Policy-Iteration-Based Finite-Horizon Approximate Dynamic Programming for Continuous-Time Nonlinear Optimal Control

Ziyu Lin, Jingliang Duan, *Member, IEEE*, Shengbo Eben Li, *Senior Member, IEEE*, Haitong Ma, Jie Li, Jianyu Chen, Bo Cheng, and Jun Ma

***Abstract*— The Hamilton–Jacobi–Bellman (HJB) equation serves as the necessary and sufficient condition for the optimal solution to the continuous-time (CT) optimal control problem (OCP). Compared with the infinite-horizon HJB equation, the solving of the finite-horizon (FH) HJB equation has been a long-standing challenge, because the partial time derivative of the value function is involved as an additional unknown term. To address this problem, this study first-time bridges the link between the partial time derivative and the terminal-time utility function, and thus it facilitates the use of the policy iteration (PI) technique to solve the CT FH OCPs. Based on this key finding, the FH approximate dynamic programming (ADP) algorithm is proposed leveraging an actor–critic framework. It is shown that the algorithm exhibits important properties in terms of convergence and optimality. Rather importantly, with the use of multilayer neural networks (NNs) in the actor–critic architecture, the algorithm is suitable for CT FH OCPs toward more general nonlinear and complex systems. Finally, the effectiveness of the proposed algorithm is demonstrated by conducting a series of simulations on both a linear quadratic regulator (LQR) problem and a nonlinear vehicle tracking problem.**

***Index Terms*— Actor critic, approximate dynamic programming (ADP), finite-horizon (FH), Hamilton–Jacobi–Bellman (HJB) equation, optimal control, policy iteration (PI).**

Manuscript received 1 August 2021; revised 18 July 2022 and 28 October 2022; accepted 23 November 2022. This work was supported in part by the International Science and Technology Cooperation Program of China under Grant 2019YFE0100200; in part by the Tsinghua University-Toyota Joint Research Center for AI Technology of Automated Vehicle; and in part by the NSF China under Grant 51575293, Grant U20A20334, and Grant 52202487. *(Ziyu Lin and Jingliang Duan contributed equally to this work.) (Corresponding author: Shengbo Eben Li.)*

Ziyu Lin, Shengbo Eben Li, Haitong Ma, Jie Li, and Bo Cheng are with the School of Vehicle and Mobility, Tsinghua University, Beijing 100084, China (e-mail: linzy17@mails.tsinghua.edu.cn; lishbo@tsinghua.edu.cn; maht19@mails.tsinghua.edu.cn; jie-li18@mails.tsinghua.edu.cn; chengbo@tsinghua.edu.cn).

Jingliang Duan is with the School of Mechanical Engineering, University of Science and Technology Beijing, Beijing 100083, China (e-mail: duanjl@ustb.edu.cn).

Jianyu Chen is with the Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing 100084, China (e-mail: jianyuchen@tsinghua.edu.cn).

Jun Ma is with the Robotics and Autonomous Systems Thrust, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou, China, and also with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Hong Kong SAR, China (e-mail: jun.ma@ust.hk).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2022.3225090.

Digital Object Identifier 10.1109/TNNLS.2022.3225090

I. Introduction

THE optimal control problem (OCP) for continuous-time (CT) systems has gathered considerable momentum and research attention with a long history because of its theoretical and practical significance. As a standard practice, the optimal value function for CT systems can be obtained by solving the well-known Hamilton-Jacobi–Bellman (HJB) equation, which gives a necessary and sufficient condition for optimality [1]. For CT linear quadratic regulator (LQR) problems, the HJB equation is simplified as the algebraic Riccati equation, which can be solved directly. On the other hand, for complex nonlinear dynamic systems, the LQR can be implemented iteratively to derive the optimal solution [2], [3], [4]. In such scenarios, the determination of the analytic solution to the HJB equation typically poses great difficulty, because a nonlinear partial differential equation remains to solve [5].

In view of the difficulty of finding the exact solution to the HJB equation for nonlinear dynamic systems, it prompts many researchers to turn their attention to investigate the near-optimal solution. As one of the seminal works in this field, Werbos [6] proposed the approximate dynamic programming (ADP), with a primitive aim to solve the large-scale and complex problems. It is known that ADP can be regarded as one of the model-based reinforcement learning algorithms [7], and it has many synonyms, such as the so-called neurodynamic programming [8] and adaptive dynamic programming [8]. The core idea behind the ADP is to solve the intractable HJB equation utilizing the policy iteration (PI) technique [9]. In addition, the training process with PI technique can be split systematically into two stages: 1) policy evaluation (PEV) and 2) policy improvement (PIM) [10].

Last few years have witnessed the successful application of ADP on solving the HJB equation for the CT infinite-horizon control problem, in which the optimal control inputs only depend on the system states [11]. In terms of the CT infinite-horizon HJB equation, there are two unknown terms. Hence, in this case, the PI technique can be deployed to determine the two unknown terms in the infinite-horizon HJB equation by solving two equations in an iterative framework. Remarkably, Vamvoudak and Lewis [12] presented an online ADP algorithm called the synchronous PI, which was implemented in the actor–critic architecture for nonlinear CT infinite-horizon systems. Vrabie et al. [13] presented a new scheme to obtain

an online optimal control solution to linear CT infinite-horizon systems based on the adaptive critics. Fu et al. [14] proposed an online robust ADP algorithm for two-player zero-sum games of CT unknown linear systems with uncertainties. The PI technique is used, yet only one iteration loop is involved. Kamalapurkar et al. [15] utilized the state following kernel method to approximate the infinite-horizon value function. He et al. [16] utilized the network linear differential inclusion technique to linearize and approximate the nonlinear term in dynamic model, and then the online H_∞ OCP with infinite-horizon quadratic function is solved. He et al. [17] provided the perspective to address CT infinite-horizon control problem with the combination of the online linearization and the online PI technique. Fang et al. [18] studied the adaptive optimal controller for nonlinear Markov jump systems via coupled linear subsystems. Deniz et al. [19] extended the adaptive control to uncertain dynamic systems and obtained the optimal model for the nonlinear reference.

It is pertinent to notice that, the CT finite-horizon (FH) control problem is essentially different from the infinite-horizon control counterpart, because the value function is time-dependent in the FH control problem. In this sense, the additional unknown term, i.e., the partial derivative of value function with respect to time, is naturally introduced. Thus in this regard, there are three unknown terms in two equations, and the PI technique is no longer suitable for the FH HJB equation [20]. Therefore, the solving of the FH HJB equation is still not explored thoroughly due to the lack of effective mathematical tools.

Recently, the development of machine learning methods renders it possible to solve the FH HJB equation leveraging parameterized functions [21], [22], [23]. As one of the representative types of parameterized functions, the neural network (NN) is commonly used in OCPs [24], [25]. In essence, the NN can also be suitably used for approximating the near-optimal solution to the value function and control input. In this case, it avoids the difficulty of solving the exact analytical solutions to the two iterative equations in the PI framework. In particular, the single-layer NN (SNN), which is also known as the linear combination method, is widely employed for the FH optimal control studies [26], [27], [28]. To handle the tractable time-varying values, a few studies utilized SNNs with time-varying weights to construct the value function and policy [29], [30], [31], [32], [33]. The optimal time-varying weights can be found by solving a nonlinear ordinary differential equation backward in time using the weighted residuals method.

However, existing methods for CT FH OCP suffer from two significant problems. First, most algorithms are restricted to affine systems as the policy is restricted to be expressed using the value function analytically [34], [35]. Therefore, the OCP can be addressed via solving the HJB equation for input-affine systems with only value NN, yet directly solving the optimal policy for nonaffine systems in this way is often intractable. Second, the performance of SNN-based algorithms relies on the design accuracy of the hand-crafted basis functions heavily, and it is generally hard to design such features for complex nonlinear systems [36], [37]. Compared with NNs with a single hidden layer, NNs with multiple hidden layers own the universal fitting ability. Hence, they can directly map the system states to the value function and control inputs without reliance on hand-crafted features [38]. Thus, it leaves an interesting problem to explore the use of NNs with multiple hidden layers to parameterize both value functions and control input separately, and then solve the FH OCP.

Hence, taking all the aforementioned matters into consideration, a parameterized ADP algorithm is proposed in this work to find the nearly optimal policy of CT FH OCPs. The main contributions of this work are threefold.

1) This study bridges the link between the partial time derivative of the value function and the terminal-time utility function. The analytical and solvable expression for the additional unknown term is given, and thus it facilitates the use of the PI technique to solve the FH OCP. With rigorous proofs and an illustrative example, the effectiveness of the proposed development is verified.
2) Based on the FH HJB, the FH ADP algorithm with the PI framework is proposed for solving CT FH OCPs. The algorithm is shown to exhibit important properties in terms of convergence and optimality.
3) Multilayer NNs are deployed in the proposed FH ADP algorithm to parameterize both approximated value function and control input with the actor–critic framework, so that the proposed methodology is not limited to affine systems only. Furthermore, the need for complex hand-crafted features is relaxed appropriately. Hence, it is suitable for more general real-world systems with nonlinearity and high complexity.

The rest of this article is organized as follows. Section II introduces the CT HJB equation and ADP method. Section III derives the value function's partial time derivative in FH HJB equation. Section IV proposes the FH ADP algorithm. Section V employs the algorithm to linear quadratic and nonlinear system's simulations with demonstrated effectiveness. The conclusion of this article is given in Section VI.

II. Preliminaries

In this part, we first introduce the CT HJB equation. Then, we discuss the challenges of solving the FH OCPs by analyzing the difference between the infinite-horizon and FH HJB equations. Besides, an FH version of PI is given to point out that the derivation of the partial time derivative of the value function is the key to solve FH OCPs.

A. CT HJB Equation

The general CT time-invariant dynamical system is denoted as

$$\dot{x} = f(x, u) \tag{1}$$

where $x \in \Omega \subset \mathbb{R}^n$ is the state vector, Ω denotes the compact set, $u \in \mathbb{R}^m$ is the control input, and $f(x, u) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is the state dynamic model. Our study assumes that the state model is controllable and the dynamic system is stabilizable

on Ω. Meanwhile, the model function $f(x,u)$ is regarded to be known, which could be any nonlinear analytical function as long as $(\partial f(x,u)/\partial u)$ and $(\partial f(x,u)/\partial x)$ are available.

The FH value function corresponding to the known policy π is defined as

$$V^{\pi}(x(t),t) = \int_t^T l(x(\tau),u(\tau))d\tau = \int_t^T l(x(\tau),\pi(x(\tau),\tau))d\tau \quad (2)$$

where $t \in [0,T]$ is the current time, T is the fixed terminal time, $l(x(\tau),u(\tau)) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ is the non-negative utility function. It is obvious that $V^{\pi}(x(t),t)$ is time-dependent. We define the policy function π as the mapping from the states-time pair $(x(t),t)$ to control inputs u. In other words, the control inputs are generated by the policy π

$$u(\tau) = \pi(x(\tau),\tau), \quad \tau \in [t,T], \ t \in [0,T]. \quad (3)$$

For system (1) with the value function (2), we introduce the corresponding Hamiltonian

$$H\left(x,\pi(x,t),\frac{\partial V^{\pi}(x,t)}{\partial x^{\top}}\right) = l(x,\pi(x,t)) + \frac{\partial V^{\pi}(x,t)}{\partial x^{\top}} f(x,\pi(x,t)) \quad (4)$$

where x is the abbreviation for the initial state $x(t)$, which is used in the this article for the sake of brevity. By taking the derivative of (2), the FH Lyapunov equation can be expressed as

$$H\left(x,\pi(x,t),\frac{\partial V^{\pi}(x,t)}{\partial x^{\top}}\right) = -\frac{\partial V^{\pi}(x,t)}{\partial t} \quad \forall t \in [0,T]. \quad (5)$$

It means that given a policy π, its associated value function V^{π} can be calculated by solving (5). It can also be regarded as the self-consistency condition.

The OCP is formulated to find a policy that minimizes the value function (2) associated with the system for all x,t. The minimum value function $V^*(x(t),t)$ is defined by

$$V^*(x,t) = \min_{\pi} V^{\pi}(x,t). \quad (6)$$

It meets the CT FH HJB equation

$$\min_{\pi}\left\{l(x,\pi(x,t)) + \frac{\partial V^*(x,t)}{\partial x^{\top}} f(x,\pi(x,t))\right\} = -\frac{\partial V^*(x,t)}{\partial t}. \quad (7)$$

Along with (7), the terminal boundary condition also needs to be satisfied by the optimal policy

$$V^*(x(T),T) = 0. \quad (8)$$

To obtain the optimal control policy of CT FH OCP, researchers first need to solve the HJB equation (7) for $V^*(x,t)$. Then, the optimal control policy $\pi^*(x,t)$ can be directly obtained by minimizing the associated Hamiltonian

$$\pi^*(x,t) = \arg\min_{\pi}\left\{H\left(x,\pi(x,t),\frac{\partial V^*(x,t)}{\partial x^{\top}}\right)\right\}. \quad (9)$$

Nevertheless, for complex nonlinear systems, it is generally difficult or impossible to directly find the solution to the HJB equation, since it is a nonlinear partial differential equation.

By inserting the optimal policy $\pi^*(x,t)$ and optimal value function $V^*(x,t)$ into the Lyapunov equation (5), we can rewrite the FH HJB equation as

$$l\big(x,\pi^*(x,t)\big) + \frac{\partial V^*(x,t)}{\partial x^{\top}} f\big(x,\pi^*(x,t)\big) = -\frac{\partial V^*(x,t)}{\partial t}. \quad (10)$$

There are three unknown terms in this equation: 1) $\pi^*(x,t)$; 2) $(\partial V^*(x,t)/\partial x)$; and 3) $(\partial V^*(x,t)/\partial t)$. For infinite-horizon OCPs, i.e., $T \to \infty$, the value function and optimal policy are time-independent. This yields the celebrated infinite-horizon HJB equation

$$l\big(x,\pi^*(x,t)\big) + \frac{\partial V^*(x,t)}{\partial x^{\top}} f\big(x,\pi^*(x,t)\big) = 0 \quad (11)$$

which only contains two unknowns: 1) $\pi^*(x,t)$ and 2) $(\partial V^*(x,t)/\partial x)$ as $(\partial V^*(x,t)/\partial t) = 0$ in this case [39]. Therefore, compared with infinite-horizon HJB equation, it is more difficult to solve the FH HJB equation because more unknown terms are involved.

B. Approximate Dynamic Programming

The basic principle of the ADP algorithm is to solve the HJB equation (7). Iterative methods can be utilized to approach the near-optimal approximation of both value function and policy. PI technique is applied in this study, which iteratively evaluates and improves the policy until convergence. Hence, the proposed algorithm for CT FH OCPs involves two iterative steps.

1) Policy Evaluation (PEV): It aims to drive the estimated value function of the current policy toward the true value function. It means that the value function $V^{\pi^k}(x,t)$ corresponding to the given policy π^k at the kth iteration is calculated by solving the Lyapunov equation (5).

2) Policy Improvement (PIM): The target for this stage is to seek an improved policy $\pi^{k+1}(x,t)$ by minimizing the Hamiltonian with respect to the current estimated value function $V^{\pi^k}(x,t)$ utilizing (9).

Algorithm 1 PI for CT Systems

Initialize policy $\pi^0(x,t)$ and an arbitrarily small positive ϵ.

repeat

1. Update value function $V^{\pi^k}(x,t)$ for $\forall x \in \Omega$ via

$$l\big(x,\pi^k(x,t)\big) + \frac{\partial V^{\pi^k}}{\partial x^{\top}} f\big(x,\pi^k(x,t)\big) = -\frac{\partial V^{\pi^k}(x,t)}{\partial t}. \quad (12)$$

2. Update policy $\pi^{k+1}(x,t)$ for $\forall x \in \Omega$ via

$$\pi^{k+1}(x,t) = \arg\min_{\pi}\left\{H\left(x,\pi^k(x,t),\frac{\partial V^{\pi^k}(x,t)}{\partial x^{\top}}\right)\right\}. \quad (13)$$

until $\|V^{\pi^{k+1}}(x,t) - V^{\pi^k}(x,t)\| \le \epsilon$.

A traditional PI framework for CT OCP is shown in Algorithm 1. Notice that the algorithm does not require the

admissibility of the initial policy. It is due to the fact that the value function is finite and bounded for the FH problem. The PEV stage contains two unknown partial derivative terms in (12): 1) $(\partial V^\pi(x,t)/\partial t)$ and 2) $(\partial V^\pi(x,t)/\partial x)$. Given a policy $\pi(x,t)$, for any different $(\partial V^\pi(x,t)/\partial t)$, one can always seek out a correspondingly different $(\partial V^\pi(x,t)/\partial x)$ making (12) hold. This indicates that no unique solution exists for (12). Nevertheless, considering the infinite-horizon OCP, $(\partial V^\pi(x,t)/\partial t) = 0$ holds. In this case, one can easily solve the PEV process with the single unknown term $(\partial V^\pi(x,t)/\partial x)$. Once $(\partial V^\pi(x,t)/\partial t)$ is known, the solution to FH HJB can be directly obtained by using PI techniques. Therefore, it is meaningful and crucial to express $(\partial V^\pi(x,t)/\partial t)$ analytically.

III. Partial Time Derivative of the Value Function

Since the partial time derivative of the value function is crucial for solving the FH OCP, this section shows that this partial time derivative is equivalent to the negative value of the terminal-time utility function. Particularly, it coverts the time-dependent unknown term into a solvable utility function. Meanwhile, this finding is verified by analyzing the optimal solution to the general LQR problem.

A. Derivation

Theorem 1: Given a CT time-invariant dynamic system (1) and its corresponding FH value function (2), the partial time derivative of the value function satisfies

$$\frac{\partial V^*(x,t)}{\partial t} = -l\left(x^*(T), \pi^*(x(T),T)\right). \tag{14}$$

Proof: The value function (2) can be partitioned as

$$\begin{aligned} V^\pi(x(t),t) &= \int_t^{t+dt} l(x(\tau),u(\tau))d\tau + V^\pi(x(t+dt),t+dt) \\ &= l(x(t),u(t))dt + V^\pi(x(t+dt),t+dt) \end{aligned} \tag{15}$$

where $u(\tau)=\pi(x(\tau),\tau)$ and $x(t+dt)$ can be expressed as

$$x(t+dt) = x(t) + f(x(t),u(t))dt. \tag{16}$$

Similarly, the introduced function $V^\pi(x(t),t+dt)$ denotes the value starting from the state $x(t)$ at time $t+dt$ following policy π, i.e.,

$$V^\pi(x(t),t+dt) = l(x(t),\hat{u}(t))dt + V^\pi(\hat{x}(t+dt),t+2dt) \tag{17}$$

where $\hat{u}(t)=\pi(x(t),t+dt)$, and $\hat{x}(t+dt)$ is generated by $\hat{u}(t)$, i.e., $\hat{x}(t+dt)=x(t)+f(x(t),\hat{u}(t))dt$. Then, one has

$$\begin{aligned} \hat{x}(t+dt) &= x(t) + f(x(t),u(t))dt + \frac{df(x(t),u(t))dt}{du}\frac{du(t)}{dt}dt \\ &= x(t+dt) + dx(t+dt) \end{aligned} \tag{18}$$

where $dx(t+dt) := (df(x(t),u(t))dt/du)(du(t)/dt)dt$. Then, we take the Taylor expansion of the right-hand side of (17), and the first term is expanded as

$$l(x(t),\hat{u}(t))dt = l(x(t),u(t))dt + \frac{dl(x(t),u(t))dt}{du(t)}\frac{du(t)}{dt}dt. \tag{19}$$

Then the Taylor expansion of the second term of (17) is

$$\begin{aligned} &V^\pi(\hat{x}(t+dt),t+2dt) \\ &= V^\pi(x(t+dt)+dx(t+dt),\ t+dt+dt) \\ &= V^\pi(x(t+dt),t+dt) + \frac{\partial V^\pi(x(t+dt),t+dt)}{\partial(t+dt)}dt \\ &\quad + \frac{\partial V^\pi(x(t+dt),t+dt)}{\partial x(t+dt)}dx(t+dt) \\ &= V^\pi(x(t+dt),t+dt) + \frac{\partial V^\pi(x(t+dt),t+dt)}{\partial(t+dt)}dt \\ &\quad + \frac{\partial V^\pi(x(t+dt),t+dt)}{\partial x(t+dt)}\frac{df(x(t),u(t))dt}{du(t)}\frac{du(t)}{dt}dt. \end{aligned} \tag{20}$$

Upon (16), we obtain $(dx(t+dt)/du(t)) = (df(x(t),u(t))dt/du(t))$. Thereafter, (20) can be further derived as

$$\begin{aligned} &V^\pi(\hat{x}(t+dt),t+2dt) \\ &\quad = V^\pi(x(t+dt),t+dt) + \frac{\partial V^\pi(x(t+dt),t+dt)}{\partial(t+dt)}dt \\ &\qquad + \frac{dV^\pi(x(t+dt),t+dt)}{du(t)}\frac{du(t)}{dt}dt. \end{aligned} \tag{21}$$

Plugging (19) and (21) into (17), one can get the Taylor expansion of $V^\pi(x(t),t+dt)$ as

$$\begin{aligned} &V^\pi(x(t),t+dt) \\ &= l(x(t),u(t))dt + V^\pi(x(t+dt),t+dt) \\ &\quad + \frac{\partial V^\pi(x(t+dt),t+dt)}{\partial(t+dt)}dt \\ &\quad + \frac{d(l(x(t),u(t))dt + V^\pi(x(t+dt),t+dt))}{du(t)}\frac{du(t)}{dt}dt \end{aligned} \tag{22}$$

where $V^\pi(x(t+dt),t+dt) = V^\pi(x,t) + (\partial V^\pi(x,t)/\partial x)f(x,u)dt + (\partial V^\pi(x,t)/\partial x)dt$. In this case, when taking the derivative with respect to the control input, one can obtain $(dV^\pi(x(t+dt),t+dt)/du(t)) = (d(\partial V^\pi(x(t),t)/\partial x)fdt/du(t))$. Then, combined with Hamilton equation (4), (22) can be further derived as

$$\begin{aligned} &V^\pi(x(t),t+dt) \\ &= l(x(t),u(t))dt + V^\pi(x(t+dt),t+dt) \\ &\quad + \frac{\partial V^\pi(x(t+dt),t+dt)}{\partial(t+dt)}dt \\ &\quad + \frac{d\left(l(x(t),u(t))dt + \frac{\partial V^\pi(x(t),t)}{\partial x}fdt\right)}{du(t)}\frac{du(t)}{dt}dt \\ &= l(x(t),u(t))dt + V^\pi(x(t+dt),t+dt) \\ &\quad + \frac{\partial V^\pi(x(t+dt),t+dt)}{\partial(t+dt)}dt + \frac{dH\left(x,\pi,\frac{\partial V^\pi(x(t),t)}{\partial x}\right)}{du(t)}\frac{du}{dt}(dt)^2. \end{aligned} \tag{23}$$

Thereafter, utilizing the definition of the partial-time derivative and then subtracting (15) from (23), one can derive

$$\begin{aligned}&\frac{\partial V^{\pi}(x(t),t)}{\partial t}\\&=\frac{V^{\pi}(x(t),t+dt)-V^{\pi}(x(t),t)}{dt}\\&=\frac{\partial V^{\pi}(x(t+dt),t+dt)}{\partial(t+dt)}+\frac{dH\left(x,\pi,\frac{\partial V^{\pi}(x(t),t)}{\partial x(t)}\right)}{du(t)}\frac{du}{dt}dt.\end{aligned}\tag{24}$$

Here, $(\partial V^{\pi}(x(t+dt),t+dt)/\partial(t+dt))$ can be expanded repeatedly by applying (24), we finally have

$$\frac{\partial V^{\pi}(x(t),t)}{\partial t}=\frac{\partial V^{\pi}(x(T),T)}{\partial(T)}+\zeta^{\pi}\tag{25}$$

where ζ^{π} is used to represent the integral term, i.e.,

$$\zeta^{\pi}=\int_t^T\frac{dH\left(x,\pi(x,\tau),\frac{\partial V^{\pi}(x(\tau),\tau)}{\partial x(\tau)}\right)}{du(\tau)}\frac{du(\tau)}{d\tau}d\tau.\tag{26}$$

Combined with the Lyapunov function (5), (25) can be further expanded as

$$\begin{aligned}&\frac{\partial V^{\pi}(x(t),t)}{\partial t}\\&=-l(x(T),u(T))-\frac{\partial V^{\pi}(x(T),T)}{\partial x^{\top}(T)}f(x(T),u(T))+\zeta^{\pi}.\end{aligned}\tag{27}$$

Since $V^{\pi}(x(T),T)=0;\ \forall x(T)$, one can obtain $(\partial V^{\pi}(x(T),T)/\partial x^{\top}(T))=0$. Therefore, we get the general condition as

$$\frac{\partial V^{\pi}(x(t),t)}{\partial t}=-l(x(T),\pi(x(T),T))+\zeta^{\pi}.\tag{28}$$

Supposing π^* is an optimal policy with respect to $V^*(x(t),t)$ and $u^*(t)=\pi^*(x,t)$, we have

$$\frac{\partial H\left(x,u^*(t),\frac{\partial V^*(x(t),t)}{\partial x^{\top}(t)}\right)}{\partial u^*(t)}=0.\tag{29}$$

Thereafter, $\zeta^{\pi}=0$. Up to now, it concludes that the value function's partial time derivative equals the negative value of the terminal-time utility function as (14). □

B. Verification With LQR Problem

Consider the general LQR control problem expressed as

$$\begin{aligned}V^{\pi}(x,t)&=\frac{1}{2}\int_t^T\left(qx^2(\tau)+ru^2(\tau)\right)d\tau\\&\text{s.t. }\dot{x}=ax+bu\end{aligned}\tag{30}$$

where stabilizable (a,b) are scalars, $q>0$ and $r>0$. In this linear case, one can solve the algebraic Riccati equation to obtain the optimal analytic value and policy function. Therefore, we are able to provide evidence for the correctness of Theorem 1 by inserting the analytic solutions in (14).

For LQR problem, the optimal value function V^* and the corresponding optimal policy π^* are given in the following form:

$$V^*(x,t)=\frac{1}{2}x^{\top}P(t)x,\quad \pi^*(x,t)=-\frac{1}{r}bP(t)x\tag{31}$$

where the matrix $P(t)$ is denoted as

$$P(t)=\frac{r}{b^2}\frac{\beta+a+\eta(\beta-a)e^{2\beta(t-T)}}{1-\eta e^{2\beta(t-T)}}\tag{32}$$

with $P(T)=0$, $\beta=((qb^2/r)+a^2)^{1/2}$, $\eta=(a+\beta/a-\beta)$. The derivation of (32) can be found in Appendix. Then, the analytic expression of the optimal terminal state $x^*(T)$ can be derived as

$$\begin{aligned}x^*(T)&=x(t)e^{\int_t^T\left(a-\frac{b^2}{r}P(\tau)\right)d\tau}\\&=x(t)\left(\frac{1-\eta e^{2\beta(T-T)}}{1-\eta e^{2\beta(t-T)}}\right)e^{-\beta(T-t)}.\end{aligned}\tag{33}$$

From (30) to (33), one has

$$\begin{aligned}\frac{\partial V^*(x(t),t)}{\partial t}&=-l\left(x^*(T),\pi^*(x(T),T)\right)\\&=\frac{1}{2}qx(t)^2\left(\frac{2\beta}{-a+\beta}\right)^2\frac{e^{2\beta(t-T)}}{\left(1-\eta e^{2\beta(t-T)}\right)^2}.\end{aligned}\tag{34}$$

The details of the derivation of (33) and (34) are provided in Appendix.

IV. CT FH ADP Algorithm

In this section, we first propose an FH PI framework based on Theorem 1, along with the proofs of convergence and optimality. Then the parameterized FH ADP algorithm based on actor–critic is proposed to solve the nearly optimal policies of CT FH OCPs.

A. FH Policy Iteration

According to Theorem 1, after replacing $(\partial V^*(x,t)/\partial t)$ in (10) by $-l(x^*(T),\pi^*(x^*(T),T))$, we can get the reshaped FH HJB equation

$$\begin{aligned}&l\left(x,\pi^*(x,t)\right)+\frac{\partial V^*(x,t)}{\partial x^{\top}}f\left(x,\pi^*(x,t)\right)\\&\qquad=l\left(x^*(T),\pi^*\left(x^*(T),T\right)\right)\end{aligned}\tag{35}$$

where $x^*(T)$ is the terminal state under the policy π^*, which is obtained by rolling out from the initial state $x(t)$. Similar to the infinite-horizon HJB (11), (35) contains two unknowns: 1) $\pi^*(x,t)$ and 2) $(\partial V^*(x,t)/\partial x)$. Therefore, we refer to (35) as the FH HJB. Correspondingly, the self-consistency condition (12) used for PEV is rewritten as

$$\begin{aligned}&l(x(t),\pi(x(t),t))+\frac{\partial V^{\pi}(x(t),t)}{\partial x^{\top}}f(x,\pi(x,t))\\&\qquad=l(x^{\pi}(T),\pi(x^{\pi}(T),T)).\end{aligned}\tag{36}$$

It is noted that (36) only provides the first-order derivative information (i.e., $(\partial V^{\pi}(x,t)/\partial x)$). Therefore, the terminal boundary condition (8) is essential to fix the value function to a setting point when training value and policy NNs.

Remark 1: From (14), $(\partial V^{\pi}(x,t)/\partial x)$ is the single unknown term in (36). Under this condition, the only difference is that $(\partial V^{\pi}(x,t)/\partial t)=0$ for the infinite-horizon ADP, whereas $(\partial V^{\pi}(x,t)/\partial t)=-l(x^{\pi}(T),\pi(x^{\pi}(T),T))$ for the FH ADP. Therefore, like existing infinite-horizon ADP algorithms, we can use the traditional PI techniques to approach

the optimal solution to (35) iteratively. It iteratively solves PEV equation based on (36) and PIM equation based on (13).

Next, the pseudocode of FH PI is demonstrated in Algorithm 2.

Algorithm 2 Finite-Horizon PI for CT Systems

Initialize policy $\pi^0(x,t)$ and small positive ϵ.
repeat
Rollout with policy π^k from $\forall x_t \in \Omega$ and receive $x^{\pi^k}(T)$,
1) Update value function $V^{\pi^k}(x,t)$ for $\forall x \in \Omega$ utilizing

$$l\left(x,\pi^k(x,t)\right)+\frac{\partial V^{\pi^k}}{\partial x^\top}f\left(x,\pi^k(x,t)\right)=l\left(x^{\pi^k}(T),\pi^k(x(T),T)\right); \tag{37}$$

2) Update policy $\pi^{k+1}(x,t)$ for $\forall x \in \Omega$ utilizing (13);
until $\|V^{\pi^{k+1}}(x,t)-V^{\pi^k}(x,t)\| \le \epsilon$.

B. Analysis on the Convergence and Optimality

In this part, Theorem 2 is presented, which shows that the FH PI algorithm is gradually convergent to the optimal solution for both policy and value function.

Theorem 2: For arbitrary initial policy $\pi^0(x,t)$, if the policy is updated with Algorithm 2, then $V^{\pi^k}(x,t) \to V^*(x,t)$, $\pi^k(x,t) \to \pi^*(x,t)$ uniformly on Ω as k goes to ∞.

Proof: Considering the PIM step (9), the following equation is satisfied:

$$\begin{aligned}
&H\left(x,\pi^{k+1}(x,t),\frac{\partial V^{\pi^k}(x,t)}{\partial x^\top}\right)\\
&\quad=\min_\pi\left\{l(x,\pi(x,t))+\frac{\partial V^{\pi^k}(x,t)}{\partial x^\top}f(x,\pi(x,t))\right\}\\
&\quad\le H\left(x,\pi^k(x,t),\frac{\partial V^{\pi^k}(x,t)}{\partial x^\top}\right).
\end{aligned} \tag{38}$$

Upon (28), given the policy π^k, we have

$$H\left(x,\pi^k(x,t),\frac{\partial V^{\pi^k}(x,t)}{\partial x^\top}\right)=l\left(x^{\pi^k}(T),\pi^k\left(x^{\pi^k}(T),T\right)\right)-\zeta^\pi. \tag{39}$$

Combining the two formulas above, we obtain

$$\begin{aligned}
&H\left(x,\pi^{k+1}(x,t),\frac{\partial V^{\pi^k}(x,t)}{\partial x^\top}\right)\\
&\qquad\le l\left(x^{\pi^k}(T),\pi^k\left(x^{\pi^k}(T),T\right)\right)-\zeta^\pi.
\end{aligned} \tag{40}$$

Taking the derivative of $V^{\pi^k}(x,t)$ and $V^{\pi^{k+1}}(x,t)$ with respect to t utilizing policy π^{k+1}, respectively, we can obtain

$$\begin{aligned}
&\frac{dV^{\pi^k}(x,t)}{dt}\\
&\quad=\frac{\partial V^{\pi^k}(x,t)}{\partial x^\top}f\left(x,\pi^{k+1}(x,t)\right)+\frac{\partial V^{\pi^k}(x,t)}{\partial t}\\
&\quad=H\left(x,\pi^{k+1}(x,t),\frac{\partial V^{\pi^k}(x,t)}{\partial x^\top}\right)-l\left(x,\pi^{k+1}(x,t)\right)\\
&\qquad+\frac{\partial V^{\pi^k}(x,t)}{\partial t}\\
&\quad\le H\left(x,\pi^k(x,t),\frac{\partial V^{\pi^k}(x,t)}{\partial x^\top}\right)-l\left(x,\pi^{k+1}(x,t)\right)\\
&\qquad+\frac{\partial V^{\pi^k}(x,t)}{\partial t}\\
&\quad=-l\left(x,\pi^{k+1}(x,t)\right)
\end{aligned} \tag{41}$$

$$\begin{aligned}
&\frac{dV^{\pi^{k+1}}(x,t)}{dt}\\
&\quad=\frac{\partial V^{\pi^{k+1}}(x,t)}{\partial x^\top}f\left(x,\pi^{k+1}(x,t)\right)+\frac{\partial V^{\pi^{k+1}}(x,t)}{\partial t}\\
&\quad=H\left(x,\pi^{k+1}(x,t),\frac{\partial V^{\pi^{k+1}}(x,t)}{\partial x^\top}\right)-l\left(x,\pi^{k+1}(x,t)\right)\\
&\qquad+\frac{\partial V^{\pi^{k+1}}(x,t)}{\partial t}\\
&\quad=-l\left(x,\pi^{k+1}(x,t)\right).
\end{aligned} \tag{42}$$

Since $l(x,u) \ge 0$, it is obvious that

$$\frac{dV^{\pi^k}(x,t)}{dt} \le \frac{dV^{\pi^{k+1}}(x,t)}{dt} \le 0 \quad \forall x \in \Omega,\ t \in [0,T]. \tag{43}$$

From Newton–Leibniz formula, we have

$$V^\pi(x,t)=V^\pi(x(T),T)-\int_t^T\frac{dV^\pi(x,t)}{dt}d\tau. \tag{44}$$

Considering the definition of the value function in (2), for arbitrary π, one can obtain

$$V^\pi(x^\pi(T),T)=0. \tag{45}$$

So, from (2) and (43), it yields

$$0 \le V^{\pi^{k+1}}(x,t) \le V^{\pi^k}(x,t) \quad \forall x \in \Omega, t \in [0,T]. \tag{46}$$

$V^{\pi^k}(x,t)$ will be convergent as $k \to \infty$. Therefore, one can obtain $\lim_{k\to\infty} V^{\pi^k}(x,t) = V^{\pi^\infty}(x,t)$, which also implies that $\lim_{k\to\infty}\pi^k(x,t)=\pi^\infty(x,t)$. Furthermore, it is clear that

$$\begin{aligned}
&\min_\pi\left(l\left(x,\pi^\infty(x,t)\right)+\frac{\partial V^{\pi^\infty}(x,t)}{\partial x^\top}f\left(x,\pi^\infty(x,t)\right)\right)\\
&\quad=l\left(x,\pi^\infty(x,t)\right)+\frac{\partial V^{\pi^\infty}(x,t)}{\partial x^\top}f\left(x,\pi^\infty(x,t)\right)\\
&\quad=l\left(x^{\pi^\infty}(T),\pi^\infty(x(T),T)\right).
\end{aligned} \tag{47}$$

Hence, $V^\infty(x,t)$ and $\pi^\infty(x,t)$ satisfy the HJB equation (10), i.e., $V^{\pi^\infty}(x,t)=V^*(x,t)$ and $\pi^\infty(x,t)=\pi^*(x,t)$. Therefore, we can conclude that $V^{\pi^k}(x,t) \to V^*(x,t)$, $\pi^k(x,t) \to \pi^*(x,t)$ uniformly on Ω as k goes to ∞. □

Remark 2: The convergence and optimality of the proposed ADP algorithm are guaranteed with the theoretical proof. Since the value function in the FH is finite and bounded, there is a feasible solution to the self-consistency condition. In addition, if the optimal policy to the OCP is stable, the policy obtained by the proposed ADP algorithm is also stable, which can drive the states toward the equilibrium.

C. CT FH ADP

Since it is difficult to get the exact solution for the two iterative equation in PI framework, one usually adopts approximate function to approach the near-optimal solution. Previous researches using the SNN suffer from the complexity of handcrafted basis function and limited application. In this article, multilayer NNs are adopted to approximate both the value function and the control policy with the following parametric architectures:

$$\begin{aligned} V^{\pi}(x,t) &\approx V(x,t;w) \\ \pi(x,t) &\approx \pi(x,t;\theta) \end{aligned} \tag{48}$$

where the value network (i.e., critic) is denoted as $V(x,t;w)$ with parameters w, and the policy network (i.e., actor) is denoted as $\pi(x,t;\theta)$ with parameters θ. Under the universal approximation theorem [40], [41], multilayer NNs own the universal fitting ability and convergence property. Hence, the value network $V(x,t;w)$ and policy network $\pi(x,t;\theta)$ can map from the system states and time t to the approximated value function and control inputs, respectively. The key process to determine the parameter vector of the parametric architecture is generally called training. For brevity, V_{w^k} and π_{θ^k} are short for $V(x,t;w^k)$ and $\pi(x,t;\theta^k)$, respectively, at the kth step in the PI process.

The process of PI works as follows. We first initialize the parameter vectors and the states. Then, at the kth step of the PEV stage, the policy π^k is known, and the critic seeks to find the corresponding value V_{w^k}. In essence, the parameterized critic is to approximate the solution of self-consistency condition (36). Hence, the critic aims to minimize the mean square error J_{Critic}

$$\begin{aligned} J_{\text{Critic}} = \mathop{\mathbb{E}}_{x,t\sim d_{x,t}} \Bigg\{ \frac{1}{2} \Bigg(& H\left(x, \pi_{\theta^k}, \frac{\partial V_{w^k}}{\partial x^{\top}}\right) \\ & - l\left(x^{\pi^k}(T), \pi\left(x^{\pi^k}(T), T; \theta^k\right)\right)\Bigg)^2 \Bigg\} \end{aligned} \tag{49}$$

where the training set consists of a batch of state-time pairs (x,t); $\mathbb{E}$ represents the mathematical expectation; $d_{x,t}$ denotes the distribution of state-time pairs over $x \in \Omega$ and $t \in \lceil 0, T \rceil$; $d_{x,t}$ can be regarded as a particular weighting function, which reflects the importance of each state and can be chosen as the uniform distribution; $x^{\pi}(T)$ represents the terminal state which is rolling out by the dynamic model using policy π^k.

As for the PIM stage according to (13), the policy network is tuned to directly minimize the expectation of Hamiltonian for whole states. Therefore, the loss can be expressed as

$$\begin{aligned} J_{\text{Actor}} &= \mathop{\mathbb{E}}_{x,t\sim d_{x,t}} \left\{ H\left(x, \pi_{\theta^k}, \frac{\partial V_{w^k}}{\partial x^{\top}}\right) \right\} \\ &= \mathop{\mathbb{E}}_{x,t\sim d_{x,t}} \left\{ l(x, \pi_{\theta^k}) + \frac{\partial V_{w^k}}{\partial x^{\top}} f(x, \pi_{\theta^k}) \right\} \end{aligned} \tag{50}$$

where V_{w^k} becomes a known variable after critic update, and π_{θ^k} has unknown policy parameter θ.

To solve the parameter vector w of the value network and θ of the policy network, the key point is to calculate the gradients of the critic and actor in (49) and (50). In this case, the gradient of the loss J_{Critic} with respect to w is calculated as

$$\begin{aligned} \nabla_w J_{\text{Critic}} = & \mathop{\mathbb{E}}_{x,t\sim d_{x,t}} \Bigg\{ \Bigg(H\left(x, \pi_{\theta^k}, \frac{\partial V_{w^k}}{\partial x^{\top}}\right) \\ & - l\left(x^{\pi^k}(T), \pi\left(x^{\pi^k}(T), T; \theta^k\right)\right)\Bigg) \frac{\partial \left(\frac{\partial V_{w^k}}{\partial x^{\top}} f(x, \pi_{\theta^k})\right)}{\partial w} \Bigg\}. \end{aligned} \tag{51}$$

Meanwhile, the gradient of J_{Actor} is expressed as

$$\nabla_{\theta} J_{\text{Actor}} = \mathop{\mathbb{E}}_{x,t\sim d_{x,t}} \left\{ \frac{\partial \pi_{\theta^k}}{\partial \theta} \left(\frac{\partial l(x, \pi_{\theta^k})}{\partial \pi} + \frac{\partial f(x, \pi_{\theta^k})}{\partial \pi} \frac{\partial V_{w^k}}{\partial x^{\top}} \right) \right\}. \tag{52}$$

In fact, the weight vector is adjusted by taking a small step in opposite direction of the gradient, which aims to minimize the critic loss and actor loss. Therefore, the updating rule for tuning the parameter w of the critic is

$$w^k = w^k - \alpha_w \cdot \nabla J_{\text{Critic}} \tag{53}$$

where α_w is the step size. Then the updating rule for the actor is

$$\theta^k = \theta^k - \alpha_{\theta} \cdot \nabla J_{\text{Actor}} \tag{54}$$

where α_{θ} is the step size. Finally, the new parameters modify the current policy until approaching the final policy. The pseudocode and schematic illustration of the proposed constrained ADP algorithm is shown in Algorithm 3.

Algorithm 3 Parameterized CT Finite-Horizon ADP

Initialize w^0, θ^0, learning rates α_w and α_{θ}.
repeat
  Rollout with policy $\pi(x,t;\theta^k)$ from $\forall x(t) \in \Omega$ to $x^{\pi^k}(T)$
  1. Calculate $\frac{\partial J_{\text{Critic}}}{\partial w}$ using (51);
     Update the value function utilizing (53)
  2. Calculate $\frac{\partial J_{\text{Actor}}}{\partial \theta}$ using (52);
     Update the policy utilizing (54)
until $\|V_{w_{k+1}}(x,t) - V_{w_k}(x,t)\| \geq \epsilon$.

Remark 3: The proposed FH ADP relaxes the requirement for hand-crafted basis functions. It employs multilayer NNs to parameterize the value function and policy, mapping the system states and time to the value and control policy directly. Two separated NNs are trained with the loss functions J_{Critic} and J_{Actor}. Therefore, it is applicable for arbitrary nonlinear nonaffine systems as it learns an independent policy network.

V. Illustrative Examples

This section aims to give the verification for the effectiveness of the proposed FH ADP algorithm. The simulations with both the LQR problem and nonlinear vehicle tracking problem have been carried out.

A. Example 1: Linear System

1) Problem Description: To verify the optimality of the proposed FH ADP algorithm numerically, we first consider the general LQR control problem shown in (30) with $T = 0.5$, $a = -1$, $b = 1$, $q = 2$, and $r = 1$. In addition, the sampling frequency in the simulation is set as 50 Hz.

2) Details of the Algorithm: Both $V(x, t; w)$ and $\pi(x, t; \theta)$ are approximated by fully connected NNs with three layers and there are 256 units in every hidden layer. The input layer for both NNs contains state and t, followed by two hidden layers whose activation function is exponential linear units (ELUs). $V(x, t; w)$ is the output of the value network with softplus as the active function. $\pi(x, t; \theta)$ is the output of the policy network with tanh as the active function. We set learning rates α_w and α_θ to 0.001 and utilize Adam optimization method to update both NNs.

3) Result Analysis: We run Algorithm 3 for ten times with different random seeds, with evaluations every 100 iterations. Each evaluation measures the policy performance by calculating the relative errors of both policy and value, denoted as u_{error} and V_{error}. Given the optimal solution $\pi^*(x, t)$ and $V^*(x, t)$, the relative errors u_{error} and V_{error} of the proposed FH ADP algorithm are calculated by the equations

$$u_{\text{error}} = \mathop{\mathbb{E}}_{x,t\sim d_{x,t}}\left[\frac{|\pi(x,t;\theta^*) - \pi^*(x,t)|}{\max\limits_{x,t\sim d_{x,t}} \pi^*(x,t) - \min\limits_{x,t\sim d_{x,t}} \pi^*(x,t)}\right] \tag{55}$$

$$V_{\text{error}} = \mathop{\mathbb{E}}_{x,t\sim d_{x,t}}\left[\frac{|V(x,t;w^*) - V^*(x,t)|}{\max\limits_{x,t\sim d_{x,t}} V^*(x,t) - \min\limits_{x,t\sim d_{x,t}} V^*(x,t)}\right] \tag{56}$$

where the optimal solution $\pi^*(x, t)$ and $V^*(x, t)$ are obtained by calculating (31). Each evaluation contains 500 states-time pairs randomly chosen from $x \in \Omega$ and $t \in \lceil 0, T \rceil$ as the $d_{x,t}$.

The learning curves are shown in Fig. 1. Solid lines are average relative value and policy errors over ten runs. After 3×10^4 iterations, both the policy error and value error are less than 1%, which indicates that the algorithm has the ability to converge policy to optimality. Fig. 2 compares the value, state, and control input trajectories of Algorithm 3 and optimal solutions (31) from three different initial conditions, including $[t = 0.02, x = 5]^\top$, $[t = 0.1, x = 4]^\top$, and $[t = 0.2, x = -1]^\top$. Obviously, the trajectories generated by the learned policy almost overlap with the optimal solutions. As shown in Fig. 2(a), the proposed policy can drive the states to bounded results in the FH. Meanwhile, the descending direction is toward the equilibrium state.

B. Example 2: Nonlinear System

1) Problem Description: As the second example, the vehicle path tracking task is selected to show the applicability of Algorithm 3 to nonlinear systems. The linear four-dimension tracking control problem has been studied in [42]. The state and control input of vehicle are described as

$$x = \left[v_y, r, \varphi, y_e\right]^\top, \quad u = \delta \tag{57}$$

where v_y is vehicle lateral speed, r is the yaw rate, φ is the angle between vehicle heading direction and road tangent, and

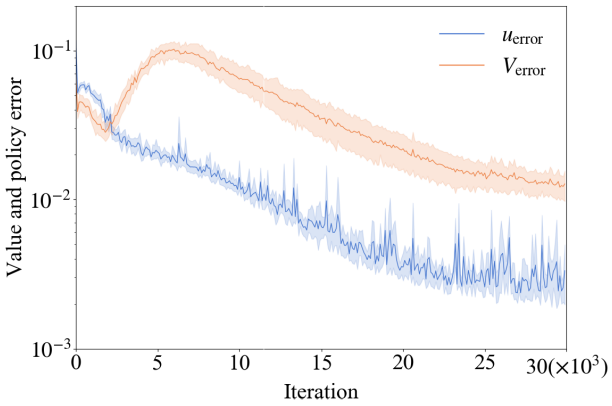

Fig. 1. Value and policy error in the training process. The solid lines correspond to the mean and the shaded regions correspond to 95% confidence interval over ten runs.

y_e is the lateral position error between the vehicle and trajectory. The position of the ego vehicle point and the reference point are denoted as $(x_{\text{ego}}, y_{\text{ego}})$ and (x_r, y_r), respectively. The position error y_e is calculated as $y_e = (y_{\text{ego}} - y_r)\cos\varphi$. δ is the steering angle of front wheels. The reference path is the superposition of the sine curves, which is designed as $y_r = 0.3\sin(x_r/12) + 0.6\sin(x_r/20) + 1.5\sin(x_r/30)$.

In this study, the nonlinear vehicle dynamics with the nonaffine saturated control input is used

$$\dot{x} = f(x, u) = \begin{bmatrix} \dfrac{F_{yf}\cos\delta + F_{yr}}{m} - v_x r \\ \dfrac{aF_{yf}\cos\delta - bF_{yr}}{I_{zz}} \\ r \\ v_x \sin\varphi + v_y \cos\varphi \end{bmatrix} \tag{58}$$

where F_{yf} and F_{yr} represent the front and rear tires' lateral tire forces, respectively. The parameters are listed in Table I. Considering the Pacejka tire model [43], [44], [45], the lateral tire forces can be approximated as

$$\begin{aligned} F_{yf} &= -Dmg\frac{b}{L}\sin\big(C\arctan\big(B\alpha_f\big)\big) \\ F_{yr} &= -Dmg\frac{a}{L}\sin(C\arctan(B\alpha_r)) \end{aligned} \tag{59}$$

where α_f and α_r represent the front and rear wheel slip angle, respectively, and $D = 0.75$, $C = 1.43$, $B = 14$ are the parameters of the Pacejka tire model. The slip angles can be calculated utilizing the geometric relationship

$$\begin{aligned} \alpha_f &= \arctan\left(\frac{v_y + ar}{v_x}\right) - \delta \\ \alpha_r &= \arctan\left(\frac{v_y - br}{v_x}\right). \end{aligned} \tag{60}$$

The aim of this task is to minimize the lateral tracking errors. Hence, the policy optimization problem of this example is given by

$$\begin{aligned} &\min_u \int_0^{0.5} \big(6y^2(\tau) + 80u^2(\tau)\big)d\tau \\ &\text{s.t. } \dot{x} = f(x, u). \end{aligned} \tag{61}$$

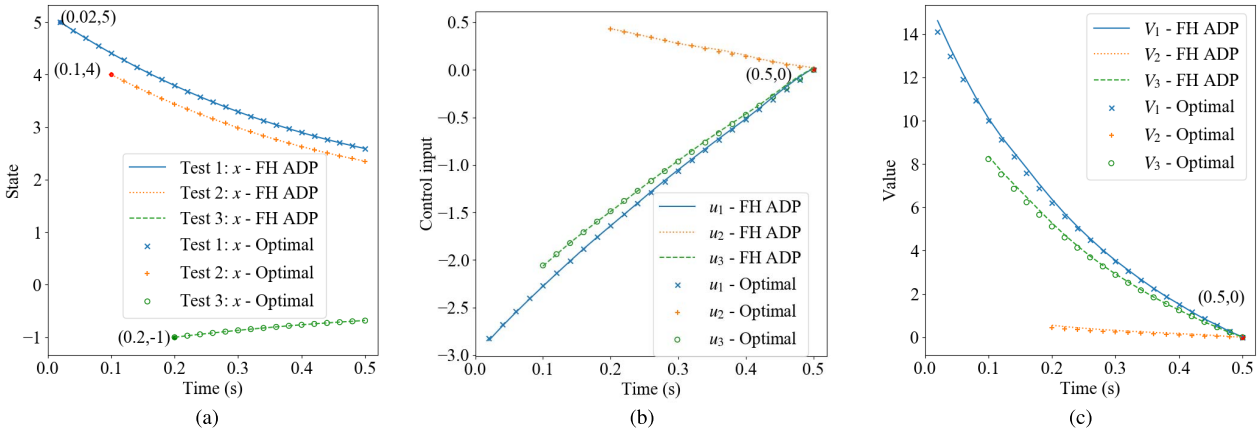

Fig. 2. State, control input, and value trajectories of FH ADP compared with the optimal LQR solution. The initial state of Test 1, 2, and 3 are $[t = 0.02, x = 5]^\top$, $[t = 0.1, x = 4]^\top$, and $[t = 0.2, x = -1]^\top$, respectively. (a) State, (b) control input, and (c) value function curves.

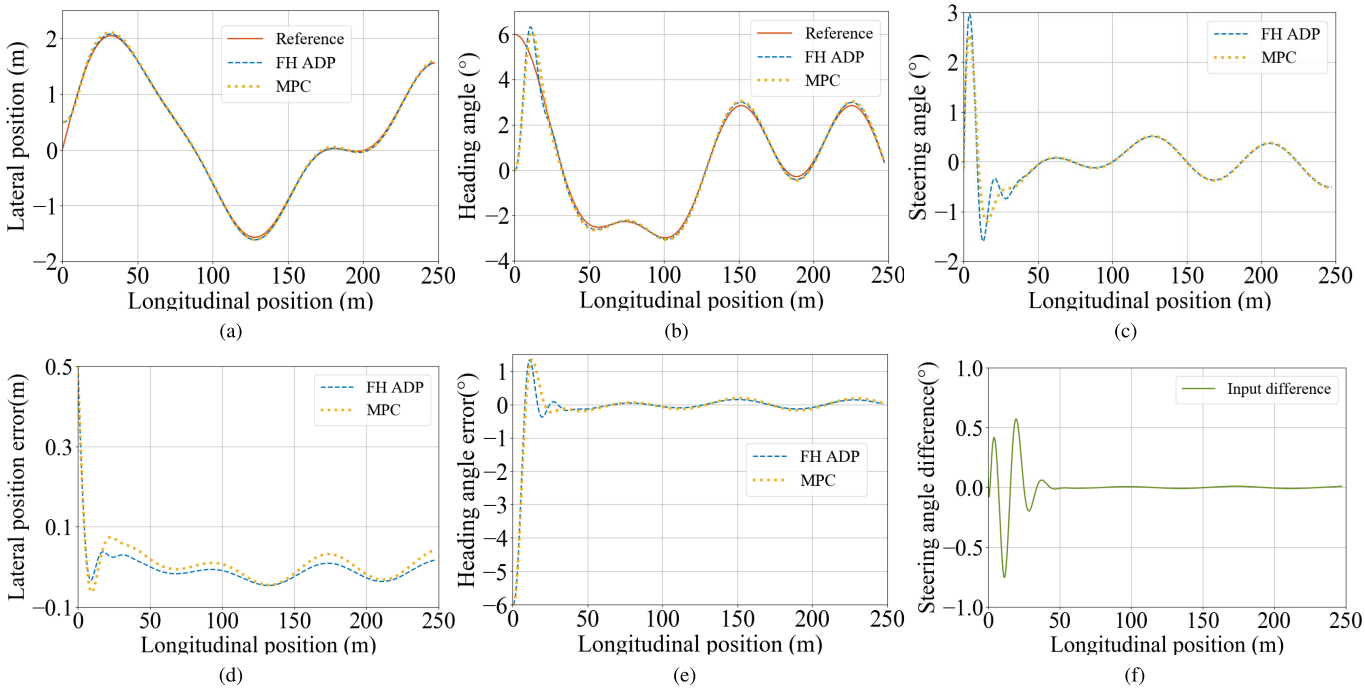

Fig. 3. Performance comparison for Example 2. (a) Lateral position. (b) Heading angle. (c) Control input. (d) Position tracking errors. (e) Heading angle tracking errors. (f) Control input difference between FH ADP and MPC.

TABLE I
VEHICLE PARAMETERS

Explanation	Symbol	Value
The cornering stiffness for the front wheel	k_1	-88000 [N/rad]
The cornering stiffness for the rear wheel	k_2	-94000 [N/rad]
Mass	m	1500 [kg]
Distance between front axle and CG	a	1.14 [m]
Distance between rear axle and CG	b	1.40 [m]
Polar moment of inertia at CG	I_{zz}	2420 [kg·m^2]
Longitudinal velocity	v_x	15 [m/s]
Sampling and simulation frequency	f	200 [Hz]

2) Details of the Algorithm: To ensure the approximation accuracy and training performance, multilayer NNs are applied to represent the value function $V(x, t; w)$ and the policy $\pi(x, t; \theta)$, which are comprised of an input layer, two hidden layers, and an output layer. The input layer for both NNs contains state x and time t, followed by two hidden layers whose activation functions are ELUs. Moreover, each hidden layer has 256 neurons. Then, the scalar $V(x, t; w)$ is the output of the value network with softplus as the activation function, and $\pi(x, t; \theta)$ is the output of the policy network with tanh as the activation function. The policy network for the front-wheel angle is further scaled to $[-0.4, 0.4]$ rad with a gain. Obviously, the architecture of the value network and value network is almost the same except for activation function for the output layer.

To accelerate the training process, a parallel exploring trick is applied [46]. By employing parallel agents to explore different state spaces, the diversity of the training set will

TABLE II
COMPARISON BETWEEN FH ADP AND MPC

Methods		FH ADP	MPC
Mean absolute error	Lateral position (m)	0.024	0.029
	Heading angle (°)	0.233	0.272

be improved, thereby avoiding overfitting. The batch size of the NNs is set as 256, which means that 256 agents are used to estimate the gradients of actor and critic for every iteration during the training. In addition, we utilize the Adam optimization method to update both NNs, and the learning rate is set as 10^{-3}.

3) Result Analysis: After 3×10^4 iterations of training, we obtain a learned policy $\pi(x, t; \theta^*)$ of Algorithm 3. Distinguished from the linear systems, we cannot get the CT optimal analytical policies for the nonlinear systems. In this case, the solver for model predictive control is utilized as a baseline to evaluate the accuracy performance for the learned policy of our algorithm, although it solves the discrete-time OCP. Consistent with (61), we can formulate the MPC problem as

$$\min_{u_t,\ldots,u_{t+T-1}\in U} V(x,t) = \sum_{i=0}^{T}\left(6y_{i|t}^2 + 80u_{i|t}^2\right)$$
$$\text{s.t. } x_{i+1|t} = x_{i|t} + f\left(x_{i|t}, u_{i|t}\right)\Delta t, \quad i = 0 : T-1 \quad (62)$$

where $\Delta t = 0.005$ and $T = 100$. The IPOPT solver with Casadi package [47] is adopted to calculate the optimal control input of the MPC problem formulated as (62).

Similar to the MPC method, we also implement the learned policy $\pi(x, t; \theta^*)$ in a receding-horizon way by setting $t = 0$. In other words, at each time step, $u = \pi(x, 0; \theta^*)$ is taken as the control input. The control results of Algorithm 3 and MPC are shown in Fig. 3. The horizontal axis (i.e., longitudinal position) represents the X-coordinate position of the vehicle center. Meanwhile, the vertical axis in Fig. 3(a) (i.e., longitudinal position) represents the Y-coordinate position of the vehicle center. It is shown that the state and control curves of Algorithm 3 are almost the same as those of MPC. Specifically, Fig. 3(a) and (d) shows the trajectories of the vehicle's lateral position. The maximum error of the ADP algorithm is 0.046 m while the one of MPC method is 0.074 m except for the initial position. In addition, the heading angle curves are shown in Fig. 3(b) and (e), where the maximum errors of both methods are less than 1.5° except for the initial position. Fig. 3(c) and (f) reflects the difference on the control input between the ADP algorithm and MPC, which is less than 0.57°. In conclusion, the performance attained by the proposed ADP algorithms is at a comparable level with the MPC. Furthermore, Table II compares the tracking accuracy of CT FH ADP and MPC for the whole tracking simulation process. We can see that the FH matches MPC in terms of tracking accuracy, which demonstrates the optimality and effectiveness of the proposed algorithm in the case of nonlinear systems.

VI. CONCLUSION

This work investigates the solving of the FH HJB equation. Essentially in the stated problem, the partial time derivative of the value function hinders the solving for CT FH OCP by the PI technique. For the first time, this work constructs the connection between the partial time derivative and the terminal-time utility function. Meanwhile, the initial-time equivalence between two FH OCPs is utilized to support the theoretical analysis and rigorous proof of this finding. Based on the FH HJB equation, we develop the FH ADP algorithm with the actor–critic framework for solving CT FH OCPs. The convergence and optimality have been proved to be guaranteed. Meanwhile, multilayer NNs are adopted for parameterizing the value function and control input, so that the proposed algorithm is suitable for the general nonlinear systems with high complexity. Finally, two simulations are carried out to demonstrate the efficacy and generality of the proposed development, and the results show that the proposed FH ADP algorithm is convergent to the nearly optimal policy for general CT FH OCPs. In the future, the practical implementation of the algorithm will be carried out on the motion control problem of vehicles, which aims to improve the online computation efficiency for solving the control policy.

APPENDIX
DERIVATION DETAILS

By deriving the HJB equation for the LQR problem, the well-known Riccati equation can be obtained as

$$\dot{P}(t) = -2aP(t) + \frac{b^2}{r}P^2(t) - q. \quad (63)$$

When solving the solution to Riccati equation (63), an intermediate variable $y(t)$ is introduced to eliminate the constant term. It is set as

$$y(t) = P(t) + \frac{r}{b^2}(-a + \beta) \quad (64)$$

where $\beta = ((qb^2/r) + a^2)^{1/2}$. Then, the Riccati equation is converted to the differential equation as

$$\dot{y}(t) + 2\beta y(t) = \frac{b^2}{r}y^2(t). \quad (65)$$

Following it, we can divide the differential equation by y^2 to get

$$y^{-2}\dot{y}(t) + 2\beta y^{-1}(t) = \frac{b^2}{r}. \quad (66)$$

Subsequently, the variable substitution can be performed as

$$z(t) = y^{-1}(t). \quad (67)$$

Then, (65) is converted to a differential equation in terms of $z(t)$ as

$$\dot{z}(t) - 2\beta z(t) = -\frac{b^2}{r}. \quad (68)$$

The solution to the linear ordinary differential equation is calculated as

$$z(t) = e^{2\beta t}\left(\int_0^t e^{-2\beta t}\left(-\frac{b^2}{r}\right)dt + C\right)$$
$$= \frac{b^2}{2\beta r}\left(1 - \eta e^{2\beta(t-T)}\right) \quad (69)$$

where $\eta = -(a+\beta/-a+\beta)$. So, the solution to (65) can be solved subsequently

$$y(t) = \frac{2\beta r}{b^2}\frac{1}{1-\eta e^{2\beta(t-T)}}. \tag{70}$$

Thereafter, by replacing y with P, we obtain the solution of (63) as

$$\begin{aligned} P(t) &= \frac{r}{b^2}\left(\frac{2\beta}{1-\eta e^{2\beta(t-T)}} - (-a+\beta)\right) \\ &= \frac{r}{b^2}\frac{\beta + a + \eta(\beta - a)e^{2\beta(t-T)}}{1-\eta e^{2\beta(t-T)}}. \end{aligned} \tag{71}$$

The following content will derive (34). According to [48], the optimal state trajectory in (30) satisfies

$$x^*(\tau) = x(t)e^{\int_t^\tau \left(a-\frac{b^2}{r}P(\mathrm{t})\right)dt} \tag{72}$$

where the variable $\tau \in [t, T]$. From (32), one can further derive the integral term in (72)

$$\begin{aligned} &\int_t^\tau \left(a - \frac{b^2}{r}P(\mathrm{t})\right)dt \\ &= \int_t^\tau \left(a - \frac{b^2}{r}\frac{r}{b^2}\left(\frac{2\beta}{1-\eta e^{2\beta(\mathrm{t}-T)}} - (-a+\beta)\right)\right)dt \\ &= \int_t^\tau \left(\beta - \frac{2\beta}{1-\eta e^{2\beta(\mathrm{t}-T)}}\right)dt \\ &= \beta(\tau - t) - 2\beta\int_t^\tau \frac{1}{1-\eta e^{2\beta(\mathrm{t}-T)}}dt \\ &= \beta(\tau - t) - 2\beta\left[(\tau - t) - \frac{1}{2\beta}\ln\left(\frac{1-\eta e^{2\beta(\tau-T)}}{1-\eta e^{2\beta(t-T)}}\right)\right] \\ &= -\beta(\tau - t) + \ln\left(\frac{1-\eta e^{2\beta(\tau-T)}}{1-\eta e^{2\beta(t-T)}}\right). \end{aligned} \tag{73}$$

Hence, the optimal state trajectory can be derived as

$$\begin{aligned} x^*(\tau) &= x(t)e^{\int_t^\tau \left(a-\frac{b^2}{r}P(\mathrm{t})\right)dt} \\ &= x(t)e^{\left[-\beta(\tau-t)+\ln\left(\frac{1-\eta e^{2\beta(\tau-T)}}{1-\eta e^{2\beta(t-T)}}\right)\right]} \\ &= x(t)\left(\frac{1-\eta e^{2\beta(\tau-T)}}{1-\eta e^{2\beta(t-T)}}\right)e^{-\beta(\tau-t)}, \quad \tau \in [t, T]. \end{aligned} \tag{74}$$

Therefore, one can get the optimal state $x^*(T)$ at the terminal time by letting $\tau = T$, i.e.,

$$x^*(T) = x(t)\left(\frac{1-\eta e^{2\beta(T-T)}}{1-\eta e^{2\beta(t-T)}}\right)e^{-\beta(T-t)}. \tag{75}$$

From (31), $(\partial V^*(x(t), t)/\partial t)$ of system (30) is

$$\frac{\partial V^*(x(t), t)}{\partial t} = \frac{1}{2}\dot{P}(t)x(t)^2. \tag{76}$$

According to (32), we have

$$\dot{P}(t) = (2\beta)^2\frac{r\eta}{b^2}\frac{e^{2\beta(t-T)}}{\left(1-\eta e^{2\beta(t-T)}\right)^2}. \tag{77}$$

Then, (76) can be further expanded as

$$\begin{aligned} &\frac{\partial V^*(x(t), t)}{\partial t} \\ &= -\frac{1}{2}(2\beta)^2\frac{r}{b^2}\frac{a+\beta}{-a+\beta}x(t)^2\frac{e^{2\beta(t-T)}}{\left(1-\eta e^{2\beta(t-T)}\right)^2} \\ &= -\frac{1}{2}(2\beta)^2\frac{r}{b^2}\frac{-a^2+\beta^2}{(-a+\beta)^2}x(t)^2\frac{e^{2\beta(t-T)}}{\left(1-\eta e^{2\beta(t-T)}\right)^2} \\ &= -\frac{1}{2}\frac{r}{b^2}\left(\frac{qb^2}{r} + a^2 - a^2\right)\frac{(2\beta)^2}{(-a+\beta)^2}x(t)^2\frac{e^{2\beta(t-T)}}{\left(1-\eta e^{2\beta(t-T)}\right)^2} \\ &= -\frac{1}{2}qx(t)^2\frac{(2\beta)^2}{(-a+\beta)^2}\frac{e^{2\beta(t-T)}}{\left(1-\eta e^{2\beta(t-T)}\right)^2}. \end{aligned} \tag{78}$$

In addition, from (31) and (75), one can derive $l(x^*(T), \pi^*(x(T), T))$ as

$$\begin{aligned} &l\left(x^*(T), \pi^*(x(T), T)\right) \\ &= \frac{1}{2}qx^*(T)^2 + \frac{1}{2}r\pi^*(x(T), T)^2 \\ &= \frac{1}{2}qx(T)^2 \\ &= \frac{1}{2}qx(t)^2\left(\frac{1-\eta e^{2\beta(T-T)}}{1-\eta e^{2\beta(t-T)}}\right)^2 e^{-2\beta(T-t)} \\ &= \frac{1}{2}qx(t)^2 e^{-2\beta(T-t)}\left(\frac{1-\eta}{1-\eta e^{2\beta(t-T)}}\right)^2 \\ &= \frac{1}{2}qx(t)^2(1-\eta)^2\frac{e^{2\beta(t-T)}}{\left(1-\eta e^{2\beta(t-T)}\right)^2} \\ &= \frac{1}{2}qx(t)^2\left(\frac{2\beta}{-a+\beta}\right)^2\frac{e^{2\beta(t-T)}}{\left(1-\eta e^{2\beta(t-T)}\right)^2}. \end{aligned} \tag{79}$$

Finally, we can get (34). In addition, it is noted that $l(x^*(T), \pi^*(x(T), T))$ is still time-dependent. It is due to that the time-dependent terminal-time state $x^\pi(T)$ is obtained by rolling forward the dynamic model from the initial state $x(t)$ with the policy π, $u(T) = \pi(x^\pi(T), T)$.

REFERENCES

[1] F. L. Lewis, D. Vrabie, and V. L. Syrmos, *Optimal Control*. Hoboken, NJ, USA: Wiley, 2012.

[2] J. Chen, W. Zhan, and M. Tomizuka, "Autonomous driving motion planning with constrained iterative LQR," *IEEE Trans. Intell. Veh.*, vol. 4, no. 2, pp. 244–254, Jun. 2019.

[3] J. Ma, Z. Cheng, X. Zhang, M. Tomizuka, and T. H. Lee, "Alternating direction method of multipliers for constrained iterative LQR in autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, early access, Aug. 17, 2022, doi: 10.1109/TITS.2022.3194571.

[4] X. Zhang, Z. Cheng, J. Ma, S. Huang, F. L. Lewis, and T. H. Lee, "Semi-definite relaxation-based ADMM for cooperative planning and control of connected autonomous vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 9240–9251, Jul. 2022.

[5] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, vol. 1, no. 2. Belmont, MA, USA: Athena Scientific, 1995.

[6] P. J. Werbos, "Approximate dynamic programming for realtime control and neural modeling," in *Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches*, D. A. White and D. A. Sofge, Eds. New York, NY, USA: Van Nostrand Reinhold, 1992.

[7] S. E. Li, *Reinforcement Learning for Sequential Decision and Optimal Control*. Singapore: Springer, 2023.

[8] W. B. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality*, vol. 703. Hoboken, NJ, USA: Wiley, 2007.

[9] W. E. Dixon, "Reinforcement learning for approximate optimal control," in *Encyclopedia of Systems and Control*. Cham, Switzerland: Springer, 2021, pp. 1863–1868.

[10] R. Howard, *Dynamic Programming and Markov Processes*. Cambridge, MA, USA: MIT Press, 1960.

[11] J. Duan, S. Eben Li, Z. Liu, M. Bujarbaruah, and B. Cheng, "Generalized policy iteration for optimal control in continuous time," 2019, *arXiv:1909.05402*.

[12] K. G. Vamvoudakis and F. L. Lewis, "Online actor–critic algorithm to solve the continuous-time infinite horizon optimal control problem," *Automatica*, vol. 46, no. 5, pp. 878–888, 2010.

[13] D. Vrabie, O. Pastravanu, M. Abu-Khalaf, and F. L. Lewis, "Adaptive optimal control for continuous-time linear systems based on policy iteration," *Automatica*, vol. 45, no. 2, pp. 477–484, Feb. 2009.
[14] Y. Fu, J. Fu, and T. Chai, "Robust adaptive dynamic programming of two-player zero-sum games for continuous-time linear systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 12, pp. 3314–3319, Dec. 2015.
[15] R. Kamalapurkar, J. A. Rosenfeld, and W. E. Dixon, "Efficient model-based reinforcement learning for approximate online optimal control," *Automatica*, vol. 74, pp. 247–258, Dec. 2016.
[16] S. He, H. Fang, M. Zhang, F. Liu, X. Luan, and Z. Ding, "Online policy iterative-based H_∞ optimization algorithm for a class of nonlinear systems," *Inf. Sci.*, vol. 495, pp. 1–13, Jan. 2019.
[17] S. He, H. Fang, M. Zhang, F. Liu, and Z. Ding, "Adaptive optimal control for a class of nonlinear systems: The online policy iteration approach," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 2, pp. 549–558, Feb. 2020.
[18] H. Fang et al., "Adaptive optimization algorithm for nonlinear Markov jump systems with partial unknown dynamics," *Int. J. Robust Nonlinear Control*, vol. 31, no. 6, pp. 2126–2140, 2021.
[19] M. Deniz, T. Yucelen, and S. N. Balakrishnan, "Adaptive control of uncertain dynamical systems with an optimal nonlinear reference model," in *Proc. AIAA SCITECH Forum*, 2022, p. 1584.
[20] Z. Lin et al., "Solving finite-horizon HJB for optimal control of continuous-time systems," in *Proc. Int. Conf. Comput., Control Robot. (ICCCR)*, 2021, pp. 116–122.
[21] A. Sahoo, H. Xu, and S. Jagannathan, "Approximate optimal control of affine nonlinear continuous-time systems using event-sampled neurodynamic programming," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 3, pp. 639–652, Mar. 2017.
[22] B. Talaei, S. Jagannathan, and J. Singler, "Boundary control of linear uncertain 1-D parabolic PDE using approximate dynamic programming," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 4, pp. 1213–1225, Apr. 2018.
[23] R. Song, F. Lewis, Q. Wei, H.-G. Zhang, Z.-P. Jiang, and D. Levine, "Multiple actor-critic structures for continuous-time optimal control using input-output data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 4, pp. 851–865, Apr. 2015.
[24] R. Kamalapurkar, L. Andrews, P. Walters, and W. E. Dixon, "Model-based reinforcement learning for infinite-horizon approximate optimal tracking," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 3, pp. 753–758, Mar. 2017.
[25] J. Ma, Z. Cheng, X. Zhang, Z. Lin, F. L. Lewis, and T. H. Lee, "Local learning enabled iterative linear quadratic regulator for constrained trajectory planning," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, May 2, 2022, doi: 10.1109/TNNLS.2022.3165846.
[26] Y. Ge, S. Li, and P. Chang, "An approximate dynamic programming method for the optimal control of alkali-surfactant-polymer flooding," *J. Process Control*, vol. 64, pp. 15–26, Apr. 2018.
[27] C. Mu, D. Wang, and H. He, "Novel iterative neural dynamic programming for data-based approximate optimal control design," *Automatica*, vol. 81, pp. 240–252, Jul. 2017.
[28] A. Heydari and S. N. Balakrishnan, "Fixed-final-time optimal control of nonlinear systems with terminal constraints," *Neural Netw.*, vol. 48, pp. 61–71, Dec. 2013.
[29] T. Cheng, F. Lewis, and M. Abu-Khalaf, "A neural network solution for fixed-final time optimal control of nonlinear systems," *Automatica*, vol. 43, no. 3, pp. 482–490, Mar. 2007.
[30] M. Abu-Khalaf and F. L. Lewis, "Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network HJB approach," *Automatica*, vol. 41, no. 5, pp. 779–791, May 2005.
[31] T. Cheng, F. L. Lewis, and M. Abu-Khalaf, "Fixed-final-time-constrained optimal control of nonlinear systems using neural network HJB approach," *IEEE Trans. Neural Netw.*, vol. 18, no. 6, pp. 1725–1737, Nov. 2007.
[32] D. M. Adhyaru, M. Gopal, and I. N. Kar, "Fixed final time optimal control approach for bounded robust controller design using Hamilton–Jacobi–Bellman solution," *IET Control Theory Appl.*, vol. 3, no. 9, pp. 1183–1195, Sep. 2009.
[33] Z. Zhao, Y. Yang, H. Li, and D. Liu, "Approximate finite-horizon optimal control with policy iteration," in *Proc. 33rd Chin. Control Conf.*, Jul. 2014, pp. 8895–8900.
[34] N. Xu, B. Niu, H. Wang, X. Huo, and X. Zhao, "Single-network ADP for solving optimal event-triggered tracking control problem of completely unknown nonlinear systems," *Int. J. Intell. Syst.*, vol. 36, no. 9, pp. 4795–4815, Sep. 2021.
[35] K. Zhang, R. Su, H. Zhang, and Y. Tian, "Adaptive resilient event-triggered control design of autonomous vehicles with an iterative single critic learning framework," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 12, pp. 5502–5511, Dec. 2021.
[36] 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.
[37] T. Nakamura-Zimmerer, Q. Gong, and W. Kang, "Adaptive deep learning for high-dimensional Hamilton–Jacobi–Bellman equations," *SIAM J. Sci. Comput.*, vol. 43, no. 2, pp. A1221–A1247, Jan. 2021.
[38] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
[39] D. Liu, Q. Wei, D. Wang, X. Yang, and H. Li, *Adaptive Dynamic Programming With Applications in Optimal Control*. Berlin, Germany: Springer, 2017.
[40] Z. Allen-Zhu, Y. Li, and Z. Song, "A convergence theory for deep learning via over-parameterization," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 242–252.
[41] S. Du, J. Lee, H. Li, L. Wang, and X. Zhai, "Gradient descent finds global minima of deep neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 1675–1685.
[42] Z. Lin, J. Duan, S. E. Li, H. Ma, Y. Yin, and B. Cheng, "Continuous-time finite-horizon ADP for automated vehicle controller design with high efficiency," in *Proc. 3rd Int. Conf. Unmanned Syst. (ICUS)*, Nov. 2020, pp. 978–984.
[43] E. Bakker, L. Nyborg, and H. B. Pacejka, "Tyre modelling for use in vehicle dynamics studies," *SAE Trans.*, vol. 10, pp. 190–204, Jan. 1987.
[44] R. Li, Y. Li, S. E. Li, C. Zhang, E. Burdet, and B. Cheng, "Indirect shared control for cooperative driving between driver and automation in steer-by-wire vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 12, pp. 7826–7836, Dec. 2021.
[45] S. E. Li, H. Chen, R. Li, Z. Liu, Z. Wang, and Z. Xin, "Predictive lateral control to stabilise highly automated vehicles at tire-road friction limits," *Vehicle Syst. Dyn.*, vol. 58, no. 5, pp. 768–786, 2020.
[46] V. Mnih et al., "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1928–1937.
[47] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi: A software framework for nonlinear optimization and optimal control," *Math. Program. Comput.*, vol. 11, no. 1, pp. 1–36, Mar. 2018.
[48] H. Kwakernaak and R. Sivan, *Linear Optimal Control Systems*, vol. 1. New York, NY, USA: Wiley, 1972.

Ziyu Lin received the B.S. degree in automotive engineering from China Agricultural University, Beijing, China, in 2017. She is currently pursuing the Ph.D. degree with the School of Vehicle and Mobility, Tsinghua University, Beijing.

She studied as a Visiting Student Researcher with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore, from 2021 to 2022. Her current research interests include reinforcement learning, approximate dynamic programming, model predictive control, and autonomous driving.

Ms. Lin was a recipient of the Best Paper Award at the IEEE 2020 3rd International Conference on Unmanned Systems and the Best Presentation Award at the IEEE 2021 International Conference on Computer Control and Robotics.

Jingliang Duan (Member, IEEE) received the Ph.D. degree from the School of Vehicle and Mobility, Tsinghua University, Beijing, China, in 2021.

He studied as a Visiting Student Researcher with the Department of Mechanical Engineering, University of California at Berkeley, Berkeley, CA, USA, in 2019, and worked as a Research Fellow with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore, from 2021 to 2022. He is currently an Associate Professor with the School of Mechanical Engineering, University of Science and Technology Beijing, Beijing. His research interests include reinforcement learning, optimal control, and self-driving decision-making.

Shengbo Eben Li (Senior Member, IEEE) received the M.S. and Ph.D. degrees from Tsinghua University, Beijing, China, in 2006 and 2009, respectively.

He worked at Stanford University, Stanford, CA, USA, the University of Michigan, Ann Arbor, MI, USA, and the University of California at Berkeley, Berkeley, CA, USA. He is currently a Tenured Professor at Tsinghua University. He is the author of over 100 journals/conference papers and the co-inventor of over 20 Chinese patents. His active research interests include intelligent vehicles and driver assistance, reinforcement learning and distributed control, and optimal control and estimation.

Dr. Li was a recipient of the Best Paper Award in 2014 IEEE ITS Symposium, the Best Paper Award in 14th ITS Asia–Pacific Forum, the National Award for Technological Invention in China in 2013, the Excellent Young Scholar of NSF China in 2016, and the Young Professorship of Changjiang Scholar Program in 2016. He serves as an Associate Editor for *IEEE Intelligent Transportation Systems Magazine* and IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS.

Haitong Ma received the B.S. degree from the School of Vehicle and Mobility, Tsinghua University, Beijing, China, in 2019, and the master's degree from the School of Vehicle and Mobility, Tsinghua University, in 2022. He is currently pursuing the Ph.D. degree with Harvard University, Cambridge, MA, USA.

His current research interests include optimal control and safe-critical systems.

Jie Li received the B.S. degree in automotive engineering from Tsinghua University, Beijing, China, in 2018, where he is currently pursuing the Ph.D. degree with the School of Vehicle and Mobility.

He studied as a Visiting Student Researcher with the Department of Mechanical Engineering, The University of British Columbia, Vancouver, BC, Canada, in 2022. His current research interests include model predictive control, adaptive dynamic programming, and robust reinforcement learning.

Jianyu Chen received the bachelor's degree from Tsinghua University, Beijing, China, in 2015, and the Ph.D. degree from the University of California at Berkeley, Berkeley, CA, USA, in 2020.

He has been an Assistant Professor with the Institute for Interdisciplinary Information Sciences (IIIS), Tsinghua University, since 2020. Prior to that, he was working with Prof. Masayoshi Tomizuka at the University of California at Berkeley. He is working in the cross fields of robotics, reinforcement learning, and control and autonomous driving. His research goal is to build advanced robotic systems with high performance and high intelligence.

Bo Cheng received the B.S. and M.S. degrees in automotive engineering from Tsinghua University, Beijing, China, in 1985 and 1988, respectively, and the Ph.D. degree in mechanical engineering from The University of Tokyo, Tokyo, Japan, in 1998.

He is currently a Professor with the School of Vehicle and Mobility, Tsinghua University, and the Dean of the Tsinghua University Suzhou Automotive Research Institute. He is the author of more than 100 peer-reviewed journals/conference papers and the co-inventor of 40 patents. His active research interests include autonomous vehicles, driver-assistance systems, active safety, and vehicular ergonomics.

Dr. Cheng is the Chairman of the Academic Board of SAE-Beijing, a member of the Council of the Chinese Ergonomics Society, and a Committee Member of the National 863 Plan.

Jun Ma received the B.Eng. degree (First Class Hons.) in electrical and electronic engineering from Nanyang Technological University, Singapore, in 2014, and the Ph.D. degree in electrical and computer engineering from the National University of Singapore, Singapore, in 2018.

From 2018 to 2021, he held several positions at the National University of Singapore; University College London, London, U.K.; the University of California at Berkeley, Berkeley, CA, USA; and Harvard University, Cambridge, MA, USA. He is currently an Assistant Professor with the Robotics and Autonomous Systems Thrust, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou, China, and an Assistant Professor with the Division of Emerging Interdisciplinary Areas and an Affiliate Assistant Professor with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Hong Kong SAR, China. His research interests include control, optimization, and machine learning with application to robotics and autonomous driving.