Soft Comput (2013) 17:2089–2099
DOI 10.1007/s00500-013-1110-y

METHODOLOGIES AND APPLICATION

A supervised Actor–Critic approach for adaptive cruise control

Dongbin Zhao · Bin Wang · Derong Liu

Published online: 22 August 2013

Abstract A novel supervised Actor–Critic (SAC) approach for adaptive cruise control (ACC) problem is proposed in this paper. The key elements required by the SAC algorithm namely Actor and Critic, are approximated by feed-forward neural networks respectively. The output of Actor and the state are input to Critic to approximate the performance index function. A Lyapunov stability analysis approach has been presented to prove the uniformly ultimate bounded property of the estimation errors of the neural networks. Moreover, we use the supervisory controller to pre-train Actor to achieve a basic control policy, which can improve the training convergence and success rate. We apply this method to learn an approximate optimal control policy for the ACC problem. Experimental results in several driving scenarios demonstrate that the SAC algorithm performs well, so it is feasible and effective for the ACC problem.

Keywords Supervised reinforcement learning · Actor–Critic · Adaptive cruise control · Uniformly ultimate bounded · Neural networks

Communicated by C. Alippi, D. Zhao and D. Liu.

This work was supported in part by National Natural Science Foundation of China under Grant Nos. 61273136 and 61034002, and Beijing Natural Science Foundation under Grant No. 4122083.

D. Zhao · B. Wang (✉) · D. Liu
The State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China
e-mail: bin.wang@ia.ac.cn

D. Zhao
e-mail: dongbin.zhao@ia.ac.cn

D. Liu
e-mail: derong.liu@ia.ac.cn

1 Introduction

To improve driving safety and comfort, advanced driver assistant systems have been proposed into practice in recent years, such as Anti-lock Braking Systems (ABS), Electronic Braking Systems (EBS), Traction Control Systems (TCS), and so on (Siciliano and Khatib 2008). After decades' development, adaptive cruise control (ACC) has been fully researched to improve driving safety, enhance driving comfort, reduce fuel consumption and increase traffic capacity (Andreas 2012). ACC system is an advanced driver assistance system (ADAS), which is developed from traditional cruise control (CC) system. In the ACC system, a range sensor such as radar is equipped to measure the distance and the relative speed to the preceding vehicle. The controller can automatically adjust the throttle and/or the brake to control speed or time headway of the vehicle by calculating appropriate throttle angle and brake pressure. It can be used in full speed range as with the function of stop-and-go (SG) (Kyongsu and Ilki 2004; Martinez and Canudas-De-Wit 2007; Naranjo et al. 2006), collision warning and collision avoidance (CW/CA) (Moon et al. 2009; Kesting et al. 2007). Nowadays, ACC systems are equipped in some luxury vehicles to improve both safety and comfort.

Because of the uncertainty of vehicle dynamics and the disturbance from the environment, ACC can be viewed as an optimal tracking control problem of the complex nonlinear system. Plenty of control theory and techniques have been implemented to design ACC controllers, of which are PID controllers (Guvenc and Kural 2006), model predictive controllers (Kural and Guvenc 2001; Li et al. 2011), sliding mode controllers (Fritz and Schiehlen 2001; Xiao and Gao 2011), fuzzy inference approaches (Naranjo et al. 2006; Milanes et al. 2012), neural networks methods (Ohno 2001; Bifulco et al. 2008) and so on. However, some methods based on

linear model of vehicles or other algebraic analytical solutions cannot deal with emergency-braking situations well, and some ACC controllers may not act adaptively for different drivers.

On the other hand, reinforcement learning (RL) theory has been fully researched for several decades. RL gets reward from the environment via interacting with it, and approximates an optimal policy through trial and error (Sutton and Barto 1998). Typical RL algorithms are Q-learning, SARSA, Actor–Critic and the Adaptive Dynamic Programming (ADP) family (Murray et al. 2002). Because of the self-adaptation ability of RL, numerous optimal control problems have been solved with this approach, some of which can be found in Si and Wang (2001) and Zhao et al. (2012b) for the under-actuated systems, the Go-Moku game problem (Zhao et al. 2012a), the freeway ramp metering problem (Zhao et al. 2011), and the urban intersection traffic signal control problem (Li et al. 2008), etc.. Nevertheless, RL algorithms usually need a long training procedure to learn the optimal policy and converge slowly, thus its inefficiency may cause trouble in some real time control requirements.

Supervised Reinforcement Learning (SRL) can be an alternative to compensate for the deficiency of RL by integrating respective advantages of Supervised Learning (SL) and RL. SRL algorithm has been successfully implemented to control the manipulator, solve the ship steering task and the peg insertion task (Barto and Dietterich 2004; Rosenstein and Barto 2004), while the weighting factor between the supervisor and Actor is a key element affecting the convergence significantly which is hard to choose. In our previous work, we presented the SRL approach with shaping rewards to solve ACC problem (Hu and Zhao 2011), which showed satisfactory performance for the speed and distance control. We also introduced a Supervised ADP (SADP) method to achieve the full range ACC in order to suit various driving scenarios (Zhao et al. 2013). The main limitation of Hu and Zhao (2011) and Zhao et al. (2013) is the incomplete sampling data for training, which is important for the adaptability of the controller.

Based on the aforementioned work, we propose a novel supervised Actor–Critic (SAC) approach for ACC problem in this paper. Different from the work of Rosenstein and Barto (2004), in which they combined Actor and the supervisor to form a "composite" actor that sent a composite action to the environment, and Time Difference (TD) error was used to update the parameters of Actor and Critic, we implement an action-dependent critic network of which the action value is an additional input. We first train Actor via the supervisor, and an exploration noise is added to the action for better performance. Furthermore, the parameters of Actor are updated by the error between the value function and the ultimate objective. It is proved in detail that the estimation errors of the neural networks for Actor and Critic are uniformly ultimate bounded (UUB) by the Lyapunov approach, which guarantees the stability of the proposed SAC algorithm. Moreover, we use the supervisory controller to pre-train Actor and get a basic control policy, which can improve the training convergence and success rate. We apply this method to learn an approximately optimal control policy for ACC problem.

The paper is organized as follows. In Sect. 2 we describe the ACC problem. Section 3 proposes the novel SAC algorithm and presents the detailed implementation in ACC. Detailed stability analysis can be seen in Sect. 4. Section 5 provides simulation examples. In Sect. 6 we get the conclusion.

2 Adaptive cruise control

In Fig. 1 we show the working principle of ACC. During driving, the following vehicle runs after the preceding vehicle under the control of the ACC system to keep a desired distance d_d and the safe speed v_f. Radar or other sensors detect the distance d_r which can be used to compute the speed of the preceding vehicle v_p. According to the instant information including v_f, v_p, d_r and d_d, the ACC controller outputs a corresponding action to control the throttle and/or the brake. In this paper we adopt a hierarchical structure to design the ACC controller which consists of the upper level and the lower level, shown in Fig. 2.

The upper level controller integrates the instant information to generate the acceleration control signal, whereas the lower level controller receives the acceleration signal and transfers it to corresponding brake and/or throttle control actions according to the current acceleration of the preceding vehicle. Our main interest in this paper is the design of the upper level controller. The implementation of the lower level controller can be achieved in our previous work (Xia and Zhao 2012), where we presented a throttle controller based on PID method and a brake controller via hybrid feed-forward & feedback control approach.

Define the relative speed Δv as

$$\Delta v = v_f - v_p \tag{1}$$

and the relative distance Δd is defined as

$$\Delta d = d_r - d_d \tag{2}$$

In the upper level, the controller tends to drive $(\Delta v, \Delta d)$ to zero simultaneously through the appropriate acceleration action. It should be noted that the driving preference and habits vary with drivers. The adaptability to different drivers should be considered when designing the ACC controller. It has been revealed that the driving behavior can be represented by a combination of a constant time gap and a constant distance (Kyongsu and Ilki 2004). The desired distance d_d in Fig. 1 is defined as

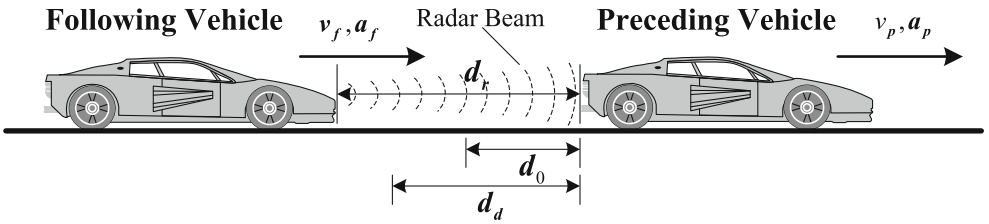

Fig. 1 The working principle of ACC

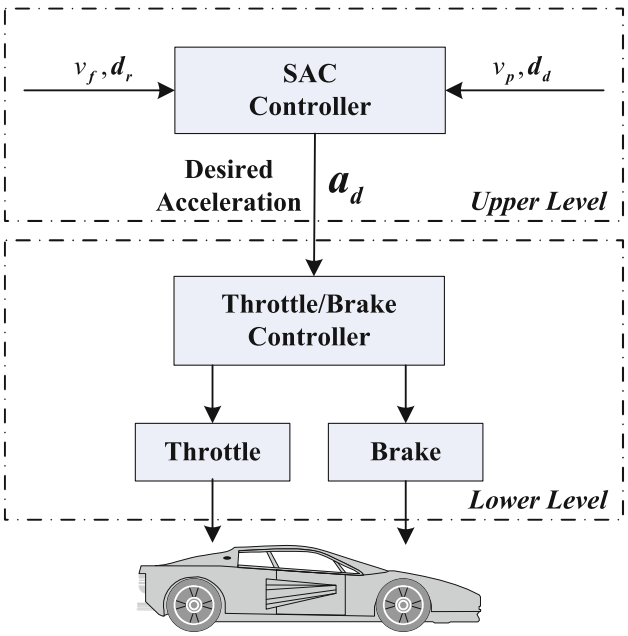

Fig. 2 The hierarchical control framework for ACC

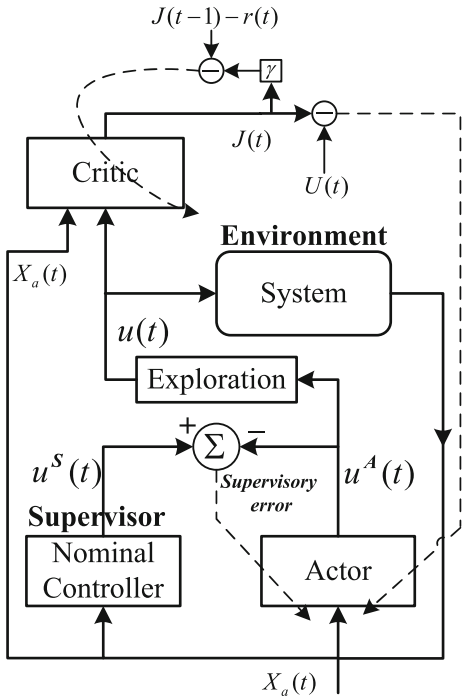

Fig. 3 The schematic of the SAC framework

$$d_d = d_0 + v_p \cdot \tau \tag{3}$$

where τ is the time gap, and d_0 is the minimum distance to be kept when the vehicle stops. According to (3), driving behavior can be classified by analyzing human driver data.

In the next section we will propose the upper level controller implemented by the SAC approach.

3 The SAC control strategy

The structure of the SAC approach is shown in Fig. 3. There are four main parts of the SAC framework, which are Actor, Supervisor, Critic and System, respectively. Actor outputs the control action. As Supervisor, we adopt a nominal controller to train Actor in order to get an initial admissible control policy. System (the *environment*) responds to the control action with a transition from the current state to the next state. Critic criticizes the control actions made by Actor. Details about the SAC algorithm will be presented as follows.

3.1 System

System responds to the control action and transits the current state $x(t)$ to the next state $x(t+1)$. System also provides an evaluation called the immediate reward, $r(t)$.

For the ACC problem, we define the state as the relative speed $\Delta v(t)$ and the relative distance $\Delta d(t)$, namely $x(t) = [\Delta v(t), \Delta d(t)]$. And the acceleration of the following vehicle $a_f(t)$ is defined as the control variable, namely $u(t) = a_f(t)$.

Take action $u(t) = a_f(t)$ under the current state $x(t) = [\Delta v(t), \Delta d(t)]$, the state transition is defined as

$$\begin{cases} v_f(t+1) = v_f(t) + a_f(t) \cdot \Delta t \\ d_f(\Delta t) = v_f(t) \cdot \Delta t + \frac{1}{2} \cdot \Delta t^2 \\ d_p(\Delta t) = (v_p(t) + v_p(t+1)) \cdot /2 \\ \Delta v(t+1) = v_f(t+1) - v_p(t+1) \\ \Delta d(t+1) = \Delta d(t) - (d_f(\Delta t) - d_p(\Delta t)) \end{cases} \tag{4}$$

where Δt is the sampling time. Note that if we want to compute the next state $x(t+1)$, the preceding vehicle speed of the next step $v_p(t+1)$ or its acceleration must be known.

3.2 Actor

The input of Actor is the system state $x(t)$, thus an action $u^A(t)$ is output to approximate the optimal action. Actor can be carried out by a parameterized method such as the neural network. A simple feed-forward neural network with one hidden layer is considered for Actor.

The output of Actor is represented by

$$u^A(t) = w_a^T(t)\sigma\left(v_a^T(t)X_a(t)\right) = w_a^T(t)\sigma_a(t) \tag{5}$$

where $X_a(t) = [x_1(t), x_2(t), \ldots, x_n(t)]$ is the input of Actor, $v_a(t)$ is the weight matrix between the input layer and the hidden layer, while $w_a(t)$ is the weight matrix between the hidden layer and the output layer. The activation function is $\sigma(\cdot)$, which is the hyperbolic tangent function as

$$\sigma(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \tag{6}$$

For simplicity, the hidden layer activation function vector $\sigma(v_a^T(t)X_a(t))$ is written as $\sigma_a(t)$, similarly hereinafter. Updating the weights of Actor is presented in the following parts.

3.3 Critic

Critic criticizes the control action taken under the current state by computing the future accumulative reward-to-go value defined as follows

$$R(t) = \sum_{k=0}^{T} \gamma^k r(t+k+1) \tag{7}$$

where $0 < \gamma < 1$ is a discount factor, t is the time, $r(t)$ is the external reinforcement value and T is the terminal step. Note that it's impossible to calculate the forward-in-time reward. Therefore, a similar three layer feed-forward neural network is used to approximate $R(t)$ in (7).

Critic outputs $J(t)$ as an approximation of $R(t)$

$$J(t) = w_c^T(t)\sigma\left(v_c^T(t)X_c(t)\right) = w_c^T(t)\sigma_c(t) \tag{8}$$

where $X_c(t) = [X_a(t), u(t)]$ is the input of Critic, $v_c(t)$ is the weight matrix between the input layer and the hidden layer, while $w_c(t)$ is the weight matrix between the hidden layer and the output layer. The activation function $\sigma(\cdot)$ has the same form in (6).

3.4 Supervised learning

With a nominal controller, Supervisor provides an admissible control action $u^S(t)$ to Actor for pre-training. In other words, Supervisor guides Actor beforehand to make sure that Actor won't output bad actions, at least not worse than Supervisor. Actually this is a Supervised Learning(SL) procedure. After exploiting the information from Supervisor, Actor explores in a small domain to expect a better action.

In order to execute the SL process with the nominal controller ($Supervisor$), we seek to minimize the supervisory error for each observed state

$$E_s(t) = \frac{1}{2}\left[u^A(t) - u^S(t)\right]^2 \tag{9}$$

To update the weights of Actor, we make an adjustment proportional to the negative gradient of the error with respect to $w_a(t)$

$$\begin{aligned}\Delta w_a^s(t) &= -\alpha\frac{\partial E_s(t)}{\partial w_a(t)} \\ &= -\alpha\sigma_a(t)\left[w_a^T(t)\sigma_a(t) - u^S(t)\right]^T \end{aligned} \tag{10}$$

$$w_a(t+1) = w_a(t) + \Delta w_a^s(t) \tag{11}$$

where α is Actor learning rate. Correspondingly, $v_a(t)$ can also be adjusted by (10) and (11), similarly hereinafter.

When the error decreases to a certain precision ε, SL is completed and Actor approximates to the property of Supervisor.

3.5 SAC learning

After SL, Actor acts an exploration as

$$u(t) = u^A(t) + N(0, \chi) \tag{12}$$

where $N(0, \chi)$ is a random variable with zero mean and variance χ. Thus, the exploratory action is simply a copy of the output of Actor with small noise. Then taking control action $u(t)$, System transits to the next state $x(t+1)$, and the reward r can be observed. Thus next J can be computed and parameters of Critic can be updated as follows.

Define the following prediction error for Critic

$$e_c(t) = \gamma J(t) + r(t) - J(t-1) \tag{13}$$

$$E_c(t) = \frac{1}{2}e_c^2(t) \tag{14}$$

The weights update by a gradient decent rule

$$\begin{aligned}\Delta w_c(t) &= -\beta\frac{\partial E_c(t)}{\partial w_c(t)} \\ &= -\beta\gamma\sigma_c(t)[\gamma w_c^T(t)\sigma_c(t) \\ &\quad -w_c^T(t-1)\sigma_c(t-1) + r(t)]^T \end{aligned} \tag{15}$$

$$w_c(t+1) = w_c(t) + \Delta w_c(t) \tag{16}$$

where β is Critic learning rate.

At the same time, the weights of Actor update in a similar way as above. Define the following performance error

$$e_a(t) = J(t) - U(t) \tag{17}$$

$$E_a(t) = \frac{1}{2}e_a^2(t) \tag{18}$$

where $U(t)$ is the desired ultimate objective.

Update is performed with error back propagation

$$\begin{aligned}\Delta w_a^c(t) &= -\alpha \frac{\partial E_a(t)}{\partial w_a(t)} \\ &= -\alpha w_{c,n+1}\sigma_a(t)\left[w_c^T(t)\sigma_c(t)\right]^T\end{aligned} \tag{19}$$

where $w_{c,n+1}$ is Critic weight connected to the control input u, and n is the number of hidden layer neurons. Then

$$w_a(t+1) = w_a(t) + \Delta w_a^c(t) \tag{20}$$

Algorithm 1 The Supervised Actor-Critic algorithm

Initialization
  Actor weights v_a,w_a
  Critic weights v_c, w_c
  Actor and Critic learning rate α,β
  The discount factor $\gamma \in [0,1]$
  The exploration size χ
Repeat for each trial
  $x(t) \leftarrow$initial state of trial
  repeat for each step of trial
    $u^A(t) \leftarrow$action given by Actor
    $u^S(t) \leftarrow$action given by Supervisor
    update the weights of Actor by (10) and (11)
    until $E_s(t) < \varepsilon$
    $u(t) \leftarrow u^A(t) + N(0,\chi)$
    take action $u(t)$, **observe** reward r, and the next
    state $x(t+1)$
    $e_c(t) \leftarrow r + \gamma \cdot J(t) - J(t-1)$
    update the weights of Critic by (15) and (16)
    $e_a(t) = J(t) - U(t)$
    update the weights of Actor by (19) and (20)
    $x(t) \leftarrow x(t+1)$
  until $x(t)$ is terminal

In sum the complete SAC algorithm is shown in Algorithm 1. It should be noted that the proposed SAC algorithm is an online learning and control approach in essence. Nevertheless, it is suitable to implement the algorithm offline to get an approximate optimal controller. Starting from the initial state vector $x(t)$, the SAC algorithm iterates. Firstly Supervisor "guides" Actor to get a stable control strategy, and with a random "exploration" the control action $u(t)$ acts on the system to get the next state $x(t+1)$. At the same time the reward from the system is given. Then the weights of Critic and Actor are updated respectively. In this way the algorithm iterates until the terminal state is occurred.We comment that Algorithm 1 updates Critic and Actor simultaneously, while the ADP algorithm in Si and Wang (2001) presented a sequential update, which needs more training time.

4 Stability of the SAC algorithm

In this section we present the Lyapunov stability analysis of the proposed SAC algorithm. Following the framework delineated in He and Jagannathan (2005), Hayakawa et al. (2008), Dierks et al. (2009), Vamvoudakis and Lewis (2009) and Liu et al. (2012), we show that the estimation errors of Actor and Critic weights in the SAC are uniformly ultimately bounded (UUB). As the weights of Actor are updated by the supervisory error and Critic error in the proposed SAC algorithm, two different weights estimation errors for Actor are brought in, which are proved to be UUB separately. First we make the following assumption under the current problem settings, which can be reasonably satisfied.

Assumption 1 (*Bounded optimal network weights*) (He and Jagannathan 2005) Let w_a^* and w_c^* be the optimal weights of Actor and Critic, respectively. Assume they are bounded so that

$$\|w_a^*\| \le w_{am}, \|w_c^*\| \le w_{cm} \tag{21}$$

where $w_{am}, w_{cm} \in \mathbb{R}$ are constant bounds on the network weights, and $\|\cdot\|$ represents the Euclidean vector 2-norm, similarly hereinafter.

Lemma 1 *Let Assumption 1 holds and take Critic setting as (8), and the updating rules in (15) and (16). Define* $\bar{w}_c(t) = w_c(t) - w_c^*$ *be the estimation error for Critic weights, and* $\xi_c(t) = \bar{w}_c^T(t)\sigma_c(t)$ *be the approximation error of Critic. Then, for*

$$L_1(t) = \frac{1}{\beta}tr\left(\bar{w}_c^T(t)\bar{w}_c(t)\right) \tag{22}$$

its first difference is given by

$$\begin{aligned}\Delta L_1(t) \le &-\gamma^2\|\xi_c(t)\|^2 - \left(1-\beta\gamma^2\|\sigma_c(t)\|^2\right) \\ &\times\|\gamma\xi_c^T(t) + \gamma w_c^{*T}(t)\sigma_c(t) + r(t) \\ &- w_c^T(t-1)\sigma_c(t-1)\|^2 + 2\|\gamma w_c^{*T}(t)\sigma_c(t) \\ &+ r(t) - w_c^{*T}\sigma_c(t-1) \\ &- \frac{2}{3}\bar{w}_c^T(t-1)\sigma_c(t-1)\|^2 + \frac{2}{9}\|\xi_c(t-1)\|^2\end{aligned} \tag{23}$$

Proof The first difference of (22) can be written as

$$\begin{aligned}\Delta L_1(t) &= \frac{1}{\beta}tr\left[\bar{w}_c^T(t+1)\bar{w}_c(t+1) - \bar{w}_c^T(t)\bar{w}_c(t)\right] \\ &= \frac{1}{\beta}tr\left[\left(\bar{w}_c(t) + \Delta w_c(t)\right)^T\left(\bar{w}_c(t) + \Delta w_c(t)\right)\right. \\ &\quad \left.-\bar{w}_c^T(t)\bar{w}_c(t)\right]\end{aligned} \tag{24}$$

with the basic properties for trace of matrix, we have

$$\Delta L_1(t) = \frac{1}{\beta}tr\left[2\bar{w}_c^T(t)\Delta w_c(t) + \Delta w_c^T(t)\Delta w_c(t)\right] \tag{25}$$

Substituting (15) into (25), we get

$$\begin{aligned}\Delta L_1(t) &= tr\Big[-2\gamma^2\xi_c(t)\xi_c^T(t) - 2\gamma\xi_c(t)P \\ &\quad +\beta\gamma^4\|\sigma_c(t)\|^2\xi_c(t)\xi_c^T(t) \\ &\quad +2\beta\gamma^3\|\sigma_c(t)\|^2\xi_c(t)P \\ &\quad +\beta\gamma^2\|\sigma_c(t)\|^2P^TP\Big] \\ &= tr\Big[-\gamma^2\xi_c(t)\xi_c^T(t) + P^TP \\ &\quad -(1-\beta\gamma^2\|\sigma_c(t)\|^2) \\ &\quad \times(\gamma\xi_c^T(t)+P)^T(\gamma\xi_c^T(t)+P)\Big] \\ &= -\gamma^2\|\xi_c(t)\|^2 - (1-\beta\gamma^2\|\sigma_c(t)\|^2) \\ &\quad \times\|\gamma\xi_c^T(t)+P\|^2 + \|P\|^2 \end{aligned} \tag{26}$$

where

$$P = \Big[\gamma w_c^{*T}(t)\sigma_c(t) + r(t) - w_c^T(t-1)\sigma_c(t-1)\Big]^T$$

Substituting it into (26), and with Cauchy-Schwarz inequality we get

$$\begin{aligned}\Delta L_1(t) &\le -\gamma^2\|\xi_c(t)\|^2 - (1-\beta\gamma^2\|\sigma_c(t)\|^2) \\ &\quad \times\|\gamma\xi_c^T(t) + \gamma w_c^{*T}(t)\sigma_c(t) + r(t) \\ &\quad - w_c^T(t-1)\sigma_c(t-1)\|^2 + 2\|\gamma w_c^{*T}(t)\sigma_c(t) \\ &\quad + r(t) - w_c^{*T}\sigma_c(t-1) \\ &\quad - \frac{2}{3}\bar{w}_c^T(t-1)\sigma_c(t-1)\|^2 + \frac{2}{9}\|\xi_c(t-1)\|^2 \end{aligned} \tag{27}$$

Proof is completed. □

Lemma 2 *Let Assumption 1 holds and take Actor setting as (12), and the updating rules in (19) and (20). Define* $\bar{w}_a(t) = w_a(t) - w_a^*$ *be the estimation error for Actor weights, and* $\xi_a(t) = \bar{w}_a^T(t)\sigma_a(t)$ *be the approximation error of Actor. Then, for*

$$L_2(t) = \frac{l}{\alpha}tr\left(\bar{w}_a^T(t)\bar{w}_a(t)\right) \tag{28}$$

where $l > 0$ *is the regulating parameter, its first difference is given by*

$$\begin{aligned}\Delta L_2(t) &\le l\|\xi_a(t)\|^2 - l\|w_{c,n+1}\|^2(1-\alpha\|\sigma_a(t)\|^2) \\ &\quad \times\|w_c^T(t)\sigma_c(t)\|^2 + 4l\|w_{c,n+1}\|^2\|\xi_c(t)\|^2 \\ &\quad +4l\|w_{c,n+1}\|^2\|w^{*T}\sigma_c(t)\|^2 \end{aligned} \tag{29}$$

Proof The first difference of (23) can be written as

$$\begin{aligned}\Delta L_2(t) &= \frac{l}{\alpha}tr\Big[\bar{w}_a^T(t+1)\bar{w}_a(t+1) - \bar{w}_a^T(t)\bar{w}_a(t)\Big] \\ &= \frac{l}{\alpha}tr\Big[\left(\bar{w}_a(t)+\Delta w_a^c(t)\right)^T\left(\bar{w}_a(t)+\Delta w_a^c(t)\right) \\ &\quad -\bar{w}_a^T(t)\bar{w}_a(t)\Big] \\ &= \frac{l}{\alpha}tr\Big[2\bar{w}_a^T(t)\Delta w_a^c(t) + \Delta w_a^{cT}(t)\Delta w_a^c(t)\Big] \end{aligned} \tag{30}$$

Substituting (19) into (30), we have

$$\begin{aligned}\Delta L_2(t) &= l\cdot tr[-2\bar{w}_a^T(t)\sigma_a(t)w_{c,n+1}[w_c^T(t)\sigma_c(t)]^T \\ &\quad +\alpha[w_c^T(t)\sigma_c(t)]w_{c,n+1}^T\sigma_a^T(t)\sigma_a(t) \\ &\quad \times w_{c,n+1}[w_c^T(t)\sigma_c(t)]^T] \\ &= l\cdot tr[-2\xi_a(t)w_{c,n+1}[w_c^T(t)\sigma_c(t)]^T \\ &\quad +\alpha\|\sigma_a(t)\|^2[w_c^T(t)\sigma_c(t)]w_{c,n+1}^T \\ &\quad \times w_{c,n+1}[w_c^T(t)\sigma_c(t)]^T] \\ &= l\cdot tr[(\xi_a(t) - w_c^T(t)\sigma_c(t)w_{c,n+1}^T) \\ &\quad \times(\xi_a(t) - w_c^T(t)\sigma_c(t)w_{c,n+1}^T)^T - \xi_a(t)\xi_a^T(t) \\ &\quad -[w_c^T(t)\sigma_c(t)]w_{c,n+1}^Tw_{c,n+1}[w_c^T(t)\sigma_c(t)]^T \\ &\quad +\alpha\|\sigma_a(t)\|^2[w_c^T(t)\sigma_c(t)]w_{c,n+1}^T \\ &\quad \times w_{c,n+1}[w_c^T(t)\sigma_c(t)]^T] \end{aligned}$$

$$\begin{aligned}\Delta L_2(t) &= l(-\|\xi_a(t)\|^2 - \|w_{c,n+1}\|^2 \\ &\quad \times(1-\alpha\|\sigma_a(t)\|^2)\|w_c^T(t)\sigma_c(t)\|^2 \\ &\quad +\|\xi_a(t) - w_c^T(t)\sigma_c(t)w_{c,n+1}^T\|^2) \end{aligned} \tag{31}$$

By Cauchy-Schwarz inequality, we get

$$\begin{aligned}\Delta L_2(t) &\le -l\|\xi_a(t)\|^2 - l\|w_{c,n+1}\|^2 \\ &\quad \times(1-\alpha\|\sigma_a(t)\|^2)\|w_c^T(t)\sigma_c(t)\|^2 \\ &\quad +2l\|\xi_a(t)\|^2 \\ &\quad +2l\|w_{c,n+1}\|^2\|w_c^T(t)\sigma_c(t)\|^2 \\ &\le l\|\xi_a(t)\|^2 - l\|w_{c,n+1}\|^2(1-\alpha\|\sigma_a(t)\|^2) \\ &\quad \times\|w_c^T(t)\sigma_c(t)\|^2 + 4l\|w_{c,n+1}\|^2\|\xi_c(t)\|^2 \\ &\quad +4l\|w_{c,n+1}\|^2\|w^{*T}\sigma_c(t)\|^2 \end{aligned} \tag{32}$$

Proof is completed. □

Lemma 3 *Let Assumption 1 holds and take the supervised Actor setting as (5), and the updating rules in (10) and (11). Similarly we define*

$$\xi_a(t) = (w_a(t) - w_a^*)\sigma_a(t) = \tilde{w}_a^T(t)\sigma_a(t)$$

Then, for

$$L_3(t) = \frac{l}{\alpha}tr\left(\tilde{w}_a^T(t)\tilde{w}_a(t)\right) \tag{33}$$

its first difference is given by

$$\begin{aligned}\Delta L_3(t) &\le -l\|\xi_a(t)\|^2 - l(1-\alpha\|\sigma_a(t)\|^2) \\ &\quad \times\|w_a^T(t)\sigma_a(t) - u^S(t)\|^2 \\ &\quad +2l\|u^S(t)\|^2 + 2l\|w_a^{*T}\sigma_a(t)\|^2 \end{aligned} \tag{34}$$

Proof

$$\begin{aligned}\Delta L_3(t) &= \frac{l}{\alpha}tr\Big[\tilde{w}_a^T(t+1)\tilde{w}_a(t+1) - \tilde{w}_a^T(t)\tilde{w}_a(t)\Big] \\ &= \frac{l}{\alpha}tr\Big[\left(\tilde{w}_a(t)+\Delta w_a^s(t)\right)^T\left(\tilde{w}_a(t)+\Delta w_a^s(t)\right) \\ &\quad -\tilde{w}_a^T(t)\tilde{w}_a(t)\Big] \\ &= \frac{l}{\alpha}tr\Big[2\tilde{w}_a^T(t)\Delta w_a^s(t) + \Delta w_a^{sT}(t)\Delta w_a^s(t)\Big] \end{aligned} \tag{35}$$

Substituting (10) into (35), we can get

$$\begin{aligned}\Delta L_3(t) &= \frac{l}{\alpha} tr\Big[-2\alpha\tilde{w}_a^T(t)\sigma_a(t)\left(w_a^T(t)\sigma_a(t)-u^S(t)\right)^T \\ &\quad +\alpha^2\left(w_a^T(t)\sigma_a(t)-u^S(t)\right)\sigma_a^T(t)\sigma_a(t) \\ &\quad \times\left(w_a^T(t)\sigma_a(t)-u^S(t)\right)^T\Big] \\ &= l\cdot tr\Big[-2\xi_a(t)\left(w_a^T(t)\sigma_a(t)-u^S(t)\right)^T \\ &\quad +\alpha\|\sigma_a(t)\|^2\left(w_a^T(t)\sigma_a(t)-u^S(t)\right) \\ &\quad \times\left(w_a^T(t)\sigma_a(t)-u^S(t)\right)^T\Big]\end{aligned}$$

$$\begin{aligned}\Delta L_3(t) &= l\cdot tr\Big[\left(\xi_a(t)-w_a^T(t)\sigma_a(t)+u^S(t)\right) \\ &\quad \times\left(\xi_a(t)-w_a^T(t)\sigma_a(t)+u^S(t)\right)^T-\xi_a(t)\xi_a^T(t) \\ &\quad -\left(w_a^T(t)\sigma_a(t)-u^S(t)\right)\left(w_a^T(t)\sigma_a(t)-u^S(t)\right)^T \\ &\quad +\alpha\|\sigma_a(t)\|^2\left(w_a^T(t)\sigma_a(t)-u^S(t)\right) \\ &\quad \times\left(w_a^T(t)\sigma_a(t)-u^S(t)\right)^T\Big] \\ &= -l\|\xi_a(t)\|^2-l(1-\alpha\|\sigma_a(t)\|^2) \\ &\quad \times\|w_a^T(t)\sigma_a(t)-u^S(t)\|^2 \\ &\quad +l\|u^S(t)-w_a^{*T}\sigma_a(t)\|^2 \qquad (36)\end{aligned}$$

By Cauchy-Schwarz inequality, we have

$$\begin{aligned}\Delta L_3(t) &\le -l\|\xi_a(t)\|^2-l(1-\alpha\|\sigma_a(t)\|^2) \\ &\quad \times\|w_a^T(t)\sigma_a(t)-u^S(t)\|^2 \\ &\quad +2l\|u^S(t)\|^2+2l\|w_a^{*T}\sigma_a(t)\|^2 \qquad (37)\end{aligned}$$

Proof is completed. □

We now present the main theorem.

Theorem 1 *Let Assumption 1 holds and consider Critic as (8) with the updating rules in (15) and (16), Actor as (12) and the updating rules (19) and (20), and the supervised Actor as (5) with the updating rules in (10) and (11). Then the estimation errors for the weights of Critic and Actor are uniformly ultimate bounded (UUB), provided that the following conditions hold*

$$\frac{\sqrt{2}}{3}<\gamma<1, 0<\alpha\|\sigma_a(t)\|^2<1, 0<\beta\gamma^2\|\sigma_c(t)\|^2<1 \qquad (38)$$

Proof Define the following Lyapunov function candidate

$$\begin{aligned}L(t) &= \frac{1}{\beta}tr\left[\tilde{w}_c^T(t)\tilde{w}_c(t)\right]+\frac{1}{\alpha}tr\left[\tilde{w}_a^T(t)\tilde{w}_a(t)\right] \\ &\quad +\frac{1}{\alpha}tr\left[\tilde{w}_a^T(t)\tilde{w}_a(t)\right]+\frac{2}{9}\|\xi_c(t-1)\|^2 \qquad (39)\end{aligned}$$

where $l>0$ is the regulating parameter. The first difference is

$$\begin{aligned}\Delta L(t) &= L(t+1)-L(t) \\ &= \Delta L_1(t)+\Delta L_2(t)+\Delta L_3(t) \\ &\quad +\frac{2}{9}(\|\xi_c(t)\|^2-\|\xi_c(t-1)\|^2) \qquad (40)\end{aligned}$$

By Lemmas 1, 2 and 3, we can get

$$\begin{aligned}\Delta L(t) &\le -\gamma^2\|\xi_c(t)\|^2-(1-\beta\gamma^2\|\sigma_c(t)\|^2) \\ &\quad \times\|\gamma\xi_c^T(t)+\gamma w^{*T}\sigma_c(t) \\ &\quad +r(t)-w_c^T(t-1)\sigma_c(t-1)\|^2 \\ &\quad +2\|\gamma w_c^{*T}\sigma_c(t)+r(t)-w_c^{*T}\sigma_c(t-1) \\ &\quad -\frac{2}{3}\bar{w}_c^T(t-1)\sigma_c(t-1)\|^2 \\ &\quad +\frac{2}{9}\|\xi_c(t-1)\|^2+l\|\xi_a(t)\|^2 \\ &\quad -l\|w_{c,n+1}\|^2(1-\alpha\|\sigma_a(t)\|^2)\|w_c^T(t)\sigma_c(t)\|^2 \\ &\quad +4l\|w_{c,n+1}\|^2\|\xi_c(t)\|^2+l\|u^S(t)-w^{*T}\sigma_a(t)\|^2 \\ &\quad +4l\|w_{c,n+1}\|^2\|w_c^{*T}\sigma_c(t)\|^2-l\|\xi_a(t)\|^2 \\ &\quad -l(1-\alpha\|\sigma_a(t)\|^2)\|w_a^T(t)\sigma_a(t)-u^S(t)\|^2 \\ &\quad +\frac{2}{9}\|\xi_c(t)\|^2-\frac{2}{9}\|\xi_c(t-1)\|^2 \\ &= -l\|w_{c,n+1}\|^2(1-\alpha\|\sigma_a(t)\|^2)\|w_c^T(t)\sigma_c(t)\|^2 \\ &\quad -l(1-\alpha\|\sigma_a(t)\|^2)\|w_a^T(t)\sigma_a(t)-u^S(t)\|^2 \\ &\quad -(\gamma^2-\frac{2}{9}-4l\|w_{c,n+1}\|^2)-(1-\beta\gamma^2\|\sigma_c(t)\|^2) \\ &\quad \times\|\gamma\xi_c^T(t)+\gamma w_c^{*T}\sigma_c(t)+r(t) \\ &\quad -w_c^T(t-1)\sigma_c(t-1)\|^2+\Delta\bar{L}^2 \qquad (41)\end{aligned}$$

where

$$\begin{aligned}\Delta\bar{L}^2 &= 2\|\gamma w_c^{*T}\sigma_c(t)+r(t)-w_c^{*T}\sigma_c(t-1) \\ &\quad -\frac{2}{3}\bar{w}_c^T(t-1)\sigma_c(t-1)\|^2 \\ &\quad +4l\|w_{c,n+1}\|^2\|w_c^{*T}\sigma_c(t)\|^2 \\ &\quad +l\|u^S(t)-w_a^{*T}\sigma_a(t)\|^2 \qquad (42)\end{aligned}$$

and choose

$$\frac{\sqrt{2}}{3}<\gamma<1, 0<\alpha\|\sigma_a(t)\|^2<1, 0<\beta\gamma^2\|\sigma_c(t)\|^2<1 \qquad (43)$$

Let the regulating parameter

$$0<l<\left(\gamma^2-\frac{2}{9}\right)/(4\|w_{c,n+1}\|^2) \qquad (44)$$

For (42), with Cauchy-Schwarz inequality we can get

$$\begin{aligned}\Delta\bar{L}^2 &\le 8\gamma^2\|w_c^{*T}\sigma_c(t)\|^2+8r^2(t)+8\|w_c^{*T}\sigma_c(t-1)\|^2 \\ &\quad +8\times\frac{4}{9}\|\bar{w}_c^T(t-1)\sigma_c(t-1)\|^2+2l\|u^S(t)\|^2\end{aligned}$$

$$
\begin{aligned}
&+4l\|w_{c,n+1}\|^2\|w_c^{*T}\sigma_c(t)\|^2+2l\|w_a^{*T}\sigma_a(t)\|^2\\
&\le(8\gamma^2+\frac{104}{9}+4l\|w_{cm,n+1}\|^2)w_{cm}^2\sigma_{cm}^2\\
&+8r_m^2+2lu_m^{S2}+2lw_{am}^2\sigma_{am}^2=\Delta\bar{L}_m^2
\end{aligned}
\tag{45}
$$

where $w_{cm}, \sigma_{cm}, w_{am}, \sigma_{am}, w_{cm,n+1}, u_m^S$ and r_m are the upper bounds of $w_c, \sigma_c, w_a, \sigma_a, w_{c,n+1}, u^S$ and r, respectively. Provided condition (43) holds, then for any

$$
\|\xi_c(t)\|>\Delta\bar{L}/\sqrt{\gamma^2-\frac{2}{9}-4l\|w_{c,n+1}\|^2}
\tag{46}
$$

there always be

$$
\Delta L(t)\le 0
\tag{47}
$$

According to the Lyapunov extension theorem, (47) demonstrates that the estimation errors for the weights of Critic and Actor are UUB. □

5 Experimental results

5.1 The design of ACC with SAC

For the ACC problem, the terminal state is defined as

$$
\begin{cases}
|\Delta v|<0.072\,\text{km/h}\\
|\Delta d|<0.2\,\text{m}
\end{cases}
\tag{48}
$$

If the terminal state is achieved, the reinforcement signal is assigned "0" for "success", or else "−1" for other states. Thus, the desired ultimate objective U in (17) can be set to "0". Moreover, in case of collision, we define a special bump state by (2) when $d_r \le 0$. If this bump state occurs, the reinforcement signal "−10" is given for punishment which means "failure".

5.2 Training process

The training process of the SAC algorithm for ACC is executed according to Algorithm 1. The discount factor γ is 0.9. The learning rate of Actor α is 0.1, and Critic β 0.3. The exploration size χ is 0.2, which means the action is added by a Gaussian noise with mean 0 and variance 0.2. Actor and Critic are all configured 8 hidden neurons experientially, and the weights of both networks are initialized randomly. Here we adopt a hybrid PD controller proposed in Moon et al. (2009) as Supervisor.

We generate 1,000 preceding vehicle speed data in [0,25] (m/s) randomly but uniformly for training (see in Fig. 4). The driving habit parameter and the zero speed clearance are chosen the same as "Driver_3" in Table 1 (Kyongsu and Ilki 2004). The following vehicle starts from the initial state $x(0) = (20, 18)$, and takes action in each step until encountering the terminal state. Figures 5 and 6 show Actor and

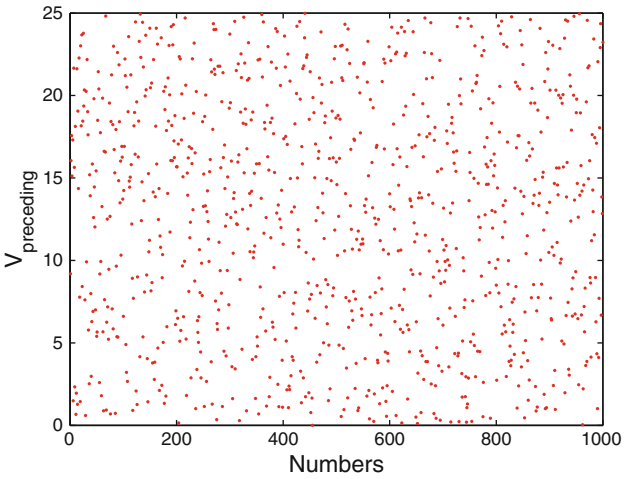

Fig. 4 Training data of the preceding vehicle speed

Table 1 Driving behavior with time gaps and minimum clearance (Kyongsu and Ilki 2004)

Human driver	Time-gap (s)	Min. clearance (m)
Driver_1	0.67	2.25
Driver_2	1.25	4.30
Driver_3	1.70	1.64

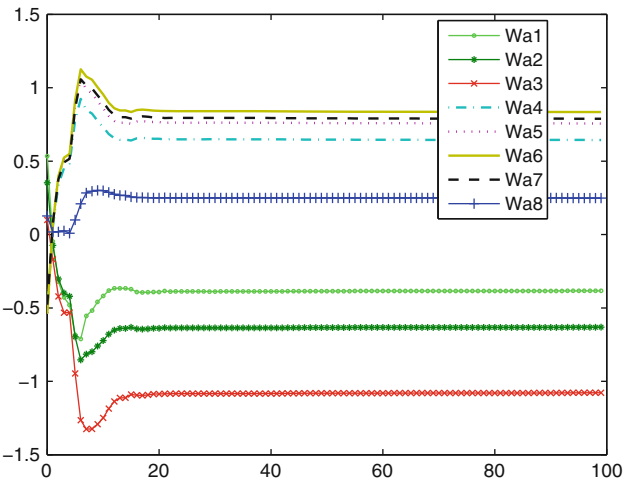

Fig. 5 Converged Actor weights

Critic weights connected the hidden layer and the output layer, which are converged to upper bounds respectively.

For comparison, we remove Supervisor and carry out an Actor–Critic (AC) learning algorithm (Sutton and Barto 1998) which has a similar training process as SAC, and we also compare with the SADP method in Zhao et al. (2013). We implement 10 experiments with 1,000 trials respectively, and a trial is said to be success when an initial state can converge to the terminal state shown in (48). Table 2 shows the success rate comparison between SAC, AC and SADP.

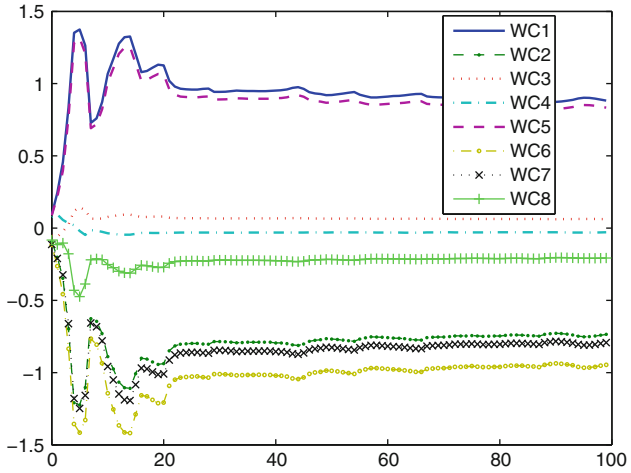

Fig. 6 Converged Critic weights

Table 2 Training results comparison among SAC, AC and SADP

Algorithms	Number of experiments	Number of trials	Success rate (%)
SAC	10	1,000	100
AC	10	1,000	4.37
SADP	10	1,000	19.88

Definitely, the presence of Supervisor greatly improves the convergence of the training process.

5.3 Adaptability test for different scenarios and drivers

After training, we get Actor to test the adaptability for typical driving scenarios and different drivers. Detailed definitions of driving scenarios can be seen in Zhao et al. (2013), such as normal driving, stop-and-go, emergency braking, cut-in, and so on. To show the generalization of the proposed SAC algorithm, we choose "Driver_2" in Table 1 as a typical driver for test. In order to simulate the real driving scenarios, we add random noise in the experiments. Moreover, the hybrid PD controller proposed above is adopted here for comparison.

Figure 7 shows the comparison result of the SG scenario. The preceding vehicle starts from 0 km/h and accelerates with 0.2 m/s^2 to 57.6 km/h, and holds about 100 s, and then decelerates to a total stop. While the following vehicle with ACC function runs with 18 km/h and 20 m away from the preceding vehicle. It can be concluded that the proposed SAC approach performs as well as the hybrid PD controller both in speed and distance, and both of them provide near optimal control action, which indicates the good learning ability of SAC approach presented in this paper.

Figure 8 shows the experimental results of the emergency braking scenario. The preceding vehicle decelerates from 80 to 0 km/h in 80 s. We can clearly see that the SAC method

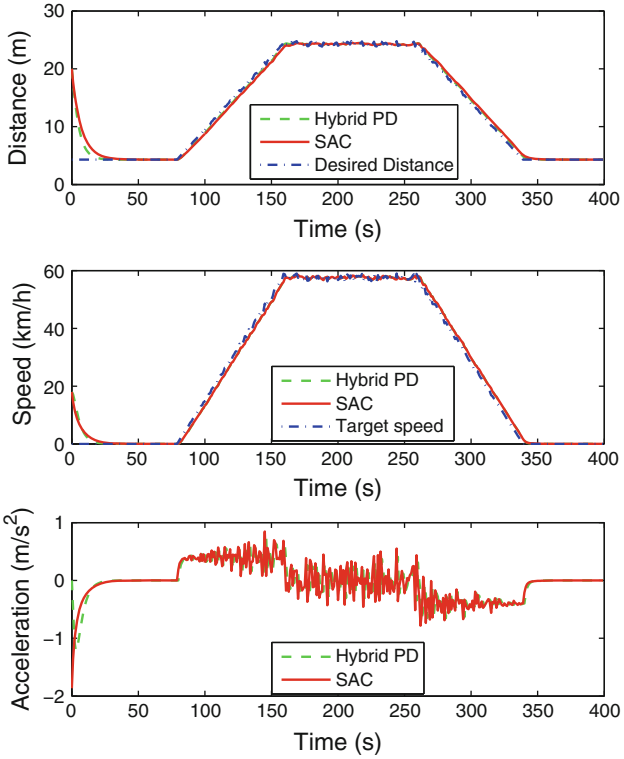

Fig. 7 Experimental results with SAC and the hybrid PD method in the SG scenario

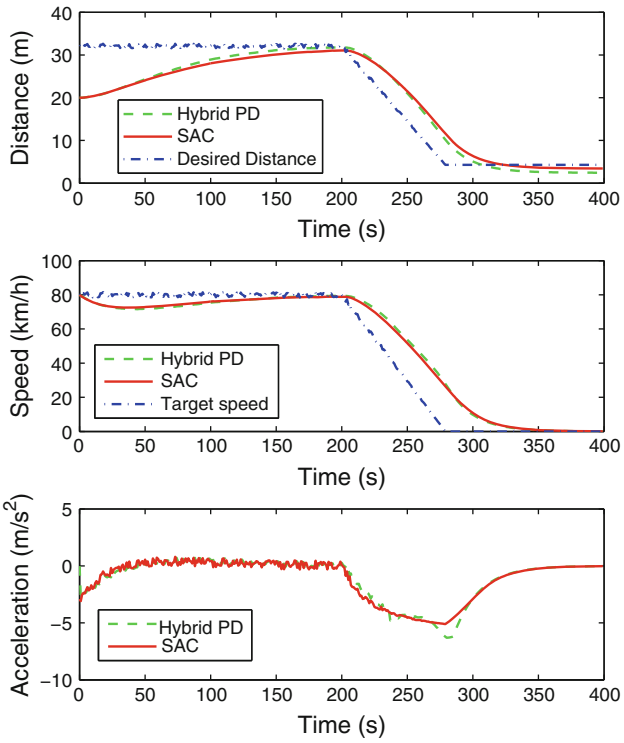

Fig. 8 Experimental results with SAC and the hybrid PD method in the emergency braking scenario

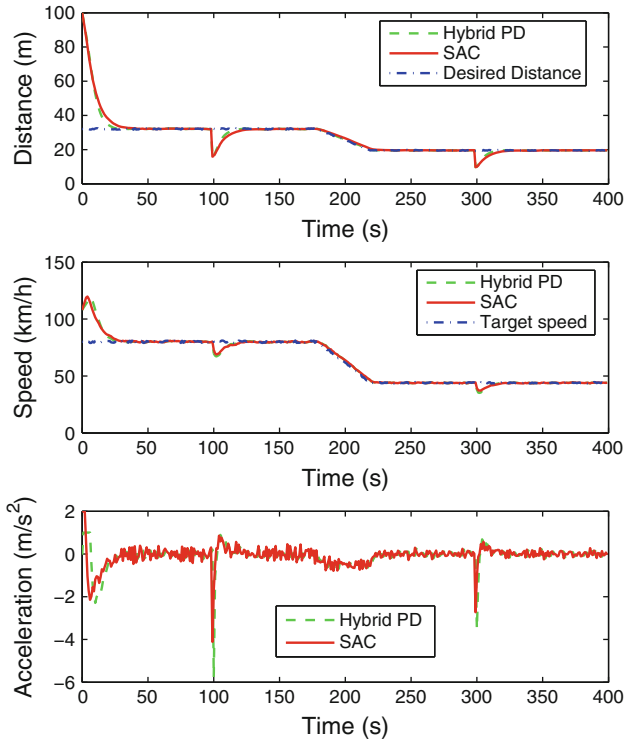

Fig. 9 Experimental results with SAC and the hybrid PD method in the cut-in scenario

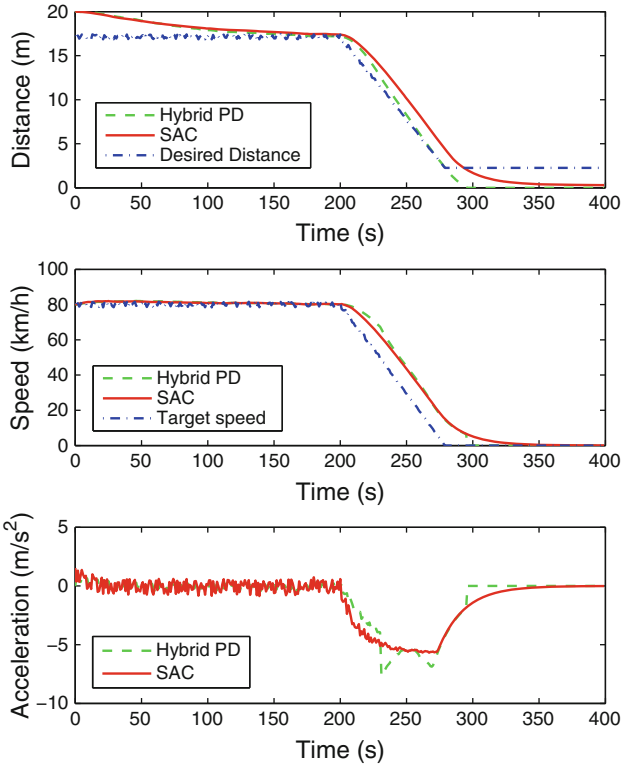

Fig. 10 Experimental results with SAC and the hybrid PD method in the emergency braking scenario for another driver

performs better than the hybrid PD control strategy with a smoother acceleration, which is very important to improve the comfort for drivers in braking process. Furthermore, the SAC method generates smaller errors to the desired distance, which means safer in avoiding collisions.

In Fig. 9, the control performance of the cut-in scenario is presented. The preceding vehicle travels by 80 km/h while the following vehicle runs after with a high speed 108 km/h. At 100 s another vehicle inserts between the preceding vehicle and the following vehicle. We comment that when another vehicle cuts in, the following vehicle brakes to avoid the crash. So there is a reduction in the safety distance, but SAC algorithm performs better in the regulation of acceleration, which also enhances the comfort for drivers.

In addition, we also test the adaptability of the SAC strategy by changing drivers according to Table 1. The result is shown in Fig. 10, where we choose "Driver_1" to test the driving performance in the emergency braking scenario with the same setting as in Fig. 8. It should be noted that the hybrid PD controller cannot deal with this driving scenario and a collision occurs. However, the SAC approach performs better and avoids the collision effectively because of exploration and exploitation. Thus we make conclusion that the proposed approach can always satisfy the driver's expectation.

Moreover, we verify the effectiveness of Critic mechanism by removing Critic in the SAC approach, which becomes a SL method. Experiments have shown that SL cannot avoid collision in emergency braking scenario, although it performs as well as SAC. Thus we can conclude that the SAC algorithm is superior with Critic mechanism.

In summary, experimental results demonstrate the effectiveness of the SAC algorithm presented in this paper for the ACC problem.

5.4 Discussions

Excellent performance of the presented SAC approach has been verified in the above experiments in dealing with ACC problem. Some advantages of the SAC approach should be summarized here, which are our main contributions as well.

1. Pre-training of Actor with the nominal controller provides appropriate initial weights, which is critical for the success rate of the training process. As is shown in Table 2, the SAC algorithm can guarantee 100 % success compared with other related algorithms such as AC and SADP.
2. Exploration of the control strategy avoids the total copy of the "Supervisor". The outstanding adaptability (avoiding collisions successfully while others fail) of the SAC approach in dealing with different situations for the ACC problem benefits from this reasonable exploration.
3. Compared with the ADP algorithm in Si and Wang (2001), we update Critic and Actor simultaneously

instead of sequentially, which can reduce the training time significantly.

4. Compared with the Supervised Learning (SL), the proposed SAC approach performs better in some extreme scenarios such as emergency braking, because of the evaluation to actions by Critic.

6 Conclusion

In this paper a novel SAC learning approach is proposed. We adopt a nominal controller to pre-train Actor and get a basic control policy. Then the exploration of such a policy tries to find a better control action in a small domain, while better or not of the action will be criticized by Critic. Critic approximates the future accumulative reward-to-go value, which can be used to update the weights of Actor so as to improve the control policy. Moreover, the Lyapunov approach is used to analyze the stability of the proposed SAC algorithm. Detailed proofs have been given to show that the estimation errors of the neural networks are uniformly ultimately bounded (UUB).

We implement this reinforcement learning approach with neural network both in Actor and Critic, and apply it for the ACC problem. The experiments verify the improved generalization performance in various scenarios and for different drivers. Thus, we can make the conclusion that the proposed SAC algorithm is feasible and effective for ACC problem.

Acknowledgments We acknowledge Dr. Cesare Alippi and Dr. Yuzhu Huang for their valuable discussions.

References

Andreas T (2012) Vehicle trajectory effects of adaptive cruise control. J Intell Trans Syst 16(1):36–44

Barto A, Dietterich T (2004) Reinforcement learning and its relationship to supervised learning. In: Si J, Barto A, Powell W, Wunsch D (eds) Handbook of learning and approximate dynamic programming. IEEE Press,Wiley, London, pp 47–63

Bifulco G, Simonelli F, Pace D (2008) Experiments toward an human-like adaptive cruise control. In: 2008 IEEE intelligent vehicles, symposium, pp 919–924

Dierks T, Thumati B, Jagannathan S (2009) Optimal control of unknown affine nonlinear discrete-time systems using offline-trained neural networks with proof of convergence. Neural Netw 22(5):851–860

Fritz A, Schiehlen W (2001) Nonlinear acc in simulation and measurement. Veh Syst Dyn 36:159–177

Guvenc B, Kural E (2006) Adaptive cruise control simulator: a low-cost, multiple-driver-in-the-loop simulator. IEEE Control Syst Mag 26(3):42–55

Hayakawa T, Haddad W, Hovakimyan N (2008) Neural network adaptive control for a class of nonlinear uncertain dynamical systems with asymptotic stability guarantees. IEEE Trans Neural Netw 19:80–89

He P, Jagannathan S (2005) Reinforcement learning-based output feedback control of nonlinear systems with input constraints. IEEE Trans Syst Man Cybern Part B Cybern 35(1):150–154

Hu Z, Zhao D (2011) Adaptive cruise control based on reinforcement leaning with shaping rewards. J Adv Comput Intell Intell Info 15(3):4645–4650

Kesting A, Treiber M, Schoenhof M, Kranke F, Helbing D (2007) Traffic and granular flow'05. In: Jam-avoiding adaptive cruise control (ACC) and its impact on traffic, dynamics. Springer, Berlin, pp 633–643

Kural E, Guvenc B (2010) Model predictive adaptive cruise control. In: 2010 IEEE international conference on systems man and cybernetics (SMC), pp 1455–1461

Kyongsu Y, Ilki M (2004) A driver-adaptive stop-and-go cruise control strategy. In: 2004 IEEE international conference on networking, sensing and, control, pp 601–606

Li T, Zhao D, Yi J (2008) Adaptive dynamic neuro-fuzzy system for traffic signal control. In: 2008 IEEE international joint conference on neural networks (IJCNN), pp 1840–1846

Li S, Li K, Rajamani R, Wang J (2011) Model predictive multi-objective vehicular adaptive cruise control. IEEE Trans Control Syst Technol 19:556–566

Liu F, Sun J, Si J, Guo W, Mei S (2012) A boundedness result for the direct heuristic dynamic programming. Neural Netw 32:229–235

Martinez J, Canudas-De-Wit C (2007) A safe longitudinal control for adaptive cruise control and stop-and-go scenarios. IEEE Trans Control Syst Technol 15:246–258

Milanes V, Villagra J, Godoy J, Gonzalez C (2012) Comparing fuzzy and intelligent pi controllers in stop-and-go maneuvers. IEEE Trans Control Syst Technol 20:770–778

Moon S, Moon I, Yi K (2009) Design, tuning, and evaluation of a full-range adaptive cruise control system with collision avoidance. Control Eng Pract 17(4):442–455

Murray J, Cox C, Lendaris G, Saeks R (2002) Adaptive dynamic programming. IEEE Trans Sys Man Cybern Part C 32(2):140–152

Naranjo J, Gonzalez C, Garcia R, Pedro d (2006) Acc plus stop&go maneuvers with throttle and brake fuzzy control. IEEE Trans Intell Transp Syst 7:213–225

Ohno H (2001) Analysis and modeling of human driving behaviors using adaptive cruise control. Appl Soft Comput 1:237–243

Rosenstein M, Barto A (2004) Supervised actor-critic reinforcement learning. In: Si J, Barto A, Powell W, Wunsch D (eds) Handbook of learning and approximate dynamic programming. IEEE Press,Wiley, London, pp 359–380

Si J, Wang Y (2001) On-line learning control by association and reinforcement. IEEE Trans Neural Netw 12(2):264–276

Siciliano B, Khatib O (2008) Springer handbook of robotics, chap. 51 intelligent vehicles. Springer, Berlin

Sutton R, Barto A (1998) Reinforcement learning: an introduction. The MIT Press, Cambridge

Vamvoudakis K, Lewis F (2009) Online actor critic algorithm to solve the continuous-time infinite horizon optimal control problem. In: 2009 international joint conference on neural networks (IJCNN), pp 3180–3187

Xia Z, Zhao D (2012) Hybrid feedback control of vehicle longitudinal acceleration. In: 2012 31st Chinese control conference (CCC), pp 7292–7297

Xiao L, Gao F (2011) Practical string stability of platoon of adaptive cruise control vehicles. IEEE Trans Intell Transp Syst 12:1184–1194

Zhao D, Bai X, Wang F, Xu J, Yu W (2011) Dhp for coordinated freeway ramp metering. IEEE Trans Intell Transp Syst 12(4):990–999

Zhao D, Zhang Z, Dai Y (2012a) Self-teaching adaptive dynamic programming for go-moku. Neurocomputing 78(1):23–29

Zhao D, Zhu Y, He H (2012b) Neural and fuzzy dynamic programming for under-actuated systems. In: 2012 international joint conference on neural networks(IJCNN), pp 1–7

Zhao D, Hu Z, Xia Z, Alippi C, Zhu Y, Wang D (2013) Full-range adaptive cruise control based on supervised adaptive dynamic programming. Neurocomputing. doi:10.1016/jneucom201209034