Letter

Deep Reinforcement Learning or Lyapunov Analysis? A Preliminary Comparative Study on Event-Triggered Optimal Control

Jingwei Lu, *Member, IEEE*, Lefei Li, *Member, IEEE*, Qinglai Wei, *Senior Member, IEEE*, and Fei-Yue Wang, *Fellow, IEEE*

Dear Editor,

This letter develops a novel method to implement event-triggered optimal control (ETOC) for discrete-time nonlinear systems using parallel control and deep reinforcement learning (DRL), referred to as Deep-ETOC. The developed Deep-ETOC method introduces the communication cost into the performance index through parallel control, so that the developed method enables control systems to learn ETOC policies directly without triggering conditions. Then, dueling double deep Q-network (D3QN) is utilized to achieve our method. In simulations, we present a preliminary comparative study of DRL and Lyapunov analysis for ETOC.

Event-triggered control (ETC) has been brought to the fore in recent years due to the increasing complexity of systems [1]–[5]. In contrast to the control mode of periodic sampling, ETC updates controls when an "event" triggers. The aperiodic sampling nature of ETC allows control systems to substantially save communication and computation resources.

While saving communication resources, many control systems still need to be designed with the consideration of optimizing the given performance index, which has resulted in the formation of ETOC [6]–[9]. The existing ETOC methods are ordinarily formulated by means of Lyapunov analysis. Through the application of the Lyapunov methodology, triggering conditions are derived by ensuring the closed-loop stability. As for optimizing the performance index, reinforcement learning (RL) is employed to solve the event-triggered Hamilton-Jacobi-Bellman equation. In addition to RL, parallel control plays a pivotal role in controlling complex systems [10]–[15]. Notably, Wang *et al*. [16] formally proposed a new parallel control framework by introducing controls into the feedback system and the new control framework has been utilized for ETOC [17]. Since these methods employ Lyapunov analysis to ensure stability, the above ETOC method can be referred to as Lyapunov stability-based ETOC methods. Considering the impressive performance of deep learning in complex tasks, it is a natural fit for applying deep RL (DRL) to achieve ETC [18]. This type of method can be termed DRL-based ETC. However, DRL-based ETC methods only convert "designing triggering conditions" to "training triggering conditions" and fail to discuss ETOC in terms of performance indices. Meanwhile, Lyapunov stability-based ETOC methods are mostly concerned with the system stability and rarely explore the performance of communication cost reduction.

To facilitate the development of ETOC, this letter develops Deep-ETOC based on parallel control and DRL, and provides a comparative discussion of DRL and Lyapunov analysis for ETOC. The main contributions are as follows.

1) The developed Deep-ETOC method introduces the communication cost into the performance index and learns the ETOC policy using this performance index.

2) Compared to existing Lyapunov stability-based ETOC methods, the developed Deep-ETOC method can deeply reduce the usage of communication resources.

3) We present a comparative study of DRL and Lyapunov stability-based ETOC methods in simulations and fully discuss their strengths and weaknesses.

Task formulation: Consider a discrete-time system

$$x_{k+1} = F(x_k, u_k) \tag{1}$$

where $x_k \in \Omega \subset \mathbb{R}^n$ and $u_k \in \mathbb{R}^m$ denote separately the state vector and the control vector with Ω being a compact set, and $F(x_k, u_k): \mathbb{R}^{n+m} \to \mathbb{R}^n$ presents the system dynamics.

Task 1: Optimal Control. Infinite-time optimal control seeks to stabilize system (1) and minimize the performance index

$$J(x_0, u_0) = \sum_{k=0}^{\infty} \gamma^k U(x_k, u_k) \tag{2}$$

with $U(x_k, u_k) \geq 0$ and $\gamma \in [0, 1]$.

Task 2: ETOC. In Task 1, the control vector u_k is required to be updated according to the state vector x_k at every time instant k. In contrast to this, ETC updates its control vector only at triggering instants, which are denoted by $\{k_i\}$ with $k_0 = 0$, $k_i < k_{i+1}$, $i \in \mathbb{N}$. During $[k_i, k_{i+1})$, the control vector is held constant. As for ETOC, we not only seek to determine triggering instants but also consider the performance index (2).

Methodology: For Tasks 1 and 2, a commonly used state-feedback control policy can be described by $u_k = K(x_k)$ where $K(\cdot)$ is a control function. Parallel control is different, and one of parallel control is [13], [16]

$$u_k = \mathcal{K}(x_k, u_{k-1}) \tag{3}$$

with $\mathcal{K}(\cdot)$ being a dynamic control policy to achieve tasks.

To achieve parallel control (3), defining the virtual control vector $v_k = u_k - u_{k-1}$ and the augmented state vector $s_k = [x_k^T, u_{k-1}^T]^T$ [13], one gets an augmented system (AS)

$$s_{k+1} = \mathcal{F}(s_k, v_k) \tag{4}$$

with $\mathcal{F}(s_k, v_k) = [(F(x_k, u_{k-1} + v_k))^T, u_{k-1}^T]^T + \mathcal{G} v_k$ and $\mathcal{G} = [0_{m\times n}, I_m]^T \in \mathbb{R}^{(n+m)\times m}$.

So far, we are able to achieve our goal by designing

$$v_k = \pi(s_k) \tag{5}$$

with $\pi(\cdot)$ being a control function for the AS (4), and we transform the design of the control policy (3) for system (1) into the design of the control policy (5) for the AS (4). The control policy (3) can be obtained by $u_k = u_{k-1} + v_k = u_{k-1} + \pi(s_k)$, and $\mathcal{K}(x_k, u_{k-1}) = u_{k-1} + \pi(s_k)$.

Based on the AS (4), we now introduce a new performance index considering the communication loss

$$\mathcal{J}(s_0, v_0) = \sum_{k=0}^{\infty} \gamma^k \mathcal{U}(s_k, v_k) \tag{6}$$

where $\mathcal{U}(s_k, v_k)$ is an improved utility function and is

$$\mathcal{U}(s_k, v_k) = U(x_k, u_k) + M(v_k) + c_e \mathbb{I}_e(v_k) \tag{7}$$

with $M(v_k)$ being a semi-positive definite function, $c_e \geq 0$ being a constant for ETC, and $\mathbb{I}_e(v_k)$ being an indicator function for ETC,

Corresponding author: Fei-Yue Wang.

Citation: J. Lu, L. Li, Q. Wei, and F.-Y. Wang, "Deep reinforcement learning or Lyapunov analysis? A preliminary comparative study on event-triggered optimal control," *IEEE/CAA J. Autom. Sinica*, vol. 11, no. 7, pp. 1702–1704, Jul. 2024.

J. Lu and L. Li are with the Department of Industrial Engineering, Tsinghua University, Beijing 100084, China (e-mail: lujingwei@tsinghua.edu.cn; lilefei@tsinghua.edu.cn).

Q. Wei and F.-Y. Wang are with The State Key Laboratory for Management and Control of Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China (e-mail: qinglai.wei@ia.ac.cn; feiyue.wang@ia.ac.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JAS.2024.124434

i.e.,

$$\mathbb{I}_e(v_k)=\begin{cases}1, & \text{if } \|v_k\|\neq 0\\ 0, & \text{else.}\end{cases}$$

Apparently, the triggering instant is determined by

$$k_{i+1}=\inf\left\{k\middle|\|v_k\|\neq 0, k>k_i\right\}. \tag{8}$$

Remark 1: It is obvious that, at the instant k, if $\|v_k\|=0$, u_k remains unchanged because of $v_k=u_k-u_{k-1}$; if $\|v_k\|\neq 0$, u_k has to be changed, that is, the event is triggered. Consequently, we can directly penalize control policies to achieve ETC according to v_k. A distinct advantage of the utility function (7) is that it effectively reflects the communication loss caused by ETC, and it is almost infeasible to construct the utility function (7) based on the control policy $u_k=K(x_k)$.

Based on the AS (4) and the performance index (6), we can now employ D3QN to directly learn the control policy (5) with the ETC attribute, without triggering conditions, i.e., a neural network learns the capabilities of real-time optimal control and triggering conditions simultaneously, and it consists of the following steps.

Step 1: Discretization of control space. In the training phase, the control space is discretized into $v_k\in\{v|v_{\min}+jv_{\text{intv}},\ j=0,1,2,\ldots,L-1, L\in\mathbb{N}^+\}$, where $L=1+(v_{\max}-v_{\min})/v_{\text{intv}}$ is the total number of discretized controls, $v_{\max}$ and $v_{\min}$ are separately the maximum and minimum values, and v_{intv} is the discretization interval.

Step 2: Training using D3QN. According to the RL theory [19], given a control policy $v_k=\pi(s_k)$, the augmented version of the Q-function for the AS (4) with the performance index (6) can be defined as

$$Q^{\pi}(s_k,v_k)=\sum_{t=k}^{\infty}\gamma^{t-k}\mathcal{U}(s_t,v_t). \tag{9}$$

Similarly, according to D3QN [20], the augmented versions of the value and advantage functions are

$$\mathcal{V}^{\pi}(s_k)=Q^{v_k\sim\pi}(s_k,v_k) \tag{10a}$$

$$\mathcal{A}^{\pi}(s_k,v_k)=Q^{\pi}(s_k,v_k)-\mathcal{V}^{\pi}(s_k). \tag{10b}$$

In our tasks, we seek to obtain the optimal augmented Q-function $Q^*(s_k,v_k)=\min_{\pi}Q^{\pi}(s_k,v_k)$. In the training phase, utilize neural networks to approximate (9), i.e., $Q(s_k,v_k|\omega)$ with parameters ω. Then, we seek to minimize the following temporal difference by updating ω at iteration i: [19]: $\mathcal{L}_i(\omega_i)=\|\mathcal{T}_i-Q(s_k,v_k|\omega_i)\|^2$ where $\mathcal{T}_i=\mathcal{U}(s_k,v_k)+\gamma Q(s_{k+1},\arg\min_{v_{k+1}}Q(s_{k+1},v_{k+1}|\omega_i)|\omega^-)$ with ω^- being the parameters of the target network.

Comparative simulations: Simulations provide two examples with a comparative discussion to show advantages and disadvantages of DRL-based and Lyapunov stability-based ETOC methods.

Example 1: Consider a linear discrete-time system

$$x_{k+1}=\begin{bmatrix}0.1 & 1.0\\ 0.007 & 0.9\end{bmatrix}x_k+\begin{bmatrix}0\\1\end{bmatrix}u_k \tag{11}$$

with $x_k=[x_{1,k},x_{2,k}]^T\in\mathbb{R}^2$ and $u_k\in\mathbb{R}$.

To evaluate ETC, define the triggering rate as $T_e=\sum_{k=0}^{N}\mathbb{I}_e(v_k)/N\times 100\%$ with the total number of running time steps in an episode N.

The training and simulation setup for the developed Deep-ETOC method is as follows. The neural network is with the structure of 3–1024–1024–1024–401; $\mathcal{U}(s_k,v_k)=0.1x_k^Tx_k+0.1u_k^Tu_k+10^{-4}v_k^Tv_k+\mathbb{I}_e(v_k)$; $v_{\max}=1$, $v_{\min}=-1$, and $v_{\text{intv}}=0.0005$; $\gamma=1$; $x_0=[-1,1]^T$; $u_{-1}=1$; $N=100$. To better exhibit training results, we terminate an episode when $\mathcal{J}(s_0,v_0)\geq 1000$. Then, Fig. 1 presents trajectories of $\mathcal{J}(s_0,v_0)$ and T_e. Fig. 1 shows that the developed Deep-ETOC method enables efficient optimization of $\mathcal{J}(s_0,v_0)$ and reduction of T_e.

In what follows, we compare the top-performing model with some popular Lyapunov stability-based ETOC methods [1], [6], [17], and comparative simulation results are depicted in Fig. 2, where Fig. 2 shows trajectories of states and controls. Table 1 presents values for triggering rates of the different methods. Results demonstrate that all

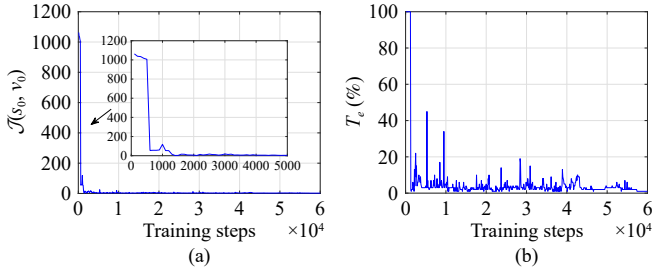

Fig. 1. Training curves. a) $\mathcal{J}(s_0,v_0)$; b) T_e.

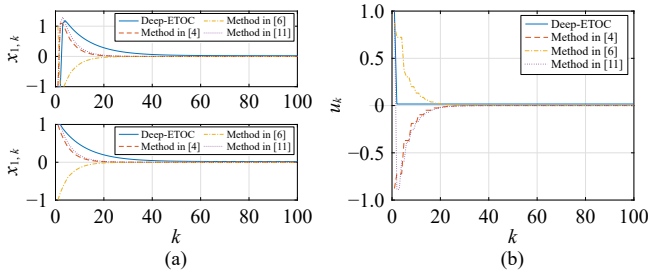

Fig. 2. Comparative results. a) x_k; b) u_k.

Table 1. Triggering Rates of Different Methods

	Deep-ETOC	Method in [1]	Method in [6]	Method in [17] ($\alpha=10^{-2}$)
T_e	**1%**	34%	51%	50%

the methods are effective in stabilizing the system (11), but the developed Deep-ETOC greatly outperforms others in reducing the communication cost. Noticing that there is a parameter "γ" in [17] could influence the triggering rate, we change it to "α" in this letter.

Example 2: Consider a torsional pendulum system [17]

$$x_{k+1}=\begin{bmatrix}x_{1,k}+0.1x_{2,k}\\ -0.49\sin(x_{1,k})+0.98x_{2,k}\end{bmatrix}+\begin{bmatrix}0\\0.1\end{bmatrix}u_k \tag{12}$$

where $x_k=[x_{1,k},x_{2,k}]^T\in\mathbb{R}^2$ and $u_k\in\mathbb{R}$.

The parameters for training and simulation are as follows. The structure of neural network is 3–1024–1024–1024–401; $\mathcal{U}(s_k,v_k)=0.1x_k^Tx_k+0.1u_k^Tu_k+0.1v_k^Tv_k+\mathbb{I}_e(v_k)$; $v_{\max}=1$, $v_{\min}=-1$, and $v_{\text{intv}}=0.005$; $\gamma=0.99$; $x_0=[-1,1]^T$, $u_{-1}=1$; Fig. 3 presents state and control trajectories of the top-performing model, and the triggering rate $T_e=5.5\%$, which means that our method applies to nonlinear systems.

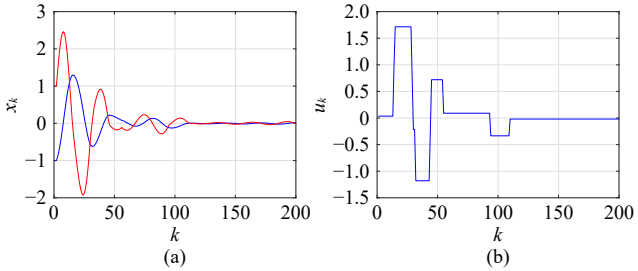

Fig. 3. Simulation results. a) x_k; b) u_k.

The above study shows that, compared with the Lyapunov stability-based ETOC methods, the developed Deep-ETOC method enables a more substantial reduction in the communication cost. It is worth noting that the method in [17] permits to influence the triggering rate by tunning α. Table 2 presents more control results with different α for Example 1. According to Table 2, when $\alpha\geq 10$, the triggering rate no longer decreases, and thus the limiting performance of the method in [17] still fails to reach the performance of the developed Deep-ETOC method in terms of the triggering rate. Thus, we can conclude that the Deep-ETOC method offers greater advantages

Table 2. Triggering Rates With Different α

α	0.05	0.1	0.5	1	10	10^2	10^5	10^8
T_e	34%	25%	20%	17%	13%	13%	13%	13%

in terms of reducing the communication cost.

However, DRL-based ETOC methods cannot provide a clear stability analysis. In addition, the performance of DRL-based ETOC methods is highly contingent on the training set. For example, if there are fewer data within the small neighborhood of the origin in the training set, then system states cannot converge strictly to the origin, see Fig. 3. This is endemic in realizing high-precision control with neural networks. Thus, an elaborate neural network and a suitable training set are critical to obtaining a desirable result.

Conclusion: This letter provides a new thought for realizing ETOC, i.e., to construct a novel performance index considering ETC penalties, and then train the ETOC policy directly based on this performance index. The developed Deep-ETOC method is established on parallel control and employs DRL to realize this method. Besides, we present a preliminary comparative study of DRL-based and Lyapunov stability-based ETOC methods and discuss their strengths and weaknesses based on the simulation results.

Acknowledgments: This work was supported by the Motion G, Inc. Collaborative Research Project for Fundamental Modeling and Parallel Drive-Control of Servo Drive Systems.

References

[1] A. Eqtami, D. V. Dimarogonas, and K. J. Kyriakopoulos, "Event-triggered control for discrete-time systems," in *Proc. Amer. Control Conf.*, 2010, pp. 4719–4724.

[2] X. Ge, S. Xiao, Q.-L. Han, X.-M. Zhang, and D. Ding, "Dynamic event-triggered scheduling and platooning control co-design for automated vehicles over vehicular ad-hoc networks," *IEEE/CAA J. Autom. Sinica*, vol. 9, no. 1, pp. 31–46, Jan. 2022.

[3] I. Ahmad, X. Ge, and Q.-L. Han, "Communication-constrained active suspension control for networked in-wheel motor-driven electric vehicles with dynamic dampers," *IEEE Trans. Intell. Veh.*, vol. 7, no. 3, pp. 590–602, Sept. 2022.

[4] D. Zhao and M. M. Polycarpou, "Fault accommodation for a class of nonlinear uncertain systems with event-triggered input," *IEEE/CAA J. Autom. Sinica*, vol. 9, no. 2, pp. 235–245, Feb. 2022.

[5] N. Zhao, X. Zhao, N. Xu, and L. Zhang, "Resilient event-triggered control of connected automated vehicles under cyber attacks," *IEEE/CAA J. Autom. Sinica*, vol. 10, no. 12, pp. 2300–2302, Dec. 2023.

[6] L. Dong, X. Zhong, C. Sun, and H. He, "Adaptive event-triggered control based on heuristic dynamic programming for nonlinear discrete-time systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 7, pp. 1594–1605, Jul. 2017.

[7] J. Lu, Q. Wei, Z. Wang, T. Zhou, and F.-Y. Wang, "Event-triggered optimal control for discrete-time multi-player non-zero-sum games using parallel control," *Inf. Sci.*, vol. 584, pp. 519–535, Jan. 2022.

[8] J. Peng, B. Fan, Z. Tu, W. Zhang, and W. Liu, "Distributed periodic event-triggered optimal control of dc microgrids based on virtual incremental cost," *IEEE/CAA J. Autom. Sinica*, vol. 9, no. 4, pp. 624–634, Apr. 2022.

[9] J. Lu, Q. Wei, Y. Liu, T. Zhou, and F.-Y. Wang, "Event-triggered optimal parallel tracking control for discrete-time nonlinear systems," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 52, no. 6, pp. 3772–3784, Jun. 2022.

[10] F.-Y. Wang, "Parallel system methods for management and control of complex systems," *Control Decis.*, vol. 19, no. 5, pp. 485–489, May 2004.

[11] J. Lu, Q. Wei, and F.-Y. Wang, "Parallel control for optimal tracking via adaptive dynamic programming," *IEEE/CAA J. Autom. Sinica*, vol. 7, no. 6, pp. 1662–1674, Nov. 2020.

[12] Q. Wei, H. Li, and F.-Y. Wang, "A novel parallel control method for continuous-time linear output regulation with disturbances," *IEEE Trans. Cybern.*, vol. 53, no. 6, pp. 3760–3770, Jun. 2023.

[13] J. Lu, L. Han, Q. Wei, X. Wang, X. Dai, and F.-Y. Wang, "Event-triggered deep reinforcement learning using parallel control: A case study in autonomous driving," *IEEE Trans. Intell. Veh.*, vol. 8, no. 4, pp. 2821–2831, Apr. 2023.

[14] F.-Y. Wang, "New control paradigm for Industry 5.0: From big models to foundation control and management," *IEEE/CAA J. Autom. Sinica*, vol. 10, no. 8, pp. 1643–1646, Aug. 2023.

[15] J. Lu, X. Wang, Q. Wei, and F.-Y. Wang, "Nearly optimal stabilization of unknown continuous-time nonlinear systems: A new parallel control approach," *Neurocomputing*, vol. 578, p. 127421, 2024.

[16] F.-Y. Wang, J. Zhang, X. Zheng, X. Wang, Y. Yuan, X. Dai, J. Zhang, and L. Yang, "Where does AlphaGo go: From church-turing thesis to AlphaGo thesis and beyond," *IEEE/CAA J. Autom. Sinica*, vol. 3, no. 2, pp. 113–120, Apr. 2016.

[17] J. Lu, Q. Wei, T. Zhou, Z. Wang, and F.-Y. Wang, "Event-triggered near-optimal control for unknown discrete-time nonlinear systems using parallel control," *IEEE Trans. Cybern.*, vol. 53, no. 3, pp. 1890–1904, Mar. 2023.

[18] D. Baumann, J.-J. Zhu, G. Martius, and S. Trimpe, "Deep reinforcement learning for event-triggered control," in *Proc. IEEE Conf. Decis. Control*, 2018, pp. 943–950.

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

[20] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," in *Proc. 33rd Int. Conf. Mach. Learn.*, 2016, pp. 1995–2003.