Neurocomputing 88 (2012) 54–66

Contents lists available at SciVerse ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Indirect adaptive tracking control of a nonholonomic mobile robot via neural networks

Omid Mohareri [a,*], Rached Dhaouadi [b], Ahmad B. Rad [c]

[a] Electrical and Computer Engineering Department, University of British Columbia, Vancouver, Canada
[b] College of Engineering, American University of Sharjah, Sharjah, United Arab Emirates
[c] School of Engineering Science, Simon Fraser University, Surrey, Canada

ARTICLE INFO

Available online 8 March 2012

Keywords:
Neural-adaptive tracking control
Backstepping control
Neural networks
Wheeled mobile robots

ABSTRACT

This paper presents the design and implementation of a novel adaptive trajectory tracking controller for a nonholonomic wheeled mobile robot (WMR) with unknown parameters and uncertain dynamics. The learning ability of neural networks is used to design a robust adaptive backstepping controller that does not require the knowledge of the robot dynamics. The kinematic controller gains are tuned on-line to minimize the velocity error and improve the trajectory tracking characteristics. The performance of the proposed control algorithm is verified and compared with the classical backstepping controller through simulation and experiments on a commercial mobile robot platform.

1. Introduction

The motion control of nonholonomic systems has received considerable attention over the last few years. Nonholonomic systems are found in different applications ranging from unicycles and car-like vehicles, possibly equipped with trailers, to systems like rolling spheres, snake-like robots, snake boards, roller racers, and wheelchairs. A wheeled mobile robot (WMR) is one of the well-known systems with nonholonomic constraints, and several papers have addressed its tracking control problem [1–5]. The challenge in the motion control of this system as illustrated by Brockett's theorem [6] is the nonexistence of pure-state feedback for asymptotic stabilization of fixed configurations.

In recent years, there has been a growing interest in the design and development of advanced feedback-control laws for mobile robots. To achieve high-precision path tracking control for the WMR, many sophisticated control approaches have been proposed in the last decade [7–10]. These methods can be divided into two research paradigms based on whether the WMR is described by a kinematic model or a dynamic model. The tracking-control problem is therefore classified as either a kinematic or a dynamic tracking-control problem. The purpose of the kinematic controller is to produce velocity outputs for the robot to make the tracking error between the real and reference trajectories converge to zero. A torque controller is next designed based on the system dynamics such that the velocity of the mobile robot converges to the generated desired velocity. The backstepping controller [11] is one of the earliest stable kinematic level controllers. In order to improve the overall performance of the backstepping controller and to guarantee the asymptotic convergence of the robot with different reference trajectories, adaptive kinematic level controllers have been reported in [12–14]. Other class of adaptive kinematic controllers employed fuzzy controllers [15,16], and QUOTE methods [17] to deal with the unknown system parameters and disturbances. There are also other kinematic control schemes presented in [1,4,18,19], which would not fall in the category of adaptive and backstepping controllers but provide stable tracking control structures dealing only with robot velocities.

The dynamic level controllers can be classified into four groups based on how the system dynamics is treated: Adaptive controllers, robust controllers, Robust Adaptive controllers and feedback linearization controllers. The dynamic adaptive controllers in [2,8,20–25], are an adaptive extension to the kinematic backstepping controller to deal with the unknown parameters. The dynamic robust controllers are designed to deal with uncertainties and disturbances [5,26] and use sliding mode [9,27,28] and H_∞ [29] techniques. There are also some proposed controllers which have both robust and adaptive properties and use a combination of an intelligent controller and sliding mode [30–32]. The last group of the dynamic controllers is the feedback linearization control scheme [7,33] to compensate for the system nonlinear dynamics. Some methods integrating both a kinematic controller and a torque controller have been also presented [9].

Due to their inherent parallelism and learning capabilities, neural network control techniques have also been very popular in robotic motion control systems [34–37]. This has led to

* Corresponding author.
E-mail addresses: omidm@ece.ubc.ca (O. Mohareri), rdhaouadi@aus.edu (A.B. Rad).

0925-2312/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2011.06.035

substantial research efforts to applying artificial neural networks to control systems dealing with the problems of nonlinearity and uncertainty of system parameters. Artificial Neural Networks (ANN) proved to be very successful in robot motion control [15]. The fundamental characteristics of neural networks are their ability to produce good models of nonlinear systems; their highly distributed and paralleled structure, which makes neural based control schemes faster than traditional algorithms; their simple implementation by software or hardware; and their ability to learn and adapt themselves to the behavior of any real process.

The motivation of this work has been to take advantage of attractive properties of ANN to address the problem of trajectory tracking control of mobile robot systems. Other researchers have already made contributions in both areas of kinematic and dynamic tracking [32–40]. In the kinematic level, controllers that combine the backstepping scheme with neural networks and GA techniques to improve its overall performance are proposed in [33,41]. In the dynamic level, robust neural network controller are proposed in [10,42–44] and adaptive controllers that integrate neural networks in [45–52], wavelet neural networks in [53,54], and a combination of fuzzy and neural network methods in [55,56], to the existing kinematic based controller in order to deal with unknown dynamics in an adaptive way are proposed in literature. The online learning ability of neural networks has been mostly used to construct the actual robot dynamics and to compensate for the nonlinear dynamics of the system in a backstepping structure.

In this paper, a new ANN-based adaptive backstepping controller is proposed, which tunes the gains of the backstepping controller online according to the robot reference trajectory and initial posture. In the proposed method, the neural network learns the dynamic model of the mobile robot and determines the future inputs that minimize the error performance index so as to compensate the backstepping controller gains. This approach retains the good features of the backstepping kinematic controller while introducing an adaptive scheme to cope with system uncertainties and unknown dynamics.

The paper is organized as follows. In Section 2, the differential drive mobile robot kinematics and dynamics are presented. The kinematic based controller, the backstepping methodology, derivation and details of the neural network based adaptive backstepping controller are presented in Section 3. The simulation results and a comparison with a dynamic based adaptive neural network controller are discussed in Section 4. The experimental verification of the proposed control algorithm and a comparative analysis is presented in Section 5.

2. Nonholonomic mobile robot kinematics and dynamics

The modeling of a differential drive mobile robot consists of three steps, namely kinematic modeling, dynamic modeling, and actuators modeling. Kinematic modeling deals with the geometric relationships that govern the system and studies the mathematics of motion without considering the affecting forces [4,7,58]. Dynamic modeling on the other hand is the study of the motion in which forces and energies are modeled and included. Actuator modeling is required to find the relationship between the control signal and the mechanical system's input. Consider a differential drive mobile robot which has two wheels, with radius R, placed at a distance L from the robot center as shown in Fig. 1. The center of mass of the platform is assumed to be located at a distance a from the center of the driving wheels axis. (A,x_r,y_r) is the reference frame fixed to the mobile robot and (o,x_I,y_I) is the global inertial reference frame. The robot platform has a total mass m and a moment of inertia I_c around its center of mass.

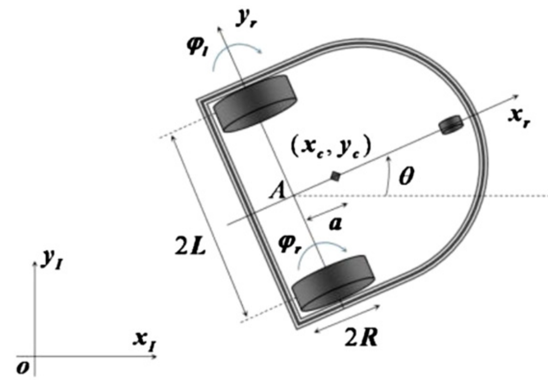

Fig. 1. Differential-drive mobile robot model and reference frames.

The configuration of the robot can be described by five generalized coordinates

$$q=[x_c \quad y_c \quad \theta \quad \varphi_r \quad \varphi_l]^T \tag{1}$$

where (x_c,y_c) are the coordinates of the center of mass in the inertial reference frame, θ is the heading angle of the robot, and (φ_r,φ_l) are the angles of the right and left driving wheels.

The translational velocity u and rotational velocity ω of the platform in the local frame are found using the velocities of the right and left driving wheels

$$u=\frac{R}{2}(\dot{\varphi}_r+\dot{\varphi}_l), \tag{2}$$

$$\omega=\dot{\theta}=\frac{R}{2L}(\dot{\varphi}_r-\dot{\varphi}_l). \tag{3}$$

The nonholonomic constraints of the differential drive wheeled mobile robot (WMR) shown in Fig. 1 are used to represent the practical conditions of no lateral slip and pure wheel rolling. The no lateral slip constraint is given by

$$\dot{y}_c\cos\theta-\dot{x}_c\sin\theta-a\dot{\theta}=0, \tag{4}$$

where $\dot{y}_c$ and $\dot{x}_c$ are the robot's center of mass velocity components in the inertial frame. This constraint means that the velocity of the robot center point A will be in the direction of the axis of symmetry and the motion along the orthogonal axis will be zero. The pure rolling constraint is expressed by two equations

$$\dot{x}_c\cos\theta+\dot{y}_c\sin\theta+L\dot{\theta}=R\dot{\varphi}_r, \tag{5}$$

$$\dot{x}_c\cos\theta+\dot{y}_c\sin\theta-L\dot{\theta}=R\dot{\varphi}_l. \tag{6}$$

This constraint shows that the driving wheels do not slip. The three nonholonomic constraints can be written in the following form:

$$A(q)\dot{q}=0, \tag{7}$$

where

$$A(q)=\begin{bmatrix}-\sin\theta & \cos\theta & -a & 0 & 0\\ \cos\theta & \sin\theta & L & -R & 0\\ \cos\theta & \sin\theta & -L & 0 & -R\end{bmatrix} \tag{8}$$

$$\dot{q}=[\dot{x}_c \quad \dot{y}_c \quad \dot{\theta} \quad \dot{\varphi}_r \quad \dot{\varphi}_l]^T. \tag{9}$$

The energy-based Lagrangian approach can be used to derive the dynamic model of the mobile robot which is represented in the following general form [9]:

$$M(q)\ddot{q}+V_m(q,\dot{q})\dot{q}+F(\dot{q})+G(q)+\tau_d=B(q)\tau-A^T(q)\lambda \tag{10}$$

where $M(q)$ is the symmetric positive definite inertia matrix, $V(q,\dot{q})$ is the centripetal and coriolis matrix, $F(\dot{q})$ is the surface friction matrix, $G(q)$ is the gravitational vector, τ_d denoted bounded unknown disturbances including unstructured unmodeled dynamics, $B(q)$ is the input transformation matrix, τ is the input vector, $A^T(q)$ is the matrix associated with the constraints and λ is the vector of the constraint forces.

The robot planar motion leads to the elimination of the gravity terms in the dynamic equation:

$$G(q)=0 \tag{11}$$

In addition, with the assumption of having negligible friction in the system we can have also

$$F(\dot{q})=0 \tag{12}$$

The above system can be transformed into a more proper representation for control and simulation purposes. In this transformation, we are trying to find a way to eliminate the constraint term from the equation. The forward kinematic equation is defined by the following transformation:

$$\dot{q}=\begin{bmatrix}\dot{x}_c\\ \dot{y}_c\\ \dot{\theta}\\ \dot{\varphi}_r\\ \dot{\varphi}_l\end{bmatrix}=S(q)v(t)=\begin{bmatrix}\cos\theta & -a\sin\theta\\ \sin\theta & a\cos\theta\\ 0 & 1\\ \frac{1}{R} & \frac{L}{R}\\ \frac{1}{R} & -\frac{L}{R}\end{bmatrix}\begin{bmatrix}u\\ \omega\end{bmatrix}, \tag{13}$$

where u is the longitudinal velocity of the robot along the axis of symmetry and $S(q)$ is the forward kinematic matrix.

It can be proved that the $S(q)$ matrix has the following relation with the $A(q)$ matrix:

$$S^T(q)A^T(q)=0. \tag{14}$$

The above equation is useful to eliminate the constraint term from the main dynamic equation as shown in the following procedure. Differentiating Eq. (13), we have

$$\ddot{q}=\dot{S}(q)v(t)+S(q)\dot{v}(t) \tag{15}$$

Substituting the above equation in (10) will result in the following equation:

$$M(q)[\dot{S}(q)v(t)+S(q)\dot{v}(t)]+V_m(q,\dot{q})S(q)v(t)+\tau_d=B(q)\tau-A^T(q)\lambda \tag{16}$$

$$M(q)\dot{S}(q)v(t)+M(q)S(q)\dot{v}(t)+V_m(q,\dot{q})S(q)v(t)+\tau_d=B(q)\tau-A^T(q)\lambda \tag{17}$$

The next step to eliminate the constraint matrix $A^T(q)\lambda$ is to multiply Eq. (17) by $S^T(q)$ as follows:

$$\begin{aligned}&[S^T(q)M(q)S(q)]\dot{v}(t)+[S^T(q)M(q)\dot{S}(q)+S^T(q)V_m(q,\dot{q})S]v(t)+S^T(q)\tau_d\\ &\quad=S^T(q)B(q)\tau-S^T(q)A^T(q)\lambda\end{aligned} \tag{18}$$

As it can be seen from the above equation, the term $S^T(q)A^T(q)$ multiplying the unknown Lagrange coefficients λ will be canceled

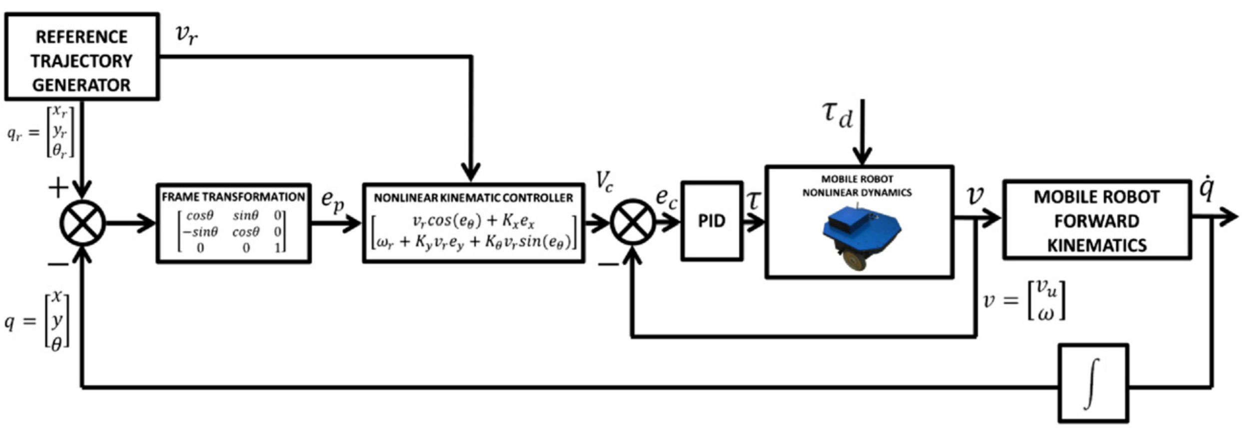

Fig. 2. Backstepping controller structure.

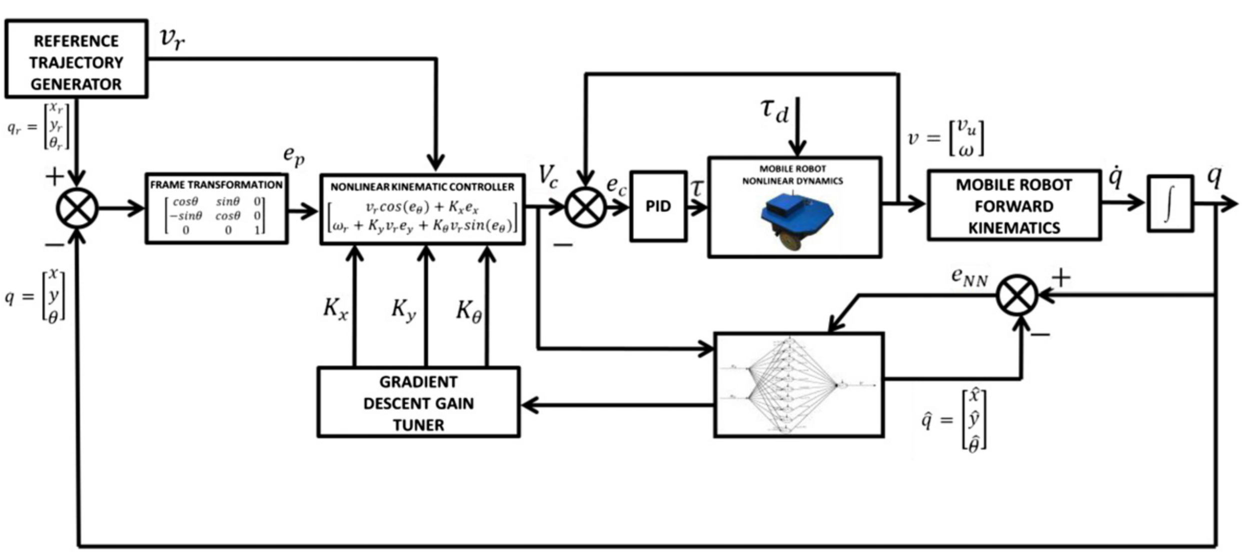

Fig. 3. NN-based adaptive backstepping controller structure.

resulting in the new dynamic equation

$$[S^T(q)M(q)S(q)]\dot{\nu}(t)+[S^T(q)M(q)\dot{S}(q)+S^T(q)V_m(q,\dot{q})S]\nu(t) + S^T(q)\tau_d = S^T(q)B(q)\tau \tag{19}$$

By the following appropriate definitions we can rewrite the above equation as follows:

$$\overline{M}(q)\dot{\nu}(t)+\overline{V}_m(q,\dot{q})\nu(t)+\overline{\tau}_d=\overline{B}(q)\tau \tag{20}$$

where

$$\overline{M}(q)=S^T(q)M(q)S(q) \tag{21}$$

$$\overline{V}_m(q,\dot{q})=S^T(q)M(q)\dot{S}(q)+S^T(q)V_m(q,\dot{q})S \tag{22}$$

$$\overline{\tau}_d=S^T(q)\tau_d \tag{23}$$

$$\overline{B}(q)=S^T(q)B(q) \tag{24}$$

The above equations represent the dynamic equations of the robot considering the nonholonomic constraints and they can easily be transformed to the following simplified matrix equations:

$$\begin{bmatrix} m & 0 \\ 0 & ma^2+I_C \end{bmatrix}\begin{bmatrix} \dot{u} \\ \ddot{\theta} \end{bmatrix}+\begin{bmatrix} 0 & -ma\dot{\theta} \\ ma\dot{\theta} & 0 \end{bmatrix}\begin{bmatrix} u \\ \dot{\theta} \end{bmatrix}=\frac{1}{R}\begin{bmatrix} 1 & 1 \\ L & -L \end{bmatrix}\begin{bmatrix} \tau_r \\ \tau_l \end{bmatrix} \tag{25}$$

3. Tracking controller design

The kinematic based controller or the so called backstepping controller is a stable tracking control rule for a nonholonomic mobile robot [10]. The controller structure is shown in Fig. 2. The input error to this controller is defined as

$$e_p=\begin{bmatrix} e_x \\ e_y \\ e_\theta \end{bmatrix}=\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x_r-x \\ y_r-y \\ \theta_r-\theta \end{bmatrix}=T_e e_r \tag{26}$$

The control law that calculates the robot velocity vector ν_c is

$$\nu_c=\begin{bmatrix} \nu_r\cos(e_\theta)+K_x e_x \\ \omega_r+K_y\nu_r e_y+K_\theta\nu_r\sin(e_\theta) \end{bmatrix}$$

$$\nu_c=f(e_p,\nu_r,K), \quad K=(K_x,K_y,K_\theta) \tag{27}$$

The additional PID control loop is required to make sure that the robot velocity follows the input reference velocity. The torques of the wheels are adjusted according to the following equation:

$$\tau=K_P e_c+K_D\frac{de_c}{dt}, \tag{28}$$

where

$$K_P=\begin{bmatrix} K_{Pr} & 0 \\ 0 & K_{Pl} \end{bmatrix} \tag{29}$$

As it can be seen in the above equations, the controller has three gains which are defined to be positive constant values.

The main advantage of the backstepping kinematic controller is its simplicity and practical application since it relies only on the kinematic model. Other techniques such as state feedback linearization, sliding-mode control, or conventional backstepping control, require knowledge of the dynamic model and their hardware implementations are not straightforward. Moreover, the performance of the standard backstepping kinematic controller with constant gains is poor and requires careful gain tuning for each reference trajectory and it does not give zero trajectory error with smooth tracking.

The proposed control algorithm enhances the backstepping controller with adaptive gains which are variable and change according to the reference trajectory. The Neural Network (NN) based adaptive backstepping controller structure is shown in Fig. 3. The proposed control structure adapts the kinematic based controller gains to minimize the following cost function:

$$F=\frac{1}{2}\sum\gamma_x e_x^2+\gamma_y e_y^2+\gamma_\theta e_\theta^2 \tag{30}$$

The kinematic model based controller gains are considered part of the above cost function and are optimized and updated according to the gradient descent method. The kinematic controller gains are represented by the set $\alpha=[K_x\ K_y\ K_\theta]$. The partial derivative of the cost function with respect to α is

$$\frac{\partial F}{\partial\alpha}=\gamma_x e_x\frac{\partial e_x}{\partial\alpha}+\gamma_y e_y\frac{\partial e_y}{\partial\alpha}+\gamma_\theta e_\theta\frac{\partial e_\theta}{\partial\alpha}=e_p^T\left(\Gamma\frac{\partial e_p}{\partial\alpha}\right) \tag{31}$$

where

$$\Gamma=\begin{bmatrix} \gamma_x & 0 & 0 \\ 0 & \gamma_y & 0 \\ 0 & 0 & \gamma_\theta \end{bmatrix}$$

Substituting Eq. (31) in the above equation, we have

$$e_p=\begin{bmatrix} e_x \\ e_y \\ e_\theta \end{bmatrix}=T_e(q_r-q)$$

$$\frac{\partial F}{\partial\alpha}=e_p^T\Gamma\frac{\partial(T_e(q_r-q))}{\partial\alpha}=-e_p^T\Gamma T_e\frac{\partial q}{\partial\alpha}. \tag{32}$$

Using the chain rule, the derivative $(\partial q/\partial\alpha)$ can be written as

$$\frac{\partial q}{\partial\alpha}=\frac{\partial q}{\partial\nu_c}\frac{\partial\nu_c}{\partial\alpha}. \tag{33}$$

The first derivative in the above equation $(\partial q/\partial\nu_c)$ is defined as the Jacobian matrix with respect to the system velocity inputs

$$\frac{\partial q}{\partial\nu_c}=J=\begin{bmatrix} \frac{\partial x}{\partial\nu_c} & \frac{\partial x}{\partial\omega_c} \\ \frac{\partial y}{\partial\nu_c} & \frac{\partial y}{\partial\omega_c} \\ \frac{\partial\theta}{\partial\nu_c} & \frac{\partial\theta}{\partial\omega_c} \end{bmatrix}=\begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \\ J_{31} & J_{32} \end{bmatrix}. \tag{34}$$

The second derivative in Eq. (33) $(\partial\nu_c/\partial\alpha)$ is calculated as follows:

$$\frac{\partial\nu_c}{\partial\alpha}=\begin{bmatrix} \frac{\partial\nu_c}{\partial K_x} & \frac{\partial\nu_c}{\partial K_y} & \frac{\partial\nu_c}{\partial K_\theta} \\ \frac{\partial\omega_c}{\partial K_x} & \frac{\partial\omega_c}{\partial K_y} & \frac{\partial\omega_c}{\partial K_\theta} \end{bmatrix}=\begin{bmatrix} e_x & 0 & 0 \\ 0 & \nu_r e_y & \nu_r\sin(e_\theta) \end{bmatrix}. \tag{35}$$

Therefore, the required derivative in Eq. (33) can be found by substituting Eqs. (34) and (35) in (33)

$$\frac{\partial q}{\partial\alpha}=J\begin{bmatrix} e_x & 0 & 0 \\ 0 & \nu_r e_y & \nu_r\sin(e_\theta) \end{bmatrix} \tag{36}$$

The derivative of the cost function with respect to the controller gains $(\partial F/\partial\alpha)$ is

$$\frac{\partial F}{\partial\alpha}=-e_p^T\Gamma T_e J\begin{bmatrix} e_x & 0 & 0 \\ 0 & \nu_r e_y & \nu_r\sin(e_\theta) \end{bmatrix}=\begin{bmatrix} \frac{\partial J}{\partial K_x} & \frac{\partial J}{\partial K_y} & \frac{\partial J}{\partial K_\theta} \end{bmatrix} \tag{37}$$

Therefore, the kinematic controller gains will change and adapt to make the cost function zero according to the gradient descent algorithm

$$K_x(n)=K_x(n-1)-\eta_x\frac{\partial F}{\partial K_x}, \tag{38}$$

$$K_y(n)=K_y(n-1)-\eta_y\frac{\partial F}{\partial K_y}, \tag{39}$$

$$K_\theta(n)=K_\theta(n-1)-\eta_\theta\frac{\partial F}{\partial K_\theta}, \tag{40}$$

where n is the iteration index and η_x, η_y, and η_θ are the learning rates of the gradient descent algorithm.

The major part in the derivation of Eq. (37) is the procedure to compute the Jacobian matrix of the system. The Jacobian matrix can be calculated using the exact equations of the system or can be provided by the neural network direct model.

3.1. Jacobian calculation using the systems exact equations

The Jacobian matrix can be calculated using the robots governing dynamic and kinematic equations. The derivative ($\partial q/\partial v_c$) can be expanded as follows:

$$J=\frac{\partial q}{\partial v_c}=\frac{\partial q}{\partial v}\frac{\partial v}{\partial \tau}\frac{\partial \tau}{\partial e_c}\frac{\partial e_c}{\partial v_c} \tag{41}$$

To calculate the first derivative in the above equation, we use the robots kinematic equation as follows:

$$\dot{q}=\begin{bmatrix}\dot{x}_c\\ \dot{y}_c\\ \dot{\theta}\end{bmatrix}=\begin{bmatrix}\cos\theta & -a\sin\theta\\ \sin\theta & a\cos\theta\\ 0 & 1\end{bmatrix}\begin{bmatrix}u\\ \omega\end{bmatrix}=Sv. \tag{42}$$

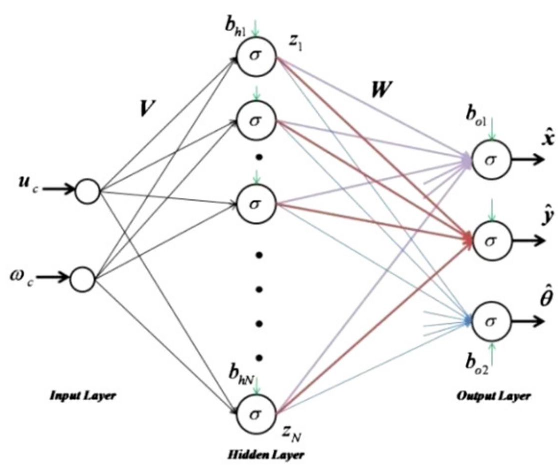

Fig. 4. Feedforward neural network structure used to model the mobile robot dynamics.

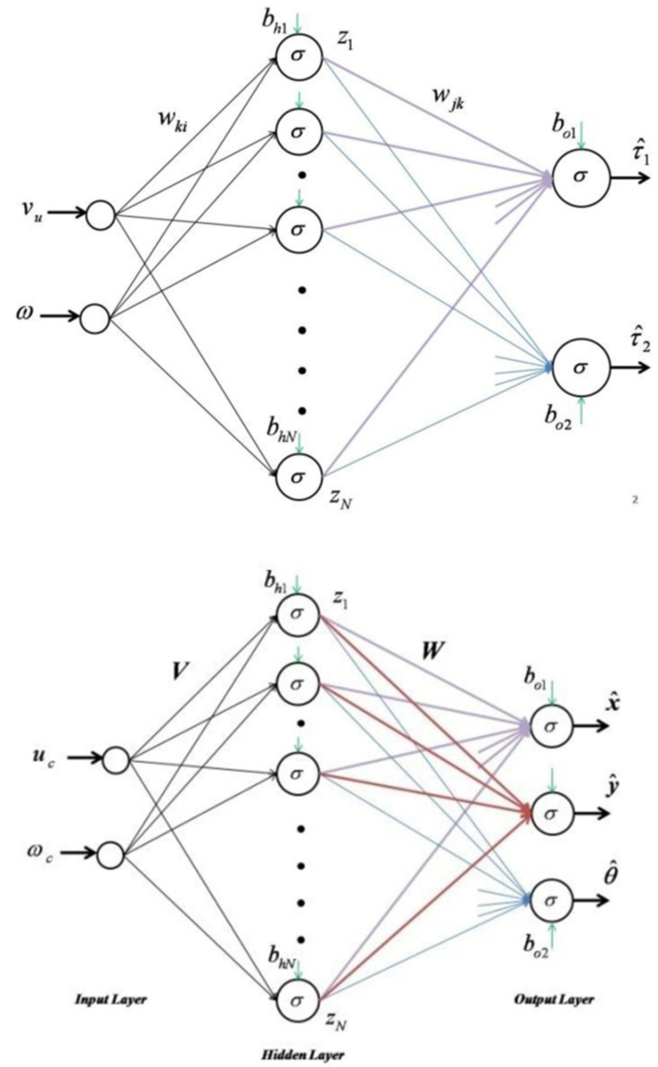

Fig. 6. A comparison between the neural network structures used in dynamic and kinematic level controllers.

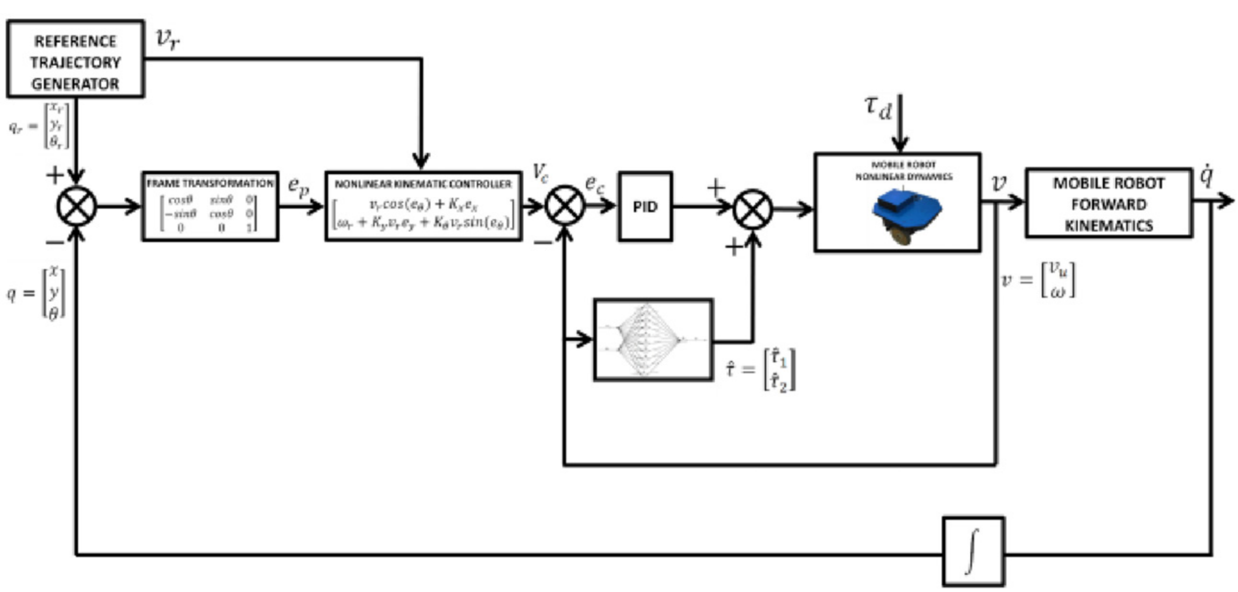

Fig. 5. The neural-adaptive dynamic level neural network control structure.

Integrating the above equation over time, we have

$$q=\int \dot{q}\,dt=\int S\nu\,dt. \tag{43}$$

Therefore the required derivative is

$$\frac{\partial q}{\partial \nu}=\int S\,dt=\int \begin{bmatrix}\cos\theta & -a\sin\theta \\ \sin\theta & a\cos\theta \\ 0 & 1\end{bmatrix} dt. \tag{44}$$

The second derivative can be found from the system's dynamic equation (20). The simplified dynamic equation of the system is rewritten here in a compact form

$$\overline{M}(q)\dot{\nu}(t)+\overline{V}_m(q,\dot{q})\nu(t)=\overline{\tau}. \tag{45}$$

where $\overline{\tau}=\overline{B}(q)\tau-\overline{\tau}_d$ is the equivalent total torque. From Eq. (45) we get

$$\dot{\nu}(t)=-\overline{M}(q)^{-1}\overline{V}_m(q,\dot{q})\nu(t)+\overline{M}(q)^{-1}\overline{\tau}. \tag{46}$$

Therefore

$$\nu(t)=\int_0^t -\overline{M}(q)^{-1}\overline{V}_m(q,\dot{q})\nu(t)dt+\int_0^t \overline{M}(q)^{-1}\overline{\tau}\,dt \tag{47}$$

This equation shows that only the second term is a function of the torque vector $\overline{\tau}$. As a result, the required Jacobian matrix is

$$\begin{aligned}\frac{\partial \nu}{\partial \overline{\tau}}&=\begin{bmatrix}\frac{\partial u}{\partial \overline{\tau}_1} & \frac{\partial u}{\partial \overline{\tau}_2} \\ \frac{\partial \omega}{\partial \overline{\tau}_1} & \frac{\partial \omega}{\partial \overline{\tau}_2}\end{bmatrix}\\ &=\begin{bmatrix}\int_0^t \overline{M}(q)^{-1}_{11}\,dt & \int_0^t \overline{M}(q)^{-1}_{21}\,dt \\ \int_0^t \overline{M}(q)^{-1}_{12}\,dt & \int_0^t \overline{M}(q)^{-1}_{22}\,dt\end{bmatrix}\\ &=\begin{bmatrix}\int_0^t \frac{1}{m}\,dt & 0 \\ 0 & \int_0^t \frac{1}{I_c+ma^2}\,dt\end{bmatrix}=\begin{bmatrix}\frac{1}{m}t & 0 \\ 0 & \frac{1}{I_c+ma^2}t\end{bmatrix}\end{aligned} \tag{48}$$

From Eq. (28), the generated input torque can be approximated by

$$\tau=K_P e_c \tag{49}$$

The derivative part of the PD controller is neglected in the above equation because it is very small. This will simplify the analysis as follows:

$$\frac{\partial \tau}{\partial e_c}=K_P=\begin{bmatrix}K_{P_r} & 0 \\ 0 & K_{P_l}\end{bmatrix} \tag{50}$$

The next derivative term needed is

$$\frac{\partial e_c}{\partial \nu_c}=\frac{\partial(\nu_c-\nu_a)}{\partial \nu_c}=1 \tag{51}$$

The required Jacobian matrix can be found by combining the previous equations

$$J=\int S\,dt\,\frac{\partial \nu}{\partial \tau}\begin{bmatrix}K_{P_r} & 0 \\ 0 & K_{P_l}\end{bmatrix} \tag{52}$$

As it can be seen in the above equation, calculation of the Jacobian using this method requires the exact knowledge of all system parameters. It is also assumed that all of the system parameters are certain and fixed which is not applicable to real life robot platforms which are subjected to parameters uncertainty. Calculation of the Jacobian matrix using the neural network direct model approach solves this problem. The neural network will be continuously learning the real robot's model online and can therefore give a better approximation of the system Jacobian.

3.2. Jacobian calculation using the neural network direct model

Fig. 4 shows the neural network topology used to approximate the robot platform. The neural network has a feedforward structure with one hidden layer having N neurons, one input layer with two inputs, and one output layer with three outputs. The output vector components of the hidden layer are

$$y_i=\sigma\left(\sum_{l=1}^{N} W_{il}z_l+b_{oi}\right) \tag{53}$$

$$z_l=\sigma\left(\sum_{j=1}^{m} V_{lj}x_j+b_{hj}\right) \tag{54}$$

where z_l is the output component of the hidden layer, b_{hj} is a bias term, and σ is the hyperbolic tangent activation function

$$\sigma(s)=\frac{e^s-e^{-s}}{e^s+e^{-s}} \tag{55}$$

In Eqs. (53) and (54), the variable name x_i refers to the neural network input components (u_c,ω_c), while the variable y_i refers to the neural network output components $(\hat{x}_c,\hat{y}_c,\hat{\theta})$.

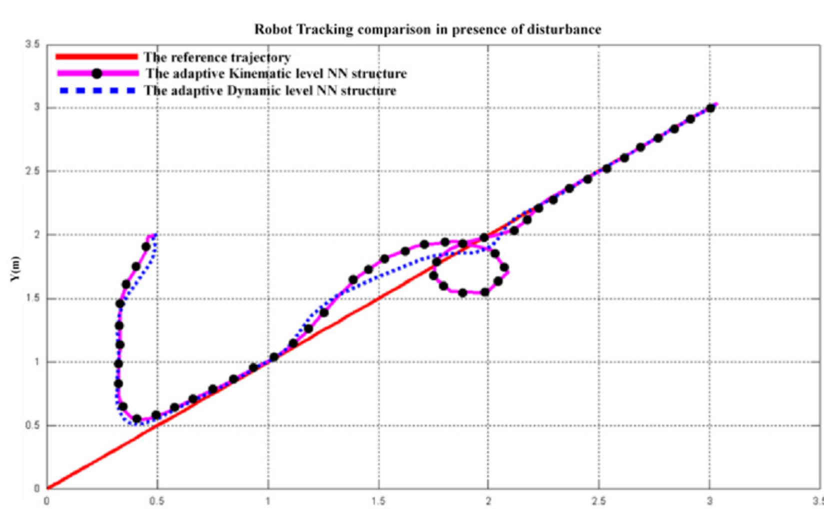

Fig. 7. The robot tracking performance in presence of disturbance in the robot path.

The Jacobian matrix can be found from the above neural network by performing the following derivations:

$$\frac{\partial \hat{q}}{\partial v_c} = \hat{J} = \begin{bmatrix} \frac{\partial \hat{x}}{\partial v_c} & \frac{\partial \hat{x}}{\partial \omega_c} \\ \frac{\partial \hat{y}}{\partial v_c} & \frac{\partial \hat{y}}{\partial \omega_c} \\ \frac{\partial \hat{\theta}}{\partial v_c} & \frac{\partial \hat{\theta}}{\partial \omega_c} \end{bmatrix} \tag{56}$$

Using the chain rule, the required derivative for the Jacobian matrix is

$$\frac{\partial y_i}{\partial x_j} = \frac{\partial y_i}{\partial z_l}\frac{\partial z_l}{\partial x_j}, \tag{59}$$

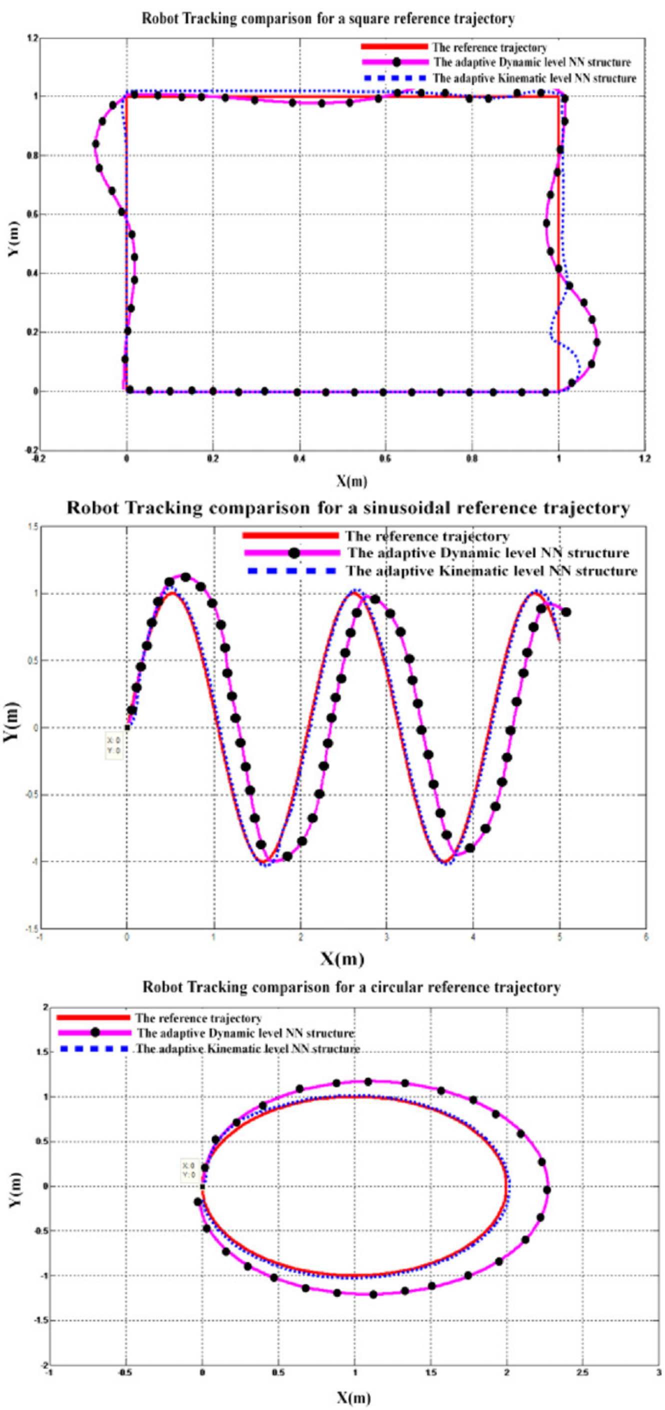

Fig. 8. The robot tracking performance with sinusoidal, square and circular reference trajectories.

Fig. 9. The ERA-MOBI robot platform.

where

$$\frac{\partial y_i}{\partial z_l} = W_{il}\acute{\sigma}\left(\sum_{l=1}^{N} W_{il} z_l\right), \tag{60}$$

$$\frac{\partial z_l}{\partial x_j} = V_{lj}\acute{\sigma}\left(\sum_{j=1}^{m} V_{lj} x_j\right). \tag{61}$$

Therefore, Eq. (59) becomes

$$\frac{\partial y_i}{\partial z_l} = W_{il}\acute{\sigma}\left(\sum_{l=1}^{N} W_{il} z_l\right) V_{lj}\acute{\sigma}\left(\sum_{j=1}^{m} V_{lj} x_j\right) \tag{62}$$

Eq. (62) will be updated online as the network is learning. This estimated Jacobian function is used to update the kinematic controller gains. The neural network to be used in this approach should be well trained so that it can perform a precise approximation of the system's Jacobian.

4. Simulation results

The proposed neural-adaptive control structure in this paper shown in Fig. 3 is classified as an adaptive controller in the kinematic level as it is mentioned before. This controller's performance is compared with the other neural-adaptive control structure in the dynamic level [10,57,58] shown in Fig. 5. Comparing Figs. 3 and 5, one can find out that the second structure directly produces torques for the dynamic system to achieve robustness. On the other hand, in the proposed control scheme the neural network's role is to tune the gains of another controller using the real model of the robot.

The topologies of the neural networks used in these two methods are shown Fig. 6.

The control structure in Fig. 5 indicates that the neural network's role in the dynamic level controller is to produce torques to be added to the backstepping controller torques by learning the actual system dynamics which contains uncertainties and disturbances. The role of the neural network in the kinematic level control structure is to learn the robot dynamics to be used in the gradient descent algorithm to tune the gains of the kinematic based controller. A comparative tracking performance of these two controllers in the presence of disturbance is shown in Fig. 7.

As it can be noted in Fig. 7, the disturbance rejection and robustness of the dynamic neural network structure is much better compared to the indirect structure. The reason behind this

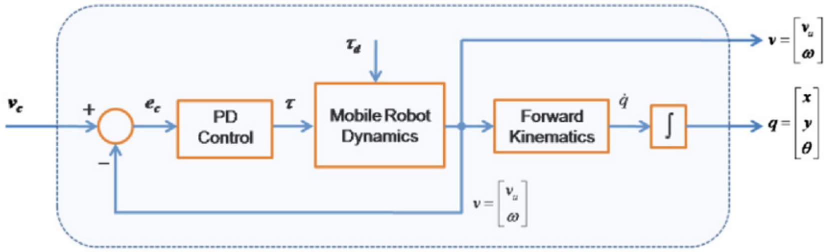

Fig. 10. Internal block diagram representation of the experimental mobile robot platform.

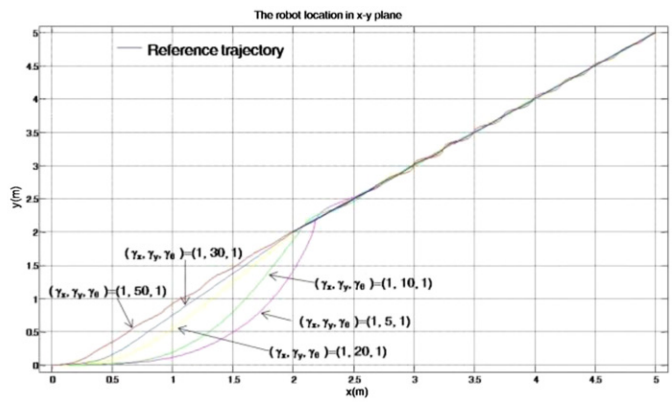

Fig. 11. Tuning of the adaptive controller gains for a linear trajectory.

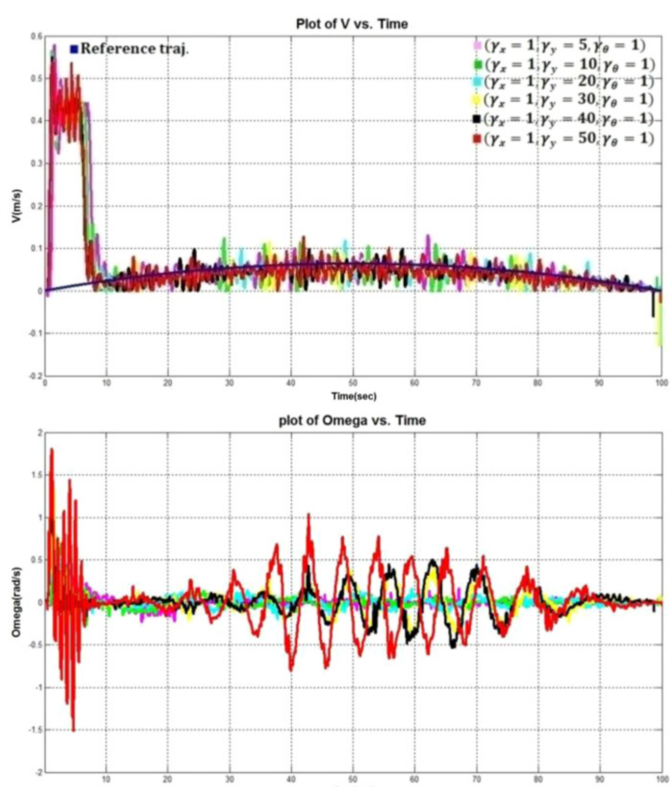

Fig. 12. Robot linear and angular velocities with different adaptive controller gains.

performance is the generated torque of the controller which helps the system reject any disturbance. The gains of the kinematic based controller for the dynamic adaptive method are tuned for a linear reference trajectory and are fixed throughout the tracking process for any different reference trajectory but and these gains are adaptively changed and tuned for the indirect method without any prior knowledge and tuning about the reference trajectory. In order to see the advantage of the proposed neural-adaptive controller, which is the online gain tuning of the kinematic based controller for any kind of reference trajectory, the tracking performance of these two controllers is tested with different reference trajectories as shown in Fig. 8.

As it can be noted from Fig. 8, the neural-adaptive controller has a more precise tracking performance. The reason behind having these results is because the gains of the kinematic controller for the direct NN structure are tuned for a linear trajectory and will not act perfect for other trajectories such as a sinusoid and the neural network in this structure does not have anything to do with the gain tuning or changing the system velocities. However, the gains of the kinematic level controller for the indirect structure are tuned continuously as the robot moves along any trajectory.

The conclusion from the above simulation results and comparison is that the neural-adaptive dynamic level controller is suitable for disturbance rejection and dealing with noise and uncertainties. However, the neural-adaptive kinematic level controller is suitable for tuning the gains at the kinematic level which will generate the required velocities for the system to have a perfect tracking with respect to any shape of the reference trajectory.

5. Experimental results and comparative analysis

The mobile robot platform used in this paper is a differential drive mobile robot (erratic ERA) [59]. This system is a fully featured commercial mobile robot platform used for research and teaching. A view of the platform is shown in Fig. 9. The platform comes with a built-in velocity controller using a PD controller as shown in the block diagram representation of Fig. 10. The robot platform software is written in Unix and gives access only to the user to tune the PD controller gains and to read the measurement of the robot velocities (u,ω) and robot posture (x, y,θ). All the other parameters of the system are not given and there is no access to the torque input variables. Therefore the platform is considered mainly as a black box with unknown parameters.

The proposed adaptive backstepping control scheme is analyzed and compared with the conventional (fixed-gains) backstepping controller. Both schemes are compared under the same operating conditions; the same reference trajectory and robot initial posture.

The optimum gains used for the adaptive controller have been found through a series of experimental tests on the robot with a linear reference trajectory. Fig. 11 shows the results of some sample tests done on the robot with different adaptive controller gains. The speed of response and smoothness of the robot trajectory are the two important factors used to select the best controller gains. The other important factor analyzed is the amount of control input to the system for each of these trajectories. The robot linear and angular velocities which represent the amount of control input to the robot are shown in Fig. 12.

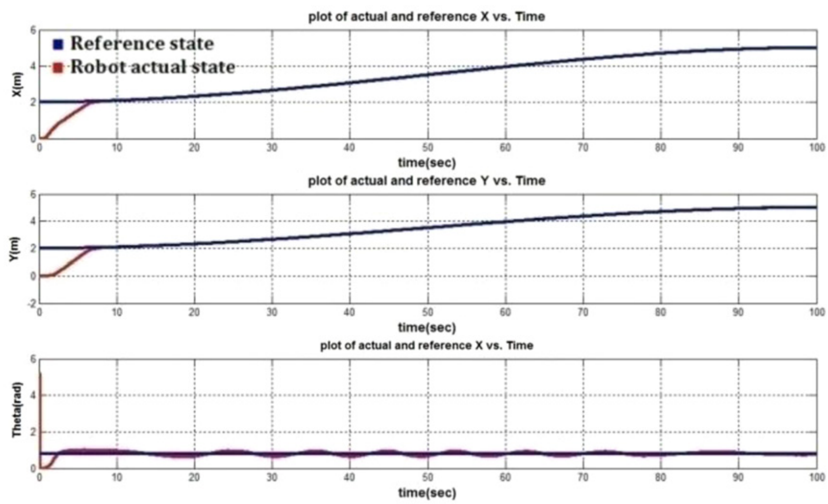

Fig. 13. Robot states errors for the linear reference trajectory.

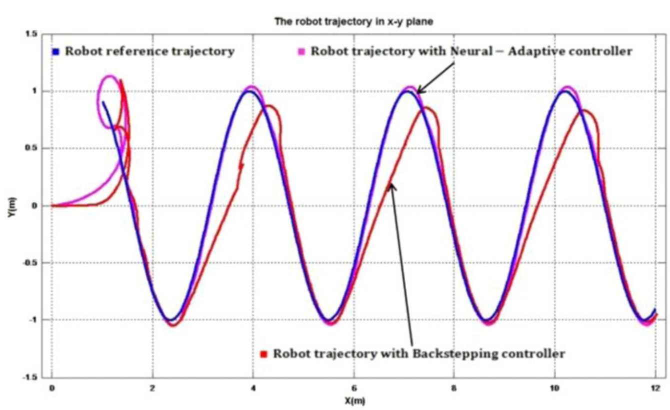

Fig. 14. Comparison between NN-adaptive backstepping controller and backstepping controller-Robot trajectory in x–y plane.

The adaptive controller gains were tuned to the optimum values (γ_x=1, γ_y=20, γ_θ=1) according to the criteria of fast response, smooth trajectory, and small amount of control input.

Fig. 13 shows the time response of the robot state errors QUOTE , QUOTE and QUOTE for a linear reference trajectory. The curves show a very good tracking performance, which confirms the effectiveness of the adaptive control scheme.

The next step is to test the controller with a sinusoidal reference trajectory, which is one of the most challenging trajectories that a robot can follow.

The backstepping controller gains chosen for comparison are the ones tuned for the sinusoidal reference trajectory. The tracking performance of the robot with NN-based adaptive backstepping and normal backstepping controller are shown in Fig. 14 with the following learning rates ($\eta_x=\eta_y=\eta_\theta=0.9$),$\beta_x=\beta_y=\beta_\theta=0.1$, and the backstepping controller gains K_x=1, K_y=120, K_θ=3.

It should be noted here that the gains of the backstepping controller are carefully tuned for the sinusoidal reference trajectory on the experimental robot platform and the above tracking performance is relatively the best response that can be achieved with the backstepping controller. The time response of the robot state errors e_x,e_y and e_θ are given in Fig. 15.

The improvement in tracking performance with the NN-adaptive backstepping controller can be seen clearly from the robot states x and y. The heading angle curve (θ) shows a longer transient time with more oscillations compared to the x and y curves. This is mainly due to the odometry error, which is based on the encoder readings. The system response could be improved by having an additional heading sensor to measure directly the robot heading angle.

The robot linear and angular velocities are shown in Fig. 16, and the adaptive controller gains are given in Fig. 17. To quantify the performance improvement achieved with the NN-based adaptive controller, the magnitudes of the trajectory position error is analyzed. The trajectory position error is calculated based on the robot position (x, y) and is shown in Fig. 15 for both control schemes

$$e_{tr}=\sqrt{{e_x}^2+{e_y}^2}=\sqrt{(x-x_r)^2+(y-y_r)^2} \tag{63}$$

The NN-based adaptive backstepping controller shows a highly reduced trajectory error compared to the backstepping controller with fixed gains.

Three performance indices are used next to analyze the trajectory error:

(1) Mean squares of trajectory errors

$$MSE=\frac{1}{N_s}\sum_{k=1}^{N_s} e_{tr}(k)^2 \tag{64}$$

(2) Power spectral density of the trajectory error signal (PSD)
(3) Sum of the modulus (SMC) of the control signals, which are the robot input velocities (u,ω)

$$SMC_u=\sum_{k=1}^{N_s}|u(k)|, \tag{65}$$

$$SMC_\omega=\sum_{k=1}^{N_s}|\omega(k)|. \tag{66}$$

Table 1 tabulates the results obtained for the sinusoidal trajectory errors as illustrated in Fig. 18.

The MSE index shows that the tracking performance of the robot is highly improved when using the proposed NN-based adaptive backstepping controller. The percentage improvement in tracking is

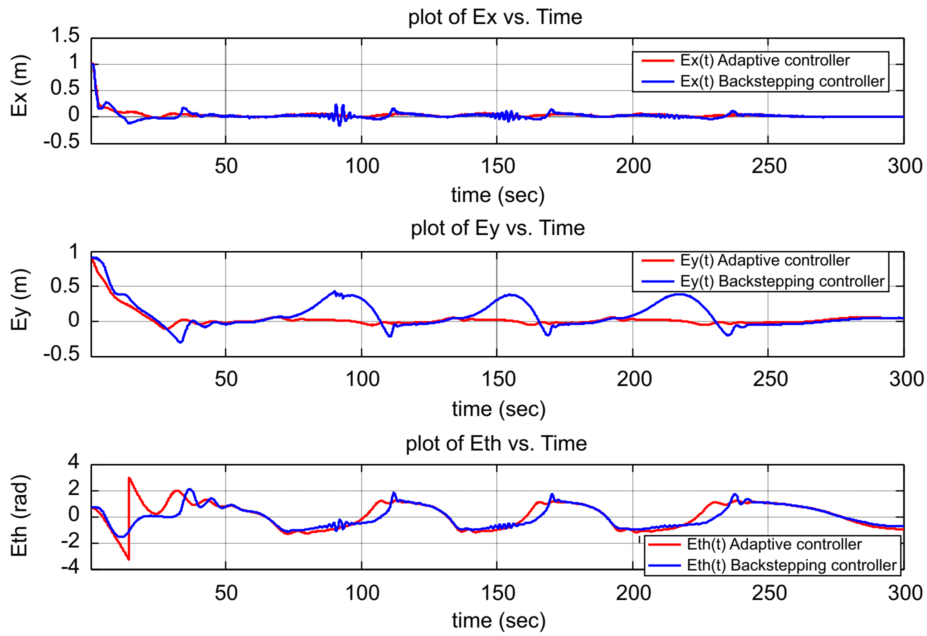

Fig. 15. Robot states errors for Sinusoidal trajectory, Comparison between NN-adaptive controller and the conventional backstepping controller.

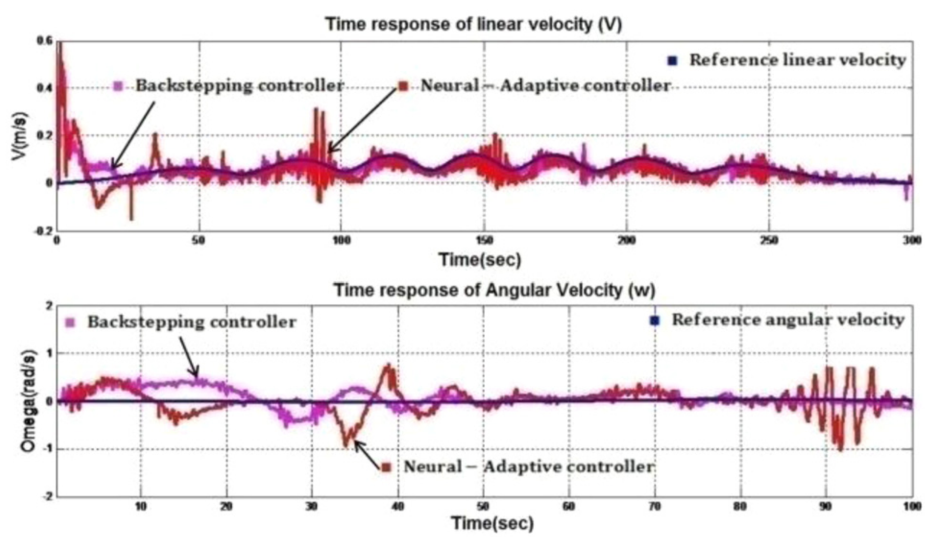

Fig. 16. Robot velocities for a Sinusoidal reference trajectory.

calculated as the relative difference between the values for each scheme. As an example, the MSE improvement index is given by

$$PI = \frac{MSE_{fixed} - MSE_{adaptive}}{MSE_{fixed}} \times 100 \tag{67}$$

The tracking performance of the NN-based adaptive controller in comparison with the fixed-gains backstepping controller is improved by 59.18% which means that the trajectory error is decreased by this percentage.

This improvement is also confirmed by the PSD index. The power spectral density plots of the trajectory position errors are shown in Fig. 19. The fixed-gain backstepping controller has a somehow flat spectrum, which means that the error contains more harmonics over a wide spectrum.

The SMC index indicates the value and power of the control signal needed for each of the controllers. According to the results in Table 1, the SMC corresponding to the linear velocity is lower for the NN-based adaptive backstepping controller by 8.64%. The SMC corresponding to the angular velocity is however higher for the NN-based adaptive controller by 6.28%. These values show that using the NN-based adaptive backstepping controller will increase the angular velocity of the robot and may cause small oscillations but decreases the linear velocity which produces smoother and more stable tracking performance.

6. Conclusion

A novel Neural Networks based adaptive backstepping controller is proposed for a wheeled mobile robot with unknown

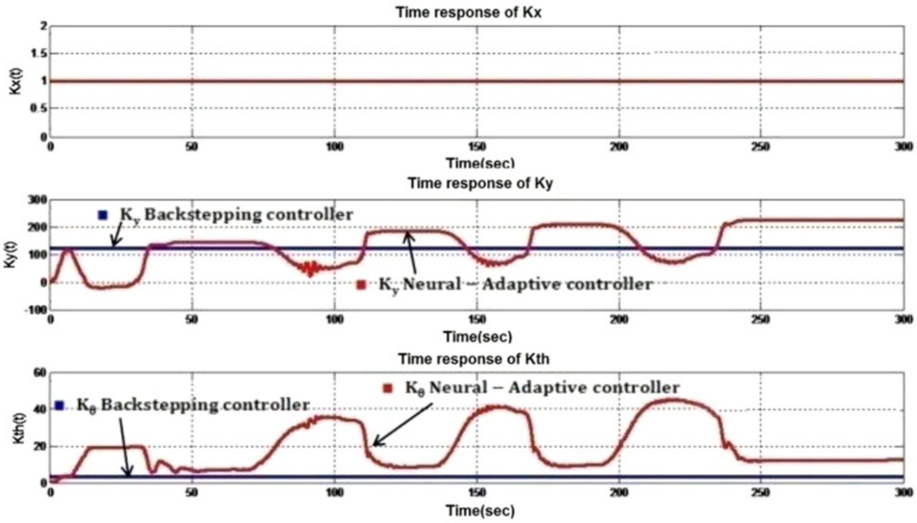

Fig. 17. The comparison between NN-adaptive backstepping controller and backstepping controller–controller gains.

Table 1
Performance indices for the NN-adaptive backstepping and the fixed-gains backstepping controllers.

	Fixed-gains backstepping	NN-adaptive backstepping	Improvement Index (PI) (%)
MSE	0.1335	0.0545	59.18
SMC_v	465.034	424.847	8.64
SMC_ω	1872.5	1990.1	−6.28

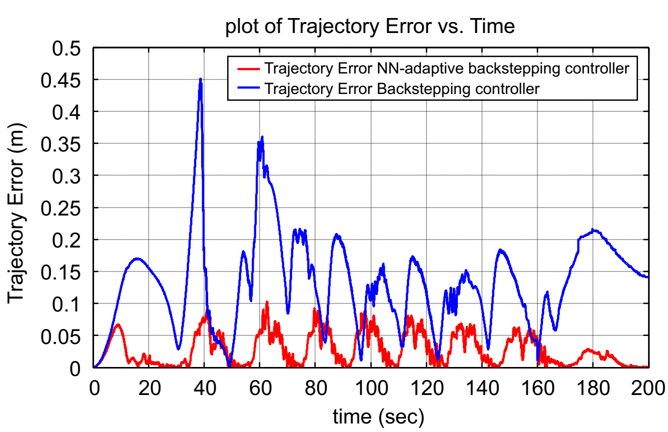

Fig. 18. Trajectory position errors for a sinusoidal reference. NN-adaptive backstepping controller (red curve) and Backstepping controller (blue curve). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

dynamics. The adaptive scheme tunes the gains of the backstepping controller online to minimize the tracking error performance index for any kind of reference trajectory. The neural network is designed to learn the direct model characteristics of the plant and to determine the system Jacobian. Experimental results and comparative analysis between the NN-based adaptive backstepping controller and the fixed-gains backstepping controller confirm the robustness of the proposed adaptive controller and its effectiveness in improving the tracking performance of the mobile robot.

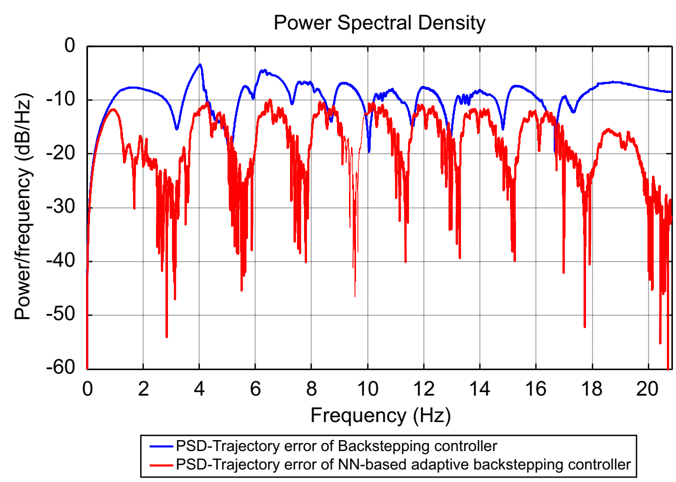

Fig. 19. PSD of the trajectory position error. NN-adaptive backstepping controller (red curve) and Backstepping controller (blue curve). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

References

[1] P. Morin, C. Samson, Control of nonholonomic mobile robots based on the transverse function approach, IEEE Trans. Robot. 25 (2009) 1058–1073.
[2] T. Fukao, H. Nakagawa, N. Adachi, Adaptive tracking control of a nonholonomic mobile robot, IEEE Trans. Robot. Autom. 16 (2000) 609–615.
[3] R.W. Brockett, Asymptotic stability and feedback stabilization, in: R.W. Brockett, R.S. Millman, H.J. Sussmann (Eds.), Differential Geometric Control Theory, Birkauser, Boston, MA, 1983.
[4] D. Buccieri, D. Perritaz, P. Mullhaupt, Z.P. Jiang, D. Bonvin, Velocity-scheduling control for a unicycle mobile robot: theory and experiments, IEEE Trans. Robot. 25 (2009) 451–458.
[5] P. Coelho, U. Nunes, Path-following control of mobile robots in presence of uncertainties, IEEE Trans. Robot. 21 (2005) 252–261.
[6] W. Dong, K.D. Kuhnert, Robust adaptive control of nonholonomic mobile robot with parameter and nonparameter uncertainties, IEEE Trans. Robot. 21 (2005) 261–266.
[7] G. Oriolo, A.D. Luca, M. Vendittelli, WMR control via dynamic feedback linearization: design, implementation, and experimental validation, IEEE Trans. Control Syst. Technol. 10 (2002) 835–852.
[8] W. Dong, W.L. Xu, Adaptive tracking control of uncertain nonholonomic dynamic system, IEEE Trans. Autom. Control 46 (2001) 450–454.

[9] J.M. Yang, J.H. Kim, Sliding mode control for trajectory tracking of nonholonomic wheeled mobile robots, IEEE Trans. Robot. Autom. 15 (1999) 578–587.
[10] R. Fierro, F.L. Lewis, Control of a nonholonomic mobile robot using neural networks, IEEE Trans. Neural Networks 9 (1998) 589–600.
[11] Y. Kanayama, et al., A stable tracking control method for an autonomous mobile robot, in: Proceedings of the IEEE Conference on Robotics Automation, 1990, pp. 384–389.
[12] T.Y. Kuc, S.M. Baek, K. Park, Adaptive learning controller for autonomous mobile robots, IEE Proc. Control Theory Appl. 148 (2001) 49–54.
[13] H. Dong, D. Sun, S.K. Tso, Tracking control of differential mobile robot using adaptive coupling scheme, in: Proceedings of the 7th International Conference on Control, Automation, Robotics and Vision, 2002, pp. 1138–1143.
[14] S. Dong, N.H. Dong, S.K. Tso, Tracking stabilization of differential mobile robots using adaptive synchronized control, in: Proceedings of the IEEE Conference on Robotics and Automation, Washington, 2002, pp. 2638–2643.
[15] T. Das, I.N. Kar, Design and implementation of an adaptive fuzzy logic-based controller for wheeled mobile robots, IEEE Trans. Control Syst. Technol. 14 (2006) 501–510.
[16] T.H. Lee, H.K. Lam, F.H.F. Leung, P.K.S. Tam, A practical fuzzy logic controller for the path tracking of wheeled mobile robots, IEEE Control Syst. Mag. 23 (2003) 60–65.
[17] Y. Miyasato, Adaptive H_∞ control of nonholonomic mobile robot based on inverse optimality, in: Proceedings of the American Control Conference, Seattle, 2008, pp. 3524–3529.
[18] W.E. Dixon, D.M. Dawson, F. Zhang, E. Zergeroglu, Global exponential tracking control of a mobile robot system via a PE condition, IEEE Trans. Syst. Man Cybern. 30 (2000) 129–142.
[19] R. Silva-Ortigoza, G. Silva-Ortigoza, V.M. Hernandez-Guzman, V.R. Barrientos-Sotelo, J.M. Albarran-Jimenez, V.M. Silva-Garcia, Trajectory tracking in a mobile robot without using velocity measurement for control of wheels, IEEE Lat. Am. Trans. 6 (2008) 598–607.
[20] T.Y. Wang, C.C. Tsai, Adaptive trajectory tracking control of a wheeled mobile robot via Lyapunov technique, in: Proceedings of the 30th IEEE Industrial Electronics Conference, 2004, pp. 389–394.
[21] T.J. Huang, Adaptive tracking control of high-order non-holonomic mobile robot systems, IET Control Theory Appl. 3 (2009) 681–690.
[22] W.E. Dixon, D.M. Dawson, E. Zargeroglu, A. Behal, Adaptive tracking control of a wheeled mobile robot via an uncalibrated camera system, IEEE Trans. Syst. Man Cybern. 31 (2001) 341–352.
[23] W.E. Dixon, M.S. de Queiroz, D.M. Dawson, T.J. Flynn, Adaptive tracking and regulation of a wheeled mobile robot with controller/update law modularity, IEEE Trans. Control Syst. Technol. 12 (2004) 138–147.
[24] Y. Liyong, X. Wei, An adaptive tracking method for nonholonomic wheeled mobile robots, in: Proceedings of the Chinese Control Conference, Hunan, 2007, pp. 801–805.
[25] W. Danwei, X. Guangyan, Full-state tracking and internal dynamics of nonholonomic wheeled mobile robots, IEEE/ASME Trans. Mechatron. 8 (2003) 203–214.
[26] W. Dong, W. Huo, S.K. Tso, W.L. Xu, Tracking control of uncertain dynamic nonholonomic system and its application to mobile robots, IEEE Trans. Robot. Automat. 16 (2000) 870–874.
[27] J.M. Yang, J.H. Kim, Sliding mode motion control of nonholonomic mobile robots, IEEE Control Syst. Mag. 19 (1999) 15–23.
[28] C. Dongkyoung, Sliding-mode tracking control of nonholonomic wheeled mobile robots in polar coordinates, IEEE Trans. Control Syst. Technol. 12 (2004) 637–644.
[29] H. Chen, M.M. Ma, H. wang, Z.Y. Liu, Z.X. Cai, Moving horizon H_∞tracking control of wheeled mobile robots with actuator saturation, IEEE Trans. Control Syst. Technol. 17 (2009) 449–457.
[30] N.A. Martins, D. Bertol, W. Lombardi, E.R. Pieri, Trajectory tracking of a nonholonomic mobile robot: a suggested neural torque controller based on the sliding mode theory, in: Proceedings of the International Workshop on Variable Structure Systems, Antalya, 2008, pp. 384–389.
[31] S. Akhavan, M. Jamshidi, A NN-based sliding mode control for nonholonomic mobile robots, in: Proceedings of the IEEE International Conference on Control Applications, Anchorage, 2000, pp. 664–667.
[32] N.A. Martins, D. Bertol, E.R. De Pieri, E.B. Castelan, M.M. Dias, Neural dynamic control of a nonholonomic mobile robot incorporating the actuator dyanmics, in: Proceedings of the International Conference on Computational Intelligence for Modeling Control and Automation, Vienna, 2008, pp. 563–568.
[33] J. Ye, Tracking control for nonholonomic mobile robots: Integrating the analog neural network into the backstepping technique, Neurocomputing 71 (2007) 3373–3378.
[34] J. Ye, Adaptive control of nonlinear PID-based analog neural networks for a nonholonomic mobile robot, Neurocomputing 71 (2008) 1561–1565.
[35] T. Das, I.N. Kar, S. Chaudhury, Simple neuron-based adaptive controller for a nonholonomic mobile robot including actuator dynamics, Neurocomputing 69 (2006) 2140–2151.
[36] K.G. Jolly, R.S. Kumar, R. Vijayakumar, An artificial neural network based dynamic controller for a robot in a multi-agent system, Neurocomputing 73 (2009) 283–294.
[37] K.H. Su, Y.Y. Chen, S.F. Su, Design of neural-fuzzy-based controller for two autonomously driven wheeled robot, Neurocomputing 73 (2010) 2478–2488.
[38] R. Fierro, F.L. Lewis, Control of a nonholonomic mobile robot: backstepping kinematics into dynamics, in: Proceedings of the 34th IEEE Conference on Decision and Control, Piscataway, NJ, IEEE Press, USA, 1995, pp. 3805–3810.
[39] Z.P. Jiang, H. Nijmeijer, Tracking control of mobile robots: a case study in backstepping, Automatica 33 (1997) 1393–1399.
[40] Z.H. Jiang, T. Ishita, A neural network controller trajectory control of industrial robot manipulators, J. Comput. 3 (8) (1997) 1–8.
[41] B. Lacevic, J. Velagic, Stable nonlinear position control law for mobile robot using genetic algorithm and neural network, in: Proceedings of the World Automation Congress, Budapest, 2006, pp. 1–7.
[42] N.A. Martins, D. Bertol, W. Lombardi, E.R. Pieri, E.B. Castelan, Trajectory tracking of a nonholonomic mobile robot with parametric and nonparametric uncertainties: a proposed neural control, in: Proceedings of the 16th Mediterranean Conference on Control and Automation, Ajaccio, 2008, pp. 315–320.
[43] N.A. Martins, D. Bertol, W. Lombardi, E.R. Pieri, M.M. Dias, Neural control applied to the problem of trajectory tracking of mobile robots with uncertainties, in: Proceedings of the 10th Brazilian Symposium on Neural Networks, Salvador, 2008, pp. 117–122.
[44] X. Dong, Z. Dongbin, Y. Jianqiang, T. Xiangmin, Trajectory tracking control of omnidirectional wheeled mobile manipulators: robust neural network-based sliding mode approach, IEEE Trans. Syst. Man Cybern. 39 (2009) 788–799.
[45] M.K. Bugeja, S.G. Fabri, L. Camilleri, Dual adaptive dynamic control of mobile robots using neural networks, IEEE Trans. Syst. Man Cybern. 39 (2009) 129–141.
[46] M.K. Bugeja, S.G. Fabri, Multi-layer perceptron adaptive dynamic control for trajectory tracking of mobile robots, in: Proceedings of the 32nd IEEE Conference on Industrial Electronics, Paris, 2006, pp. 3798–3803.
[47] Z.P. Wang, S.S. Ge, T.H. Lee, Adaptive neural network control of a wheeled mobile robot violating the pure nonholonomic constraint, in: Proceedings of the 43rd IEEE Conference on Decision and Control, vol. 5, 2004, pp. 5198–5203.
[48] M.K. Bugeja, S.G. Fabri, Multi-layer perceptron dual adaptive control for mobile robots, in: Proceedings of the Mediterranean Conference on Control and Automation, Athens, 2007, pp. 1–6.
[49] Z.P. Wang, S.S. Ge, T.H. Lee, X.C. Lai, Adaptive smart neural network tracking control of wheeled mobile robots, in: Proceedings of the 9th International Conference on Control, Automation, Robotics and Vision, Singapore, 2006, pp. 1–6.
[50] B.S. Park, S.J. yoo, J. b. Park, Y.H. Choi, Adaptive neural sliding mode control of nonholonomic wheeled mobile robots with model uncertainty, IEEE Trans. Control Syst. Technol. 17 (2009) 207–214.
[51] K. Shojaei, A. Tarakameh, A.M. Shahri, Adaptive trajectory tracking of WMRs based on feedback linearization technique, in: Proceedings of the International Conference on Mechatronics and Automation, Changchun, 2009, pp. 729–734.
[52] M. Oubbati, M. Schanz, P. Levi, Kinematic and dynamic adaptive control of a nonholonomic mobile robot using a RNN, in: Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation, 2005, pp. 27–33.
[53] J.Y. Song, H.C. Yoon, B.P. Jin, Generalized predictive control based on self-recurrent wavelet neural network for stable path tracking of mobile robots: adaptive learning rates approach, IEEE Trans. Circuits Syst. 53 (2006) 1381–1394.
[54] S. Purwar, N. Gupta, I.N. Kar, Trajectory tracking control of mobile robots using wavelet networks, in: Proceedings of the IEEE 22nd International Symposium On Intelligent Control, Singapore, 2007, pp. 550–555.
[55] R.J. Wai, C.M. Liu, Design of dynamic petri recurrent fuzzy neural network and its application to path-tracking control of nonholonomic mobile robot, IEEE Trans. Ind. Electron. 56 (2009) 2667–2683.
[56] K. Watanabe, J. Tang, M. Nakamura, S. Koga, T. Fukuda, A fuzzy-Gaussian neural network and its application to mobile robot control, IEEE Trans. Control Syst. Technol. 4 (1996) 193–199.
[57] Y. chung, C. Park, F. Harashima, A position control differential drive wheeled mobile robot, IEEE Trans. Ind. Control 48 (2001) 853–863.
[58] G. Campion, G. Bastin, B. d'Andrea-Novel, Structural properties and classification of kinematic and dynamic models of wheeled mobile robots, IEEE Trans. Robot. Autom. 12 (1996) 47–62.
[59] 〈http://www.videredesign.com/index.php?id=45〉.

Omid Mohareri received his Bachelor of Science degree in Mechanical Engineering from Shiraz University, Iran, his first Master of Science degree in Mechatronics Engineering from American University of Sharjah, school of engineering. He is currently working through his second master's degree in Simon Fraser University. His research interests are in the areas of robotics and control.

Rached Dhaouadi received his Ph.D. in Electrical Engineering from the University of Minnesota in 1990. From 1990 to 1994 he worked as a Visiting Researcher with the Hitachi Research Laboratory, Hitachi, Ltd., Japan, where he was engaged in the design and development of motor drive systems for rolling mills. From 1994 to 2000 he was with the Polytechnic School of Tunisia, University of Tunis. He also held Visiting Scholar positions at Trondheim Institute of Technology, Trondheim, Norway, and at Rice University, Houston, Texas. He is currently an Associate Professor of Electrical Engineering at the College of Engineering, American University of Sharjah, UAE. His research interests are in the areas of motor drives, modeling and control of complex electromechanical systems, and intelligent motion control systems, with applications to rolling mill drives and mobile robots. Dr. Dhaouadi is a Senior Member of the IEEE Industrial Electronics Society.

Ahmad A. Rad received the B.S. degree in Engineering from Abadan Institute of Technology, Abadan, Iran, the M.S. degree and the Ph.D. degree in Control Engineering from the University of Bradford, Bradford, UK and the University of Sussex, Brighton, UK, in 1977, 1986 and 1988, respectively. He is currently a Professor of Mechatronics at the School of Engineering Science at Simon Fraser University, Canada. His current interests include autonomous systems, SLAM, Map building, Localization in mobile robots, intelligent control, time delay systems and system identification.