Real-time Motion Control of a Nonholonomic Mobile Robot with Unknown Dynamics*

TIEMIN HU and SIMON X. YANG†
ARIS (Advanced Robotics & Intelligent Systems) Lab
School of Engineering, University of Guelph
Guelph, Ontario, N1G 2W1, Canada

Abstract

In this paper, real-time fine motion control of a nonholonomic mobile robot is investigated, where the robot dynamics is completely unknown and is subject to significant uncertainties. A novel neural network based real-time controller is developed. The neural network assumes a single layer structure, by taking advantage of the robot regressor dynamics that express the highly nonlinear robot dynamics in a linear form in terms of the known and unknown robot dynamic parameters. The learning algorithm is computationally efficient. The system stability and the converge of tracking errors to zero are rigorously proved using a Lyapunov theory. The real-time fine control of mobile robot is achieved through the on-line learning of the neural network without any off-line learning procedures. In addition, the developed controller is capable of compensating sudden changes of robot dynamics or disturbance. The effectiveness and efficiency of the proposed controller is demonstrated by simulation studies.

1 Introduction

A mobile robot that can move intelligently without human intervention, has a broad range of applications. How to *efficiently* control a mobile robot with unknown robot dynamics and subject significantly uncertainties and/or unmodelled disturbances is an open question in robotics. Many control methods (e.g., [1]-[11]) have been proposed for motion control of a mobile robot. The computed torque control approach is capable to achieving fine control of mobile robot, but it requires the exact dynamics model which is almost impossible in practice. Adaptive controllers [1]-[3] can achieve fine control with partially unknown mobile robot dynamics, however, complicated on-line estimation of these unknown dynamics is needed. There are many neural network based controllers that are quite successful in some areas where the model-based approaches failed. Fierro and Lewis [4] developed a neural network based model by combining the backstepping tracking technique and a torque controller, using a multi-layer feedforward neural network, where the neural network can learn the dynamics of the mobile robot by its on-line learning. But the control algorithm and the neural network learning algorithm are very complicated and it is computationally expensive.

*This work was supported by Natural Sciences and Engineering Research Council (NSERC) of Canada under grant RGPIN227684 to S. X. Yang.

†To whom all correspondence should be addressed. Email: syang@uoguelph.ca

Yang and Meng [5] proposed a single-layer neural network based controller for robot manipulator. This approach does not include any nonholonomic kinematics therefor it can not be used for robot manipulators with kinematic constraints. In addition, its stability is proved under the assumption of no unmodeled disturbance.

In this paper, a novel single-layer neural network based controller is proposed for real-time fine control of a nonholonomic mobile robot. First, a specific velocity control input is selected from the robot kinematics to guarantee the position error asymptotically stable. Then the neural network is used to generate the control torque that drives the mobile robot, such that the linear and angular velocities of the mobile robot converge to the desired velocities. No knowledge of the robot dynamics is required. In addition, no off-line learning procedures are needed for the neural network. The fine motion control of mobile robot is achieved by the on-line learning ability of the neural network. The stability and convergence of the robot control system is rigorously proved using a Lyapunov theory, subject to unmodeled disturbance and bounded unstructured dynamics. By taking advantage of the robot regressor dynamics formulation that can express the nonlinear robot dynamics into a linear form in terms of the robot dynamics parameters, the neural network assumes a single-layer structure. The learning algorithm is derived from stability analysis of a Lyapunov function candidate, which is much simpler than the most commonly used neural network learning algorithm. Therefor the proposed controller is computationally *efficient*.

2 The Model Algorithm

In this section, the model of a nonholonomic mobile robot will be introduced, including the geometry, kinematics, the nonholonomic constraint, error dynamics, robot dynamics, and regress dynamics. Then the proposed controller will be introduced. The stability and convergence of the robot system are rigorously proved, and the learning algorithm of the neural network is derived from the stability analysis.

2.1 A Nonholonomic Mobile Robot

Consider a mobile robot system with n generalized coordinates q, subject to m constraints, its dynamic equation can be described by

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + F(\dot{q}) + G(q) + \tau_d = E(q)\tau - A^T(q)\lambda, \tag{1}$$

where $M(q)$ is an $n \times n$ symmetric, positive definite matrix; $C(q,\dot{q})$ is the $n \times n$ centripetal coriolis matrix; $F(\dot{q})$ is the $n \times 1$ friction vector; $G(q)$ is an $n \times 1$ gravitational vector; τ_d is an $n \times 1$ bounded unknown disturbance including unstructured unmodeled dynamics; $E(q)$ is an $n \times r$ input transformation matrix; τ is an $r \times 1$ input vector; $A(q)$ is an $m \times n$ Jacobian matrix associated with the constraints and λ is an $m \times 1$ constraints forces vector. Assume that the m constraints are independent of time and can be expressed as

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

Let $S(q)$ be a full rank matrix $n \times (n-m)$ made up by a set of smooth and linearly independent vector spanning the null space of $A(q)$, i.e.,

$$A(q)S(q) = 0. \tag{3}$$

Since the constrained velocity is always in the null space of $A(q)$, from Eq.s (2) and (3), it is possible to find $(n-m)$ velocities $v(t)$ such that

$$\dot{q} = S(q)v(t). \tag{4}$$

Assume the trajectory of the mobile robot is constrained on the horizontal plane and there is no friction, then $G(q) = 0$ and $F(\dot{q}) = 0$. Now system Eq. (1) is transformed into a more appropriated representation for control purpose. Left multiply Eq. (1) by S^T, differentiate Eq. (4) and substitute into Eq. (1), the dynamic equation of the nonholonomic mobile robot are

$$\dot{q} = Sv, \tag{5}$$

and

$$S^T M S\dot{v} + S^T(M\dot{S} + CS)v + S^T\tau_d = S^T B\tau. \tag{6}$$

Define $\bar{M} = S^T MS$, $\bar{C} = S^T(M\dot{S} + CS)$, $\bar{\tau_d} = S^T\tau_d$, $\bar{\tau} = S^T\tau$, Eq. (6) can be rewritten as

$$\bar{M}\dot{v} + \bar{C}v + \bar{\tau}_d = \bar{\tau}. \tag{7}$$

Eq.s (5) and (7) present the nonholonomic mobile robot system in a local coordinates, actually $S(q)$ is a Jacobian matrix that transform independent velocities v in local coordinates attached to the robot to the constrained velocities $\dot{q}$ in global coordinates. Therefore, the properties of the original dynamics hold for the new set of coordinates. One important property about Eq. (7) should be indicated here is that $(\dot{\bar{M}} - 2\bar{C})$ is skew symmetric. In fact,

$$\dot{\bar{M}} - 2\bar{C} = \dot{S}^T MS - (\dot{S}^T MS)^T + S^T(\dot{M} - 2C)S. \tag{8}$$

Since $(\dot{M} - 2C)$ is skew symmetric, it is obvious that $(\dot{\bar{M}} - 2\bar{C})$ is also skew symmetric.

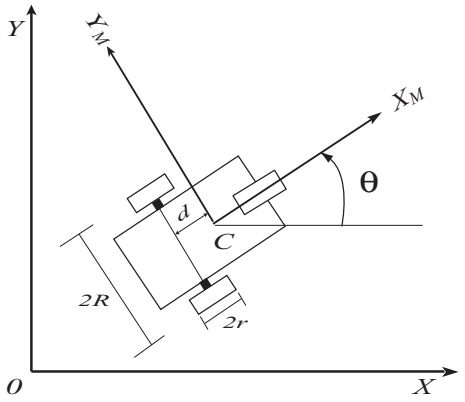

Figure 1: *The typical model of a nonholonomic mobile robot.*

A typical model of a nonholonomic mobile robot is shown in Fig. 1. The robot has two driving wheels mounted on the same axis and a free front wheel. The two driving wheel are independently driven by two actuators to achieve the motion and orientation. The position of the mobile robot in the global frame $\{X, O, Y\}$ cab be described by position C which is the mass center of the mobile robot system, and the orientation θ between robot local frame $\{X_M, C, Y_M\}$ and the global frame.

On the condition of pure rolling and non-slipping, the mobile robot can only move in the direction normal to the driving wheels axis. Therefor the nonholonomic constraint of the mobile robot can be expressed as,

$$\dot{y}_c \cos\theta - \dot{x}_c \sin\theta - d\dot{\theta} = 0. \tag{9}$$

From Eq. (2), obviously $A(q)$ can be chosen as,

$$A(q) = [-\sin\theta \;\; \cos\theta \;\; -d]. \tag{10}$$

It is easy to verify that the following $S(q)$ satisfies all the conditions mentioned above

$$S(q) = \begin{bmatrix} \cos\theta & -d\sin\theta \\ \sin\theta & d\cos\theta \\ 0 & 1 \end{bmatrix}. \tag{11}$$

Selecting the robot longitudinal velocity v_1 and the orientation angler velocity ω as the independent local velocities $v = [v_1, \omega]^T$, the kinematic equation in Eq. (4) can be described as

$$\dot{q} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta_c & -d\sin\theta_c \\ \sin\theta_c & d\cos\theta_c \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_1 \\ \omega \end{bmatrix}. \tag{12}$$

Now the constraints associated with $A^T\lambda$ in Eq. (1) is eliminated, the nonlinear mobile robot dynamics in Eq. (7) can be rewritten into a linear form similar to the regressor dynamics formulation of robot manipulators,

$$Y(v, \dot{v})\phi + \bar{\tau}_d = \bar{\tau}, \tag{13}$$

where ϕ is an $r \times 1$ vector consisting of the known and unknown robot dynamics, such as mass, moment of inertia, etc; $Y(v, \dot{v})$ is an $(n-m) \times r$ coefficient matrix consisting of the known functions of the robot velocity v and acceleration $\dot{v}$; r is the number of the known and unknown robot dynamics parameters. Let m be the mass of the mobile robot, I be the moment of inertia about the mass center C, the dynamic equation of the mobile robot in Fig. 1 can be expressed as Eq. (1), where

$$\begin{aligned}
M(q) &= \begin{bmatrix} m & 0 & md\sin\theta \\ 0 & m & -md\cos\theta \\ md\sin\theta & -md\cos\theta & I \end{bmatrix}, \\
C(q, \dot{q}) &= \begin{bmatrix} 0 & 0 & md\dot{\theta}\cos\theta \\ 0 & 0 & md\dot{\theta}\sin\theta \\ 0 & 0 & 0 \end{bmatrix}, \\
B(q) &= \frac{1}{r}\begin{bmatrix} \cos\theta & \cos\theta \\ -\sin\theta & \sin\theta \\ R & -R \end{bmatrix}, \\
\lambda &= -m(\dot{x}_c\cos\theta + \dot{y}_c\sin\theta)\dot{\theta}, \\
\tau &= \begin{bmatrix} \tau_r \\ \tau_l \end{bmatrix}.
\end{aligned} \tag{14}$$

Plug Eq. (14) into the definitions of $\bar{M}$ and $\bar{C}$ in the robot dynamics equation in Eq. (7), the following simple, interesting results can be obtained,

$$\begin{aligned}
\bar{M} &= \begin{bmatrix} m & 0 \\ 0 & I - md^2 \end{bmatrix}, \\
\bar{C} &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}.
\end{aligned} \tag{15}$$

Now it is easy to rewrite the nonlinear robot dynamics in Eq. (7) in a linear form as

$$\bar{M}\dot{v} + \bar{C}v = Y\phi, \tag{16}$$

where ϕ is a vector consisting of the known and unknown robot dynamic parameters, such as geometric size, mass, moments of inertias, etc.; Y is a coefficient matrix consisting of the known functions of robot position, velocity and acceleration, which is referred as the robot *regressor*. For a typical nonholonomic mobile robot shown in Fig. 1, the robot regressor Y and the coefficient vector ϕ are given as

$$\begin{aligned} Y &= \begin{bmatrix} \dot{v}_1 & 0 & 0 \\ 0 & \dot{\omega} & -\dot{\omega} \end{bmatrix}, \\ \phi &= \begin{bmatrix} m & I & -md^2 \end{bmatrix}^T. \end{aligned} \tag{17}$$

2.2 Control Algorithm

The function of a controller is to implement a mapping between the known information (e.g. reference position, velocity and sensory information) and the actuator commands designed to achieve the robot's task. For mobile robot, the controller design problem cab be described as: given the reference position $q_r(t)$ and velocity $\dot{q}_r(t)$, design a control law for the actuator torques, which drive the mobile robot to move, so the mobile robot velocity tracking a smooth velocity control input and the reference position. Let the velocity and position of a reference robot be

$$\begin{aligned} q_r &= [x_r \ y_r \ \theta_r]^T, \\ \dot{x}_r &= v_r \cos\theta_r, \\ \dot{y}_r &= v_r \sin\theta_r, \\ \dot{\theta}_r &= \omega_r, \end{aligned} \tag{18}$$

where $v_r > 0$ for all t is the reference linear velocity, ω_r is the reference angular velocity. Then the tracking position error between the reference robot and the actual robot can be expressed in the robot local frame as

$$e_M = \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r - x \\ y_r - y \\ \theta_r - \theta \end{bmatrix}. \tag{19}$$

The position error dynamics can be obtained from the time derivative of above equation as

$$\dot{e}_M = \begin{bmatrix} \dot{e}_1 \\ \dot{e}_2 \\ \dot{e}_3 \end{bmatrix} = \begin{bmatrix} \omega e_2 - v_1 + v_r \cos e_3 \\ -\omega e_1 + v_r \sin e_3 \\ \omega_r - \omega \end{bmatrix}. \tag{20}$$

There are many ways in the literature to select the smooth velocity input v_c, Here, very simple backstepping model from [10] is selected.

$$v_c = \begin{bmatrix} v_r \cos e_3 + k_1 e_1 \\ \omega_r + k_2 v_r e_2 + k_3 v_r \sin e_3 \end{bmatrix}, \tag{21}$$

where k_1, k_2, k_3 are positive parameters. Now define the velocity tracking error as

$$e_c = \begin{bmatrix} e_4 \\ e_5 \end{bmatrix} = v_c - v, \tag{22}$$

where v is the actual velocities of the mobile robot. Differentiate Eq. (22) and replace the result into Eq. (7), the mobile robot dynamics using the velocity tracking error can be rewritten as

$$\bar{M}\dot{e}_c = -\bar{C}e_c - \bar{\tau} + Y\phi + \bar{\tau}_d, \tag{23}$$

where

$$Y\phi = \bar{M}\dot{v}_c + \bar{C}v_c, \tag{24}$$

which is the regressor dynamics introduced in Eq. (16). Assign the torque controller for Eq. (7) as

$$\bar{\tau} = \tau_{NN} + k_4 e_c - \gamma, \tag{25}$$

where k_4 is a diagonal positive define design matrix and γ is a robustifying term to compensate the unmodeled unstructured disturbances, τ_{NN} is the output torque of the single-layer neural network, which is given by

$$\tau_{NN} = Y(v_c, \dot{v}_c)W, \tag{26}$$

where W is a 3×1 vector represents the connection weights of the neural network, and is also the approximation of ϕ in Eq. (24). The input of the neural network is the regressor Y. The variables for computing Y are the vector $\left[v_c^T, \dot{v}_c^T\right]^T$, which can be gotten from Eq. (21) or measured. Substitute Eq. (25) into Eq. (23), the closed-loop error dynamics can be expressed as

$$\bar{M}\dot{e}_c = -(k_4 + \bar{C})e_c + Y\tilde{\phi} + \bar{\tau}_d + \gamma, \tag{27}$$

where

$$\tilde{\phi} = \phi - W, \tag{28}$$

which is the estimation error of ϕ. Assume the unmodeled unstructured disturbances are bounded, such that $\|\bar{\tau}_d\| \leq b_d$, the robustifying term is

$$\gamma(t) = -k_d - [e_4, \frac{k_1 e_5}{k_3 k_3 v_r}]^T = -k_d - e_c' \tag{29}$$

where k_d is a diagonal positive define matrix, and

$$\min(\mathrm{diag}(k_d)) > b_d. \tag{30}$$

2.3 Learning Algorithm and Stability Analysis

For the controller in Eq. (25), a learning algorithm for the neural network should be developed, such that it guarantees that the control system is stable, and both the position and velocity tracking errors converge to zeros. Consider the Lyapunov function candidate

$$L = k_1(e_1^2 + e_2^2) + \frac{2k_1}{k_2}(1 - \cos e_3) + \frac{1}{2}(e_c^T \bar{M} e_c + \tilde{\phi}^T \Gamma^{-1} \tilde{\phi}), \tag{31}$$

where Γ is an $r \times r$ positive constant design matrix. Obviously, $L \geq 0$, and $L = 0$ if and only if $e_p = 0$, $e_c = 0$ and $\tilde{\phi} = 0$. Differentiate L, and substitute the error dynamics Eq. (27) into Eq. (31), $\dot{L}$ is given as

$$\begin{aligned} \dot{L} &= 2k_1 e_1 \dot{e}_1 + 2k_1 e_2 \dot{e}_2 + \frac{2k_1}{k_2}\sin e_3 - e_c^T k_4 e_c \\ &\quad + \frac{1}{2}e_c^T(\bar{M} - 2\bar{C})e_c + \tilde{\phi}^T(Y^T e_c + \Gamma^{-1}\dot{\tilde{\phi}}) + e_c^T \bar{\tau}_d + e_c^T \gamma. \end{aligned} \tag{32}$$

Note $(\bar{M} - 2\bar{C})$ is skew symmetric, and choose

$$Y^T e_c + \Gamma^{-1}\dot{\tilde{\phi}} = 0. \tag{33}$$

Then Eq. (32) can be simplified as,

$$\dot{L} = 2k_1 e_1 \dot{e}_1 + 2k_1 e_2 \dot{e}_2 + \frac{2k_1}{k_2}\sin e_3 - e_c^T k_4 e_c + e_c^T \bar{\tau}_d + e_c^T \gamma. \tag{34}$$

Substitute the robustifying term in Eq. (29), then

$$\begin{aligned}\dot{L} &= 2k_1 e_1 \dot{e}_1 + 2k_1 e_2 \dot{e}_2 + \frac{2k_1}{k_2}\sin e_3 - e_c^T e_c' - e_c^T k_4 e_c - e_c(k_d - \bar{\tau}_d) \\ &\leq 2k_1 e_1 \dot{e}_1 + 2k_1 e_2 \dot{e}_2 + \frac{2k_1}{k_2}\sin e_3 - e_c^T k_4 e_c - k_{4\min}\|e_c\|^2 - \|e_c\|(k_{d\min} - b_d).\end{aligned} \tag{35}$$

where $k_{4\min}$ and $k_{d\min}$ are the minimum singular values of k_4 and k_d, respectively. By substituting the position error dynamics from Eq. (20), the time derivative of L is given as,

$$\begin{aligned}\dot{L} \leq\ & 2k_1 e_1(\omega e_2 - v_1 + v_r \cos e_3) - e_c^T e_c' + 2k_1 e_2(v_r \sin e_3 - \omega e_1) \\ & + 2k_3 v_r(\omega_r - \omega)\sin e_3 - k_{4\min}\|e_c\|^2 - \|e_c\|^2(k_{d\min} - b_d).\end{aligned} \tag{36}$$

Replace e_c' from robust term Eq. (29), $\dot{L}$ becomes

$$\begin{aligned}\dot{L} \leq\ & -k_1^2 e_1^2 - \frac{k_1 k_3}{k_2} v_r \sin {e_3}^2 - (k_1 e_1 + e_4)^2 \\ & - \frac{k_1}{k_2 k_3 v_r}(k_3 v_r \sin e_3 + e_5)^2 - \|e_c\|^2 k_{4\min} - \|e_c\|(k_d - b_d)\end{aligned} \tag{37}$$

Since $k_d > b_d$ (Eq. (30)), $\dot{L}$ is guaranteed negative. According to a standard Lyapunov theory and LaSalle extension, all $\|e_p\|$, $\|e_c\|$ and $\tilde{\phi}$ are uniformly ultimately bounded.

Since $\tilde{\phi} = \phi - W$, then $\dot{\tilde{\phi}} = -\dot{W}$. Therefor, the learning law for the neural network is obtained as

$$\dot{W} = -\Gamma Y^T e_c. \tag{38}$$

Obviously, this learning law is much simpler than mostly used learning law like the famous multi-layer back-propagation learning rule.

In summary, Eq.s (25), (26), (29) and (38) define the proposed control algorithm.

The proposed controller structure is shown in Fig. 2. In the proposed, a specific velocity control input $v_c(t)$ is used to derive the suitable torque $\tau(t)$, therefor it does not need to assume "perfect velocity tracking", which is unrealistic in practice. By taking advantage of robot regressor dynamics formulation which transform the highly nonlinear robot dynamics into a linear form in term of the robot dynamics regressor, the neural network in this paper is a single layer structure. There is no off-line training procedures needed, the neural network reconstruct the robot dynamics by learning it on-line. The learning algorithm of the neural network, which is very simple and computationally efficient, is derived from the Lyapunov stability analysis. Unlike robot manipulators, where tedious analysis is needed to derive the regressor because of the complexity and variety of manipulator, mobile robots are comparatively simpler in computing the regressor and there are only a few models needed to simulate the realistic mobile robot for control purpose.

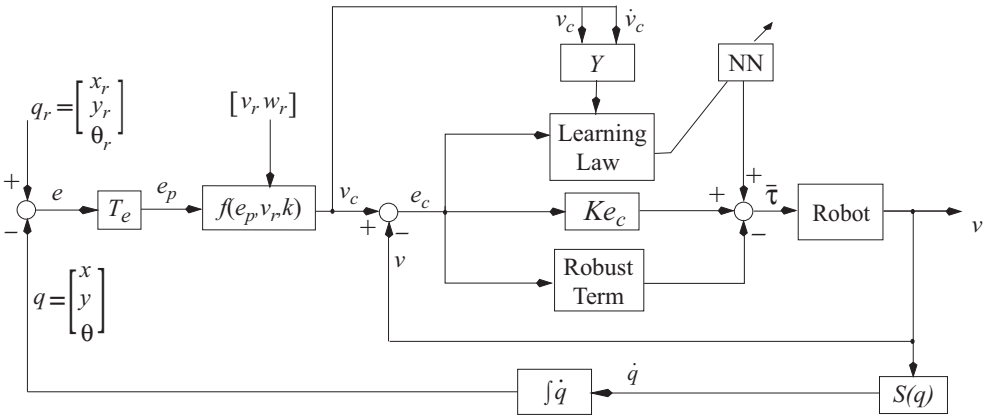

Figure 2: *Structure of the proposed controller.*

3 Simulation Results

The single layer neural network based control algorithm developed in this paper is defined by Eq.s (26) and (37). In this section, the proposed controller is applied to a typical mobile robot shown in Fig. 1. The mobile robot parameters are $m = 10kg$, $I = 5kgm^2$, $R = 0.5m$, and $r = 0.05m$. Three cases are investigated. First, the robot is to track a straight line with the disturbance of coulumb friction that subject to sudden changes. Then, the robot is to track a circular path. Finally, the robot is to track an ellipse, where there is a sudden change of robot dynamics. The simulation results demonstrate the effectiveness of the proposed controller. Since the neural network assumes a simple single layer structure, it is computationally efficient.

3.1 Tracking a Straight Line

The reference trajectory is a straight line with initial coordinates $(1, 2)$ and slop of 25.56° respectively. The initial position of the robot is $[x_0, y_0, \theta_0] = [2, 1, 0]$. The design parameters of the controller are chosen as: $k_1 = 1$, $k_2 = 3$, $k_3 = 2$, $k_4 = 18I_2$, and $\Gamma = I_2$, where I_k is a $k \times k$ identity matrix. A coulumb friction term,

$$F = [(f_1 + f(t))\text{sign}(v_1), (f_2 + f(t))\text{sign}(\omega)]^T \tag{39}$$

is added to the robot system as unmodeled disturbance, where $f_1 = 0.3$, $f_2 = 0.5$. Function $f(t)$ is nonlinear function, defined as: $f(t) = 0$ if $t < 8$; $f(t) = 0.2$, if $t \geq 8$. Thus the disturbance is subject to a sudden change at time goes to 8 *sec*. The real-time tracking performance is shown in Fig. 3A, where the desired path is shown by solid line, while the actual robot traveling route is shown by dashdot line. The tracking errors in the X- and Y-directions and in the orientation are shown in Fig. 3B by solid line, dashed line and dashdot line, respectively. The actual linear and angular velocities of the mobile robot are shown Fig. 3C by solid and dashed lines, respectively. Fig. 4D show the total control torques. It shows that the tracking errors sharply drops to zero. The robot velocities and the control torques quickly converge to its steady state. Note that there is no prior knowledge of the robot system dynamics, the controller learns the system dynamics on-line without any off-line training. In addition, when the time goes to 8 *sec*, there is a sudden change of friction

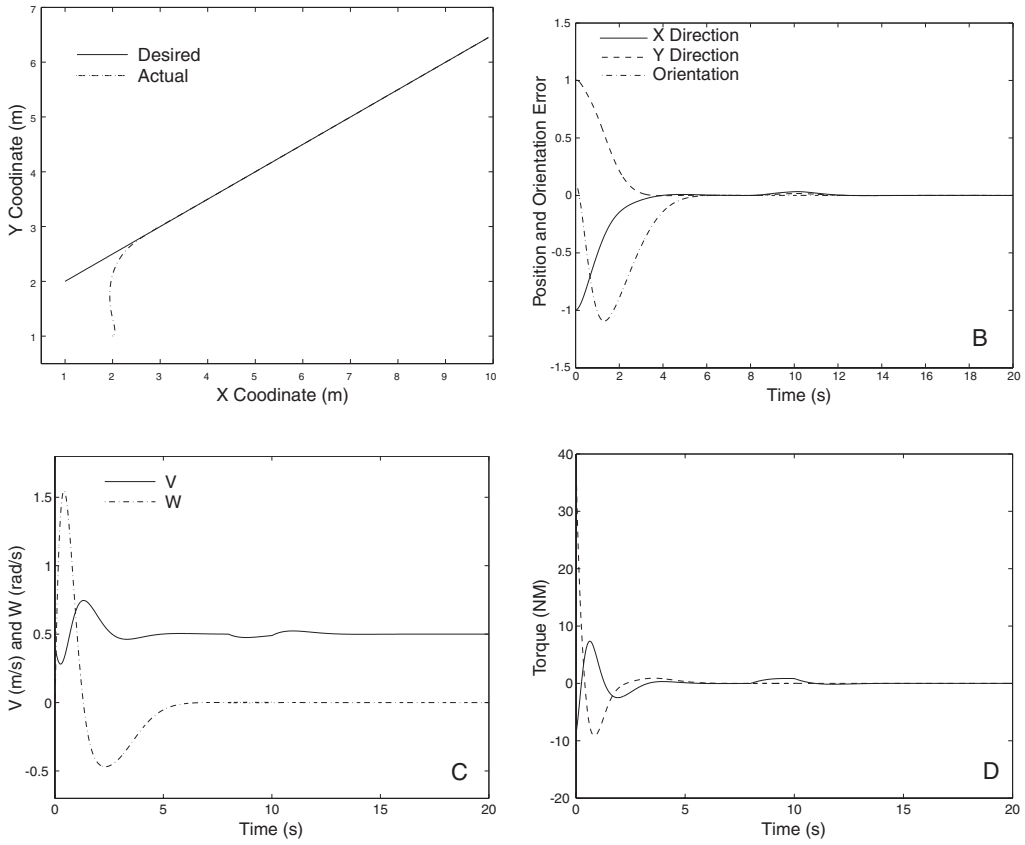

Figure 3: *Control performance to track a straight line.* A: *The real-time control performance;* B: *The tracking errors in X- and Y-directions and in orientation;* C: *The linear and angular velocities;* D: *The total control torques.*

due to the change of road conditions. It shows the position error and orientation error also have an increase when the sudden disturbance occur. However, the tracking errors converge back to zero very quickly because the proposed controller is capable of compensating the sudden change of the robot dynamics through the fast on-learning of the neural network.

In comparison to Fierro and Lewis's model, this case study chooses the same mobile robot model and the tracking conditions in their case study in Fig. 7 [4]. Unlike the result in their Fig. 7 where the robot has a sharp change of direction at the initial phase, the robot controlled by the proposed model has a much smoother navigation. This is probably due to the complicated multilayer neural network that requires a much longer time for the neural network to learn the unknown robot dynamics, and for their controller to achieve a satisfactory performance. In addition, no sudden disturbance is considered in [4].

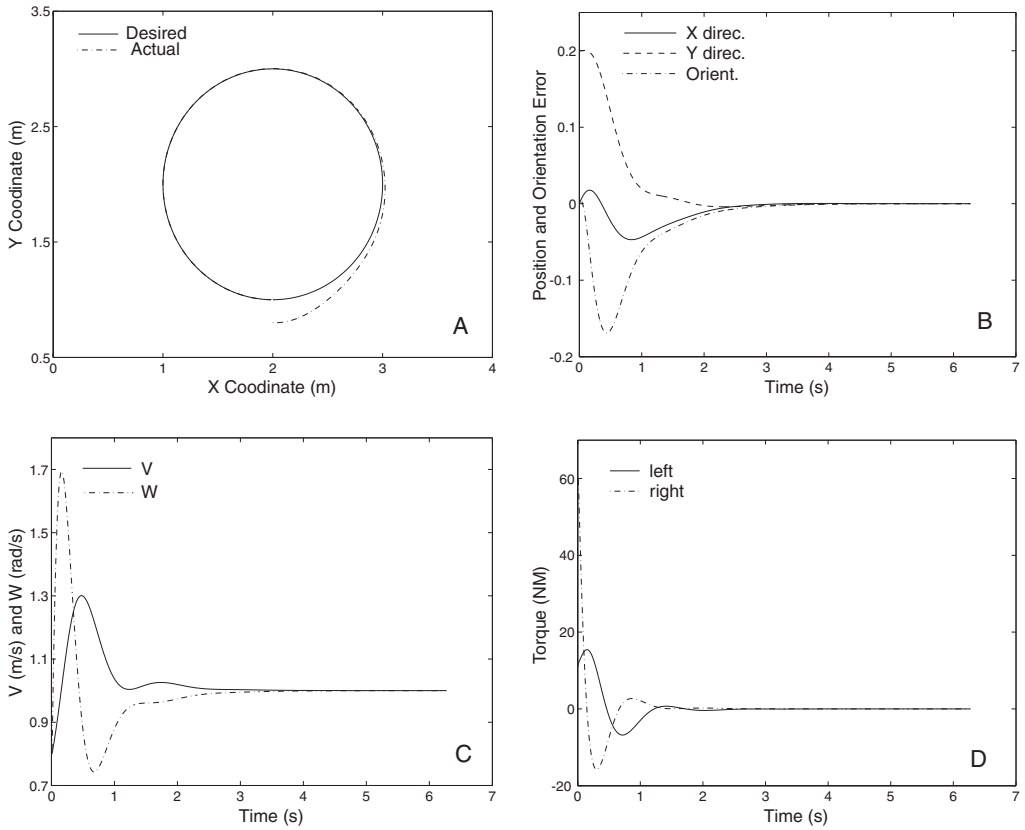

Figure 4: *Control performance to track a circular path.* A: *The real-time control performance;* B: *The tracking errors in X- and Y-directions and in orientation;* C: *The linear and angular velocities;* D: *The total control torques.*

3.2 Tracking a Circular Path

Then, the proposed controller is applied to track a circular path, which is defined by $(x-2)^2+(y-2)^2=1$ and the reference robot moves on the path with a constant velocity. The mobile robot starts at $[x_0, y_0, \theta_0] = [2, .8, 0]$ and initial velocity $v(0) = 0.8v_d$, i.e., there are large initial errors in both position and velocity. The parameters of the proposed controller are chosen as same in previous case with a straight line. In the simulation, the robot takes 628 steps to track the whole circle. The simulation results are shown in Fig. 4. It shows that the mobile robot naturally describes a smooth path tracking the circle. The tracking error quickly drops to near zero from the initial error. The tracking velocity increases at the beginning then converge to the the desired velocity.

3.3 Tracking an Elliptic Path

The proposed controller is capable of compensating any sudden change of the the robot dynamics because its on-line learning ability. The proposed controller is applied to a complicated case, where the robot is to track an elliptic path, i.e., unlike the previous cases with a straight line and the circular

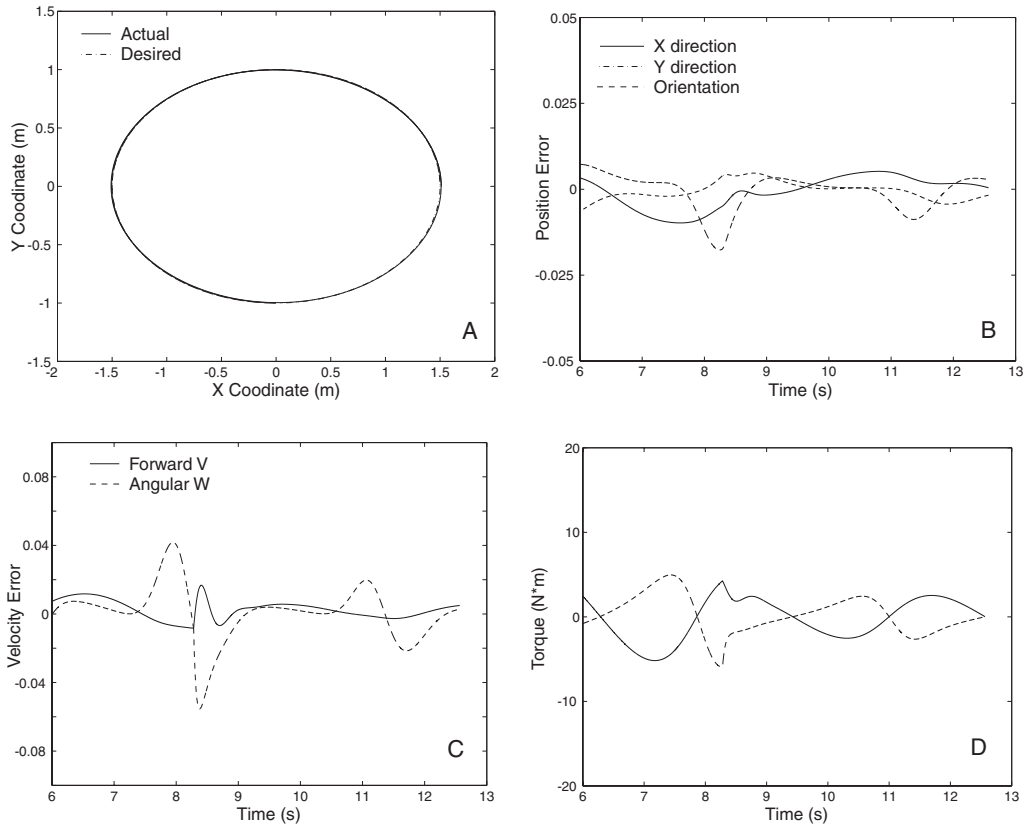

Figure 5: *Control performance to track an elliptic path with sudden dynamics change.* A: *The real-time control performance;* B: *The tracking errors in X- and Y-directions and in orientation;* C: *The linear and angular velocities errors;* D: *The total control torques.*

path, the reference velocity and acceleration are not constants. Thus the controller has to generate changing control torques in order to track the reference path even at the steady state. Therefore, the capacity to learn the robot dynamics of the proposed controller cannot be demonstrated by using a straight line or a circular path. The reference elliptic path is defined as $(x/1.5)^2 + y^2 = 1$. The reference robot moves on the path with a constant angular velocity about the path origin. The initial position of the path is set as $q_0 = [0, -1.5, 0]$. The mobile starts at $q = 0.8q_0$. When the mobile robot passes about a quarter of the whole ellipse path at the second round, the mobile robot suddenly dropped off an object of 2.5 kg, i.e., a quarter of its original mass, The simulation results are shown in Fig. 5. It shows that there is a sudden change in torque output of the controller and a sudden increase of the tracking errors after the sudden change of the robot dynamics. Such a change is result from the sudden change of robot dynamics. Due to the fast learning ability of the neural network, the proposed controller can quick compensate the dynamics changes. Thus the tracking errors drop back to near zero very quickly, and the mobile robot performs very well in tracking the desired trajectory.

4 Conclusion

In this paper, a novel control algorithm is proposed for a nonholonomic mobile robot with completely unknown robot dynamics, and subject to bounded unknown disturbance including unstructured, unmodeled dynamics. By taking advantage of the robot regressor dynamics, the neural network has a single layer structure with only three neurons. The learning algorithm derived from Lyapunov stability analysis is much simpler than the most commonly used neural network learning rules. The control algorithm is computationally *efficient* resulting from the very simply neural network structure and its very simple learning rule. The proposed controller is capable of achieving precise motion control of robot with kinematics constraint, such as a nonholonomic mobile robot, through the on-line learning ability of the neural network without any off-line training procedures. The stability of the proposed controller and the convergence of tracking errors to zero are rigorously proved by Lyapunov stability analysis in the presence of bounded unknown disturbance.

References

[1] J. E. Slotine and W. Li (1987) "On the adaptive control of robot manipulators", *Intl. J. Robotics Research*, 6 (3): 49-59.

[2] A. M. Bolch and N. H. McClamroch (1989) "Control of mechanical system with classical nonholonomic constraints", in: Proc. 28th Conf. Decision and Control. Tampa, USA, pp. 210-205.

[3] W. S. Lu and Q. H. Meng (1993) "Regressor formulation of robot dynamics: computation and application", *IEEE Trans. Robotics and Automation*, 9 (3): 323-333.

[4] R. Fierro and F. L. Lewis (1998) "Control of a nonholonomic mobile robot using neural networks", *IEEE Trans. Neural Networks*, 9 (4): 389-400.

[5] S. X. Yang and M. Meng (2001) "Real-time fine motion control of robot manipulators with unknown dynamics", Dynamics in Continuous, Discrete and Impulse Systems, Series B. In press.

[6] L. Boquete, R. Garcia, R. Barea and M. Mazo (1999) "Neural control of the movements of a wheelchair", *J. Intelligent and Robotic Systems*, 25: 213-226.

[7] E. Zalama, P. Gaudiano and J. López Coronado (1995) "A real-time, unsupervised neural network for the low-level control of a mobile robot in a nonstationary environment", *Neural Networks*, 8: 103-123.

[8] Y. Yamamoto and X. Yun (1993) "Coordinating locomotion and manipulation of a mobile manipulator", in: *Recent Trends in Mobile Robots*, Edited by Y. F. Zheng. Singapore: World Scientific, pp. 157-181.

[9] D.-H. Kim and J.-H. Oh (1999) "Tracking control of a two-wheeled mobile robot using input-output linearization", *Control Engineering Practice*, 7: 369-373.

[10] Y. Kanayama, Y. Kimura, F. Miyazaki and T. Noguchi (1990) "A stable tracking control method for an autonomous mobile robot", in: *Proc. IEEE Intl. Conf. on Robotics Automation*, pp. 1722-1727.

[11] T. S. Liu and W. S. Lee (2000) "A repetitive learning method based on sliding mode for robot control", *Trans. ASME*, 122 (3): 40-48.