Adaptive critic anti-slip control of wheeled autonomous robot

W.-S. Lin, L.-H. Chang and P.-C. Yang

Abstract: When a wheeled autonomous robot drives with wheel slips, the velocity and posture control becomes difficult. An ideal automatic driving control system should be able to comply with changes in slip conditions so as to optimise the control performance. Using dual heuristic programming and multi-layer perceptron neural networks, an adaptive critic anti-slip control design is developed to achieve this goal. The critic structure enables neural network learning by satisfying the Bellman equation so that the inclination of the action performance can be assessed to improve the control parameters. A slip model of the robot vehicle is derived. The adaptive critic anti-slip control system is verified extensively by computer simulation. The result shows that the performance is significantly better than that of using traditional fuzzy control.

1 Introduction

Wheeled autonomous robots may drive without prior knowledge of slip conditions. An automatic driving controller with fixed parameters may in this situation perform poorly or even go out of control. Ideally, the controller should be able to learn the slip conditions, assess the robot's states and then compensate for the slip effect. This article demonstrates an adaptive critic anti-slip control design that fulfils this requirement by dual heuristic programming (DHP)-based neural networks. The adaptive critic method is a technological attempt to implement human processes of learning and applying control to achieve a future goal [1]. Humans are motivated by love, fortune, power and so on. The critic structure shapes the controller to satisfy the Bellman equation, the motivation equivalence. The critic method essentially uses computational entity to criticise actions. Then in accordance with the inclination of the action performance, the control unit is improved to approximate optimal control.

Dynamic programming is a mathematical formalism to design an optimal controller for a nonlinear system. Bellman's [2] principle of optimality allows us to take every step as a starting point to look for the optimal solution. However, the nature of requesting future information makes dynamic programming not directly feasible in practical applications. Alternatively, DHP has been developed as an approximator to implement dynamic programming [3, 4]. Other than DHP, heuristic dynamic programming (HDP) and global dual heuristic programming (GDHP) serve the same purpose. They are differentiated by the output of the critic entity. In the DHP method, the critic produces the derivative of the Bellman equation [5, 6]. However, in HDP, the critic's output is the criteria function of the Bellman equation. In GDHP, both the Bellman equation and its derivative are calculated. However, DHP was demonstrated in Prokhorov *et al*. [7] and Venayagamoorthy *et al*. [8] to have a superior performance to HDP and there was no observable improved performance by GDHP.

This article embodies adaptive critic anti-slip control with DHP and multi-layer perceptron (MLP) neural networks. For anti-slip control, it is assumed that the frictional forces at the wheel–road contacts are dependent on surface condition and location. Therefore a driving wheel may purely roll, roll with slip, or spin to challenge the control design. The adaptive critic anti-slip control is shown as able to learn the action performance so as to modify the network parameters and approximate optimal control. Optimal anti-slip control optimises a utility function under variant slips. The adaptive critic anti-slip design is integrated with the velocity control as a single system. A slip model of the robot vehicle is derived. Extensive simulation studies verify the usefulness of the proposed design. Control performance is compared with that using the traditional fuzzy control method [9].

2 Slip model of the robot vehicle and the anti-slip control problem

The robot vehicle under consideration has two passive front wheels and two independent, motorised rear wheels. Fig. 1 shows a schematic top view. Manoeuvring the driving wheels (rear wheels) can change the linear and angular velocities of the vehicle. Such a robot vehicle equipped with stereovision guidance for autonomous navigation has been implemented and demonstrated by Lin *et al*. [10]. Without considering the slip condition, a mathematical model of the vehicle has been derived and verified experimentally in Lin *et al*. [11]. Lin *et al*. [9] have developed a hierarchical fuzzy control system for automatic drive. Using MLP neural networks and DHP, this article presents the adaptive critic anti-slip control system for automatic drive.

doi:10.1049/iet-cta:20050341

Paper first received 13th September 2005 and in revised form 23rd January 2006

The authors are with the Department of Electrical Engineering, National Taiwan University, Taiwan, Republic of China

E-mail: weisong@cc.ee.ntu.edu.tw

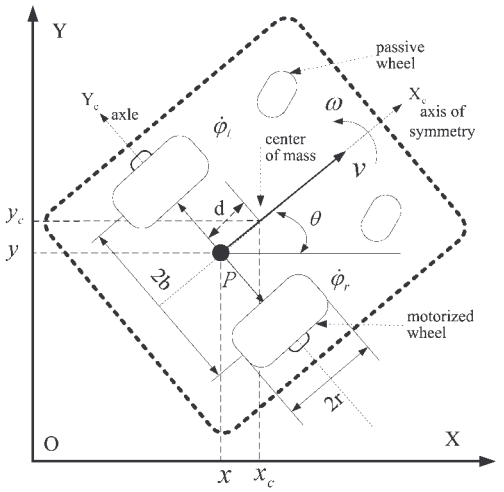

Fig. 1 *Schematic top view of the robot vehicle*

2.1 General model of the robot vehicle

A wheeled robot vehicle is a typical non-holonomic mechanical system [12] and the literature [11, 13–15] has shown that vehicle dynamics can generally be described as

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{F}(\dot{\boldsymbol{q}}) + \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{\tau}_d = \boldsymbol{B}(\boldsymbol{q})\boldsymbol{u} + \boldsymbol{A}^{\mathrm{T}}(\boldsymbol{q})\boldsymbol{\lambda} \tag{1}$$

and constrained by

$$\boldsymbol{A}(\boldsymbol{q})\dot{\boldsymbol{q}} = 0 \tag{2}$$

where $\boldsymbol{q} \in \Re^{n\times 1}$ is the generalised coordinate vector, $\boldsymbol{M}(\boldsymbol{q}) \in \Re^{n\times n}$ is a symmetric, positive definite inertia matrix, $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \in \Re^{n\times n}$ is the centripetal and Coriolis matrix, $\boldsymbol{F}(\dot{\boldsymbol{q}}) \in \Re^{n\times 1}$ denotes the surface friction, $\boldsymbol{G}(\boldsymbol{q}) \in \Re^{n\times 1}$ is the gravitational vector, $\boldsymbol{\tau}_d \in \Re^{n\times 1}$ denotes the bounded unknown disturbance including unstructured, unmodelled dynamics, $\boldsymbol{B}(\boldsymbol{q}) \in \Re^{n\times r}$ represents the input transformation matrix, $\boldsymbol{u} \in \Re^{n\times 1}$ denotes the input torques, $\boldsymbol{A}(\boldsymbol{q}) \in \Re^{m\times n}$ is a full rank matrix associated with the constraints and $\boldsymbol{\lambda} \in \Re^{m\times 1}$ is the Lagrange multiplier or the vector of constraint forces.

Assume $\boldsymbol{Z}(\boldsymbol{q}) \in \Re^{n\times(n-m)}$ is a set of smooth linearly independent vector fields spanning the null space of $\boldsymbol{A}(\boldsymbol{q})$. Then there exists an auxiliary vector time function $\boldsymbol{R}(t) \in \Re^{n\times(n-m)}$, such that for all t

$$\dot{\boldsymbol{q}} = \boldsymbol{Z}(\boldsymbol{q})\boldsymbol{R}(t) \tag{3}$$

where $\boldsymbol{R}$ does not necessarily have any physical significance. With (1), (2) and (3), the following reduced order dynamic model without the Lagrange multiplier is obtained

$$\bar{\boldsymbol{M}}(\boldsymbol{q})\dot{\boldsymbol{R}} + \bar{\boldsymbol{C}}(\boldsymbol{q}, \dot{\boldsymbol{q}})\boldsymbol{R} + \bar{\boldsymbol{F}}(\dot{\boldsymbol{q}}) + \bar{\boldsymbol{G}}(\boldsymbol{q}) + \bar{\boldsymbol{\tau}}_d = \bar{\boldsymbol{B}}(\boldsymbol{q})\boldsymbol{u} \tag{4}$$

where

$$\begin{aligned}
\bar{\boldsymbol{M}}(\boldsymbol{q}) &= \boldsymbol{Z}'(\boldsymbol{q})\boldsymbol{M}(\boldsymbol{q})\boldsymbol{Z}(\boldsymbol{q}) \in \Re^{r\times r} \\
\bar{\boldsymbol{C}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) &= \boldsymbol{Z}'(\boldsymbol{q})(\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{Z}}(\boldsymbol{q}) + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\boldsymbol{Z}(\boldsymbol{q})) \in \Re^{r\times r} \\
\bar{\boldsymbol{F}}(\dot{\boldsymbol{q}}) &= \boldsymbol{Z}'(\boldsymbol{q})\boldsymbol{F}(\dot{\boldsymbol{q}}) \in \Re^{r\times 1} \\
\bar{\boldsymbol{G}}(\boldsymbol{q}) &= \boldsymbol{Z}'(\boldsymbol{q})\boldsymbol{G}(\boldsymbol{q}) \in \Re^{r\times 1} \\
\bar{\boldsymbol{\tau}}_d &= \boldsymbol{Z}'(\boldsymbol{q})\boldsymbol{\tau}_d \in \Re^{r\times 1}
\end{aligned}$$

2.2 Slip model of the robot vehicle

Consider the vehicle shown in Fig. 1 and use the following notations: $\boldsymbol{p} = [x\ y\ \theta]'$ denotes a posture vector; b is the half-width of the axle of the driving wheels; d is the displacement from the point P along the X_c axis to the centre of mass; r is the radius of driving wheels; m_c is the weight of the body (i.e. excluding the driving wheels and their associated rotors); m_w is the mass of a single driving wheel (i.e. taking the associated rotor into account); I_c is the moment of inertia of the body; I_w is the moment of inertia of each driving wheel about the axle; and I_m is the moment of inertia of each driving wheel about a wheel diameter. Assume the vehicle moves by satisfying the following conditions.

Condition 1: The vehicle only moves in the direction normal to the axle of driving wheels (i.e. no lateral motion).

Condition 2: Each driving wheel rolls with slip only in the longitudinal direction.

Then the slip model of the vehicle can be described in the forms of (3) and (4) and the following parameters are obtained:

1. *Kinematics of the slip model:* Conditions 1 and 2 put the following constraints

$$\dot{y}\cos\theta - \dot{x}\sin\theta = 0 \tag{5}$$

$$\dot{x}\cos\theta + \dot{y}\sin\theta - b\dot{\theta} = r\rho_l\dot{\varphi}_l \tag{6}$$

$$\dot{x}\cos\theta + \dot{y}\sin\theta + b\dot{\theta} = r\rho_r\dot{\varphi}_r \tag{7}$$

where ρ_l and ρ_r, $0 < \rho_l, \rho_r \leq 1$ represent the anti-slip factors associated with left and right driving wheels, respectively. Anti-slip factor is defined as the percentage of a wheel's driving force reflected effectively by the road friction. Generally, it is dependent on wheel and road conditions and may vary with locations. When an anti-slip factor varies within [0, 1], the corresponding driving wheel may purely roll, roll with slip, or spin. However, in deriving the slip model, $0 < \rho_l, \rho_r \leq 1$ is assumed.

The linear velocity v and angular velocity ω are computed as

$$v = \dot{x}\cos\theta + \dot{y}\sin\theta, \quad \omega = \dot{\theta} \tag{8}$$

The relationship between $(\dot{\varphi}_l, \dot{\varphi}_r)$ and (v, ω) is obtained by substituting (8) into (6) and (7)

$$r\rho_l\dot{\varphi}_l = v - b\omega, \quad r\rho_r\dot{\varphi}_r = v + b\omega \tag{9}$$

Take the generalised coordinate vector $\boldsymbol{q} = [x, y, \theta, \varphi_l, \varphi_r]^{\mathrm{T}}$ and construct the auxiliary vector $\boldsymbol{R} = [v\ \omega]^{\mathrm{T}}$. The constraints (5)–(7) can be organised in the following matrix form

$$\boldsymbol{A}(\boldsymbol{q}) = \begin{bmatrix} \sin\theta & -\cos\theta & 0 & 0 & 0 \\ \cos\theta & \sin\theta & -b & -r\rho_l & 0 \\ \cos\theta & \sin\theta & b & 0 & -r\rho_r \end{bmatrix} \tag{10}$$

Then the parameter matrix in (3) of the slip model is

$$\boldsymbol{Z}(\boldsymbol{q}) = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \\ \dfrac{1}{r\rho_l} & -\dfrac{b}{r\rho_l} \\ \dfrac{1}{r\rho_r} & \dfrac{b}{r\rho_r} \end{bmatrix} \tag{11}$$

2. *Dynamics of the slip model:* Using Lagrange formalism, the parametric matrices $\boldsymbol{M}$, $\boldsymbol{C}$ and $\boldsymbol{B}$ in (1) of the slip model

are obtained as follows (the derivations are not presented)

$$\boldsymbol{M}(\boldsymbol{q}) = \begin{bmatrix} m & 0 & -m_c d\sin\theta & 0 & 0 \\ 0 & m & m_c d\cos\theta & 0 & 0 \\ -m_c d\sin\theta & m_c d\cos\theta & I & 0 & 0 \\ 0 & 0 & 0 & I_w & 0 \\ 0 & 0 & 0 & 0 & I_w \end{bmatrix}$$

$$\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \begin{bmatrix} 0 & 0 & -m_c d\dot{\theta}\cos\theta & 0 & 0 \\ 0 & 0 & -m_c d\dot{\theta}\sin\theta & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\boldsymbol{B}(\boldsymbol{q}) = \frac{1}{r}\begin{bmatrix} \rho_l\cos\theta & \rho_r\cos\theta \\ \rho_l\sin\theta & \rho_r\sin\theta \\ -\rho_l b & \rho_r b \\ r & 0 \\ 0 & r \end{bmatrix} \tag{12}$$

The unmodelled dynamics and disturbance is

$$\boldsymbol{h}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \boldsymbol{F}(\dot{\boldsymbol{q}}) + \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{\tau}_d \tag{13}$$

Accordingly, the parametric matrices in (4) of the slip model are

$$\bar{\boldsymbol{M}}(\boldsymbol{q}) = \begin{bmatrix} m + \frac{I_w}{r^2}\left(\frac{1}{\rho_l^2}+\frac{1}{\rho_r^2}\right) & \frac{bI_w}{r^2}\left(\frac{-1}{\rho_l^2}+\frac{1}{\rho_r^2}\right) \\ \frac{bI_w}{r^2}\left(\frac{-1}{\rho_l^2}+\frac{1}{\rho_r^2}\right) & I + \frac{b^2 I_w}{r^2}\left(\frac{1}{\rho_l^2}+\frac{1}{\rho_r^2}\right) \end{bmatrix} \tag{14}$$

$$\bar{\boldsymbol{C}}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \begin{bmatrix} \frac{I_w}{r^2\rho_l}\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{1}{\rho_l}\right) + \frac{I_w}{r^2\rho_r}\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{1}{\rho_r}\right) \\ m_c d\dot{\theta} - \frac{bI_w}{r^2\rho_l}\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{1}{\rho_l}\right) + \frac{bI_w}{r^2\rho_r}\frac{d}{\mathrm{d}t}\left(\frac{1}{\rho_r}\right) \end{bmatrix}$$

$$\begin{bmatrix} -m_c d\dot{\theta} - \frac{bI_w}{r^2\rho_l}\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{1}{\rho_l}\right) + \frac{bI_w}{r^2\rho_r}\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{1}{\rho_r}\right) \\ \frac{b^2 I_w}{r^2\rho_l}\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{1}{\rho_l}\right) + \frac{b^2 I_w}{r^2\rho_r}\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{1}{\rho_r}\right) \end{bmatrix} \tag{15}$$

$$\bar{\boldsymbol{B}}(\boldsymbol{q}) = \begin{bmatrix} \frac{1}{r}\left(\rho_l + \frac{1}{\rho_l}\right) & \frac{1}{r}\left(\rho_r + \frac{1}{\rho_r}\right) \\ \frac{-b}{r}\left(\rho_l + \frac{1}{\rho_l}\right) & \frac{b}{r}\left(\rho_r + \frac{1}{\rho_r}\right) \end{bmatrix} \tag{16}$$

The unmodelled term (13) becomes

$$\bar{\boldsymbol{h}}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \boldsymbol{Z}'\boldsymbol{h}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \boldsymbol{Z}'[\boldsymbol{F}(\dot{\boldsymbol{q}}) + \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{\tau}_d] \tag{17}$$

The linear and angular velocities (v, ω) are estimated as below

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} \frac{r}{2} & \frac{r}{2} \\ \frac{-r}{2b} & \frac{r}{2b} \end{bmatrix}\begin{bmatrix} \rho_l\omega_l \\ \rho_r\omega_r \end{bmatrix} \tag{18}$$

where ω_l and ω_r denote the angular velocities of left and right driving wheels respectively. Equations (11), (14)–(16) and (18) show that the slip model is nonlinearly dependent on the anti-slip factors. Furthermore, the anti-slip factors are unknown, unmeasurable and generally vary with the vehicle location and road condition. The anti-slip control problem is stated to find an optimal velocity controller for a system described by the slip model in which the slip is unknown and unmeasurable. For the convenience of computer simulation, we assume that the anti-slip factor is a function $\rho(x, y)$ of simply the vehicle location. Then ρ_l and ρ_r can be calculated as follows

$$\rho_l = \rho(x_l, y_l) = \rho(x + b\sin\theta, y - b\cos\theta) \tag{19}$$

$$\rho_r = \rho(x_r, y_r) = \rho(x - b\sin\theta, y + b\cos\theta) \tag{20}$$

Specifically, when $\rho(x, y) =$ constant, (15) becomes

$$\bar{\boldsymbol{C}}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \begin{bmatrix} 0 & -m_c\mathrm{d}\dot{\theta} \\ m_c\mathrm{d}\dot{\theta} & 0 \end{bmatrix} \tag{21}$$

3 Adaptive critic anti-slip control design

The automatic drive of a wheeled autonomous robot needs basically a navigation system to learn the environment and plan the desired path, a posture controller to infer the desired linear and angular velocities and a velocity controller to ensure that the vehicle drives with the desired linear and angular velocities. The navigation system could request the vehicle to accomplish a task such as avoiding an obstacle or tracking a trajectory, as illustrated in Fig. 2. However, unknown wheel slips challenge severely the velocity control system. They may reduce the control performance or even make the system fail to accomplish the desired task. Adaptive critic anti-slip design aims at optimising the velocity control under variant wheel slips.

In the optimal control context, the control objective is to optimise the primary utility function and Bellman's principle of optimality allows us to take every step as a starting point to look for the optimal solution. No matter the wheel slips are, the objective of the adaptive critic anti-slip control system is to follow the velocity command as closely as possible. Therefore the primary utility function is chosen as

$$U(t) = \sigma_v\{v(t) - v_d(t)\}^2 + \sigma_\omega\{\omega(t) - \omega_d(t)\}^2 \tag{22}$$

where (v, ω) and (v_d, ω_d) are velocities and desired velocities, respectively, and $(\sigma_v, \sigma_\omega)$ are weights to balance the linear and angular velocity errors. The secondary utility function, Bellman's equation and Bellman recursion, is accordingly

$$J(t) = \sum_{k=0}^{\infty}\gamma^k U(t+k) = U(t) + \gamma J(t+1) \tag{23}$$

where $\gamma, 0 < \gamma \leq 1$ discounts the significance of the utility in the future [2]. By satisfying the Bellman equation, the adaptive critic anti-slip control system approximates optimal velocity control.

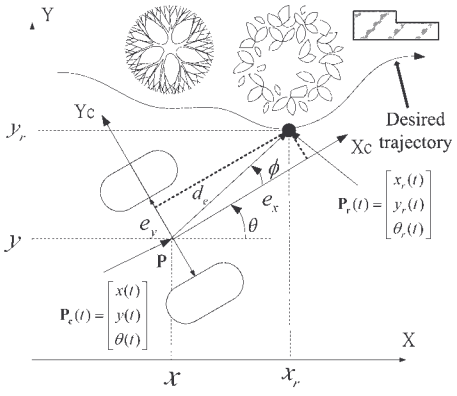

Fig. 2 *Trajectory tracking problem*

3.1 Neural networks and the learning structure

As shown in Fig. 3, the DHP algorithm is implemented with an on-line learning structure of neural networks. This structure consists of the action network, critic network, verification network and vehicle model (not the slip model). The MLP neural network and backpropagation algorithm [16] are chosen to implement the action, critic and verification networks. The architecture of the action network is designed as illustrated in Fig. 4. It has four inputs ($v(t)$, $\omega(t)$, $v_d(t)$, $\omega_d(t)$) and two outputs ($\tau_l(t)$, $\tau_r(t)$) corresponding to the torque commands. The hidden layer has three nodes (neurons) and each with a hyperbolic-tangent activation function. The activation function has a gain value of 30. Computational formulas of the action network are listed as below

$$r_1 = v, \quad r_2 = \omega, \quad r_3 = v_d, \quad r_4 = \omega_d \tag{24}$$

$$f_1(x) = \tanh(x), \quad f_2(x) = 30\tanh(x) \tag{25}$$

$$\text{net}_{j1} = \sum_{k=1}^{4} W_{1(j,k)} r_k + B_{1(j,1)},$$

$$T_j = f_1(\text{net}_{j1}), \quad j = 1, 2, 3 \tag{26}$$

$$\text{net}_{j2} = \sum_{k=1}^{3} W_{2(j,k)} T_k + B_{2(j,1)},$$

$$u_i = f_2(\text{net}_{i2}), \quad i = 1, 2 \tag{27}$$

$$u_1 = \tau_l, \quad u_2 = \tau_r \tag{28}$$

The architecture of the verification network is shown in Fig. 5. It has four inputs ($v(t)$, $\omega(t)$, $v_d(t)$, $\omega_d(t)$), the same as those of the action network. There are three hidden layer nodes, each with a hyperbolic-tangent activation function and a bias. The output node has a linear activation function. The outputs of the verification network are $\lambda_1(t) \cong \partial J(t)/\partial v(t)$ and $\lambda_2(t) \cong \partial J(t)/\partial \omega(t)$. Computational

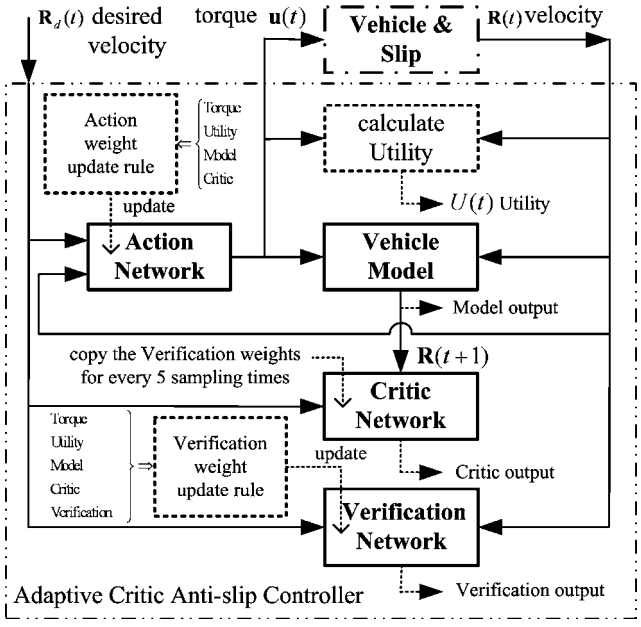

Fig. 3 *Adaptive critic anti-slip control system with the DHP learning algorithm; the action and verification networks each has a weight update rule; the critic network copies weight values from the verification network*

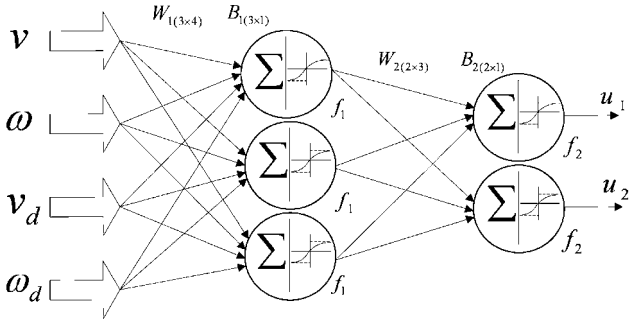

Fig. 4 *Architecture of the action network*

formulas of the verification network are listed as

$$r_1 = v, \quad r_2 = \omega, \quad r_3 = v_d, \quad r_4 = \omega_d \tag{29}$$

$$f_1(x) = \tanh(x), \quad f_2(x) = x \tag{30}$$

$$\text{net}_{j1} = \sum_{k=1}^{4} W_{1(j,k)} r_k + B_{1(j,1)},$$

$$T_j = f_1(\text{net}_{j1}), \quad j = 1, 2, 3 \tag{31}$$

$$\lambda_i = \sum_{k=1}^{3} W_{2(i,k)} T_k + B_{2(i,1)}, \quad i = 1, 2 \tag{32}$$

$$\lambda_1 = \frac{\partial J(t)}{\partial v(t)}, \quad \lambda_2 = \frac{\partial J(t)}{\partial \omega(t)} \tag{33}$$

The critic network is identical to the verification network, except the inputs and outputs. The critic network has ($v(t+1)$, $\omega(t+1)$, $v_d(t)$, $\omega_d(t)$) as inputs, and produces the predictive quantities $\lambda_1(t+1) \cong \partial J(t+1)/\partial v(t+1)$ and $\lambda_2(t+1) \cong \partial J(t+1)/\partial \omega(t+1)$.

The vehicle model predicts the quantities $\boldsymbol{R}(t+1)$, $\partial \boldsymbol{R}(t+1)/\partial \boldsymbol{R}(t)$ and $\partial \boldsymbol{R}(t+1)/\partial \boldsymbol{u}(t)$, where $\boldsymbol{R}(t) = [v(t), \omega(t)]^T$. The vehicle model can be of mathematical or neural/fuzzy type. In this article, the mathematical model is used. The model equation of the vehicle (no wheel slip) is obtained by substituting $\rho_l = \rho_r = 1$ into the slip model. The linearisation, sampled-data form of the model equation with on-line parameter update implements the vehicle model.

3.2 Weight update rules

The backpropagation algorithm and gradient-descent method are adopted to develop the weight update rules of the action and verification networks. The critic network does not have weight update rules but copies weight values from the verification network for every fifth sampling time. As we have the vehicle model to predict the states one-step ahead, the Bellman recursion can

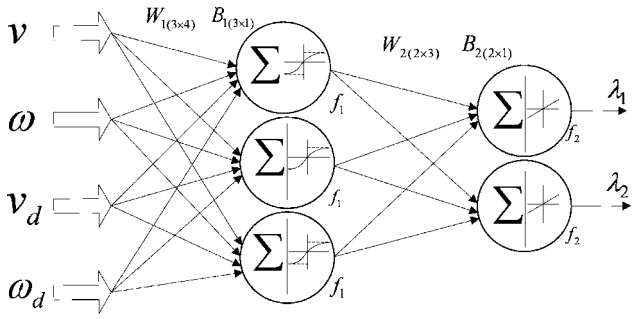

Fig. 5 *Architecture of the verification network*

substitute for the error measure required in the backpropagation algorithm [6]. The weight update rule of the action network is

$$\Delta w_a(t) = -\alpha \frac{\partial J(t)}{\partial w_a(t)} = -\alpha \sum_{j=1}^{2} \frac{\partial J(t)}{\partial u_j(t)} \frac{\partial u_j(t)}{\partial w_a(t)} \quad (34)$$

where α is the learning rate, w_a denotes a weight of the action network

$$\frac{\partial J(t)}{\partial u_j(t)} = \frac{\partial U(t)}{\partial u_j(t)} + \gamma \frac{\partial J(t+1)}{\partial u_j(t)}$$

and where

$$\frac{\partial J(t+1)}{\partial u_j(t)} = \sum_{s=1}^{2} \underbrace{\frac{\partial J(t+1)}{\partial R_s(t+1)}}_{\text{Critic output}} \underbrace{\frac{\partial R_s(t+1)}{\partial u_j(t)}}_{\text{Model output}}$$

The weight update rule of the verification network is obtained by supervised learning through backpropagation. Based on the Bellman recursion, the desired output λ° of the verification network is estimated as follows

$$\lambda_s^\circ(t) = \frac{\partial J(t)}{\partial R_s(t)} = \frac{\partial U(t)}{\partial R_s(t)} + \gamma \frac{\partial J(t+1)}{\partial R_s(t)}, \quad s = 1, 2 \quad (35)$$

where

$$\frac{\partial J(t+1)}{\partial R_s(t)} = \sum_{k=1}^{2} \underbrace{\frac{\partial J(t+1)}{\partial R_k(t+1)}}_{\text{Critic output}} \underbrace{\frac{\partial R_k(t+1)}{\partial R_s(t)}}_{\text{Model output}} + \sum_{k=1}^{2} \sum_{j=1}^{2} \underbrace{\frac{\partial J(t+1)}{\partial R_k(t+1)}}_{\text{Critic output}} \underbrace{\frac{\partial R_k(t+1)}{\partial u_j(t)}}_{\text{Model output}} \frac{\partial u_j(t)}{\partial R_s(t)}$$

Then the error measure is taken as

$$e(t) = \sum_{s=1}^{2} \{\lambda_s(t) - \lambda_s^\circ(t)\}^2 \quad (36)$$

where $\lambda_1(t) = \partial J(t)/\partial v(t)$ and $\lambda_2(t) = \partial J(t)/\partial \omega(t)$ are the verification outputs. The weight update rule is

$$\Delta w_v(t) = -\frac{\eta}{2} \frac{\partial e(t)}{\partial w_v(t)} = -\eta \sum_{s=1}^{2} \left\{ \underbrace{\lambda_s(t)}_{\substack{\text{verification} \\ \text{output}}} - \lambda_s^\circ(t) \right\} \frac{\partial \lambda_s(t)}{\partial w_v(t)} \quad (37)$$

where η is the learning rate, and w_v denotes a weight of the verification network. The training algorithm of the adaptive critic anti-slip control system is summarised in the following steps:

Step 1. Obtain $(v(t), \omega(t), v_d(t), \omega_d(t))$ from the vehicle and posture controller, and apply to the action network to produce the torque command $(\tau_l(t), \tau_r(t))$;
Step 2. Apply the torque command $(\tau_l(t), \tau_r(t))$ to the vehicle;
Step 3. Measure $(v(t), \omega(t))$ and apply $(\tau_l(t), \tau_r(t))$ to run the vehicle model to evaluate $\boldsymbol{R}(t+1)$, $\partial \boldsymbol{R}(t+1)/\partial \boldsymbol{R}(t)$ and $\partial \boldsymbol{R}(t+1)/\partial \boldsymbol{u}(t)$;
Step 4. Apply $(v(t+1), \omega(t+1), v_d(t), \omega_d(t))$ to the critic network to obtain $\lambda(t+1)$, and apply $(v(t), \omega(t), v_d(t), \omega_d(t))$ to the verification network to obtain $\lambda(t)$;
Step 5. Calculate the desired output $\lambda^\circ(t)$ of the verification network;
Step 6. Calculate weight updates of the action network and update the weights according to (34);
Step 7. Calculate weight updates of the verification network and update the weights according to (37);
Step 8. The critic network copies the weight values of the verification network for every fifth sampling time.

4 Simulation results

The vehicle parameters in Lin *et al.* [10] are listed in Table 1 and adopted in the following computer simulations. Anti-slip factor is expressed as a known function $\rho(x, y)$. For comparison, the posture controller has a fuzzy logic design, the same as in Lin *et al.* [9]. It accepts position error (e_x, e_y) as inputs to produce the desired linear and angular velocities (v_d, ω_d).

4.1 Simulation 1: Self-learning from scratch

Ability of learning from scratch means updating to the neural weights can begin and continue without human help. This is essential in an autonomous robot. In this simulation, all the neural weights are initialised randomly in the range $[-0.1, 0.1]$. The primary utility function is (22) with $(\sigma_v, \sigma_\omega) = (0.25, 0.25)$. The discount factor in (23) is $\gamma = 1$. The velocity commands are $v_d = 2 \sin\{(\pi/600)kT\}$, $\omega_d = 1.5 \cos\{(\pi/1200)kT\}$. The sampling interval is $T = 0.01$ s. The learning process takes 10 000 sampling times. The results show that the linear and angular velocity errors decrease quickly along with increase in sampling times. The weight values in the action and verification networks converge after training for 3000 sampling times. Thereafter the system demonstrates good velocity tracking.

After the self-learning from scratch, the vehicle is commanded to drive on roads with variant slip conditions. The road is divided into five zones and each zone has a specified anti-slip factor as shown in Figs. 6 and 7. Three roads are studied in the following simulations:

Road 1: Anti-slip factors in all zones equal 1 (no slip).
Road 2: In zones 2 and 4, the anti-slip factor equals 0.6 and in other zones equals 1.
Road 3: In zones 1, 3 and 5, the anti-slip factors equal 0.6 and in other zones equals 1.

The desired trajectory attempts to lead the vehicle to make a left turn. The results obtained from the adaptive critic anti-slip control are compared with those of the hierarchical fuzzy control [9].

Table 1: Mechanical figures of the vehicle

b, m	d, m	r, m	w_c, m	m_c, kg	m_w, kg	I_c, kg m^2	I_w, kg m^2	I_m, kg m^2
0.265	0.1	0.125	0.8	110	5	1.057	0.004	0.002

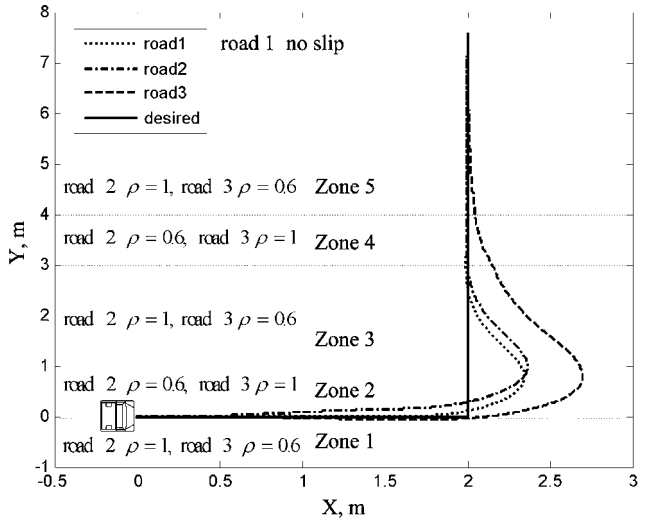

Fig. 6 *Left-turn trajectories with the fuzzy velocity control in simulation 2*

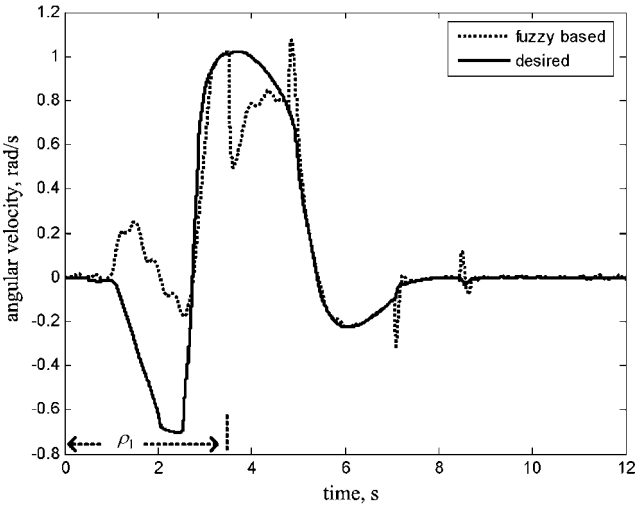

Fig. 8 *Angular velocity of road 2 case with the fuzzy velocity control in simulation 3*

4.2 Simulation 2: Trajectory tracking under variant wheel slips

Fig. 6 shows the left-turn trajectories obtained by applying the hierarchical fuzzy control to drive the vehicle on roads 1, 2 and 3 respectively. In spite of maintaining stable moving, the tracking errors vary in different roads and the maximum distance between two trajectories is as large as 1.08 m. This reveals that fuzzy control can handle unknown, nonlinear dynamics to obtain stable control but lacks an ability to maintain it.

In the adaptive critic anti-slip control, the neural weights obtained in simulation 1 are used as initial values and then the learning and control begins. Fig. 7 shows the left-turn trajectories of the vehicle driving on roads 1, 2 and 3, respectively. The results show that the responsive trajectories in all three roads are very close. The maximum distance between two trajectories is 0.23 m, much smaller than that of the hierarchical fuzzy control. This confirms that the adaptive critic anti-slip control can comply with changes in the wheel slip to approximate optimal control.

4.3 Simulation 3: Velocity response under variant wheel slips

According to the slip conditions, each trajectory in Figs. 6 and 7 is divided into five segments. For road 2 case, the five segments from the beginning to the end of a trajectory have slip values (ρ_l, ρ_r) as $\rho 1 = (0.6, 1.0)$, $\rho 2 = (0.6, 0.6)$, $\rho 3 = (1.0, 1.0)$, $\rho 4 = (0.6, 0.6)$, $\rho 5 = (1.0, 1.0)$, respectively. In segment $\rho 1$, left and right driving wheels have unequal anti-slip factors. This difference disturbs mainly the angular velocity control. Examining segment $\rho 1$ in Fig. 8, it is found that the fuzzy velocity control leaves large errors uncorrected, as no slip is assumed. In contrast, segment $\rho 1$ in Fig. 9 shows the adaptive critic anti-slip control can adapt the action parameters to correct the error quickly.

In segments $\rho 2$ to $\rho 5$, the anti-slip factors at the left and right driving wheels are equal, but values change from segment to segment. When the vehicle goes from one segment to another, changes in the anti-slip factors disturb mainly the linear velocity control. Fig. 10 shows that at the beginning of each $\rho 2$ to $\rho 5$ segment, large linear velocity

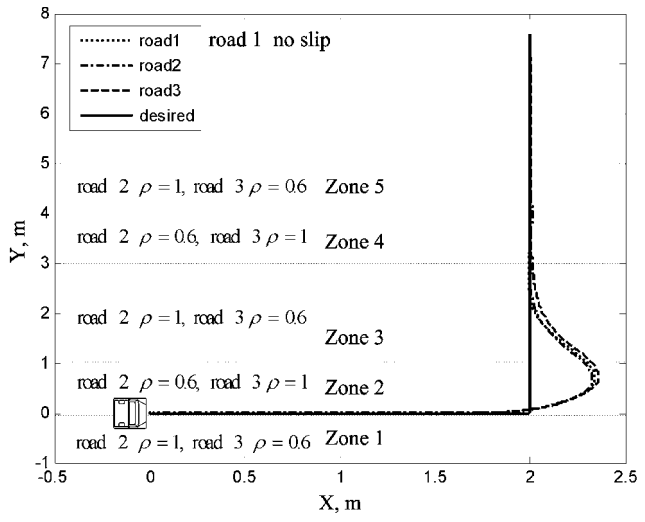

Fig. 7 *Left-turn trajectories with the adaptive critic anti-slip control in simulation 2*

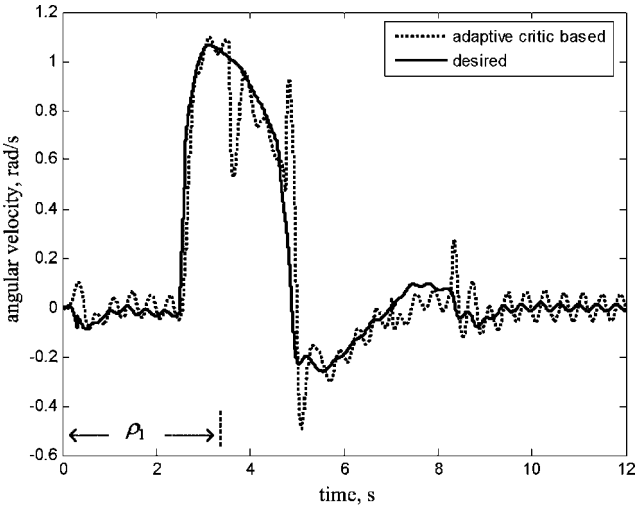

Fig. 9 *Angular velocity of road 2 case with the adaptive critic anti-slip control in simulation 3*

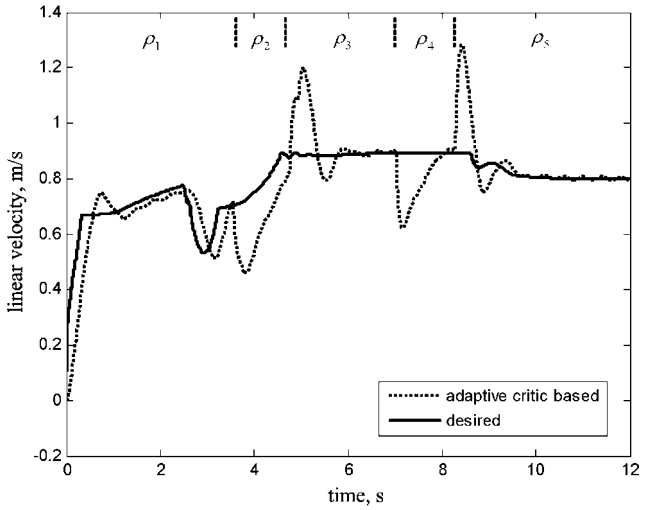

Fig. 10 *Linear velocity of road 2 case with the adaptive critic anti-slip control in simulation 3*

error presents. But the error dies out very soon after the action parameters being improved by the DHP learning algorithm. In other words, the adaptive critic anti-slip control system can surely comply with changes in the wheel slip.

5 Conclusion

A wheeled autonomous robot may encounter road conditions resulting in wheel slips. The wheel slip modifies the commanded forces unpredictably and challenges the accuracy and stability of the motion control. Without appropriate anti-slip control, the robot may lose tracking the desired trajectory. In this context, it has been shown that the DHP adaptive critic design enabled the robot control system to adjust its control parameters automatically by learning to satisfy the Bellman equation. The DHP adaptive critic design was implemented with an MLP neural network structure to achieve anti-slip velocity control. The resulting system was demonstrated to be able to improve the control performance significantly under variant wheel slips. In contrast, in spite of stable control, traditional fuzzy velocity control was shown to be unable to maintain high performance under such conditions. Ideally, the DHP adaptive critic design aimed at learning and control to satisfy the Bellman equation. But in practice, the parameters in the action network were updated merely a small step for each sampling time to obtain learning convergence. Therefore optimal control obtained only in the steady-state situation, in which the control environment did not change and the action parameters reached stable values. Otherwise, the DHP adaptive critic design simply kept on improving the action parameters and no optimal control was guaranteed in the immediate sampling time.

6 Acknowledgments

The financial support for this research from the National Science Council of Taiwan, Republic of China under grants NSC93-2218-E002-106 and NSC94-2218-E002-049 is gratefully acknowledged.

7 References

1 Werbos, P.J.: 'Approximate dynamic programming for real-time control and neural modeling' in White, D.A., and Sofge, D.A. (Eds.): 'Handbook of intelligent control: neural, fuzzy and adaptive approaches' (Van Nostrand Reinhold, New York, NY, USA, 1992), pp. 493–525
2 Bellman, R.E.: 'Dynamic programming' (Princeton University Press, 1957)
3 Prokhorov, D., and Wunsch, D.: 'Adaptive critic designs', *IEEE Trans. Neural Netw.*, 1997, **8**, pp. 997–1007
4 Prokhorov, D.L., and Feldkamp, L.: 'Analyzing for Lyapunov stability with adaptive critics', *IEEE Trans. Syst. Man Cybern.*, 1998, **2**, pp. 1658–1661
5 Lendaris, G., Paintz, C., and Shannon, T.: 'More on training strategies for critic and action neural nets in dual heuristic programming method'. Proc. IEEE Conf. on Systems, Man and Cybernetics, Orlando, October 1997, pp. 3067–3072
6 Lendaris, G., and Shannon, T.: 'Application considerations for the DHP methodology'. Proc. IEEE Int. Joint Conf. on Neural Networks, Anchorage, AK, 1998, pp. 1013–1018
7 Prokhorov, D., Santiago, R., and Wunsch, D.: 'Adaptive critic designs: a case study for neurocontrol', *Neural Netw.*, 1995, **8**, pp. 1367–1372
8 Venayagamoorthy, G.K., Harley, R.G., and Donald, C.W.D.C.: 'Comparison of heuristic dynamic programming and dual heuristic programming adaptive critics for neurocontrol of a turbogenerator', *IEEE Trans. Neural Netw.*, 2002, **13**, (3), pp. 764–773
9 Lin, W.-S.L., Huang, C.-L., and Chuang, M.-K.: 'Hierarchical fuzzy control for autonomous navigation of wheeled robots', *IEE Proc., Control Theory Appl.*, 2005, **152**, (5), pp. 598–606
10 Lin, W.-S.L., Chuang, M.-K., and Tien, G.: 'Autonomous mobile robot navigation using stereovision'. Proc. IEEE Int. Conf. on Mechatronics, Taipei, Taiwan, 2005, pp. 410–415
11 Lin, W.-S.L., Huang, C.-L., Chuang, M.-K., and Liu, G.-C.: 'Modeling a wheeled mobile robot for autonomous navigation design'. Proc. IASTED Int. Conf. on Modeling, Identification and Control, Grindelwald, Switzerland, February 2004, pp. 275–280
12 Greenwood, D.T.: 'Principles of dynamics' (Prentice-Hall, 1988)
13 Tsai, P.-S., Wu, T.-F., Chang, F.-R., and Wang, L.-S.W.: 'Tracking control of nonholonomic mobile robot using hybrid structure'. Presented at 6th World Multiconf. on Systemics, Cybernetics and Informatics, Orlando, Florida, 2002
14 Yun, X., and Yamamoto, Y.: 'Internal dynamics of a wheeled mobile robot'. Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, 1993, pp. 1288–1293
15 Lewis, F.L., Abdallah, C.T., and Dawson, D.M.: 'Control of robot manipulators' (MacMillan, New York, 1993)
16 Haykin, S.: 'Neural networks: a comprehensive foundation' (Prentice-Hall International, Inc., 1990), Ch. 4