2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

# Adaptive Optimal Control of Nonlinear Systems with Multiple Time-scale Eligibility Traces

Jun Rao, Jingcheng Wang, Jiahui Xu, and Shunyu Wu

*Abstract*— **Adaptive dynamic programming (ADP) is one of the main methods to solve the optimal control problem of nonlinear systems. Eligibility traces are utilized in recent years to reduce the computing burden of the value function, but the existing fixed eligibility trace is difficult to ensure stable convergence especially when facing environmental changes and complex neural network structures. To solve the above issues, a novel off-policy algorithm, $T$-HDP($\lambda$) with Multiple Time-scale Eligibility Traces (MET), is proposed. By utilizing MET, the new algorithm can adaptively accumulate gradients and include more gradient information, which guides the control faster in the optimal direction. $T$-step Truncated $\lambda$-returns are utilized to solve the infinite-horizon optimal control problems, and a new importance sampling ratio is designed to correct the value function. Furthermore, the convergence and boundedness of the algorithm are proved. Based on the actor-critic network architecture, the optimal value function and policy are well approximated. Finally, compared with the original algorithm by a simulation example, the proposed algorithm has a faster convergence speed and lower variance.**

## I. Introduction

Adaptive dynamic programming (ADP) is a traditional method to solve the optimal control problems of nonlinear systems. It can speed up the process of the solution by avoiding the "curse of dimensionality" [1]. Policy iteration (PI) and value iteration (VI) are two traditionally used iterative ADP algorithms. During the iteration process, the performance index and the control law will finally converge to the optimal value respectively [2]–[4].

The accurate numerical solution of the Hamilton-Jacobi-Bellman (HJB) equation is still the main obstacle. The key step of the ADP method is approximating the performance index so as to determine the optimal control policy, which exactly meets the goal of reinforcement learning (RL). Thus, the optimal control problems of discrete systems combing with RL have sprung up in recent years [5]–[8]. Lots of advanced achievements and technology are introduced into ADP to realize the optimal controller design of the system.

There are many derived approaches with ADP or actor-critic framework, such as heuristic dynamic programming (HDP), dual heuristic dynamic programming (DHP), globalized dual heuristic dynamic programming (GDHP), deep deterministic policy gradient (DDPG) algorithm, and so on. All the above algorithms are derived by the temporal difference (TD) learning, which is used to approximate the value function that depends on future values under a given policy. Affected by harsh environments or destructive tests, in many cases, the sampling data of the control system is limited. Designing a control algorithm with limited information is another problem that needs to be solved emphatically.

Although there are many methods to expand the data set, such as data augmentation and generative adversarial networks [9]–[12], the special properties of timing and transition probability involved in the Markov decision process (MDP) make these methods no longer applicable. Hence, eligibility traces are introduced in the optimal control theory, which records the past and current gradients to speed up the learning process and solve the problem of sparse data. Lots of new algorithms appeared by combining eligibility traces. HDP($\lambda$) [13] combined the eligibility trace with the HDP framework. In [14], a group of expected eligibility algorithms was proposed to make full use of the counterfactual sequences which could also have led to the desired state or goal. Moreover, some eligibility traces algorithms were proposed to solve optimal control problems for discrete-time systems, such as gradient compensation eligibility traces [15] and GDHP schemes with eligibility traces [16]. Sarsa($\lambda$) algorithms are utilized to solve the optimal control problems for real physical systems such as the community energy storage operation problem [17], static ship path planning problem [18] and underwater vehicles optimal control in dynamic environments [19].

In most of these related works, however, the update of the value function using eligibility traces depends on returns which are quite far in the future. For the infinite-horizon optimal control problems, the above methods are no longer suitable, which is still a significant intractable issue [20], [21]. Moreover, the existing fixed eligibility trace calculation cannot realize dynamic adjustment, and it is difficult to ensure stable convergence when fitting environmental changes and complex neural network structures.

Motivated by the aforementioned works, the main contributions of this paper can be listed as follows:

(1) A novel model-based algorithm, named $T$-step Truncated HDP with Multiple Time-scale Eligibility Traces ($T$-HDP($\lambda$) with MET), is proposed. The algorithm utilizes the adaptive eligibility traces to avoid the problem that the approximation using a deep neural network may not converge. Compared with [13], our algorithm guides the control faster in the optimal direction without increasing the computational cost.

This work was supported by the National Natural Science Foundation of China under Grant 62273234. (Corresponding author: Jingcheng Wang )

Jun Rao, Jingcheng Wang, Jiahui Xu, and Shunyu Wu are with the Department of Automation, Key Laboratory of System Control and Information Processing, Shanghai Jiao Tong University, Shanghai, 200240, China. E-mail:{raojun,jcwang,xu.jiahui,shunyuwu}@sjtu.edu.cn

(2) It is an off-policy algorithm to get better policy exploration results and improve data utilization. A new importance sampling ratio is designed to better match multiple time-scale eligibility traces.

(3) $T$-step truncated finite terms of the optimal performance index function are used to solve the infinite-horizon optimal control problems. The convergence and boundedness of finite terms in the optimal performance index are proved.

(4) A novel convergence analysis is shown to guarantee that the iterative value function of $T$-HDP($\lambda$) with MET algorithm converges to the optimal performance index.

The next sections are organized as follows: In Section II, the preliminaries and basic concepts are introduced. Section III presents the details of $T$-HDP($\lambda$) with MET algorithm, including the novel multiple time-scale eligibility traces and the convergence analysis of the state-value function sequence. In Section IV, the simulation results show the effectiveness of the proposed algorithm Finally, the conclusion is given.

## II. Problem Statement and Preliminaries

Consider a general discrete-time dynamic system given by

$$x_{k+1} = f(x_k) + g(x_k)u_k \tag{1}$$

where $x_k \in R^n$ is the state and $u_k \in R^m$ is the control signal. $n$ and $m$ denote the dimensions of state and control space. $f(x)$ and $g(x)$ are continuous functions with $f(0) = 0$. It is assumed that system (1) is stabilizable on the set $\Omega$.

In optimal control problems, the performance index, as well as the cost function from the initial time $k = 0$ is denoted by $\mathcal{J}(x_0) = \sum_{k=0}^{\infty} \gamma^k \left(x_k^T Q x_k + u_k^T R u_k\right) = \sum_{k=0}^{\infty} \gamma^k \mathcal{R}(x_k, u_k)$, where $0 < \gamma < 1$ is a discount factor, $x_0$ denotes the initial state, and $\mathcal{R}(x_k, u_k) = x_k^T Q x_k + u_k^T R u_k$ represents a utility function with $Q$ and $R$ being symmetric and positive definite matrices. Denote $\mathcal{J}(x_k)$ with only respect to $x_k$ and $\mathcal{R}(x_k, u_k)$ as $\mathcal{J}_k$ and $\mathcal{R}_k$ at time $k$ respectively.

The objection of the optimal control is to find an admissible control law series $u_k$ $(k = 0, 1, 2, \dots)$ which minimizes the cost function $\mathcal{J}(x_0)$

$$\mathcal{J}^*(x_0) = \mathcal{J}(x_0, u_k^*) = \min_{u_k} \left(\sum_{k=0}^{\infty} \gamma^k \mathcal{R}(x_k, u_k)\right) \tag{2}$$

where the optimal control law $u_k^*$ can be defined as

$$u_k^* = \arg\min_{u_k} \left(\sum_{k=0}^{\infty} \gamma^k \mathcal{R}(x_k, u_k)\right) \tag{3}$$

The iteration form of computing $\mathcal{J}(x_k)$ from the initial state $x_k$ can be rewritten as

$$\begin{aligned} \mathcal{J}(x_k) &= \mathcal{R}(x_k, u_k) + \gamma \sum_{s=k+1}^{\infty} \gamma^{s-k-1} \mathcal{R}(x_s, u_s) \\ &= \mathcal{R}(x_k, u_k) + \gamma \mathcal{J}(x_{k+1}) \end{aligned} \tag{4}$$

Using Bellman's optimality principle in optimal control theorem, the iteration form of optimality is shown as

$$\mathcal{J}^*(x_k) = \min_{u_k} \left(\mathcal{R}(x_k, u_k) + \gamma \mathcal{J}^*(x_{k+1})\right) \tag{5}$$

*Definition 1 (State-value function):* $v_\pi(x_k)$ defined in Eq.(6) as the expected return and following policy $\pi$ thereafter, is called the state-value function for policy $\pi$.

$$v_\pi(x_k) = \mathbb{E}_\pi[\mathcal{J}(x_k)] = \mathbb{E}_\pi\left[\sum_{s=k}^{\infty} \gamma^{s-k} \mathcal{R}(x_s, u_s)\right] \tag{6}$$

In fact, the accurate true value of state-value function $v_\pi(x_k)$ is hard to be observed. Here, an approximate value of state-value function $\mathcal{V}_\pi(x_k)$ estimated by neural networks is used. The Eq.(4) can be transferred to approximate state-value function form as follows

$$\mathcal{V}_\pi(x_k) = \min_{u_k} \left(\mathcal{R}(x_k, u_k) + \gamma \mathcal{V}_\pi(x_{k+1})\right) \tag{7}$$

*Definition 2 (n-step return):* The return $\mathcal{J}_{k:k+n}(x_k)$ defined in Eq.(8), which accumulates the reward from time $k$ to time $k + n$, is called $n$-step return.

$$\begin{aligned} \mathcal{J}_{k:k+n}(x_k) =& \mathcal{R}(x_k, u_k) + \gamma \mathcal{R}(x_{k+1}, u_{k+1}) + \gamma^n \mathcal{V}_\pi(x_{k+n}) \\ =& \sum_{s=k}^{k+n-1} \gamma^{s-k} \mathcal{R}(x_s, u_s) + \gamma^n \mathcal{V}_\pi(x_{k+n}) \end{aligned} \tag{8}$$

Similar to the one-step return, $n$-step returns accumulate the next $n$-steps rewards and obtain more information about the future. Being truncated after $n$ steps and corrected for the remaining missing terms by $\mathcal{V}_\pi(x_{k+n})$, all $n$-step returns can be considered as the approximation to the full return.

*Definition 3 ($\lambda$-return):* The return $\mathcal{J}_k^\lambda(x_k)$ defined in Eq.(9) is called $\lambda$-return. It contains all the $n$-step returns, and each return weighted proportionally to $\lambda^{n-1}$(where $\lambda \in [0, 1]$) is normalized by a factor of $1 - \lambda$ to ensure that the weights sum up to 1.

$$\mathcal{J}_k^\lambda(x_k) = (1 - \lambda) \sum_{n=1}^{\infty} \lambda^{n-1} \mathcal{J}_{k:k+n}(x_k) \tag{9}$$

The $\lambda$-return is composed of all the $n$-step returns and gives each of them weighted to make sure the convergence of computing itself. It possesses an error reduction property and thus the error convergence can be guaranteed.

Using $\lambda$-return, the update rule of estimating $\mathcal{V}_\pi(x_k)$ is shown as $\mathcal{V}_\pi^{i+1}(x_k) = \mathcal{V}_\pi^i(x_k) + \alpha[\mathcal{J}_k^\lambda(x_k) - \mathcal{V}_\pi^i(x_k)]$, where $i$ is the iteration time and $\mathcal{V}_\pi^{i+1}(x_k)$ is the next iteration of $\mathcal{V}_\pi^i(x_k)$. After all, the value of $\mathcal{V}_\pi^i(x_k)$ converges to a finite estimated value $\mathcal{V}_\pi(x_k)$ iterated time by time.

## III. Adaptive Dynamic Programming with Multiple Time-scale Eligibility Traces

Inspired by the idea of HDP($\lambda$) method [13], we proposed an off-policy algorithm, named $T$-step Truncated HDP with Multiple Time-scale Eligibility Traces ($T$-HDP($\lambda$) with MET) for discrete-time dynamic systems. It can solve the optimal control problem faster and ensure stable convergence while using complex neural network structures without more computing resources else.

### A. T-step Truncated λ-return Method

The $\lambda$-return method is limited just to episodic tasks, where each episode eventually has a termination state. When solving the infinite-horizon optimal problems, the method can not be applied directly. In the continuing tasks, the $\lambda$-return is technically never known since it depends on $n$-step returns for arbitrarily large $n$ in the future.

However, as the time index $k$ gets larger, the dependence becomes weaker and weaker for longer-delayed rewards falling by $\gamma\lambda$. Truncating the sequence after enough number of steps is a natural approximation method. Here, we proposed the $T$-step Truncated $\lambda$-return Method to estimate the state-value function $\mathcal{V}_\pi(x_k)$. The convergence and boundedness of $T$-step Truncated $\lambda$-return in the optimal performance index are proved in the part of Convergence Analysis of the Algorithm.

$T$-step truncated $\lambda$-return is denoted as $\mathcal{J}^\lambda_{k:T}(x_k) = (1-\lambda)\sum_{n=1}^{T-k-1}\lambda^{n-1}\mathcal{J}_{k:k+n}(x_k) + \lambda^{T-k-1}\mathcal{J}_{k:T}(x_k)$, where $T$ is the truncated time during the whole control process. The last term $\lambda^{T-k-1}\mathcal{J}_{k:T}(x_k)$ is the approximation to replace the remaining real values.

### B. Multiple Time-scale Eligibility Traces

The overview of the traditional forward view algorithm is summarized as follows. $T$-step Truncated $\lambda$-return without eligibility traces is used to accumulate the future rewards to update the current $\mathcal{V}_\pi(x_k)$. Different from the forward view algorithm, a special incremental mechanism is defined as the backward view. The eligibility traces are used to assist in the whole learning process.

With function approximation, the standard eligibility traces are defined in Eq.(10) as a vector $\mathbf{z}_k \in \mathbb{R}^{m\times n}$ with the same number of components as the weight vector of neural network $\boldsymbol{\theta}_v \in \mathbb{R}^{m\times n}$.

$$\begin{aligned}\mathbf{z}_k &= \gamma\lambda\mathbf{z}_{k-1} + \mathbf{g}_k, \mathbf{z}_0 = 0\\ \mathbf{g}_k &= \nabla\mathcal{V}_\pi(x_k|\boldsymbol{\theta}_v)\end{aligned} \tag{10}$$

where $\gamma$ is the discount factor, and $\lambda$ is the trace-decay parameter mentioned before.

Different from the standard eligibility traces, the replacing eligibility traces [9] have been only for the tabular case or for binary feature vectors. It is defined on a component-by-component basis depending on whether the component of the feature vector was 1 or 0. The original replacing eligibility traces can be extended for the general approximation of value functions as follows:

$$\mathbf{z}_k = \begin{cases}\mathbf{g}_k, & \|\mathbf{g}_k\| > \|\mathbf{z}_{k-1}\|\\ \gamma\lambda\mathbf{z}_{k-1}, & otherwise\end{cases} \tag{11}$$

Research has shown that gradient divergence is one possible reason why the eligibility traces fail in deep neural networks. Here, we designed an adaptively decaying eligibility traces to better accumulate the information of gradients.

Define $\mathbf{z}^i_k (i = 1,...,K)$ with $K > 1$ as multiple time-scale eligibility traces(MET). They are with a K-series-layered structure, and the maximum decaying factor of $i^{th}$ layer, $\lambda^i_{max}$, is given.

$$\mathbf{z}^i_k = \begin{cases}\mathbf{z}^{i-1}_k, & i \neq 1 \ and \ \Delta\mathbf{z}^i\mathbf{z}^{i-1}_k > 0\\ \gamma\lambda^i_{max}\mathbf{z}^i_{k-1} + \beta^i\mathbf{g}_k, & otherwise\end{cases} \tag{12}$$

where $\Delta\mathbf{z}^i = \mathbf{z}^{i-1}_k - \mathbf{z}^i_{k-1}$, and $\beta^i$ is the sequence with three conditions [22]: (1) $\beta^i \leq \beta^{i+1}$, (2) $\sum_{i=1}^K \beta^i = 1$, and (3) $\beta^K = 0$. Here, we choose $\beta^i = \frac{2(K-i)}{K(K-1)}$ and $K = 2$.

The MET keeps tracking which components of the weight vector have contributed, maybe positively or negatively, to the recent state-value function $\mathcal{V}_\pi(x_k)$, where "recent" is represented in terms of $\gamma\lambda$.

### C. The novel proposed Algorithm: T-HDP($\lambda$) with MET

In this part, the novel Algorithm 1, $T$-HDP($\lambda$) with MET is proposed to transfer the infinite terms of cost function into finite terms. Moreover, the Multiple Time-scale Eligibility Traces are utilized to approximate the cost function and guides the control process faster in the optimal direction without increasing the computational burden.

The basic architecture of $T$-HDP($\lambda$) with MET is similar to the actor-critic framework. The forward view of Algorithm 1 is shown in Fig.1 to better show the inner principle, though in actual calculations the backward view based on MET is utilized to simplify the calculation. In addition to the traditional actor and critic networks, target networks are used here to improve the stability of learning $\mathcal{V}_\pi(x_k)$.

The neural networks are used to approximate the state-value function $\mathcal{V}_\pi(x_k)$ and the control signal $u_k$ under the policy $\pi$. The actor network and critic network are adopted to approximate them through optimizing their parameters $\boldsymbol{\theta}_v$ and $\boldsymbol{\theta}_a$ respectively.

The critic network and target critic network is a three-layer neural network with one hidden layer. The inputs of them are both the state $x_k$. The output is the estimated state-value function $\mathcal{V}_\pi(x_k|\boldsymbol{\theta}^i_v)$, where $\boldsymbol{\theta}^i_v$ is the weight of critic neural network. Similarly, $\mathcal{V}_\pi(x_k|\boldsymbol{\theta}^i_{v'})$ is output of the target critic network, where $\boldsymbol{\theta}^i_{hv'}$ is the weight of target critic neural network.

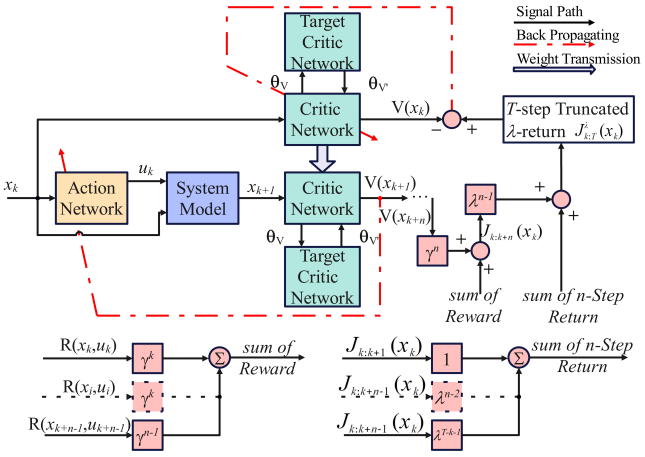


Fig. 1. The neural networks structure of $T$-HDP($\lambda$) with MET

$T$-HDP($\lambda$) with MET is an off-policy algorithm, so the target policy $\pi(x_k)$ is different from the behavior policy $b(x_k)$. The behavior policy $b(x_k)$ is fixed during the whole process to generate the sampled sequence $\{(x_k, u_k, r_k, x_{k+1})\}^T_{k=1}$, and the target policy $\pi(x_k)$ is our final goal which converges to optimal policy.

In order to use episodes from $b(x_k)$ to estimate values for $\pi(x_k)$, we require that every action taken under $\pi(x_k)$

is also taken under $b(x_k)$ at least occasionally. Here, a new importance sampling ratio $\rho_{t:T}^{\lambda} = \prod_{i=k+1}^{T} \frac{Pr\{u_k^a|x_k\}}{Pr\{u_k^b|x_k\}}$ where $u_k^a$ and $u_k^b$ denote the control signal under target policy $\pi(x_k)$ and behavior policy $b(x_k)$ during the state $x_k$ respectively, is designed to transform the returns to have the right expected value and better match the proposed algorithm.

Hence, the MET with importance sampling ratio $\rho_{t:T}^{\lambda}$ can be denoted as

$$\mathbf{z}_k^i = \begin{cases} \mathbf{z}_k^{i-1}, & i \neq 1 \ and \ \Delta\mathbf{z}^i \mathbf{z}_k^{i-1} > 0 \\ \rho_{t:T}^{\lambda}(\gamma\lambda_{max}^i \mathbf{z}_{k-1}^i + \beta^i \mathbf{g}_k), & otherwise \end{cases} \tag{13}$$

where $\Delta\mathbf{z}^i = \mathbf{z}_k^{i-1} - \mathbf{z}_{k-1}^i$, and $\beta^i = \frac{2(K-i)}{K(K-1)}$.

The loss function of critic network is constructed as

$$\delta_v^i(x_k) = \mathcal{R}(x_k, u_k) + \gamma\mathcal{V}_\pi(x_{k+1}|\boldsymbol{\theta}_v^{i-1}) - \mathcal{V}_\pi(x_k|\boldsymbol{\theta}_v^{i-1}) \tag{14}$$

Employing the gradient method, the critic network and target critic network can be updated at iteration time $i$ by eligibility traces $\mathbf{z}_k^i$: $\boldsymbol{\theta}_v^i = \boldsymbol{\theta}_v^{i-1} - \alpha_1 \delta_v^i(x_k)\mathbf{z}_k^i$, $\boldsymbol{\theta}_{v'}^i = (1-\tau)\boldsymbol{\theta}_{v'}^{i-1} + \tau\boldsymbol{\theta}_v^i$, where $0 < \tau < 1$ is the updating rate.

The actor network is approximated by a three-layer neural network with one hidden layer as well. The input is the state $x_k$ and the output is the control signal $u_k$ under the policy $\pi$. The control signal at iteration time $i-1$ can be represented as $u_k^{i-1} = \pi(x_k|\boldsymbol{\theta}_a^{i-1})$, where $\boldsymbol{\theta}_a^i$ is the weight of actor neural network.

When the system can be expressed as an affine nonlinear form of Eq.(1), the target optimal control law can be expressed as follows by solving Bellman's optimality equation $\hat{u}_k^{i-1} = -\frac{\gamma}{2}R^{-1}g^T(x_k)\frac{\partial\mathcal{V}_\pi(x_{k+1}|\boldsymbol{\theta}_v^{i-1})}{\partial x_{k+1}}$. The actor network can be updated by the error of critic network at iteration time $i$ by: $E_a^i(x_k) = \left\|\hat{u}_k^{i-1} - u_k^{i-1}\right\|^2$, $\boldsymbol{\theta}_a^i = \boldsymbol{\theta}_a^{i-1} - \alpha_2\nabla_{\theta_a}E_a^i(x_k)$.

**Algorithm 1** The Implementation of $T$-HDP($\lambda$) with MET

**Input:** An initial state $x_0$, target policy $\pi_0(x_k)$ and behavior policy $b(x_k)$, truncated step $T$, initial state-value function $\mathcal{V}_{\pi_0}^0(x_k)$, iteration time $i = 0$ and the given threshold $\epsilon$.

**Output:** Optimal control policy $\pi^*(x_k)$.

1: Initial network parameters $\boldsymbol{\theta}_v^i$ and $\boldsymbol{\theta}_{v'}^i$, $\boldsymbol{\theta}_a^i$, and the learning rate $\alpha_1$, $\alpha_2$.
2: Update the parameters of critic network by $\boldsymbol{\theta}_v^i \leftarrow \boldsymbol{\theta}_{v'}^i$.
3: Get action $u_k$ under target policy $\pi_i(x_k)$ , reward $r_k$ and next new state $x_{k+1}$ until $k = T$.
4: Store the sampled sequence $\{(x_k, u_k, r_k, x_{k+1})\}_{k=1}^T$ under behavior policy $b(x_k)$.
5: Compute $T$-step truncated $\lambda$-return $\mathcal{J}_{k:T}^{\lambda}(x_k)$.
6: Compute eligibility traces $\mathbf{z}_k^i$ by Eq.(13).
7: Update the weights of the critic network and actor network respectively.
8: **if** $\left\|\mathcal{V}_{\pi_i}^{i+1}(x_k) - \mathcal{V}_{\pi_i}^i(x_k)\right\| \leq \epsilon$ **then**
9: Stop iteration and output the optimal control policy $\pi^*(x_k)$.
10: **else**
11: Let $i \leftarrow i+1$, and go to Step 2.
12: **end if**

A random admissible policy is adopted as an initialized policy. Sample the controlled system through the initial policy and store sampled sequence $\{(x_k, u_k, r_k, x_{k+1})\}_{k=1}^T$ under behavior policy $b(x_k)$ to calculate the $T$-step truncated $\lambda$-return. Update the weights of the critic network and actor network to obtain the optimal control policy.

### D. Convergence Analysis of the Algorithm

In this section, the convergence of the sequence $\mathcal{V}_\pi^i(x_k)$ generated by Algorithm 1 ($T$-HDP($\lambda$) with MET) is proved.

*Theorem 1:* Let the estimated state-value function sequence $\{\mathcal{V}_\pi^i(x_k)\}_{i=0}^{\infty}$ be obtained by Algorithm 1.

If the following condition holds:

$$\mathcal{V}_\pi^0(x_k) \geqslant \min_{u_k}\left(\mathcal{R}(x_k, u_k) + \gamma\mathcal{V}_\pi^0(x_{k+1})\right) \tag{15}$$

then, 1) for arbitrary $i$

$$\mathcal{V}_\pi^{i+1}(x_k) \leqslant \min_{u_k}\left(\mathcal{R}(x_k, u_k) + \gamma\mathcal{V}_\pi^i(x_{k+1})\right) \leqslant \mathcal{V}_\pi^i(x_k) \tag{16}$$

2) $\lim_{i\to\infty}\mathcal{V}_\pi^i(x_k) = \mathcal{J}(x_k, u_k^*) = \mathcal{J}^*(x_k)$

*Proof:* Prove the inequation (16) by using mathematical induction method.

$$\begin{aligned}
&\mathcal{V}_\pi^1(x_k) = (1-\lambda)\sum_{n=1}^{T-k-1}\lambda^{n-1}\mathcal{J}_{k:k+n}^0 + \lambda^{T-k-1}\mathcal{J}_{k:T}^0 \\
&= \sum_{n=1}^{T-k-1}\lambda^{n-1}\mathcal{J}_{k:k+n}^0 - \sum_{n=1}^{T-k-1}\lambda^n\mathcal{J}_{k:k+n}^0 + \lambda^{T-k-1}\mathcal{J}_{k:T}^0 \\
&= \sum_{n=0}^{T-k-2}\lambda^n\mathcal{J}_{k:k+n+1}^0 - \sum_{n=1}^{T-k-1}\lambda^n\mathcal{J}_{k:k+n}^0 + \lambda^{T-k-1}\mathcal{J}_{k:T}^0 \\
&= \mathcal{J}_{k:k+1}^0 + \sum_{n=1}^{T-k-2}\lambda^n\mathcal{J}_{k:k+n+1}^0 - \lambda^{T-k-1}\mathcal{J}_{k:T-1}^0 \\
&\quad - \sum_{n=1}^{T-k-2}\lambda^n\mathcal{J}_{k:k+n}^0 + \lambda^{T-k-1}\mathcal{J}_{k:T}^0 \\
&= \mathcal{R}(x_k, \hat{u}_k^0) + \gamma\mathcal{V}_\pi^0(x_{k+1}) + \sum_{n=1}^{T-k-1}\lambda^n(\mathcal{J}_{k:k+n+1}^0 - \mathcal{J}_{k:k+n}^0) \\
&= \mathcal{R}(x_k, \hat{u}_k^0) + \gamma\mathcal{V}_\pi^0(x_{k+1}) + \sum_{n=1}^{T-k-1}(\lambda\gamma)^n\delta_n^0(x_{k+n})
\end{aligned} \tag{17}$$

$$\delta_n^0(x_{k+n}) = \mathcal{R}(x_{k+n}, \hat{u}_{k+n}^0) + \gamma\mathcal{V}_\pi^0(x_{k+n+1}) - \mathcal{V}_\pi^0(x_{k+n}) \tag{18}$$

According to condition(15), one has

$$\begin{aligned}
\delta_n^0(x_{k+n}) &= \mathcal{R}(x_{k+n}, \hat{u}_{k+n}^0) + \gamma\mathcal{V}_\pi^0(x_{k+n+1}) - \mathcal{V}_\pi^0(x_{k+n}) \\
&= \min_{u_{k+n}}\left(\mathcal{R}(x_{k+n}, u_{k+n}) + \gamma\mathcal{V}_\pi^0(x_{k+n+1}\right) - \mathcal{V}_\pi^0(x_{k+n}) \\
&\leqslant \mathcal{V}_\pi^0(x_{k+n}) - \mathcal{V}_\pi^0(x_{k+n}) = 0
\end{aligned} \tag{19}$$

which means that $\delta_n^0(x_{k+n})$ is non-positive. Therefore,

$$\begin{aligned}
\mathcal{V}_\pi^1(x_k) &= \mathcal{R}(x_k, \hat{u}_k^0) + \gamma\mathcal{V}_\pi^0(x_{k+1}) + \sum_{n=1}^{T-k-1}(\lambda\gamma)^n\delta_n^0(x_{k+n}) \\
&= \min_{u_k}\left(\mathcal{R}(x_k, u_k) + \gamma\mathcal{V}_\pi^0(x_{k+1})\right) \leqslant \mathcal{V}_\pi^0(x_k)
\end{aligned} \tag{20}$$

Thus, the inequation (16) holds on for $i = 0$. Now assume that the inequation (16) is true for index $i-1$ , then it suffices to prove that inequation (16) is true for index $i$. To this end, we need first to prove that

$$\mathcal{J}_{k:k+n}^{i-1}(x_k) \geqslant \mathcal{R}(x_{k+n}, \hat{u}_{k+n}^{i-1}) + \gamma\mathcal{J}_{k:k+n}^{i-1}(x_{k+1}) \tag{21}$$

holds for $n = 1, 2, \ldots, T$.

In fact, according to the assumption by inequation (16), for all $n = 1, 2, \ldots, T$, we get

$$\begin{aligned}\mathcal{J}_{k:k+n}^{i-1}(x_k) &= \sum_{s=0}^{n-1} \gamma^s \mathcal{R}(x_{k+s}, \hat{u}_{k+s}^{i-1}) + \gamma^n \mathcal{V}_\pi^{i-1}(x_{k+n}) \\ &\geqslant \gamma^n \min_{u_{k+n}} \left( \mathcal{R}(x_{k+n}, u_{k+n}) + \gamma \mathcal{V}_\pi^{i-1}(x_{k+n+1}) \right) \\ &\quad + \sum_{s=0}^{n-1} \gamma^s \mathcal{R}(x_{k+s}, \hat{u}_{k+s}^{i-1}) \\ &= \sum_{s=0}^{n} \gamma^s \mathcal{R}(x_{k+s}, \hat{u}_{k+s}^{i-1}) + \gamma \mathcal{V}_\pi^{i-1}(x_{k+n+1}) \\ &= \mathcal{R}(x_k, u_k^*) + \gamma \mathcal{J}_{k:k+n}^{i-1}(x_{k+1})\end{aligned} \tag{22}$$

Next, we have

$$\begin{aligned}&\mathcal{V}_\pi^i(x_k) = (1-\lambda) \sum_{n=1}^{T-k-1} \lambda^{n-1} \mathcal{J}_{k:k+n}^{i-1}(x_k) + \lambda^{T-1} \mathcal{J}_{k:T}^{i-1}(x_k) \\ &\geqslant \gamma(1-\lambda) \sum_{n=1}^{T-k-1} \lambda^{n-1} \mathcal{J}_{k:k+n}^{i-1}(x_k) \\ &\quad + \mathcal{R}(x_k, \hat{u}_k^{i-1}) + \gamma \lambda^{T-1} \mathcal{J}_{k:T}^{i-1}(x_{k+1}) \\ &= \mathcal{R}(x_k, \hat{u}_k^{i-1}) + \gamma \mathcal{V}_\pi^i(x_{k+1}) = \min_{u_k} \left( \mathcal{R}(x_k, u_k) + \gamma \mathcal{V}_\pi^i(x_{k+1}) \right)\end{aligned} \tag{23}$$

Using the similar approach with Eq.(19), we get

$$\begin{aligned}&\mathcal{V}_\pi^{i+1}(x_k) = (1-\lambda) \sum_{n=1}^{T-k-1} \lambda^{n-1} \mathcal{J}_{k:k+n}^{i}(x_k) + \lambda^{T-1} \mathcal{J}_{k:T}^{i}(x_k) \\ &= \sum_{n=1}^{T-k-1} (\gamma\lambda)^n \left( \mathcal{R}(x_{k+n}, \hat{u}_{k+n}^i) + \gamma \mathcal{V}_\pi^i(x_{k+n+1}) - \mathcal{V}_\pi^i(x_{k+n}) \right) \\ &\quad + \mathcal{R}(x_k, \hat{u}_k^i) + \gamma \mathcal{V}_\pi^i(x_{k+1}) \\ &= \mathcal{R}(x_k, \hat{u}_k^i) + \gamma \mathcal{V}_\pi^i(x_{k+1}) + \sum_{n=1}^{T-1} (\gamma\lambda)^n \delta_n^i(x_{k+n}) \\ &\leqslant \min_{u_k} \left( \mathcal{R}(x_k, u_k) + \gamma \mathcal{V}_\pi^i(x_{k+1}) \right) \leqslant \mathcal{V}_\pi^i(x_k)\end{aligned} \tag{24}$$

This proves that the conclusion (16) is true for index $i$. Thus, the conclusion (16) is valid.

The previous part of this theorem shows that $\{\mathcal{V}_\pi^i(x_k)\}_{i=0}^\infty$ is a non-increasing and non-negative sequence. As a bounded monotone sequence, considering that $\{\mathcal{V}_\pi^i(x_k)\}_{i=0}^\infty$ must have a limit denoted by $\mathcal{V}_\pi^\infty(x_k, u_k) = \lim_{i\to\infty} \mathcal{V}_\pi^i(x_k)$. Take limit on both sides of the inequation (16) with respect to $i$, and we get $\mathcal{V}_\pi^\infty(x_k) \leqslant \min_{u_k} \left( \mathcal{R}(x_k, u_k) + \gamma \mathcal{V}_\pi^\infty(x_{k+1}) \right) \leqslant \mathcal{V}_\pi^\infty(x_k)$, which implies $\mathcal{V}_\pi^\infty(x_k) = \min_{u_k} \left( \mathcal{R}(x_k, u_k) + \gamma \mathcal{V}_\pi^\infty(x_{k+1}) \right)$.

It is obvious that the above equation is essentially the Bellman equation (5) , that is $\lim_{i\to\infty} \mathcal{V}_\pi^i(x_k) = \mathcal{J}(x_k, u_k^*) = \mathcal{J}^*(x_k)$. This completes the proof of the convergence of Algorithm 1. ■

## IV. Simulation Results

In this section, one simulation example is utilized to show the effectiveness of the proposed Algorithm 1. The nonlinear discrete-time system is considered as follows:

$$x_{k+1} = \begin{bmatrix} -x_1(k) + x_2(k) + 2x_2^3(k) \\ -0.5(x_1(k) + x_2(k)) \end{bmatrix} + \begin{bmatrix} 1 \\ sin(x_1(k)) \end{bmatrix} u_k \tag{25}$$

where $x_{k+1} \in R^2$ and $u_k \in R$. The utility function is expressed as $U(x_k, u_k) = x_k^T Q x_k + u_k^T R u_k$, where $Q = I$, $R = 5I$ and $I$ is the identity matrix with suitable dimensions.

A three-layer neural network with the structure of $2-8-1$ is used to bulid up the actor network. And the critic network is with the structure of $2-12-1$. The truncated step is $T = 9$ and $\lambda = 0.9$. Train the networks for 450 episodes.

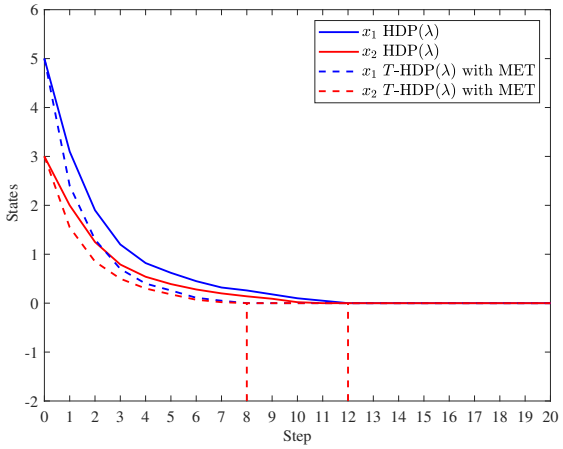


Fig. 2. The states of HDP($\lambda$) and $T$-HDP($\lambda$) with MET ($T = 9$ and $\lambda = 0.9$)

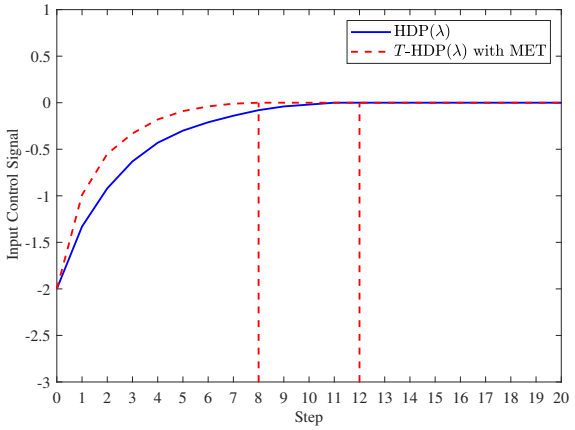


Fig. 3. The input control signal of HDP($\lambda$) and $T$-HDP($\lambda$) with MET ($T = 9$ and $\lambda = 0.9$)

From Fig.2-3, it is more remarkably shown that our proposed algorithm converges much faster compared with HDP($\lambda$) algorithm [13]. The proposed algorithm only uses 9 steps to achieve the optimal control rather than 12 steps. The sum of reward in $T$-HDP$\lambda$) with MET converges faster and has better training stability than the original HDP($\lambda$) algorithm in Fig.4.

In order to show that MET has the same effect as replay buffer to solve the problem of data scarceness, the reward of each step between HDP with different capacities of replay buffer ($N$=100 and $N$=200) and with MET method is shown in Fig.5. As the capacity of the replay buffer increases from 100 to 200, the converge speed is faster in traditional HDP. $T$-HDP($\lambda$) with MET method proposed in this paper has a higher convergence speed and quicker time than HDP($\lambda$) without using the replay buffer. The conclusion is that MET plays a similar role as the replay buffer and our method has better effectiveness to expand the data and speed up the control process.

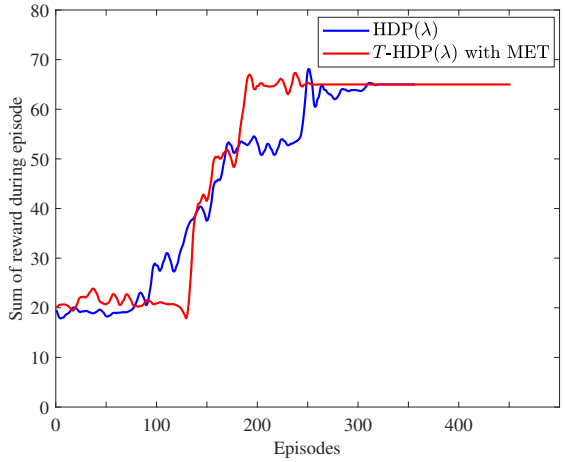


Fig. 4. The sum of reward between HDP($\lambda$) and $T$-HDP($\lambda$) with MET

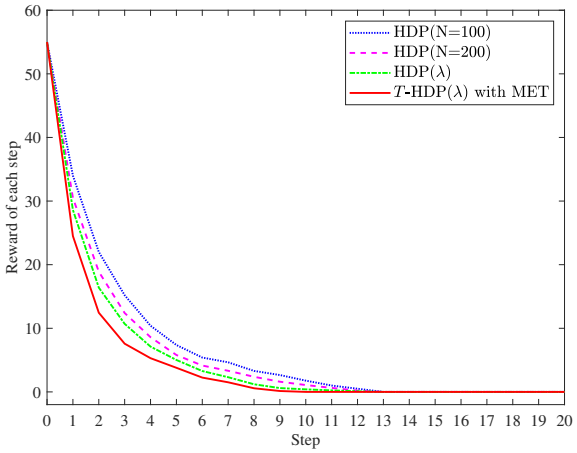


Fig. 5. Reward of each step between HDP with the capacity of replay buffer ($N$ = 100 and $N$ = 200), HDP($\lambda$) and $T$-HDP($\lambda$) with MET

## V. Conclusion

To solve the optimal control problem of the discrete-time nonlinear systems, a novel model-based $T$-HDP($\lambda$) with MET algorithm has been proposed. Different from the traditional accumulating eligibility traces, the multiple time-scale eligibility traces can be adaptively adjusted to avoid the problem that the approximation using deep neural networks may not converge. Moreover, it can make full use of the limited data and speed up the control process. The $T$-step truncated method can address the infinite-horizon optimal problems and has the same convergence property. Since $T$-HDP($\lambda$) with MET is an off-policy algorithm, a new importance sampling ratio $\rho_{t:T}^{\lambda}$ is designed to correct the right expected state-value function between the target policy $\pi(x_k)$ and behavior policy $b(x_k)$. The simulation has shown that the proposed algorithm converges faster than the original algorithm, and the MET plays a significant role to expand the data.

For the proposed $T$-HDP($\lambda$) with MET algorithm, more prospects for future research could be adding the event-triggered scheme, input constraints and so on.

## References


[1] D. Liu, S. Xue, B. Zhao, B. Luo, and Q. Wei, "Adaptive dynamic programming for control: A survey and recent advances," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 1, pp. 142–160, 2020.

[2] 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, 2009.

[3] H. Zhang, Q. Wei, and Y. Luo, "A novel infinite-time optimal tracking control scheme for a class of discrete-time nonlinear systems via the greedy HDP iteration algorithm," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 38, no. 4, pp. 937–942, 2008.

[4] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. USA: MIT press, 2018.

[5] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 26–38, 2017.

[6] R. Cui, C. Yang, Y. Li, and S. Sharma, "Adaptive neural network control of AUVs with control input nonlinearities using reinforcement learning," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 6, pp. 1019–1029, 2017.

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

[8] Q. Wei, R. Song, Z. Liao, B. Li, and F. L. Lewis, "Discrete-time impulsive adaptive dynamic programming," *IEEE Transactions on Cybernetics*, vol. 50, no. 10, pp. 4293–4306, 2019.

[9] S. Liu, B. Niu, G. Zong, X. Zhao, and N. Xu, "Data-driven-based event-triggered optimal control of unknown nonlinear systems with input constraints," *Nonlinear Dynamics*, pp. 1–19, 2022.

[10] M. Bayer, M.-A. Kaufhold, and C. Reuter, "A survey on data augmentation for text classification," *ACM Computing Surveys*, 2021.

[11] J. Nalepa, M. Marcinkiewicz, and M. Kawulok, "Data augmentation for brain-tumor segmentation: a review," *Frontiers in computational neuroscience*, vol. 13, p. 83, 2019.

[12] A. Antoniou, A. Storkey, and H. Edwards, "Data augmentation generative adversarial networks," *arXiv preprint arXiv:1711.04340*, 2017.

[13] T. Li, D. Zhao, and J. Yi, "Heuristic dynamic programming strategy with eligibility traces," in *2008 American Control Conference*. IEEE, 2008, pp. 4535–4540.

[14] H. van Hasselt, S. Madjiheurem, M. Hessel, D. Silver, A. Barreto, and D. Borsa, "Expected eligibility traces," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 11, 2021, pp. 9997–10 005.

[15] W. Bi, L. Xuelian, G. Zhiqiang, and C. Yang, "Gradient compensation traces based temporal difference learning," *Neurocomputing*, vol. 442, pp. 221–235, 2021.

[16] J. Ye, Y. Bian, B. Xu, Z. Qin, and M. Hu, "Online optimal control of discrete-time systems based on globalized dual heuristic programming with eligibility traces," in *2021 3rd International Conference on Industrial Artificial Intelligence (IAI)*. IEEE, 2021, pp. 1–6.

[17] E. M. S. Duque, J. S. Giraldo, P. P. Vergara, P. Nguyen, A. van der Molen, and H. Slootweg, "Community energy storage operation via reinforcement learning with eligibility traces," *Electric Power Systems Research*, vol. 212, p. 108515, 2022.

[18] J. Yuan, J. Wan, X. Zhang, Y. Xu, Y. Zeng, and Y. Ren, "A second-order dynamic and static ship path planning model based on reinforcement learning and heuristic search algorithms," *EURASIP Journal on Wireless Communications and Networking*, vol. 2022, no. 1, pp. 1–29, 2022.

[19] P. Padrao, A. Dominguez, L. Bobadilla, and R. N. Smith, "Towards learning ocean models for long-term navigation in dynamic environments," in *OCEANS 2022-Chennai*. IEEE, 2022, pp. 1–6.

[20] Q. Wei, D. Liu, and X. Yang, "Infinite horizon self-learning optimal control of nonaffine discrete-time nonlinear systems," *IEEE transactions on neural networks and learning systems*, vol. 26, no. 4, pp. 866–879, 2015.

[21] B. Luo, D. Liu, T. Huang, X. Yang, and H. Ma, "Multi-step heuristic dynamic programming for optimal control of nonlinear discrete-time systems," *Information Sciences*, vol. 411, pp. 66–83, 2017.

[22] T. Kobayashi, "Adaptive and multiple time-scale eligibility traces for online deep reinforcement learning," *Robotics and Autonomous Systems*, vol. 151, p. 104019, 2022.