psi^*, \mathbf{u}^*, \ldots)$].

In addition, define the impulsive utility function $\mathcal{U}_{\varkappa,\psi,\mathbf{u}}(\varsigma)$, impulsive value function $V^{r_0}(\varsigma)$, and impulsive performance index function $J^{h,r_0}(\varsigma)$ as $\mathcal{U}_{\varkappa,\psi,\mathbf{u}}(\cdot), V^{r_0}(\cdot), J^{h,r_0}(\cdot)\colon \hat{X} \to \mathbb{R}$, respectively. Particularly, $\mathcal{U}_{\varkappa,\psi,\mathbf{u}}(\varsigma)$ is the accumulated cost of one impulsive control cycle, during which the impulsive state at the decision-making instant equals to ς, that is,

$$\mathcal{U}_{\varkappa,\psi,\mathbf{u}}(\varsigma) = E\left\{\sum_{\kappa=\theta_l}^{\theta_l+\varphi(\varsigma)-1} U(e(\kappa), \varpi)\middle| e(\theta_l) = \begin{bmatrix} r_0 \\ \varsigma \end{bmatrix}\right\}.$$

$V^{r_0}(\varsigma)$ denotes the value of the impulsive state ς at the decision-making instant where the expanded system state is $e = \begin{bmatrix} r_0 \\ \varsigma \end{bmatrix} = \begin{bmatrix} 0 \\ \varsigma \end{bmatrix}$. Given the policy h, $J^{h,r_0}(\varsigma)$ is the corresponding performance index regarding to the current impulsive state ς with the current time being an decision-making instant. The corresponding vectorized forms $\boldsymbol{\mathcal{U}}_{\varkappa,\psi,\mathbf{u}}, \mathbf{V}^{r_0}$ and $\mathbf{J}^{h,r_0}$ are all $|\hat{X}|$-dimensional vectors whose ith elements are $\mathcal{U}_{\varkappa,\psi,\mathbf{u}}(\sigma_i), V^{r_0}(\sigma_i)$ and $J^{h,r_0}(\sigma_i)$, respectively.

III. Event-Triggered Impulsive ADP Algorithm

A. Derivation of the ETIADP Algorithm

Define the set of all "impulsive" equilibrium points as $\Omega_{\mathrm{o}} = \{\varsigma \,|x = 0, \hat{r}'' = 0, \hat{r}' \in \mathcal{G}\}$ (it is guaranteed that any admissible ETIC can drive the impulsive system into the impulsive states $\varsigma \in \Omega_{\mathrm{o}}$, and once the impulsive trajectory enters inside the space Ω_{o}, it remains there and never leaves). Let the elements of the initial value function $\mathbf{V}_0^{r_0}$ satisfy

$$V_0^{r_0}(\varsigma) = \begin{cases} 0, & \varsigma \in \Omega_{\mathrm{o}} \\ > 0, & \text{otherwise.} \end{cases} \tag{11}$$

Then, the initial triggering strategy $\varkappa_0(\varsigma)$, delay function $\psi_0(x)$ and control law cluster $\mathbf{u}_0(x)$ are calculated by

$$(\varkappa_0, \psi_0, \mathbf{u}_0) = \arg \min_{\varkappa\in\mathcal{K},\psi\in\mathcal{M},\mathbf{u}\in\mathbf{C}} \left\{\boldsymbol{\mathcal{U}}_{\varkappa,\psi,\mathbf{u}} + \mathscr{P}_{\varkappa,\psi,\mathbf{u}}\mathbf{V}_0^{r_0}\right\}$$

$\forall i = 1, 2, \ldots,$ obtain the iterative value functions $\mathbf{V}_i^{r_0}$ via

$$\mathbf{V}_i^{r_0} = \boldsymbol{\mathcal{U}}_{\varkappa_{i-1},\psi_{i-1},\mathbf{u}_{i-1}} + \mathscr{P}_{\varkappa_{i-1},\psi_{i-1},\mathbf{u}_{i-1}}\mathbf{V}_{i-1}^{r_0}. \tag{12}$$

The iterative triggering strategy $\varkappa_i(\varsigma)$, delay function $\psi_i(x)$, and control law cluster $\mathbf{u}_i(x)$ are derived through

$$(\varkappa_i, \psi_i, \mathbf{u}_i) = \arg \min_{\varkappa\in\mathcal{K},\psi\in\mathcal{M},\mathbf{u}\in\mathbf{C}} \left\{\boldsymbol{\mathcal{U}}_{\varkappa,\psi,\mathbf{u}} + \mathscr{P}_{\varkappa,\psi,\mathbf{u}}\mathbf{V}_i^{r_0}\right\}. \tag{13}$$

Then, ETIADP iterates through (12) and (13) with $i \to \infty$.

B. Convergence and Monotonicity Analysis

Theorem 1: For $i = 0, 1, \ldots,$ let $\mathbf{V}_i^{r_0}$, $\varkappa_i$, ψ_i, and $\mathbf{u}_i$ be obtained by (11)–(13). Choose the constants $\zeta \in \mathbb{R}_{>0}$ and $\underline{\eta}, \overline{\eta} \in \mathbb{R}_{\geq 0}$, such that

$$\underline{\eta} \leq 1 \leq \overline{\eta} < \infty, \tag{14}$$

$$\mathscr{P}_{\varkappa,\psi,\mathbf{u}}\mathbf{J}^{h^*,r_0} \leq \zeta\boldsymbol{\mathcal{U}}_{\varkappa,\psi,\mathbf{u}} \tag{15}$$

and

$$\underline{\eta}\mathbf{J}^{h^*,r_0} \leq \mathbf{V}_0^{r_0} \leq \overline{\eta}\mathbf{J}^{h^*,r_0} \tag{16}$$

hold for any $\varkappa \in \mathcal{K}$, $\psi \in \mathcal{M}$ and $\mathbf{u} \in \mathbf{C}$. Then, the iterative value function $\mathbf{V}_i^{r_0}$ converges to the optimum, that is,

$$\lim_{i\to\infty} \mathbf{V}_i^{r_0} = \mathbf{J}^{h^*,r_0}. \tag{17}$$

Remark 2: It is obvious that there must exist constants ζ, $\underline{\eta}$, and $\overline{\eta}$ satisfying (14)–(16). In other words, as long as the initial value function is chosen as (11), the sequence of iterative value functions obtained by the ETIADP algorithm converges to the optimal performance index function. In terms of the monotonicity property, Theorem 1 shows that the value sequence may be neither monotonically decreasing nor monotonically increasing. Nevertheless, under several conditions, the value sequence can increase (decrease) monotonically, which is illustrated by the following theorem.

Theorem 2: If the initial value function of the ETIADP algorithm satisfies $\mathbf{V}_1^{r_0} \le \mathbf{V}_0^{r_0}$ ($\mathbf{V}_1^{r_0} \ge \mathbf{V}_0^{r_0}$), then, for $i = 0, 1, \ldots,$ we have $0 \le \mathbf{V}_{i+1}^{r_0} \le \mathbf{V}_i^{r_0}$ ($\mathbf{V}_{i+1}^{r_0} \ge \mathbf{V}_i^{r_0} \ge 0$).

C. Neural Network Implementation

The proposed ETIADP approximates the theoretical iterative value functions and policies by adopting neural networks. In detail, the value function $\mathbf{V}_i^{r_0}$ is approximated by the neural network $\hat{V}_i(\varsigma)$ (critic network), while the control law cluster $\mathbf{u}_i(x) = [u_i(x, \mathcal{T}_1), \ldots, u_i(x, \mathcal{T}_{\max})]$, delay function $\psi_i(x)$ and triggering strategy $\varkappa_i(\varsigma)$ are approximated by the neural networks $\hat{\mathbf{u}}_i(x) = [\hat{u}_i(x, \mathcal{T}_1), \ldots, \hat{u}_i(x, \mathcal{T}_{\max})]$, $\hat{\psi}_i(x)$ and $\hat{\varkappa}_i(\varsigma)$, respectively ($\hat{\mathbf{u}}_i(x)$, $\hat{\psi}_i(x)$ and $\hat{\varkappa}_i(\varsigma)$ together form the "action network group").

All the neural networks mentioned above consist of three layers, namely the input, hidden, and output layers. Furthermore, the mathematical descriptions of the action network group and critic network are

$$\begin{aligned}
\hat{u}_i(x,\rho) &= \mathscr{W}_{i,\rho}^{\mathsf{T}}\left\{g\left(\overline{\mathscr{W}}_{i,\rho}^{\mathsf{T}}x + \overline{\mathscr{O}}_{i,\rho}\right)\right\} + \mathscr{O}_{i,\rho}, \quad \rho \in \mathcal{G}\\
\hat{\psi}_i(x) &= \gamma\left\{W_i^{\mathsf{T}}\left\{g\left(\overline{W}_i^{\mathsf{T}}x + \overline{O}_i\right)\right\} + O_i\right\}\\
\hat{\varkappa}_i(\varsigma) &= \mu\left\{\mathcal{W}_i^{\mathsf{T}}\left\{g\left(\overline{\mathcal{W}}_i^{\mathsf{T}}\varsigma + \overline{\mathcal{O}}_i\right)\right\} + \mathcal{O}_i\right\}
\end{aligned}$$

and

$$\hat{V}_i(\varsigma) = \mathfrak{W}_i^{\mathsf{T}}\left\{g(\overline{\mathfrak{W}}_i^{\mathsf{T}}\varsigma + \overline{\mathfrak{O}}_i)\right\} + \mathfrak{O}_i$$

respectively. The input layer to hidden layer weight matrices are denoted by $\overline{\mathscr{W}}_{i,\rho}$ ($\overline{W}_i, \overline{\mathcal{W}}_i, \overline{\mathfrak{W}}_i$) while the hidden layer to output layer weight matrices are denoted by $\mathscr{W}_{i,\rho}$ ($W_i, \mathcal{W}_i, \mathfrak{W}_i$). The bias vectors attached to the hidden and output layers are represented by $\overline{\mathscr{O}}_{i,\rho}$ ($\overline{O}_i, \overline{\mathcal{O}}_i, \overline{\mathfrak{O}}_i$) and $\mathscr{O}_{i,\rho}$ ($O_i, \mathcal{O}_i, \mathfrak{O}_i$), respectively. $g(\cdot)$, which is the tan-sigmoid function, is the transfer function of the hidden layers. The transfer functions of the output layers of the networks $\hat{\psi}_i$ and $\hat{\varkappa}_i$ are $\gamma(\cdot)$ and $\mu(\cdot)$, respectively. Let $\mathsf{w} = W_i^{\mathsf{T}}\{g(\overline{W}_i^{\mathsf{T}}x + \overline{O}_i)\} + O_i$ and $w = \mathcal{W}_i^{\mathsf{T}}\{g(\overline{\mathcal{W}}_i^{\mathsf{T}}\varsigma + \overline{\mathcal{O}}_i)\} + \mathcal{O}_i$. Then, we have

$$\gamma(\mathsf{w}) = \begin{cases} \mathcal{T}_1, & 0 < \mathsf{w} \le \mathcal{T}_1\\ \mathcal{T}_2, & \mathcal{T}_1 < \mathsf{w} \le \mathcal{T}_2\\ \vdots & \\ \mathcal{T}_{\max}, & \mathcal{T}_{\max-1} < \mathsf{w} \le \mathcal{T}_{\max} \end{cases} \tag{18}$$

$$\mu(w) = \begin{cases} 0, & -1 < w \le 0\\ 1, & 0 < w \le 1. \end{cases} \tag{19}$$

At each iteration of ETIADP, the weight matrices and bias vectors are properly tuned by employing the gradient decent and error back propagation techniques such that the neural networks precisely approximate the targets. Overall, the implementation structure of the ETIADP algorithm is illustrated by Fig. 4.

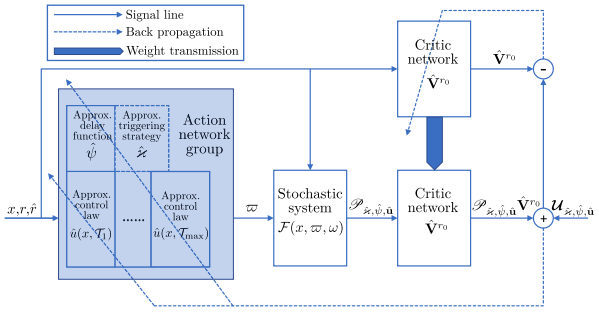


Fig. 4. Implementation structure of the ETIADP algorithm.

IV. High-Efficiency Event-Triggered Impulsive ADP Algorithm

A. Derivation of the HEIADP Algorithm

Modern computing devices are usually equipped with multiple processors and the neural processing unit (NPU, which accelerates neural network operations such as convolutions and matrix multiplications). For example, the Apple A14 Bionic system on a chip (SoC), designed by Apple Inc., features six central processors and includes the dedicated NPU that is called "Neural Engine." Meanwhile, the Kirin 9000 SoC introduced by Huawei Technologies Company is an octa-core chipset integrating a tri-core NPU. The simplified architecture of the above MPSs is illustrated by Fig. 5(a), where, without loss of generality, the MPS is assumed to have four central processors c_1, c_2, c_3, and c_4. The MPSs, along with computing techniques such as multithread programming and parallel processing, enable the ETIADP algorithm to be executed in a concurrent manner. Specifically, with the MPS shown in Fig. 5(a) employed, the concurrent ETIADP algorithm at the ith iteration step divides the original and impulsive global system state spaces, i.e., X and $\hat{X}$, into smaller disjoint nonempty subsets $X_{i,1}, X_{i,2}, X_{i,3}, X_{i,4}$ and $\hat{X}_{i,1}, \hat{X}_{i,2}, \hat{X}_{i,3}, \hat{X}_{i,4}$, respectively, where

$$X = \cup_{j=1}^4 X_{i,j}, \quad \hat{X} = \cup_{j=1}^4 \hat{X}_{i,j}. \tag{20}$$

and

$$\hat{X}_{i,j} = \left\{\varsigma \,|\, \varsigma = \begin{bmatrix} x \\ \hat{r} \end{bmatrix}, x \in X_{i,j}, \hat{r} \in \hat{\Re}\right\} \subseteq \hat{X}. \tag{21}$$

Then, $X_{i,j}$ and $\hat{X}_{i,j}$ are sent to the processor c_j, which generates and gathers the required sample data for training ($\hat{\mathbf{u}}_i, \hat{\psi}_i, \hat{\varkappa}_i, \hat{\mathbf{V}}_{i+1}^{r_0}$) at $X_{i,j}/\hat{X}_{i,j}$ according to (12) and (13). After the NPU receives all the sampled data which are obtained by all processors and associated with the global spaces X and $\hat{X}$, it updates the neural networks $\hat{\mathbf{V}}_{i+1}^{r_0}, \hat{\mathbf{u}}_i, \hat{\varkappa}_i$ and $\hat{\psi}_i$. Implemented in such a concurrent manner, the ETIADP algorithm iteratively executes the above steps with $i \to \infty$ and converges to the optimum. The task scheduling for the MPS under the concurrent ETIADP algorithm is demonstrated in Fig. 5(b). However, this algorithm may result in the unsynchronization of the task progresses across processors. Since the complexity levels of the assigned tasks are different and the processors may operate at different speeds, the consumed time for each processor to complete its sample generation task of the current iteration step usually differs from each other. Specifically, as shown in Fig. 5(b), c_1 is the first processor among the MPS

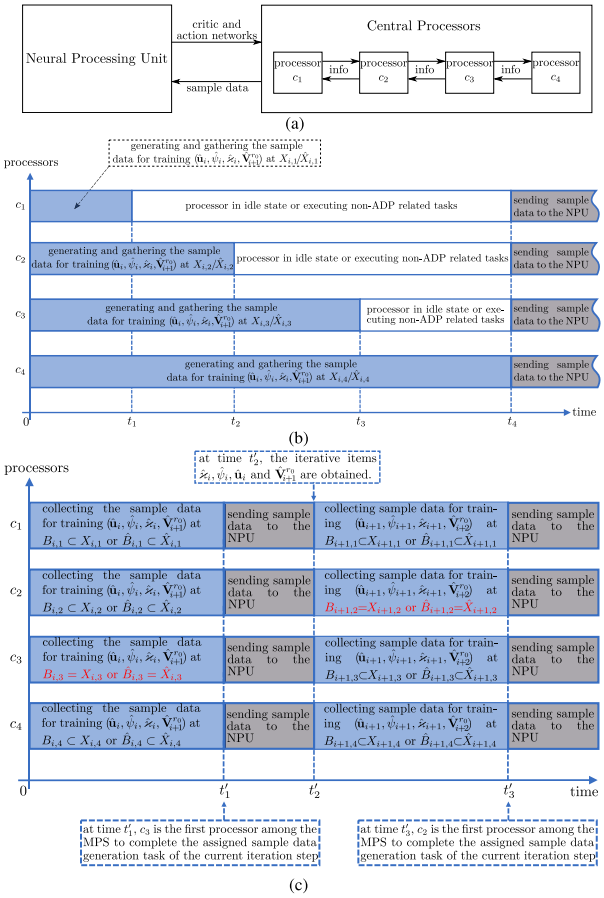


Fig. 5. Comparison between the concurrent ETIADP and HEIADP algorithms. (a) Simplified architecture of the MPS. (b) Task scheduling for processors with the concurrent ETIADP algorithm implemented in the MPS. (c) Task scheduling for processors with the concurrent HEIADP algorithm implemented in the MPS (for illustration purpose only).

to finish collecting the required sample set at the ith iteration step, then, it is in the idle state during $[t_1, t_4]$ waiting for the other processors to catch up. Similarly, c_2 and c_3, as the second and third processors to complete their assigned tasks of the current iteration, they also have to be in the idle state until c_4 gets its job done. Only at time t_4, when all the sampled data associated with the global spaces are gathered, can the NPU kick in and train the neural networks $(\hat{\mathbf{u}}_i, \hat{\psi}_i, \hat{\varkappa}_i, \hat{\mathbf{V}}^{r_0}_{i+1})$. Overall, due to the unsynchronization problem, the concurrent ETIADP algorithm leaves a number of processors of the MPS unused at some time intervals during execution, and causes the computational resources of the MPSs not being fully utilized.

To address this deficiency, a concurrent HEIADP algorithm is proposed, which realizes synchronization by forcing the current task of the MPS to transition from sample set generation to neural networks training, when the first processor to finish the current assigned sample generation task occurs. Fig. 5(c) shows the task scheduling for the processors with the concurrent HEIADP algorithm implemented in the MPS. According to Fig. 5(c), at time t'_1, the first processor c_3 among the MPS completes the assigned sample generation task at the ith iteration step. At the same time, all current sample generation tasks running on the other processors are forced to terminate simultaneously, and all processors proceed to send the already generated sample data to the NPU. Let

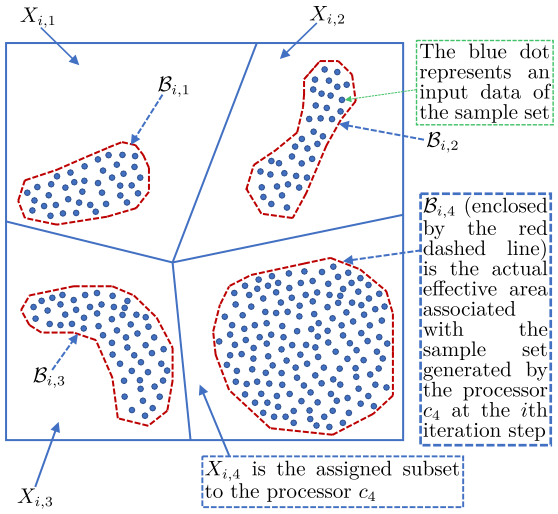


Fig. 6. Effective areas associated with the sample sets (for illustration purpose only).

$\mathcal{B}_{i,j} \subseteq X_{i,j}$ and $\hat{\mathcal{B}}_{i,j} \subseteq \hat{X}_{i,j}$ denote the actual "effective areas" associated with the sample dataset generated by c_j at the ith iteration step. Suppose the sample dataset generated by c_j at ith iteration is $\mathcal{S}_{i,j} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_{|\mathcal{S}_{i,j}|}, y_{|\mathcal{S}_{i,j}|})\}$, where $x_1, \ldots, x_{|\mathcal{S}_{i,j}|} \in X_{i,j}$ are the input data and $y_1, \ldots, y_{|\mathcal{S}_{i,j}|}$ are the corresponding expected output data. Then, by choosing a relatively small positive constant ξ, we define the set $\{x \| \|x - x_1\| \leq \xi\} \cup \{x \| \|x - x_2\| \leq \xi\} \cup \cdots \cup \{x \| \|x - x_{|\mathcal{S}_{i,j}|}\| \leq \xi\} = \mathcal{B}_{i,j}$ as the "effective area" associated with $\mathcal{S}_{i,j}$. The concept of "effective areas" is illustrated by Fig. 6. Then, the NPU trains the neural networks $(\hat{\mathbf{u}}_i, \hat{\psi}_i, \hat{\varkappa}_i, \hat{\mathbf{V}}^{r_0}_{i+1})$ at $\mathcal{B}_i = \cup_{j=1}^{4} \mathcal{B}_{i,j}$ or $\hat{\mathcal{B}}_i = \cup_{j=1}^{4} \hat{\mathcal{B}}_{i,j}$, while letting them inherit the values of their corresponding predecessors at $X \backslash \mathcal{B}_i$ or $\hat{X} \backslash \hat{\mathcal{B}}_i$. Afterward, the concurrent HEIADP algorithm repeat similar steps for the next iteration, and so on. The above synchronization mechanism ensures that the processors in the MPS are properly loaded during the whole execution, thus making full use of the multiprocessor computing power.

Another advantage of the HEIADP algorithm over the ETIADP approach lies in its low memory usage. As for the concurrent ETIADP approach, noticing that before the NPU training the neural networks, the corresponding computing devices or the MPSs have to store all the sample data associated with the global state spaces, i.e., X and $\hat{X}$, into the physical memory spaces. Therefore, when dealing with complexed/large-scale controlled systems, or if the control action space and system state space are of huge volume, the required vast amounts of sample data may not be totally fit into the limited physical memory spaces (i.e., the memory overflow problems), causing the algorithms unfeasible. In contrast, HEIADP at each iteration only locally updates the iterative $\hat{\varkappa}_i$, $\hat{\psi}_i$, $\hat{\mathbf{u}}_i$ and $\hat{\mathbf{V}}^{r_0}_{i+1}$ with respect to the local state spaces $\mathcal{B}_i$ and $\hat{\mathcal{B}}_i$. Consequently, the size of the sample datasets which the algorithm collects and stores at each iteration is also reduced significantly, meaning that the memory burden and the memory requirement for the concurrent execution are relaxed. In summary, the present approach improves the utilization rate of the computational resources of the MPS, while reducing the memory requirement for large-scale problems. Next, the mathematical abstraction of the concurrent HEIADP algorithm is given.

B. Mathematical Abstraction of the Concurrent HEIADP

Given the set sequences $\{\mathcal{B}_i\}$ and $\{\hat{\mathcal{B}}_i\}$, $i = 0, 1, \ldots$, which according to (21), should satisfy $\mathcal{B}_i \subseteq X$ and

$$\hat{\mathcal{B}}_i = \left\{ \varsigma \,|\, \varsigma = \begin{bmatrix} x \\ \hat{r} \end{bmatrix}, x \in \mathcal{B}_i, \hat{r} \in \hat{\mathfrak{R}} \right\} \subseteq \hat{X} \tag{22}$$

respectively. Define $\{\alpha_i(\varsigma)\}$, $i = 0, 1, \ldots$, as the sequence of weight functions, where $\alpha_i(\cdot)\colon \hat{X} \to [0, 1]$ satisfies

$$\begin{cases} 0 < \alpha_i(\varsigma) \leq 1, & \varsigma \in \hat{\mathcal{B}}_i \\ \alpha_i(\varsigma) = 0, & \varsigma \notin \hat{\mathcal{B}}_i. \end{cases} \tag{23}$$

Then, let $\{\mathcal{A}_i\}$, $i = 0, 1, \ldots$, be a sequence of diagonal matrices, such that $\mathcal{A}_i = \operatorname{diag}(\alpha_i(\sigma_1), \ldots, \alpha_i(\sigma_m), \ldots, \alpha_i(\sigma_{|\hat{X}|}))$. The algorithm is initialized by the value function $\mathbf{V}_0^{r_0}$ which is chosen according to (11), and the initial $\varkappa_0$, ψ_0 and $\mathbf{u}_0$ are computed by

$$(\varkappa_0, \psi_0, \mathbf{u}_0) = \arg \min_{\varkappa \in \mathscr{K}_0, \psi \in \mathscr{M}_0, \mathbf{u} \in \mathscr{C}_0} \left\{ \mathcal{U}_{\varkappa,\psi,\mathbf{u}} + \mathscr{P}_{\varkappa,\psi,\mathbf{u}} \mathbf{V}_0^{r_0} \right\} \tag{24}$$

where the constrained searching spaces of $\varkappa$, ψ and $\mathbf{u}$ are expressed as $\mathscr{K}_0 = \{\varkappa(\cdot) | \varkappa(\varsigma) = \varkappa_{-1}(\varsigma), \forall \varsigma \in \hat{X} \backslash \hat{\mathcal{B}}_0\} \subseteq \mathcal{K}$, $\mathscr{M}_0 = \{\psi(\cdot) | \psi(x) = \psi_{-1}(x), \forall x \in X \backslash \mathcal{B}_0\} \subseteq \mathcal{M}$ and $\mathscr{C}_0 = \{\mathbf{u}(\cdot) | u(x, \rho) = u_{-1}(x, \rho), \forall x \in X \backslash \mathcal{B}_0, \forall \rho \in \mathcal{G}\} \subseteq \mathbf{C}$, respectively. Particularly, $\varkappa_{-1}$, ψ_{-1} and $u_{-1}(\cdot, \rho)$ are chosen arbitrarily in the sets $\mathcal{K}, \mathcal{M}$ and C_ρ, respectively. For any $i = 1, 2, \ldots$, obtain the iterative value functions $\mathbf{V}_i^{r_0}$ via

$$\begin{aligned} \mathbf{V}_i^{r_0} &= (\mathbf{E} - \mathcal{A}_{i-1}) \mathbf{V}_{i-1}^{r_0} \\ &\quad + \mathcal{A}_{i-1} \left(\mathcal{U}_{\varkappa_{i-1}, \psi_{i-1}, \mathbf{u}_{i-1}} + \mathscr{P}_{\varkappa_{i-1}, \psi_{i-1}, \mathbf{u}_{i-1}} \mathbf{V}_{i-1}^{r_0} \right) \end{aligned} \tag{25}$$

where $\mathbf{E}$ is the $\left|\hat{X}\right|$-dimensional identity matrix. The iterative triggering strategy $\varkappa_i(\varsigma)$, control law cluster $\mathbf{u}_i(x)$, and delay function $\psi_i(x)$ are derived through

$$(\varkappa_i, \psi_i, \mathbf{u}_i) = \arg \min_{\varkappa \in \mathscr{K}_i, \psi \in \mathscr{M}_i, \mathbf{u} \in \mathscr{C}_i} \left\{ \mathcal{U}_{\varkappa,\psi,\mathbf{u}} + \mathscr{P}_{\varkappa,\psi,\mathbf{u}} \mathbf{V}_i^{r_0} \right\} \tag{26}$$

where $\mathscr{K}_i = \{\varkappa(\cdot) | \varkappa(\varsigma) = \varkappa_{i-1}(\varsigma), \forall \varsigma \in \hat{X} \backslash \hat{\mathcal{B}}_i\} \subseteq \mathcal{K}$, $\mathscr{M}_i = \{\psi(\cdot) | \psi(x) = \psi_{i-1}(x), \forall x \in X \backslash \mathcal{B}_i\} \subseteq \mathcal{M}$, and $\mathscr{C}_i = \{\mathbf{u}(\cdot) | u(x, \rho) = u_{i-1}(x, \rho), \forall x \in X \backslash \mathcal{B}_i, \forall \rho \in \mathcal{G}\} \subseteq \mathbf{C}$.

Then, HEIADP iterates through (25) and (26) with $i \to \infty$.

C. Property Analysis

Theorem 3: For $i = 0, 1, \ldots$, let $\mathbf{V}_i^{r_0}$, $\varkappa_i$, ψ_i, and $\mathbf{u}_i$ be obtained by (24)–(26). Define the function $\phi(\delta)\colon \mathbb{N}_{\geq 0} \to \mathbb{N}_{\geq 0}$, where $\phi(0) = 0$, such that

$$\cup_{i=\phi(\delta)}^{\phi(\delta+1)-1} \hat{\mathcal{B}}_i = \hat{X} \quad \forall \delta = 0, 1, \ldots, \infty. \tag{27}$$

Given the constants $\zeta \in \mathbb{R}_{>0}$ and $\underline{\eta}, \overline{\eta} \in \mathbb{R}_{\geq 0}$ satisfying (14)–(16), respectively. Then, the iterative value function $\mathbf{V}_i^{r_0}$ converges to the optimum $\mathbf{J}^{h^*, r_0}$, as i increases to infinity.

Proof: First, we prove that

$$\mathbf{V}_{\phi(\delta)}^{r_0} \leq \left(1 + (\overline{\eta} - 1) \prod_{j=0}^{\delta-1} \left(1 - \frac{\beta_\delta}{1+\zeta} \right) \right) \mathbf{J}^{h^*, r_0} \tag{28}$$

holds for $\delta = 0, 1, \ldots$, where $\prod_{j=0}^{-1}(\cdot) = 1$ and

$$\beta_\delta = \min_{\phi(\delta) \leq i \leq \phi(\delta+1)-1} \left\{ \min_{\varsigma \in \hat{\mathcal{B}}_i} \{\alpha_i(\varsigma)\} \right\}. \tag{29}$$

According to (16), (28) obviously holds for $\delta = 0$. Based on the left-hand side of (16), we can obtain

$$\begin{aligned} \mathbf{V}_1^{r_0} &= (\mathbf{E} - \mathcal{A}_0) \mathbf{V}_0^{r_0} + \mathcal{A}_0 \min_{\varkappa \in \mathscr{K}_0, \psi \in \mathscr{M}_0, \mathbf{u} \in \mathscr{C}_0} \left\{ \mathcal{U}_{\varkappa,\psi,\mathbf{u}} + \mathscr{P}_{\varkappa,\psi,\mathbf{u}} \mathbf{V}_0^{r_0} \right\} \\ &= (\mathbf{E} - \mathcal{A}_0) \mathbf{V}_0^{r_0} + \mathcal{A}_0 \min_{\varkappa \in \mathcal{K}, \psi \in \mathcal{M}, \mathbf{u} \in \mathbf{C}} \left\{ \mathcal{U}_{\varkappa,\psi,\mathbf{u}} + \mathscr{P}_{\varkappa,\psi,\mathbf{u}} \mathbf{V}_0^{r_0} \right\} \\ &\leq (\mathbf{E} - \mathcal{A}_0) \mathbf{V}_0^{r_0} + \mathcal{A}_0 \min_{\varkappa \in \mathcal{K}, \psi \in \mathcal{M}, \mathbf{u} \in \mathbf{C}} \left\{ \mathcal{U}_{\varkappa,\psi,\mathbf{u}} + \overline{\eta} \mathscr{P}_{\varkappa,\psi,\mathbf{u}} \mathbf{J}^{h^*, r_0} \right\}. \end{aligned} \tag{30}$$

By adding $\zeta(\overline{\eta} - 1/1 + \zeta) \mathcal{U}_{\varkappa,\psi,\mathbf{u}}$ to and subtracting the same term from (30), it becomes

$$\begin{aligned} \mathbf{V}_1^{r_0} &\leq (\mathbf{E} - \mathcal{A}_0) \mathbf{V}_0^{r_0} \\ &\quad + \mathcal{A}_0 \min_{\varkappa \in \mathcal{K}, \psi \in \mathcal{M}, \mathbf{u} \in \mathbf{C}} \Bigg\{ \mathcal{U}_{\varkappa,\psi,\mathbf{u}} + \overline{\eta} \mathscr{P}_{\varkappa,\psi,\mathbf{u}} \mathbf{J}^{h^*, r_0} \\ &\qquad + \zeta \frac{\overline{\eta} - 1}{1+\zeta} \mathcal{U}_{\varkappa,\psi,\mathbf{u}} + \zeta \frac{1 - \overline{\eta}}{1+\zeta} \mathcal{U}_{\varkappa,\psi,\mathbf{u}} \Bigg\}. \end{aligned} \tag{31}$$

Since $\mathscr{P}_{\varkappa,\psi,\mathbf{u}} \mathbf{J}^{h^*, r_0} \leq \zeta \mathcal{U}_{\varkappa,\psi,\mathbf{u}}$, (31) can be developed into

$$\begin{aligned} \mathbf{V}_1^{r_0} &\leq (\mathbf{E} - \mathcal{A}_0) \mathbf{V}_0^{r_0} \\ &\quad + \mathcal{A}_0 \min_{\varkappa,\psi,\mathbf{u}} \Bigg\{ \mathcal{U}_{\varkappa,\psi,\mathbf{u}} + \overline{\eta} \mathscr{P}_{\varkappa,\psi,\mathbf{u}} \mathbf{J}^{h^*, r_0} \\ &\qquad + \zeta \frac{\overline{\eta} - 1}{1+\zeta} \mathcal{U}_{\varkappa,\psi,\mathbf{u}} + \frac{1 - \overline{\eta}}{1+\zeta} \mathscr{P}_{\varkappa,\psi,\mathbf{u}} \mathbf{J}^{h^*, r_0} \Bigg\} \\ &= (\mathbf{E} - \mathcal{A}_0) \mathbf{V}_0^{r_0} \\ &\quad + \mathcal{A}_0 \min_{\varkappa,\psi,\mathbf{u}} \Bigg\{ \left(1 + \zeta \frac{\overline{\eta} - 1}{1+\zeta} \right) \mathcal{U}_{\varkappa,\psi,\mathbf{u}} \\ &\qquad + \left(\overline{\eta} - \frac{\overline{\eta} - 1}{1+\zeta} \right) \mathscr{P}_{\varkappa,\psi,\mathbf{u}} \mathbf{J}^{h^*, r_0} \Bigg\} \\ &\leq \overline{\eta} (\mathbf{E} - \mathcal{A}_0) \mathbf{J}^{h^*, r_0} + \left(1 + \zeta \frac{\overline{\eta} - 1}{1+\zeta} \right) \mathcal{A}_0 \mathbf{J}^{h^*, r_0} \\ &= \left(\mathbf{E} + (\overline{\eta} - 1) \left(\mathbf{E} - \frac{\mathcal{A}_0}{1+\zeta} \right) \right) \mathbf{J}^{h^*, r_0}. \end{aligned} \tag{32}$$

Similar to (30)–(32), for any $1 \leq i \leq \phi(1)$, we can obtain

$$\mathbf{V}_i^{r_0} \leq \left(\mathbf{E} + (\overline{\eta} - 1) \left(\mathbf{E} - \frac{\mathcal{A}_{i-1}}{1+\zeta} \right) \right) \mathbf{J}^{h^*, r_0}. \tag{33}$$

By (27), (29) and (33), we get

$$\mathbf{V}_{\phi(1)}^{r_0} \leq \left(1 + (\overline{\eta} - 1) \left(1 - \frac{\beta_0}{1+\zeta} \right) \right) \mathbf{J}^{h^*, r_0}. \tag{34}$$

Assume (28) holds for $\delta = l - 1, l = 1, 2, \ldots$ Then, for $i = \phi(l-1) + 1$, we have (35), shown at the bottom of the page. By adding $\zeta(\overline{\eta} - 1/(1+\zeta)) \prod_{j=0}^{l-2}(1 - (\beta_j/1+\zeta)) \mathcal{U}_{\varkappa,\psi,\mathbf{u}}$

$$\begin{aligned} &\mathbf{V}_{\phi(l-1)+1}^{r_0} \\ &= \mathcal{A}_{\phi(l-1)} \min_{\varkappa \in \mathscr{K}_{\phi(l-1)}, \psi \in \mathscr{M}_{\phi(l-1)}, \mathbf{u} \in \mathscr{C}_{\phi(l-1)}} \left\{ \mathcal{U}_{\varkappa,\psi,\mathbf{u}} + \mathscr{P}_{\varkappa,\psi,\mathbf{u}} \mathbf{V}_{\phi(l-1)}^{r_0} \right\} + \left(\mathbf{E} - \mathcal{A}_{\phi(l-1)} \right) \mathbf{V}_{\phi(l-1)}^{r_0} \\ &\leq \left(\mathbf{E} - \mathcal{A}_{\phi(l-1)} \right) \mathbf{V}_{\phi(l-1)}^{r_0} + \mathcal{A}_{\phi(l-1)} \min_{\varkappa \in \mathcal{K}, \psi \in \mathcal{M}, \mathbf{u} \in \mathbf{C}} \left\{ \mathcal{U}_{\varkappa,\psi,\mathbf{u}} + \left(1 + (\overline{\eta} - 1) \prod_{j=0}^{l-2} \left(1 - \frac{\beta_j}{1+\zeta} \right) \right) \mathscr{P}_{\varkappa,\psi,\mathbf{u}} \mathbf{J}^{h^*, r_0} \right\} \end{aligned} \tag{35}$$

to and subtracting the same term from (35), it becomes

$$\begin{aligned}
&\mathbf{V}^{r_0}_{\phi(l-1)+1}\\
&\le \left(1+(\overline{\eta}-1)\prod_{j=0}^{l-2}\left(1-\frac{\beta_j}{1+\zeta}\right)\right)(\mathbf{E}-\mathcal{A}_{\phi(l-1)})\mathbf{J}^{h^*,r_0}\\
&\quad+\mathcal{A}_{\phi(l-1)}\min_{\varkappa,\psi,\mathbf{u}}\left\{\left(1+\zeta\frac{\overline{\eta}-1}{(1+\zeta)}\prod_{j=0}^{l-2}\left(1-\frac{\beta_j}{1+\zeta}\right)\right)\mathcal{U}_{\varkappa,\psi,\mathbf{u}}\right.\\
&\qquad+\frac{1-\overline{\eta}}{(1+\zeta)}\prod_{j=0}^{l-2}\left(1-\frac{\beta_j}{1+\zeta}\right)\mathscr{P}_{\varkappa,\psi,\mathbf{u}}\mathbf{J}^{h^*,r_0}\\
&\qquad\left.+\left(1+(\overline{\eta}-1)\prod_{j=0}^{l-2}\left(1-\frac{\beta_j}{1+\zeta}\right)\right)\mathscr{P}_{\varkappa,\psi,\mathbf{u}}\mathbf{J}^{h^*,r_0}\right\}.
\end{aligned}\tag{36}$$

Combining similar terms of (36), we can obtain

$$\begin{aligned}
&\mathbf{V}^{r_0}_{\phi(l-1)+1}\\
&\le\left(\mathbf{E}+(\overline{\eta}-1)\prod_{j=0}^{l-2}\left(1-\frac{\beta_j}{1+\zeta}\right)\left(\mathbf{E}-\frac{\mathcal{A}_{\phi(l-1)}}{1+\zeta}\right)\right)\mathbf{J}^{h^*,r_0}.
\end{aligned}\tag{37}$$

Similar to (35)–(37), for any $\phi(l-1)+1\le i\le\phi(l)$, we get

$$\mathbf{V}^{r_0}_{i}\le\left(\mathbf{E}+(\overline{\eta}-1)\prod_{j=0}^{l-2}\left(1-\frac{\beta_j}{1+\zeta}\right)\left(\mathbf{E}-\frac{\mathcal{A}_{i-1}}{1+\zeta}\right)\right)\mathbf{J}^{h^*,r_0}\tag{38}$$

which indicates

$$\mathbf{V}^{r_0}_{\phi(l)}\le\left(1+(\overline{\eta}-1)\prod_{j=0}^{l-1}\left(1-\frac{\beta_j}{1+\zeta}\right)\right)\mathbf{J}^{h^*,r_0}.\tag{39}$$

Using the same techniques as in (30)–(39), we can prove

$$\mathbf{V}^{r_0}_{\phi(\delta)}\ge\left(1+\left(\underline{\eta}-1\right)\prod_{j=0}^{\delta-1}\left(1-\frac{\beta_j}{1+\zeta}\right)\right)\mathbf{J}^{h^*,r_0},\ \delta=0,1,\ldots\tag{40}$$

Since

$$\begin{aligned}
\ln\left(\prod_{j=0}^{\infty}\left(1-\frac{\beta_j}{1+\zeta}\right)\right)&=\sum_{j=0}^{\infty}\ln\left(1-\frac{\beta_j}{1+\zeta}\right)\\
&\le-\sum_{j=0}^{\infty}\frac{\beta_j}{1+\zeta}\to-\infty
\end{aligned}\tag{41}$$

we have

$$\lim_{\delta\to\infty}\prod_{j=0}^{\delta-1}\left(1-\frac{\beta_j}{1+\zeta}\right)=\exp(-\infty)=0.\tag{42}$$

Equation (42) indicates that

$$\begin{aligned}
&\lim_{\delta\to\infty}\left(1+(\overline{\eta}-1)\prod_{j=0}^{\delta-1}\left(1-\frac{\beta_j}{1+\zeta}\right)\right)\mathbf{J}^{h^*,r_0}\\
&=\lim_{\delta\to\infty}\left(1+\left(\underline{\eta}-1\right)\prod_{j=0}^{\delta-1}\left(1-\frac{\beta_j}{1+\zeta}\right)\right)\mathbf{J}^{h^*,r_0}\\
&=\mathbf{J}^{h^*,r_0}.
\end{aligned}\tag{43}$$

From (43) and based on the facts (39) and (40), we immediately derive that

$$\lim_{i\to\infty}\mathbf{V}^{r_0}_{i}=\lim_{\delta\to\infty}\mathbf{V}^{r_0}_{\phi(\delta)}=\mathbf{J}^{h^*,r_0}.\tag{44}$$

The proof is completed. ■

Remark 3: The convergence proof of Theorem 1 can be obtained from that of Theorem 3 if we let $\mathcal{B}_i=X,\hat{\mathcal{B}}_i=\hat{X},\forall i=0,1,\ldots$ (thereupon, the ETIADP algorithm is in fact a special case of the HEIADP algorithm, since if the sequence of effective areas satisfy $\mathcal{B}_i=X,\hat{\mathcal{B}}_i=\hat{X},\forall i=0,1,\ldots$, the HEIADP algorithm reduces to the ETIADP algorithm). Moreover, Theorem 3 shows that as long as the iterative impulsive policies and iterative value functions get updated infinite times for any $\varsigma\in\hat{X}$, then, the HEIADP algorithm is guaranteed to converge to the global optimum. However, in practical applications, it is impossible for the algorithm to execute for infinite iteration steps on the computing devices. Instead, the convergence termination condition $\|\mathbf{V}^{r_0}_{\phi(\delta)}-\mathbf{V}^{r_0}_{\phi(\delta-1)}\|<\varepsilon$ is used to terminate the algorithm, and the iterative value function which satisfies the termination condition for the first time during algorithm execution, is treated as the "optimal" performance index function. To guarantee the admissibility of the corresponding obtained controller, an admissibility criterion identifying the admissible policies is provided via the following theorems, which also show the HEIADP algorithm can obtain an admissible policy within finite iteration steps.

Theorem 4: For $i=0,1,\ldots$, let $\mathbf{V}^{r_0}_{i}$, $\varkappa_i$, ψ_i and $\mathbf{u}_i$ be obtained by (24)–(26). If there exists $\delta\in\mathbb{N}_{\ge0}$, such that for any $\phi(\delta)\le i\le\phi(\delta+1)-1$, the condition

$$V^{r_0}_{i+1}(\varsigma)-V^{r_0}_{i}(\varsigma)<\alpha_i(\varsigma)\mathcal{U}_{\varkappa_i,\psi_i,\mathbf{u}_i}(\varsigma)\quad\forall\varsigma\in\hat{\mathcal{B}}_i\backslash\Omega_{\mathrm{o}}\tag{45}$$

holds, then, the iterative triggering strategy $\varkappa_{\phi(\delta+1)-1}$, delay function $\psi_{\phi(\delta+1)-1}$, and control law cluster $\mathbf{u}_{\phi(\delta+1)-1}$ are admissible.

Theorem 5: For $i=0,1,\ldots$, let $\mathbf{V}^{r_0}_{i}$, $\varkappa_i$, ψ_i, and $\mathbf{u}_i$ be obtained by (24)–(26). Then, there must exist a finite $\delta\in\mathbb{N}_{\ge0}$, with which (45) holds for any $\phi(\delta)\le i\le\phi(\delta+1)-1$.

Proof: Assume the conclusion is false. Then, for any $\delta\in\mathbb{N}_{\ge0}$, there always exist an integer $0\le\overline{\phi}(\delta)\le\phi(\delta+1)-\phi(\delta)-1$ and an impulsive state $\varsigma\in\hat{\mathcal{B}}_{\phi(\delta)+\overline{\phi}(\delta)}\backslash\Omega_{\mathrm{o}}$ satisfying

$$V^{r_0}_{\phi(\delta)+\overline{\phi}(\delta)+1}(\varsigma)-V^{r_0}_{\phi(\delta)+\overline{\phi}(\delta)}(\varsigma)\ge\alpha_{\phi(\delta)+\overline{\phi}(\delta)}(\varsigma)\mathcal{U}_{\phi(\delta)+\overline{\phi}(\delta)}(\varsigma)$$

where $\mathcal{U}_i$ is used to represent $\mathcal{U}_{\varkappa_i,\psi_i,\mathbf{u}_i}$. With $\delta\to\infty$, we get $\lim_{\delta\to\infty}\|\mathbf{V}^{r_0}_{\phi(\delta)+\overline{\phi}(\delta)+1}-\mathbf{V}^{r_0}_{\phi(\delta)+\overline{\phi}(\delta)}\|=0$ according to Theorem 3. The above facts indicate

$$\lim_{\delta\to\infty}\alpha_{\phi(\delta)+\overline{\phi}(\delta)}(\varsigma)\mathcal{U}_{\phi(\delta)+\overline{\phi}(\delta)}(\varsigma)=0,\quad\varsigma\notin\Omega_{\mathrm{o}}.\tag{46}$$

It contradicts the positive definiteness of $\mathcal{U}$. Hence, the assumption is false and the conclusion holds. ■

According to the above convergence and admissibility analysis, the HEIADP algorithm in the form of mathematical abstraction is summarized in Algorithm 1.

Since the approximation structures such as neural networks are used in the ETIADP and HEIADP algorithms, there exist errors between the approximated and theoretical values. The following theorem analyzes the error dynamic characteristics of the proposed methods, and establishes the approximation error bound of the critic networks, by which the approximated iterative value functions fall in a small neighborhood of the optimum as $i\to\infty$. In other words, the following theorem addresses the connection between the ideal and neural-network-based realizations of the proposed methods.

Theorem 6: Let the operators $L_i, \mathscr{L}_\delta\colon \mathbb{R}^{|\hat{X}|} \to \mathbb{R}^{|\hat{X}|}$ be expressed by

$$L_i\big(\mathbf{V}^{r_0}\big) = (\mathbf{E} - \mathcal{A}_{i-1})\mathbf{V}^{r_0} + \mathcal{A}_{i-1}\min_{\varkappa,\psi,\mathbf{u}}\big\{\mathcal{U}_{\varkappa,\psi,\mathbf{u}} + \mathscr{P}_{\varkappa,\psi,\mathbf{u}}\mathbf{V}^{r_0}\big\} \tag{47}$$

and

$$\mathscr{L}_\delta\big(\mathbf{V}^{r_0}\big) = L_{\phi(\delta+1)-1}\cdots L_{\phi(\delta)+1}L_{\phi(\delta)}\big(\mathbf{V}^{r_0}\big) \tag{48}$$

respectively, where $i = 0, 1, \ldots, \delta = 0, 1, \ldots$ Given the finite positive constants $\underline{\epsilon} \le 1$ and $\overline{\epsilon} \ge 1$, such that

$$\underline{\epsilon}\mathscr{L}_\delta\Big(\hat{\mathbf{V}}^{r_0}_{\phi(\delta)}\Big) \le \hat{\mathbf{V}}_{\phi(\delta+1)} \le \overline{\epsilon}\mathscr{L}_\delta\Big(\hat{\mathbf{V}}^{r_0}_{\phi(\delta)}\Big) \tag{49}$$

holds for any $\delta = 0, 1, \ldots$ Then, the approximate iterative value functions $\hat{\mathbf{V}}^{r_0}_i$ satisfy

$$\begin{aligned}\underline{\epsilon}&\left(1 + \sum_{j=1}^{\delta}\hat{\zeta}^j\underline{\epsilon}^{j-1}\frac{\underline{\epsilon}-1}{(1+\zeta)^j} + \hat{\zeta}^\delta\underline{\epsilon}^\delta\frac{\underline{\eta}-1}{(1+\zeta)^\delta}\right)\mathbf{J}^{h^*,r_0} \le \hat{\mathbf{V}}^{r_0}_{\phi(\delta+1)}\\ &\le \overline{\epsilon}\left(1 + \sum_{j=1}^{\delta}\hat{\zeta}^j\overline{\epsilon}^{j-1}\frac{\overline{\epsilon}-1}{(1+\zeta)^j} + \hat{\zeta}^\delta\overline{\epsilon}^\delta\frac{\overline{\eta}-1}{(1+\zeta)^i}\right)\mathbf{J}^{h^*,r_0}\end{aligned} \tag{50}$$

where $\hat{\zeta} = \max_{\delta\in\mathbb{N}_{\ge 0}}\{\zeta + 1 - \beta_\delta\}$ and $\sum_{j=1}^{<1}(\cdot) = 0$.

Proof: First, we prove that

$$\hat{\mathbf{V}}^{r_0}_{\phi\delta+1)} \le \overline{\epsilon}\left(1 + \sum_{j=1}^{\delta}\hat{\zeta}^j\overline{\epsilon}^{j-1}\frac{\overline{\epsilon}-1}{(1+\zeta)^j} + \hat{\zeta}^\delta\overline{\epsilon}^\delta\frac{\overline{\eta}-1}{(1+\zeta)^\delta}\right)\mathbf{J}^{h^*,r_0}. \tag{51}$$

Let $\delta = 0$. Based on (34) and (49), we obtain

$$\hat{\mathbf{V}}^{r_0}_{\phi(1)} \le \overline{\epsilon}\mathscr{L}_1\big(\hat{\mathbf{V}}^{r_0}_0\big) = \overline{\epsilon}\mathscr{L}_1\big(\mathbf{V}^{r_0}_0\big) = \overline{\epsilon}\mathbf{V}^{r_0}_{\phi(1)} \le \overline{\epsilon}\overline{\eta}\mathbf{J}^{h^*,r_0}. \tag{52}$$

Define $\Gamma_{\phi(\delta)+m} = L_{\phi(\delta)+m-1}\cdots L_{\phi(\delta)+1}L_{\phi(\delta)}(\hat{\mathbf{V}}^{r_0}_{\phi(\delta)}), \forall m = 1, \ldots, \phi(\delta+1) - \phi(\delta)$, and assume (51) holds for $\delta = l-1$, $l = 1, 2, \ldots$ Then, for $i = \phi(l) + 1$, we have (53), shown at the bottom of the page.

Similar to (36)–(37), by adding $\zeta(\sum_{j=1}^{l}\hat{\zeta}^{j-1}\overline{\epsilon}^{j-1}(\overline{\epsilon} - 1/(1+\zeta)^j) + \hat{\zeta}^{l-1}\overline{\epsilon}^l(\overline{\eta} - 1/(1+\zeta)^l))\mathcal{U}_{\varkappa,\psi,\mathbf{u}}$ to and subtracting the same term from (53), we get

$$\begin{aligned}\Gamma_{\phi(l)+1} \le &\left\{\mathbf{E} + \sum_{j=1}^{l}\hat{\zeta}^{j-1}\overline{\epsilon}^{j-1}\frac{\overline{\epsilon}-1}{(1+\zeta)^{j-1}}\left(\mathbf{E} - \frac{\mathcal{A}_{\phi(l)}}{1+\zeta}\right)\right.\\ &\left.+\hat{\zeta}^{l-1}\overline{\epsilon}^l\frac{\overline{\eta}-1}{(1+\zeta)^{l-1}}\left(\mathbf{E} - \frac{\mathcal{A}_{\phi(l)}}{1+\zeta}\right)\right\}\mathbf{J}^{h^*,r_0}.\end{aligned} \tag{54}$$

In fact, it can further be derived that, for any $m = 1, \ldots, \phi(l+1) - \phi(l)$

$$\begin{aligned}\Gamma_{\phi(l)+m} \le &\left\{\mathbf{E} + \sum_{j=1}^{l}\hat{\zeta}^{j-1}\overline{\epsilon}^{j-1}\frac{\overline{\epsilon}-1}{(1+\zeta)^{j-1}}\left(\mathbf{E} - \frac{\mathcal{A}_{\phi(l)+m-1}}{1+\zeta}\right)\right.\\ &\left.+\hat{\zeta}^{l-1}\overline{\epsilon}^l\frac{\overline{\eta}-1}{(1+\zeta)^{l-1}}\left(\mathbf{E} - \frac{\mathcal{A}_{\phi(l)+m-1}}{1+\zeta}\right)\right\}\mathbf{J}^{h^*,r_0}.\end{aligned} \tag{55}$$

(55) yields

$$\begin{aligned}\Gamma_{\phi(l+1)} &\le \left\{1 + \sum_{j=1}^{l}\hat{\zeta}^{j-1}\overline{\epsilon}^{j-1}\frac{\overline{\epsilon}-1}{(1+\zeta)^{j-1}}\left(1 - \frac{\beta_l}{1+\zeta}\right)\right.\\ &\quad\left.+\hat{\zeta}^{l-1}\overline{\epsilon}^l\frac{\overline{\eta}-1}{(1+\zeta)^{l-1}}\left(1 - \frac{\beta_l}{1+\zeta}\right)\right\}\mathbf{J}^{h^*,r_0}\\ &\le \left(1 + \sum_{j=1}^{l}\hat{\zeta}^j\overline{\epsilon}^{j-1}\frac{\overline{\epsilon}-1}{(1+\zeta)^j} + \hat{\zeta}^l\overline{\epsilon}^l\frac{\overline{\eta}-1}{(1+\zeta)^l}\right)\mathbf{J}^{h^*,r_0}.\end{aligned} \tag{56}$$

Algorithm 1 HEIADP Algorithm

Initialization:

Give the calculation precision ε;
Construct the initial $\mathbf{V}^{r_0}_0$ according to (11);
Use $\mathscr{P}^{\langle\varsigma\rangle} = \mathscr{P}^{\langle\sigma_m\rangle}$ to denote the mth row vector of $\mathscr{P}$;
Define the function $\phi(\delta)$ and denote the sequences of "effective areas" as $\{\hat{\mathcal{B}}_\kappa\} \subseteq \hat{X}$ and $\{\mathcal{B}_\kappa\} \subseteq X, \kappa = 0, 1, \ldots,$ which satisfy (22) and (27).

Iteration:

1: Let $i = 0, \delta = 1$;
2: Obtain the initial triggering strategy $\varkappa_0$, delay function ψ_0, and control law cluster $\mathbf{u}_0$ by (24);
3: Let $i = i + 1$;
4: Obtain the iterative value function $\mathbf{V}^{r_0}_i$ as follows
If $\varsigma \in \hat{\mathcal{B}}_{i-1}$, **then**
$$V^{r_0}_i(\varsigma) = (1 - \alpha_{i-1}(\varsigma))V^{r_0}_{i-1}(\varsigma) + \alpha_{i-1}(\varsigma)\big(\mathcal{U}_{\varkappa_{i-1},\psi_{i-1},\mathbf{u}_{i-1}}(\varsigma) + \mathscr{P}^{\langle\varsigma\rangle}_{\varkappa_{i-1},\psi_{i-1},\mathbf{u}_{i-1}}\mathbf{V}^{r_0}_{i-1}\big),$$
else
$$V^{r_0}_i(\varsigma) = V^{r_0}_{i-1}(\varsigma),$$
end if;
5: If $i < \phi(\delta)$, then, go to Step 8. Else, go to Step 6;
6: If $V^{r_0}_{\iota+1}(\varsigma) - V^{r_0}_\iota(\varsigma) < \alpha_\iota(\varsigma)\mathcal{U}_{\varkappa_\iota,\psi_\iota,\mathbf{u}_\iota}(\varsigma)$ holds for any $\varsigma \in \hat{\mathcal{B}}_\iota\backslash\Omega_{\mathrm{o}}$ and $\phi(\delta-1) \le \iota \le i-1$, then, the current iterative controller is admissible. Go to Step 7. Else, let $\delta = \delta + 1$ and go to Step 8;
7: If $\|\mathbf{V}^{r_0}_i - \mathbf{V}^{r_0}_{\phi(\delta-1)}\| < \varepsilon$, then, the algorithm converges. Go to Step 10. Else, let $\delta = \delta + 1$ and go to Step 8;
8: Derive the iterative triggering strategy $\varkappa_i(\varsigma)$, control law cluster $\mathbf{u}_i(x)$ and delay function $\psi_i(x)$ by
$$(\varkappa_i, \psi_i, \mathbf{u}_i) = \arg\min_{\varkappa\in\mathscr{K}_i,\psi\in\mathscr{M}_i,\mathbf{u}\in\mathscr{C}_i}\big\{\mathcal{U}_{\varkappa,\psi,\mathbf{u}} + \mathscr{P}_{\varkappa,\psi,\mathbf{u}}\mathbf{V}^{r_0}_i\big\};$$
9: Go to Step 3;
10: **return** $\mathbf{V}^{r_0}_i$, $\mathbf{u}_{i-1}(x)$, $\psi_{i-1}(x)$ and $\varkappa_{i-1}(\varsigma)$.

$$\begin{aligned}&\Gamma_{\phi(l)+1}\\ &= \big(\mathbf{E} - \mathcal{A}_{\phi(l)}\big)\hat{\mathbf{V}}^{r_0}_{\phi(l)} + \mathcal{A}_{\phi(l)}\min_{\varkappa,\psi,\mathbf{u}}\Big\{\mathcal{U}_{\varkappa,\psi,\mathbf{u}} + \mathscr{P}_{\varkappa,\psi,\mathbf{u}}\hat{\mathbf{V}}^{r_0}_{\phi(l)}\Big\}\\ &\le \big(\mathbf{E} - \mathcal{A}_{\phi(l)}\big)\hat{\mathbf{V}}^{r_0}_{\phi(l)} + \mathcal{A}_{\phi(l)}\min_{\varkappa,\psi,\mathbf{u}}\left\{\mathcal{U}_{\varkappa,\psi,\mathbf{u}} + \left(\overline{\epsilon} + \sum_{j=1}^{l-1}\hat{\zeta}^j\overline{\epsilon}^j\frac{\overline{\epsilon}-1}{(1+\zeta)^j} + \hat{\zeta}^{l-1}\overline{\epsilon}^l\frac{\overline{\eta}-1}{(1+\zeta)^{l-1}}\right)\mathscr{P}_{\varkappa,\psi,\mathbf{u}}\mathbf{J}^{h^*,r_0}\right\}\end{aligned} \tag{53}$$

Combining (56) with the fact $\hat{\mathbf{V}}^{r_0}_{\phi(l+1)} \leq \overline{\epsilon}\mathscr{L}_l(\hat{\mathbf{V}}^{r_0}_{\phi(l)}) = \overline{\epsilon}\Gamma_{\phi(l+1)}$, we have

$$\hat{\mathbf{V}}^{r_0}_{\phi(l+1)} \leq \overline{\epsilon}\left(1+\sum_{j=1}^{l}\hat{\zeta}^j\overline{\epsilon}^{j-1}\frac{\overline{\epsilon}-1}{(1+\zeta)^j}+\hat{\zeta}^l\overline{\epsilon}^l\frac{\overline{\eta}-1}{(1+\zeta)^l}\right)\mathbf{J}^{h^*,r_0}. \tag{57}$$

Using the same techniques as (52)–(57), we can prove

$$\hat{\mathbf{V}}^{r_0}_{\phi(\delta+1)} \geq \underline{\epsilon}\left(1+\sum_{j=1}^{\delta}\hat{\zeta}^j\underline{\epsilon}^{j-1}\frac{\underline{\epsilon}-1}{(1+\zeta)^j}+\hat{\zeta}^\delta\underline{\epsilon}^\delta\frac{\underline{\eta}-1}{(1+\zeta)^\delta}\right)\mathbf{J}^{h^*,r_0}. \tag{58}$$

Let $\underline{q} = (\hat{\zeta}\underline{\epsilon}/1+\zeta)$. Then, as $\delta \to \infty$, (57) and (58) become

$$\begin{aligned}\lim_{\delta\to\infty}\underline{\epsilon}&\left(1+\hat{\zeta}\frac{\underline{\epsilon}-1}{1+\zeta}\frac{1-\underline{q}^\delta}{1-\underline{q}}+\hat{\zeta}^\delta\underline{\epsilon}^\delta\frac{\underline{\eta}-1}{(1+\zeta)^\delta}\right)\mathbf{J}^{h^*,r_0}\\ &=\frac{\underline{\epsilon}(1+\zeta-\hat{\zeta})}{1+\zeta-\hat{\zeta}\underline{\epsilon}}\mathbf{J}^{h^*,r_0}\leq\hat{\mathbf{V}}^{r_0}_{\infty}\leq\frac{\overline{\epsilon}(1+\zeta-\hat{\zeta})}{1+\zeta-\hat{\zeta}\overline{\epsilon}}\mathbf{J}^{h^*,r_0}\end{aligned} \tag{59}$$

which completes the proof. ■

V. Simulation Example

Consider the stochastic process $\{x(k)\}$ generated by the dynamics $\mathcal{F}(x,a,\omega)$ which is expressed as

$$\begin{aligned}\begin{bmatrix}x_1(k+1)\\x_2(k+1)\end{bmatrix} = \ &\Delta T\begin{bmatrix}-x_1(k)+a(k)x_2(k)\\ \sin(a(k))\cos^2(x_1(k))-x_2(k)\end{bmatrix}\\ &+\begin{bmatrix}x_1(k)+\Delta T\omega_1(k)\\ x_2(k)+\Delta T(a(k)+\sin(a(k))+\omega_2(k))\end{bmatrix}.\end{aligned} \tag{60}$$

The symbol $x = \begin{bmatrix}x_1\\x_2\end{bmatrix}$ represents the system state, a represents the control action (a is the output of $\varpi^{\langle 2\rangle}$, and $\overline{a}$ is the output of $\varpi^{\langle 1\rangle}$), $\omega = \begin{bmatrix}\omega_1\\\omega_2\end{bmatrix}$ represents the random variable and $\Delta T = 0.01$. Due to the existence of ω, (60) is essentially a stochastic system (see Chapters 3–5 of [10] for more explanations). Both the state and control spaces are finite and countable. All possible impulsive intervals of ETIC are given by $\mathcal{G} = \{\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_4, \mathcal{T}_5\}$, where $\mathcal{T}_1 = 1, \mathcal{T}_2 = 2, \mathcal{T}_3 = 3$, $\mathcal{T}_4 = 4$ and $\mathcal{T}_5 = 5$. By constructing the corresponding register r, recorder $\hat{r}$ and the "expanded" state e, the utility function is defined by $U(e,[\overline{a},a]^\mathsf{T}) = Q(e,a) + \pi(\overline{a})\varrho(e,a)$ where

$$Q(e,a) = x^\mathsf{T}\mathbf{Q}x + a^\mathsf{T}\mathbf{Z}_{e,a}a \tag{61}$$

$$\varrho(e,a) = x^\mathsf{T}\mathbb{Q}x + a^\mathsf{T}\mathbb{Z}a. \tag{62}$$

$\mathbf{Q} = 0.8 * \Upsilon, \mathbb{Q} = \mathbb{Z} = 0.2 * \Upsilon$ (Υ is the two dimensional unit matrix) and $\mathbf{Z}_{e,a}$ is

$$\mathbf{Z}_{e,a} = \begin{cases}0.95*\Upsilon, & \hat{r}'=5, r=1\\ 1.11*\Upsilon, & \hat{r}'=4, r=1\\ 1.6*\Upsilon, & \hat{r}'=3, r=1\\ 2.65*\Upsilon, & \hat{r}'=2, r=1\\ 6.51*\Upsilon, & a\neq 0, r=0\\ \mathbf{0}, & \text{otherwise.}\end{cases} \tag{63}$$

$Q(e,a)$ is the immediate cost regarding the current expanded state e and the corresponding impulsive action a at each time k. In particular, $\mathbf{Z}_{e,a}$ in Q is constructed in order to suppress the abuse of high frequency impulsive actions, thus guaranteeing the healthy operations of the impulsive controllers/actuators and systems. Notice that the value of the recorder $\hat{r}'$ represents the time interval from the current impulsive action to the following one. Hence, with $\hat{r}'$ stepping down, the frequency of the impulsive actions goes up, and the penalty $\mathbf{Z}_{e,a}$ is accordingly strengthened and increased. Besides, $\varrho(e,a)$ is the computational and communication cost caused by the ETIC resampling the system state and updating its output.

By utilizing the MPS task scheduling scheme in Algorithm 2, it is guaranteed that condition (27) is satisfied. Then, according to Theorem 3, the concurrent HEIADP should theoretically approach to the impulsive optimum of (60).

Algorithm 2 MPS Task Scheduling Scheme for the Concurrent HEIADP Algorithm

Initialization:

Let $i = 0, \delta = 0$;
Denote the processors in the MPS as $c_1, c_2, \ldots, c_n$;
Let $\phi(0) = 0$;
Construct the sequences of state subsets as $\{\hat{\mathscr{B}}_\kappa\} \subseteq \hat{X}$ and $\{\mathscr{B}_\kappa\} \subseteq X, \kappa = 0, 1, \ldots$, where $\mathscr{B}_0 = \hat{\mathscr{B}}_0 = \emptyset$;
Divide the global original state space X into disjoint nonempty subsets $X_{0,1}, X_{0,2}, \ldots, X_{0,n}$, such that $X = \cup_{j=1}^n X_{0,j}$;
Divide the global impulsive state space $\hat{X}$ into subsets $\hat{X}_{0,1}, \hat{X}_{0,2}, \ldots, \hat{X}_{0,n}$, according to (21);
$\forall j = 1, 2, \ldots, n$, send $X_{0,j}$ and $\hat{X}_{0,j}$ to the central processor c_j.

Iteration:

1: $\forall j = 1, \ldots, n$, processor c_j generates and gathers the required sample set[1] for training $(\hat{\mathbf{u}}_i, \hat{\psi}_i, \hat{x}_i, \hat{\mathbf{V}}^{r_0}_{i+1})$ at $X_{i,j}/\hat{X}_{i,j}$ according to (12) and (13);
2: The first processor in the MPS to complete the assigned sampled data generation task informs the others to terminate their ongoing sampled data generation tasks simultaneously;
3: Obtain the corresponding effective areas $\mathcal{B}_i$ and $\hat{\mathcal{B}}_i$ of the currently acquired sample set from all processors;
4: **If** $\mathscr{B}_i \cup \mathcal{B}_i = X$ and $\hat{\mathscr{B}}_i \cup \hat{\mathcal{B}}_i = \hat{X}$, **then**

$$\mathscr{B}_{i+1} = \hat{\mathscr{B}}_{i+1} = \emptyset$$
$$\delta = \delta + 1$$
$$\phi(\delta) = i + 1$$

else

$$\mathscr{B}_{i+1} = \mathscr{B}_i \cup \mathcal{B}_i$$
$$\hat{\mathscr{B}}_{i+1} = \hat{\mathscr{B}}_i \cup \hat{\mathcal{B}}_i$$

end if;
5: The NPU trains $(\hat{\mathbf{u}}_i, \hat{\psi}_i, \hat{x}_i, \hat{\mathbf{V}}^{r_0}_{i+1})$ at $\mathcal{B}_i$ or $\hat{\mathcal{B}}_i$, while letting them inherit the values of their corresponding predecessors at $X\backslash\mathcal{B}_i$ or $\hat{X}\backslash\hat{\mathcal{B}}_i$;
6: Let $i = i + 1$;
7: Construct the subsets $X_{i,1}, X_{i,2}, \ldots, X_{i,n}$, such that $X\backslash\mathscr{B}_i \subseteq \cup_{j=1}^n X_{i,j}$;
8: Construct the subsets $\hat{X}_{i,1}, \hat{X}_{i,2}, \ldots, \hat{X}_{i,n}$, according to (21);
9: $\forall j = 1, 2, \ldots, n$, send $X_{i,j}$ and $\hat{X}_{i,j}$ to the central processor c_j, and go to Step 1.

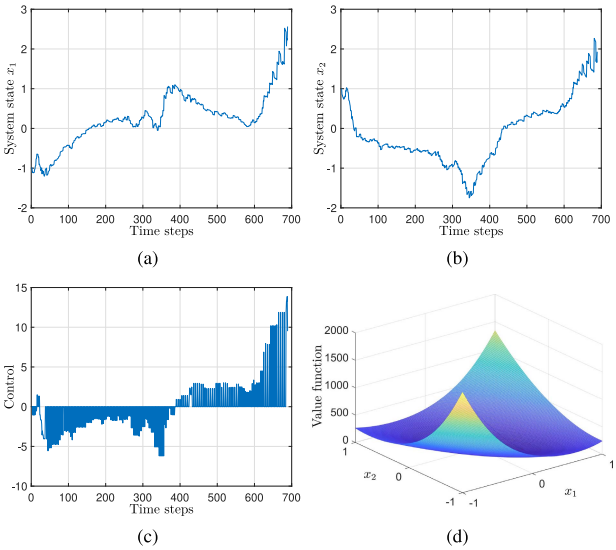


Fig. 7. (a) Initial x_1 trajectory. (b) Initial x_2 trajectory. (c) Initial ETIC control trajectory. (d) Initial impulsive value function.

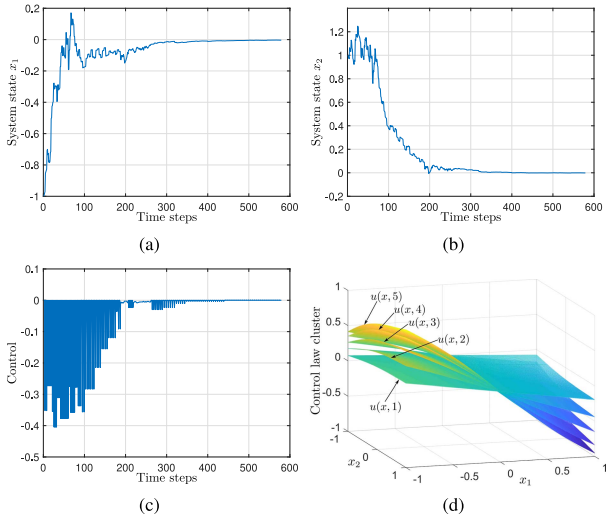


Fig. 8. Converged items by HEIADP. (a) Optimal x_1 trajectory. (b) Optimal x_2 trajectory. (c) Optimal ETIC control trajectory. (d) Optimal control law cluster.

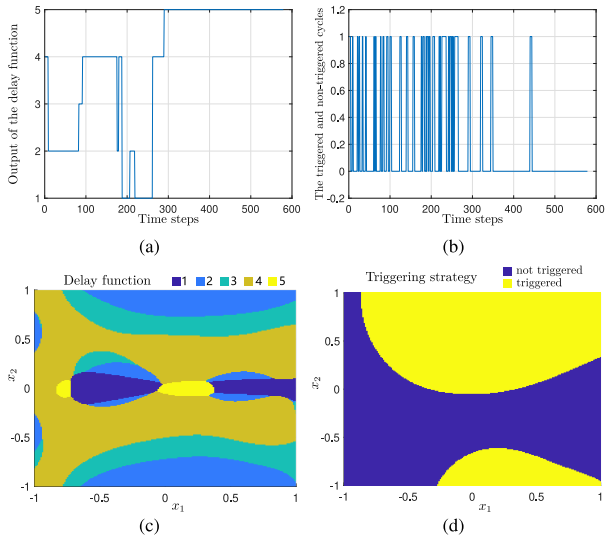


Fig. 9. Optimal ψ^* and $\varkappa^*$ by HEIADP. (a) Output trajectory of ψ^*. (b) Triggered and nontriggered cycles. (c) Mapping rule of ψ^*. (d) Mapping rule of $\varkappa^*$.

The trajectories of the system states with $x(0) = [-1, 1]^\mathsf{T}$ are demonstrated in Fig. 7(a) and (b) which are subject to the initial iterative ETIC. Fig. 7(c) illustrates the trajectory of the control input by the initial ETIC, which is obviously not admissible judging from Fig. 7(a)–(c). Fig. 7(d) shows the initial impulsive value function $\mathbf{V}_0^{r_0}$ with the recorder $\hat{r}$ fixed as $\hat{r} = [1, 0]^\mathsf{T}$. Fig. 8(a) and (b) give the trajectories of the system states controlled by the optimal ETIC obtained by HEIADP. In addition, Fig. 8(c) represents the control trace of the optimal ETIC, while Fig. 8(d) shows the optimal control law cluster obtained by HEIADP. The trace of the optimal ψ^* is shown by Fig. 9(a). Fig. 9(b) shows the triggered and nontriggered impulsive control cycles as the system is running, where the values 1 and 0 indicate the triggered and nontriggered cycles, respectively. Fig. 9(c) displays the mapping rule of the optimal ψ^* (each state x is assigned to a color which represents the output $\psi^*(x) \in \mathcal{G} = \{1, 2, 3, 4, 5\}$) which is responsible for determining the optimal impulsive control cycle for the current state. Similarly, Fig. 9(d) shows the mapping rule of $\varkappa^*$, where the recorder $\hat{r}$ is specified by $\hat{r} = [1, 0.25]^\mathsf{T}$, and the blue and yellow colors indicate "CUE is not triggered" and "CUE is triggered," respectively, for the corresponding state. The converged impulsive performance index functions of ETIADP and HEIADP are demonstrated in Fig. 10(a) and (b). Through Fig. 10(a) and (b), we get the conclusion that with the MPS task scheduling scheme in Algorithm 2 utilized, the condition (27) is satisfied and HEIADP approaches the same impulsive optimum as ETIADP does, thus verifying the results in Theorem 3 along with Remark 3. In the experiment, the algorithm is terminated at the iteration step $i^* = \phi(\delta^*)$. Fig. 10(c) shows that $V_{i+1}^{r_0}(\varsigma) - V_i^{r_0}(\varsigma) - \mathcal{U}_i(\varsigma) < 0$ holds when $i = \phi(\delta^* - 1)$ and $\varsigma \in \{\varsigma | \hat{r} = [1, 0]^\mathsf{T}, \varsigma \notin \Omega_o\}$. In fact, the experimental results also show that the above inequality is true for any $\phi(\delta^* - 1) \le i \le \phi(\delta^*) - 1$ and $\varsigma \notin \Omega_o$. This fact indicates that the admissibility criteria (45) holds, which guarantees that the obtained ETIC is admissible.

Now let $\varkappa \equiv 1$. Then, the ETIADP or HEIADP turns into the time-triggered ADP algorithm. Under this condition, the converged value function and state and control trajectories are illustrated in Figs. 10(d)–(f). Once the triggering strategy is fixed as $\varkappa \equiv 1$, then, the controller updates periodically, consuming more computational and communication resources, and the policy space becomes $\Xi^t = \mathcal{M}_0 \times \mathbf{C}_0 \times \mathcal{M}_1 \times \mathbf{C}_1 \times \cdots$ in which the time-triggered ADP finds the optimal time-triggered controller. However, Ξ^t has a lower volume than the event-triggered impulsive policy space associated with the proposed methods, i.e., $\Xi^t \subset \Xi = \mathcal{K}_0 \times \mathcal{M}_0 \times \mathbf{C}_0 \times \mathcal{K}_1 \times \mathcal{M}_1 \times \mathbf{C}_1 \times \cdots$. Therefore, the performance of the optimal ETIC in Ξ should theoretically be better than the optimal time-triggered controller in Ξ^t, which is validated in Fig. 10(b).

Fig. 11 compares the memory/CPU usage of the concurrent ETIADP with that of the concurrent HEIADP. From Fig. 11(a), it is noticed that the CPU utilization rate of the ETIADP algorithm significantly drops at some time periods, during which some processors in the MPS finish their sample generation tasks of the current iteration earlier than the others, thus transitioning to the idle state and waiting for the others to catch up. This unsynchronization across processors cause

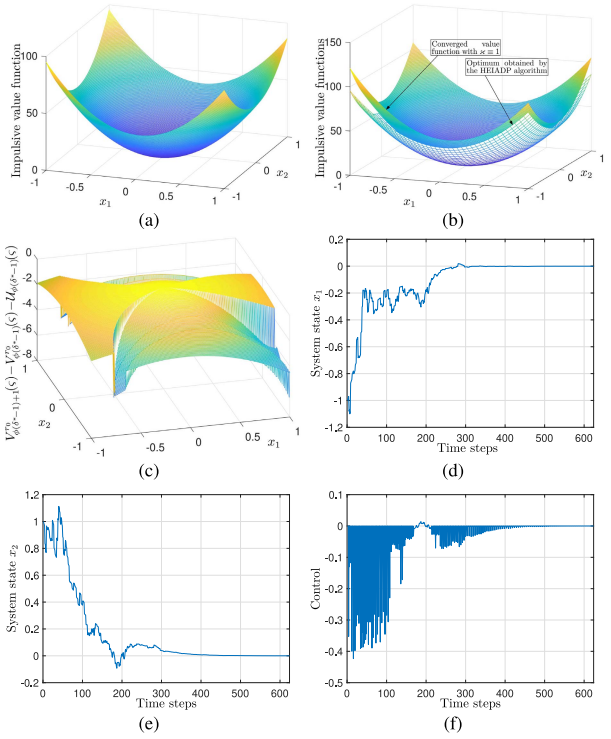


Fig. 10. (a) Optimal impulsive value function obtained by ETIADP. (b) Converged impulsive value functions by HEIADP. (c) Admissibility criteria. (d) Optimal state trajectory when $\varkappa \equiv 1$. (e) Optimal state trajectory when $\varkappa \equiv 1$. (f) Optimal control trajectory when $\varkappa \equiv 1$.

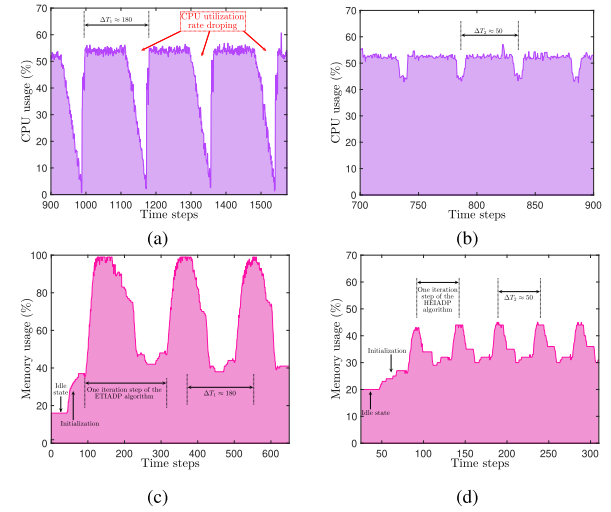


Fig. 11. (a) CPU usage as the concurrent ETIADP is running. (b) CPU usage as the concurrent HEIADP is running. (c) Memory usage as the concurrent ETIADP is running. (d) Memory usage as the concurrent HEIADP is running.

the algorithm not fully utilizing the computing resources of the MPS. In contrast, in Fig. 11(b), the CPUs are properly loaded the whole execution time of the concurrent HEIADP algorithm, due to the novel MPS task scheduling scheme. As for the memory consumption, Fig. 11(c) and (d) show that ETIADP (which represents the traditional ADP-based methods wherein the iterative items are updated globally) has a peak memory usage of nearly 100%. In contrast, HEIADP introduces a novel updating mechanism which reduces the memory usage to around 20%–25%, thus more suitable to run on computing devices with limited memory sizes. Therefore, based on the above experimental results, HEIADP can effectively improves the operating efficiency (in terms of CPU utilization and memory footprint) compared to the traditional ADP-based approaches, while approaching the optimum.

VI. Conclusion

To obtain the optimal ETIC of the stochastic systems, the ETIADP is proposed with its convergence and error boundedness properties analyzed. The HEIADP is also developed to fully utilize the computing resources of MPSs. The effectiveness of the methods is verified by the numerical study.

References

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

[2] D. Liu and Q. Wei, "Policy iteration adaptive dynamic programming algorithm for discrete-time nonlinear systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 3, pp. 621–634, Mar. 2014.

[3] D. Liu, S. Xue, B. Zhao, B. Luo, and Q. Wei, "Adaptive dynamic programming for control: A survey and recent advances," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 1, pp. 142–160, Jan. 2021.

[4] D. Liu, Y. Xu, Q. Wei, and X. Liu, "Residential energy scheduling for variable weather solar energy based on adaptive dynamic programming," *IEEE/CAA J. Autom. Sinica*, vol. 5, no. 1, pp. 36–46, Jan. 2018.

[5] D. Liu, Q. Wei, D. Wang, X. Yang, and H. Li, *Adaptive Dynamic Programming With Applications in Optimal Control*. Cham, Switzerland: Springer, 2017.

[6] 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. Neural Netw. Learn. Syst.*, vol. 25, no. 2, pp. 418–428, Feb. 2014.

[7] Q. Wei, D. Liu, Y. Liu, and R. Song, "Optimal constrained self-learning battery sequential management in microgrid via adaptive dynamic programming," *IEEE/CAA J. Autom. Sinica*, vol. 4, no. 2, pp. 168–176, Apr. 2017.

[8] D. Liu, X. Yang, D. Wang, and Q. Wei, "Reinforcement-learning-based robust controller design for continuous-time uncertain nonlinear systems subject to input constraints," *IEEE Trans. Cybern.*, vol. 45, no. 7, pp. 1372–1385, Jul. 2015.

[9] A. Al-Tamimi, F. L. Lewis, and M. Abu-Khalaf, "Discrete-time nonlinear HJB solution using approximate dynamic programming: Convergence proof," *IEEE Trans. Syst., Man, Cybern., B, Cybern.*, vol. 38, no. 4, pp. 943–949, Aug. 2008.

[10] W. B. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality*, 2nd ed. Hoboken, NJ, USA: Wiley, 2011.

[11] Q. Wei, D. Liu, and H. Lin, "Value iteration adaptive dynamic programming for optimal control of discrete-time nonlinear systems," *IEEE Trans. Cybern.*, vol. 46, no. 3, pp. 840–853, Mar. 2016.

[12] Q. Wei, F. L. Lewis, D. Liu, R. Song, and H. Lin, "Discrete-time local value iteration adaptive dynamic programming: Convergence analysis," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 48, no. 6, pp. 875–891, Jun. 2018.

[13] Q. Wei, D. Liu, and Q. Lin, "Discrete-time local value iteration adaptive dynamic programming: Admissibility and termination analysis," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 11, pp. 2490–2502, Nov. 2017.

[14] B. Pang and Z.-P. Jiang, "Adaptive optimal control of linear periodic systems: An off-policy value iteration approach," *IEEE Trans. Autom. Control*, vol. 66, no. 2, pp. 888–894, Feb. 2021.

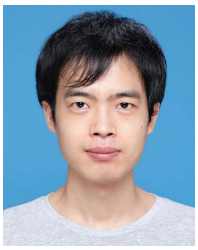
[15] S. Bhattacharya, S. Badyal, T. Wheeler, S. Gil, and D. Bertsekas, "Reinforcement learning for POMDP: Partitioned rollout and policy iteration with application to autonomous sequential repair problems," *IEEE Robot. Autom. Lett.*, vol. 5, no. 3, pp. 3967–3974, Jul. 2020.

[16] R. Song and L. Zhu, "Stable value iteration for two-player zero-sum game of discrete-time nonlinear systems based on adaptive dynamic programming," *Neurocomputing*, vol. 340, pp. 180–195, May 2019.

[17] Y. Zhu and D. Zhao, "Online minimax Q network learning for two-player zero-sum Markov games," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 3, pp. 1228–1241, Mar. 2022.

[18] A. G. Khiabani and A. Heydari, "Optimal torque control of permanent magnet synchronous motors using adaptive dynamic programming," *IET Power Electron.*, vol. 13, no. 12, pp. 2442–2449, Sep. 2020.

[19] T. Sardarmehni and A. Heydari, "Sub-optimal switching in anti-lock brake systems using approximate dynamic programming," *IET Control Theory Appl.*, vol. 13, no. 9, pp. 1413–1424, Jun. 2019.
[20] A. Heydari, "Optimal switching of DC–DC power converters using approximate dynamic programming," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 3, pp. 586–596, Mar. 2018.
[21] W. M. Haddad, V. Chellaboina, and N. A. Kablar, "Non-linear impulsive dynamical systems. Part I: Stability and dissipativity," *Int. J. Control*, vol. 74, no. 17, pp. 1631–1658, 2001.
[22] X. Li, J. Cao, and D. W. C. Ho, "Impulsive control of nonlinear systems with time-varying delay and applications," *IEEE Trans. Cybern.*, vol. 50, no. 6, pp. 2661–2673, Jun. 2020.
[23] V. Lakshmikantham, D. D. Bainov, and P. S. Simeonov, *Theory of Impulsive Differential Equations*. Singapore: World Scientific, 1989.
[24] F. Dufour, M. Horiguchi, and A. B. Piunovskiy, "Optimal impulsive control of piecewise deterministic Markov processes," *Stochastics*, vol. 88, no. 7, pp. 1073–1098, Jun. 2016.
[25] A. Miller, B. Miller, and K. Stepanyan, "A numerical approach to joint continuous and impulsive control of Markov chains," *IFAC-PapersOnLine*, vol. 51, no. 32, pp. 462–467, 2018.
[26] A. Basu and Ł. Stettner, "Zero-sum Markov games with impulse controls," *SIAM J. Control Optim.*, vol. 58, no. 1, pp. 580–604, Feb. 2020.
[27] F. Dufour and A. B. Piunovskiy, "Impulsive control for continuous-time Markov decision processes," *Adv. Appl. Probab.*, vol. 47, no. 1, pp. 106–127, Jan. 2016.
[28] A. Heydari, "Optimal impulsive control using adaptive dynamic programming and its application in spacecraft rendezvous," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 10, pp. 4544–4552, Oct. 2021.
[29] Q. Wei, R. Song, Z. Liao, B. Li, and F. L. Lewis, "Discrete-time impulsive adaptive dynamic programming," *IEEE Trans. Cybern.*, vol. 50, no. 10, pp. 4293–4306, Oct. 2020.
[30] X. Wang and S. N. Balakrishnan, "Optimal neurocontroller synthesis for impulse-driven systems," *Neural Netw.*, vol. 23, no. 1, pp. 125–134, Jan. 2010.
[31] X. Wang, Y. Huang, H. Wang, and S. N. Balakrishnan, "Variable time impulse system optimization with continuous control and impulse control," *Asian J. Control*, vol. 16, no. 1, pp. 107–116, Jan. 2014.
[32] K. Avrachenkov, O. Habachi, A. Piunovskiy, and Y. Zhang, "Infinite horizon optimal impulsive control with applications to internet congestion control," *Int. J. Control*, vol. 88, no. 4, pp. 703–716, Nov. 2014.
[33] F. Cacace, V. Cusimano, and P. Palumbo, "Optimal impulsive control with application to antiangiogenic tumor therapy," *IEEE Trans. Control Syst. Technol.*, vol. 28, no. 1, pp. 106–117, Jan. 2020.
[34] S. H. Hou and K. H. Wong, "Optimal impulsive control problem with application to human immunodeficiency virus treatment," *J. Optim. Theory Appl.*, vol. 151, no. 2, pp. 385–401, Apr. 2011.
[35] K. G. Vamvoudakis, "Event-triggered optimal adaptive control algorithm for continuous-time nonlinear systems," *IEEE/CAA J. Autom. Sinica*, vol. 1, no. 3, pp. 282–293, Jul. 2014.
[36] D. Wang, C. Mu, H. He, and D. Liu, "Event-driven adaptive robust control of nonlinear systems with uncertainties through NDP strategy," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 47, no. 7, pp. 1358–1370, Jul. 2017.
[37] B. Luo, Y. Yang, D. Liu, and H. Wu, "Event-triggered optimal control with performance guarantees using adaptive dynamic programming," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 1, pp. 76–88, Jan. 2020.
[38] C. Mu, K. Wang, and T. Qiu, "Dynamic event-triggering neural learning control for partially unknown nonlinear systems," *IEEE Trans. Cybern.*, vol. 52, no. 4, pp. 2200–2213, Apr. 2022.
[39] B. Zhao and D. Liu, "Event-triggered decentralized tracking control of modular reconfigurable robots through adaptive dynamic programming," *IEEE Trans. Ind. Electron.*, vol. 67, no. 4, pp. 3054–3064, Apr. 2020.
[40] S. Xue, B. Luo, and D. Liu, "Event-triggered adaptive dynamic programming for unmatched uncertain nonlinear continuous-time systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 7, pp. 2939–2951, Jul. 2021.
[41] S. Xue, B. Luo, D. Liu, and Y. Gao, "Event-triggered ADP for tracking control of partially unknown constrained uncertain systems," *IEEE Trans. Cybern.*, vol. 52, no. 9, pp. 9001–9012, Sep. 2022.



Mingming Liang (Member, IEEE) received the B.S. degree in automation from the Dalian University of Technology, Dalian, China, in 2015, and the Ph.D. degree in control theory and control engineering from the University of Chinese Academy of Sciences, Beijing, China, in 2020.

From 2020 to 2022, he was a Post-Doctoral Fellow with the School of Automation, Guangdong University of Technology, Guangzhou, China. He is currently an engineer with BYD Auto Industry Co., Ltd., Shenzhen, China. His current research interests include adaptive dynamic programming, optimal control, and stochastic processes.

Derong Liu (Fellow, IEEE) received the B.S. degree in mechanical engineering from the Nanjing University of Science and Technology, Nanjing, China, in 1982, the M.S. degree in automatic control theory and applications from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 1987, and the Ph.D. degree in electrical engineering from the University of Notre Dame, Notre Dame, IN, USA, in 1994.

He was a Product Design Engineer with the China North Industries Corporation, Jilin, China, from 1982 to 1984. He was an Instructor with the Graduate School of Chinese Academy of Sciences, Beijing, from 1987 to 1990. He was a Staff Fellow with the General Motors Research and Development Center, from 1993 to 1995. He was an Assistant Professor with the Department of Electrical and Computer Engineering, Stevens Institute of Technology, from 1995 to 1999. He joined the University of Illinois Chicago, Chicago, IL, USA, in 1999, and became a Full Professor of electrical and computer engineering and computer science in 2006. He was selected for the 100 Talents Program by the Chinese Academy of Sciences in 2008, and he served as the Associate Director for The State Key Laboratory of Management and Control for Complex Systems with the Institute of Automation, from 2010 to 2016. He is currently a Chair Professor with the Southern University of Science and Technology, Shenzhen, China. He has published 13 books and 270 papers in international journals.

Dr. Liu is a fellow of the International Neural Network Society and the International Association for Pattern Recognition and a member of Academia Europaea (The Academy of Europe). He was elected three times AdCom Member of the IEEE Computational Intelligence Society in 2006, 2015, and 2022, respectively. He was the Editor-in-Chief of the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, from 2010 to 2015. He was elected twice Distinguished Lecturer of the IEEE Computational Intelligence Society in 2012 and 2016, respectively. He served as a member for the Council of International Federation of Automatic Control from 2014 to 2017 and he served as the President for Asia Pacific Neural Network Society in 2018. He was a General Chair of the 2014 IEEE World Congress on Computational Intelligence, the 2016 World Congress on Intelligent Control and Automation, and the 2017 International Conference on Neural Information Processing. He received the Faculty Early Career Development Award from the National Science Foundation in 1999, the University Scholar Award from University of Illinois from 2006 to 2009, the Overseas Outstanding Young Scholar Award from the National Natural Science Foundation of China in 2008, and the Outstanding Achievement Award from Asia Pacific Neural Network Assembly in 2014. He received the International Neural Network Society's Gabor Award in 2018; the IEEE Systems, Man and Cybernetics Society Andrew P. Sage Best Transactions Paper Award in 2018; the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS Outstanding Paper Award in 2018; the IEEE/CCA J. Automatica Sinica Hsue-Shen Tsien Paper Award in 2018. He is a recipient of the IEEE CIS Neural Network Pioneer Award in 2022. He has been named as a highly cited researcher consecutively for six years from 2017 to 2022 by Clarivate. He was a plenary/keynote speaker at 35 international conferences. He is currently the Editor-in-Chief of *Artificial Intelligence Review*, the Deputy Editor-in-Chief of the IEEE/CAA JOURNAL OF AUTOMATICA SINICA, the Deputy Editor-in-Chief of the *CAAI Artificial Intelligence Research*, and the Chair IEEE Guangzhou Section.