D Narendra Kumar, S LalithaKumari , Veeravasantarao D / International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 www.ijera.com
Vol. 2, Issue 5, September- October 2012, pp.1376-1382

Dynamic Adaptive Control of Mobile Robot UsingRBF Networks

D Narendra Kumar[#], S LalithaKumari [#], Veeravasantarao D*
[#]Department of Power and Industrial Drives, GMRIT, Rajam, Andra Pradesh, India
[*]Indian Institute of Technology, Kanpur, India

Abstract

In this paper, an adaptive neuro-control systemwith two levels is proposed for the motion control of anonholonomic mobile robot. In the first level, a PD controller is designed to generate linear and angular velocities, necessary to tracka reference trajectory. In the second level, a neural network converts the desired velocities, provided by the first level, intoa torque control. The advantage of the control approach is that, no knowledge about the dynamic model is required, and no synaptic weight changing is needed in presence of robot's parameter's variation (mass or inertia).By introducing appropriate Lyapunovfunctions asymptotic stability of state variables and stability of system is guaranteed. The tracking performance of neural controller under disturbances is compared with PD controller. Sinusoidal trajectory and lamniscate trajectories are considered for this comparison.

Keywords— Direct Adaptive Control, RBF Networks, Trajectory tracking, Set point tracking, Lyapunov stability

I. INTRODUCTION

Navigation control of mobile robots has been studied by many authors in the last decade, since they are increasingly used in wide range of applications. At the beginning, the research effort was focused only on the kinematic model, assuming that there is *perfect* velocity tracking [1]. Later on, the research has been conducted to design navigation controllers, including also the dynamics of the robot [2], [3]. Taking into account the specific robot dynamics is more realistic, because the assumption "perfect velocity tracking" does not hold in practice. Furthermore, during the robot motion, the robot parameters may change due to surface friction, additional load, among others. Therefore, it is desirable to develop a robust navigation control, which has the following capabilities: i) ability to successfully handle estimation errors and noise in sensor signals, ii) "perfect" velocity tracking, and iii) adaptation ability, in presence of time varying parameters in the dynamical model.

Artificial neural networks are one of the most popular intelligent techniques widely applied in engineering. Their ability to handle complex input-output mapping, without detailed analytical model, and robustness for noise environment make them an ideal choice for real implementations.

The robot studied in this research is a kind of a simplenonholonomic mechanical system. Nonholonomic property is seen inmany mechanical and robotic systems, particularly thoseusing velocity inputs. Smaller control space compared withconfiguration space (lesser control signals than independentcontrolling variables) causes conditional controllability ofthese systems. So the feasible trajectory is limited. Thismeans that a mobile robot with parallel wheels can't movelaterally. Nonholonomic constraint is a differential equationon the base of state variables, it's not integrable. Rolling butnot sliding is a source of this constraint.

The control strategy proposed on this paper addressesthe dynamic compensation of mobile robotsand only requires information about the robot localization.The problem statement is presented onsection 2 and the kinematic and dynamic model ofthe considered robot, on section 3 and 4. Neural controller design as well as the main control systemdesign is presented on section 5 and 6. Some results and final considerations are also presented on section 6.

II. PROBLEM STATEMENT

The dynamics of a mobile robot is time variant and changes with disturbances. The dynamic model is composed of two consecutive part; kinematic model and equations of linear and angular torques. By transforming dynamic error equationsof kinematic model to mobile coordinates, the trackingproblem changes to stabilization. In the trajectory tracking problem, the robot mustreach and follow a trajectory in the Cartesian spacestarting from a given initial configuration.The trajectory tracking problem is simpler than thestabilisation problem because there is no need to controlthe robot orientation: it is automatically compensatedas the robot follows the trajectory, provided that the specified trajectory respects the non-holonomicconstraints of the robot. Controller is designed in twoconsecutive parts: in the first part kinematic stabilization isdone using simple PD control laws, in the second one, direct adaptive control using RBF Networks has been used for exponentialstabilization of linear and angular velocities. Uncertainties inthe parameters of dynamic model (mass and inertia) havebeen compensated using model reference adaptive control.

III. KINEMATIC CONTROL

In this paper the mobile robot with differential driveis used(Fig. 1). The robot has two driving wheels mounted on thesame axis and a free front wheel. The two driving wheels areindependently driven by two actuators to achieve both thetransition and orientation. The position of the mobile robot inthe global frame {X,O,Y} can be defined by the position of themass center of the mobile robot system, denoted by C, oralternatively by position A, which is the center of mobile robotgear, and the angle between robot local frame $\{x_m, C, y_m\}$andglobal frame.

A. Kinematic model

Kinematic equations [9] of the two wheeled mobile robot are:

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{bmatrix} = \begin{bmatrix}\cos(\theta) & 0\\ \sin(\theta) & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}v\\ \omega\end{bmatrix}, \quad (1)$$

$$\text{And}\begin{bmatrix}v\\ \omega\end{bmatrix} = \begin{bmatrix}\frac{r}{2} & \frac{r}{2}\\ \frac{r}{D} & -\frac{r}{D}\end{bmatrix}\begin{bmatrix}v_R\\ v_L\end{bmatrix}, \quad (2)$$

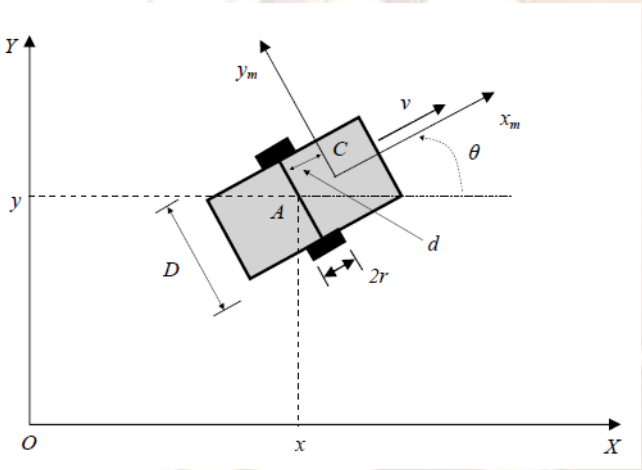

Fig.1 The representation of a nonholonomic mobile robot

Where x and y are coordinates of the center of mobile robotgear, $\theta$is the angle that represents the orientation of thevehicle, v and ω are linear and angular velocities of thevehicle, v_R and v_L are velocities of right and left wheels, r is awheel diameter and $D$is the mobile robot base length.Inputs of kinematic model of mobile robot are velocities of right and left wheels $v_R$and v_L.

The mainfeature of this model for wheeled mobile robots isthe presence of nonholonomic constraints, due to therolling without slipping condition between the wheelsand the ground.The nonholonomic constraints imposethat the system generalized velocities cannot assume independent values.

In order to reduce the model complexity [5], one couldrewrite it in terms of the robot linear and angular displacement,s andθ, so that $\dot{s} = v$and $\dot{\theta} = \omega$. One could easily design a control system basedon the block diagram on Fig. 2, if s and θ are measurableand s_{ref} and θ_{ref} are defined. This controllercan be based on any of the classic design techniquesfor linear systems where the controller receives the error signal and generates the input to the plant (a PD,for example).

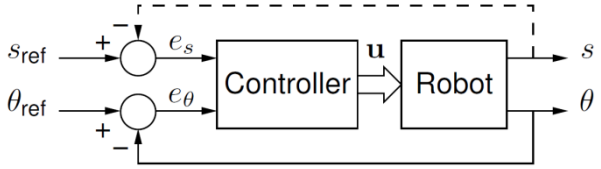

Fig. 2 Kinematic Control system block diagram

As the design of such a controller is simple, thismodel has been used for the control system design, despite of two problems that still hold: the linear displacementsalong a trajectory is practically unmeasurableand s_{ref} is meaningless. However, these problemscan be contoured, as will be shown on the nextsection.

B. Kinematic controller design

The robot stabilisation problem can be divided into two different control problems: robot positioning control and robot orientating control. The robot positioningcontrol must assure the achievement of a desiredposition (x_{ref} ; y_{ref}), regardless of the robot orientation.The robot orientating control must assurethe achievement of the desired position and orientation(x_{ref} ; y_{ref} ; θ_{ref}). In this paper we only consider the positioning control.

Fig. 3 illustrates the positioning problem, where$\Delta l$is the distance between the robot and the desired reference(x_{ref} ;y_{ref}), in the Cartesian space. The robotpositioning control problem will be solved if we assure $\Delta l \to 0$. This is not trivial since the $l$variable does not appear in the model of equation 1.

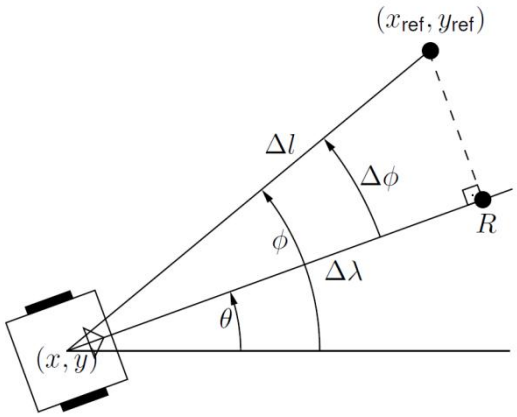

Fig. 3Robot positioning problem

To overcome this problem, we can define two newvariables, Δλand ϕ. Δλ¸ is the distance to R, thenearest point from the desired reference that lies onthe robot orientation line; ϕis the angle of the vectorthat binds the robot position to the desired reference.We can also define Δϕ as the difference between theϕ angle and the robot orientation:$\Delta\phi = \phi - \theta$.We can now easily conclude that:

$$\Delta l = \frac{\Delta\lambda}{cos(\Delta\phi)} \qquad (3)$$

So, if $\Delta\lambda \to 0$ and $\Delta\phi \to 0$then$\Delta l \to 0$. That is,if we design a control system that assures the , and$\Delta\lambda$ and $\Delta\phi$ converges to zero, then the desired reference, x_{ref} and y_{ref} is achieved. Thus, the robot positioningcontrol problem can be solved by applying any controlstrategy that assures such convergence.

The block diagram in Fig. 2 suggests that the systemcan be controlled using linear and angular references, s_{ref} and θ_{ref} , respectively. We will generatethese references in order to ensure the converge of$\Delta\lambda$and$\Delta\phi$to zero, as required by equation 3. In otherwords, we want $e_s = \Delta\lambda$¸ and $e_\theta = \Delta\phi$. Thus, if thecontroller assures the errors convergence to zero, therobot positioning control problem is solved.To make $e_\theta = \Delta\phi$,we just need to define $\theta_{ref} = \phi$,so $e_\theta = \theta_{ref} - \theta = \phi - \theta = \Delta\phi$. For this, we make:

$$\theta_{ref} = tan^{-1}\left(\frac{y_{ref} - y}{x_{ref} - x}\right) = tan^{-1}\left(\frac{\Delta y_{ref}}{\Delta x_{ref}}\right) \qquad (4)$$

To calculate e_s is generally not very simple, because s output signal cannot be measured and wecannot easily calculate a suitable value for s_{ref} . Butif we define the R point in Fig. 3 as the referencepoint for the s controller, only in this case it is truethat $e_s = s_{ref} - s = \Delta\lambda$. So:

$$e_s = \Delta\lambda = \Delta l \,.\cos(\Delta\phi) = \qquad (5)$$
$$\sqrt{(\Delta x_{ref})^2 + (\Delta y_{ref})^2}.cos\left[tan^{-1}\left(\frac{\Delta y_{ref}}{\Delta x_{ref}} - \theta\right)\right]$$

The complete robot positioning controller, based on the diagram of Fig. 2 and the equations 4 and 5, is presented on Fig. 4. It can be used as a stand-alonerobot control system if the problem is just to drive torobot to a given position $(x_{ref}; y_{ref})$, regardless of the final robot orientation.

Controller

$$u = \begin{bmatrix} v_d \\ \omega_d \end{bmatrix} = \begin{bmatrix} k_s e_s + k_{sd}\dot{e}_s \\ k_\theta e_\theta + k_{\theta d}\dot{e}_\theta \end{bmatrix} \qquad (6)$$

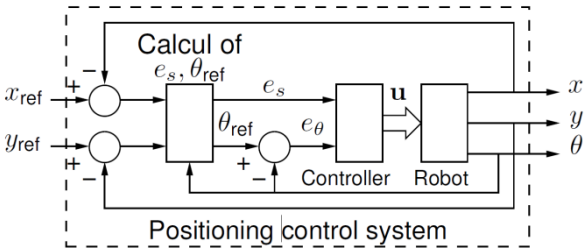

Fig. 4Robot positioning controller

C. Set point tracking

On Fig. 5 a simulation of the robot stabilization control problem is shown, where the initial position of robot is different and the desired position is fixed. A simple PD controller has been implemented as positioning controller.

Fig. 6 shows the linear and angular errors convergenceto zero, thus, assuring the achievement ofthe control objective.

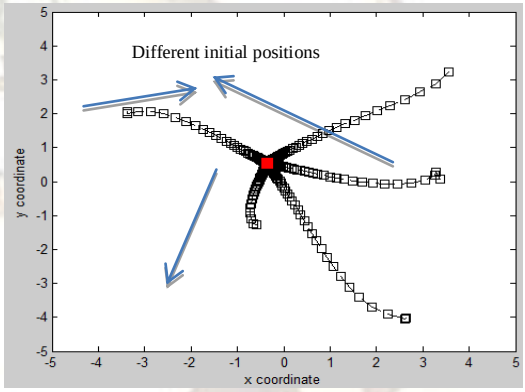

Fig. 5 Robot stabilization for different initial conditions

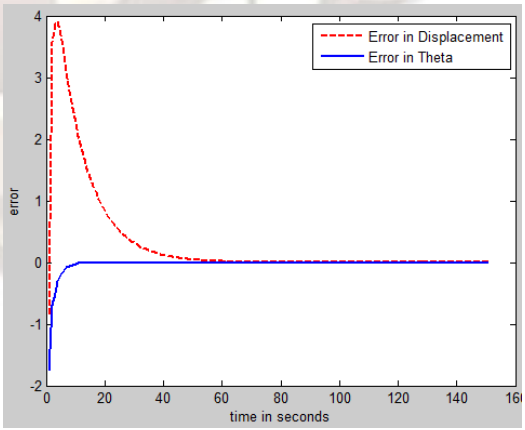

Fig. 6 Linear and angular errors

IV. DYNAMIC CONTROL

In this section, a dynamic model of anonholonomic mobile robot with motor torques will be derived first.

A. Dynamic model

The dynamic equations of motion can be expressed as [10]

$$A\ddot{\theta}_R + B\ddot{\theta}_L = \tau_R - K_1\dot{\theta}_R \quad (7)$$

$$B\ddot{\theta}_R + A\ddot{\theta}_L = \tau_L - K_1\dot{\theta}_L \quad (8)$$

Where

$$A = \left[\frac{Mr^2}{4} + \frac{(I_A + Md^2)r^2}{4R^2} + I_0\right]$$

$$B = \left[\frac{Mr^2}{4} - \frac{(I_A + Md^2)r^2}{4R^2}\right] (9)$$

Here M is the mass of the entire vehicle, I_A is the moment of inertia of the entire vehicle considering point A,$I_0$is the moment of inertia of the rotor/wheel and $d\theta_R/dt$ and $d\theta_L/dt$ are angular velocities of the right and left wheelrespectively. τ_R, τ_L are right and left wheel motor torques.$\frac{K_1}{A} = 0.5$.

B. State space model

Substitute $\dot{\theta}_R, \dot{\theta}_L$ as ω_R, ω_L respectively in equations (7), (8) and convert these velocities into linear and angular velocities using equation (2). Then the state space model will become

$$\begin{bmatrix}\dot{v}\\ \dot{\omega}\end{bmatrix} = A_X\begin{bmatrix}v\\ \omega\end{bmatrix} + B_U\begin{bmatrix}\tau_R\\ \tau_L\end{bmatrix} \quad (10)$$

Where A_X, B_U are functions of parameters A and B.

C. Feedback linearization

The above model (equation 10) is similar to a general state space model of nonlinear system as follows

$$\dot{X} = f(X) + g(X)U \quad (11)$$

When the nonlinearities $f(X)$ and $g(X)$ are completely known, feedback linearization can be used to design controller for a system, where the controller may have a form [8]:

$$U = g^{-1}(X)\left[-f(X) + \dot{X}_d + K\,e\right] \quad (12)$$

Here, $e = X_d - X$ where X_d represents desired state vector. The above mentioned control law makes the closed loop error dynamics linear as well as stable thus the error converges to zero with time.

But these nonlinear parameters are unknown in reality. So neural network models are used to estimate these functions and use it in control structure.

V. NEURAL CONTROLLER

Feedback linearization is a useful control design technique in control systems literature where a large class of nonlinear systems can be made linear by nonlinear state feedback. The controller can be proposed in such a way that the closed loop error dynamics become linear as well as stable. The main problem with this control scheme is that cancellation of the nonlinear dynamics depends upon the exact knowledge of system nonlinearities. When system nonlinearities are not known completely they can be approximated either by neural networks or by fuzzy systems. The controller then uses these estimates to linearize the system. The parameters of the controller are updated such that the output tracking error converges to zero with time while the closed loop stability is maintained. The design technique is popularly known as direct adaptive control technique.

A. Function approximation

The control problem becomes difficult when $g(X)$ is unkown because the fact that the approximation of $g(X)$ can be zero at times which makes controller unbounded. For simplicity, we have considered $f(X)$ as unknown function and $g(X)$ as known function. Radial basis function network (RBFN) is used to approximate $f(X)$. Fig. 7 shows RBF network. The weight update law of the RBF network is derived such a way that the closed loop system is Lyapunov stable and the output tracking error converges to zero with time.

In the equation (12), $f(X)$ can be approximated as $\hat{f}(X) = \hat{W}^T\emptyset(X)$ using a radial basis function network. Then the control law $U = g^{-1}(X)\left[-\hat{f}(X) + \dot{X}_d + Ke\right]$ will stabilize the system (equation 10) in the sense of Lyapunov provided $\hat{W}$ is updated using the update law$\dot{\hat{W}} = -F\emptyset e^T$. Where $\emptyset(X) = e^{-\frac{(||X-C||)^2}{2\sigma}}$.

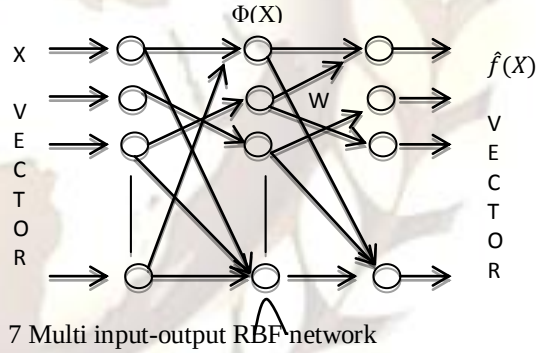

Fig. 7 Multi input-output RBF network

B. Weight update law

Let us assume that there exists an ideal weight W such that the original function $f(X)$ can be represented as$f(X) = W^T\emptyset(X)$.

Control Uin the system (equation 11) we get,

$$\dot{X} = f(X) + g(X)g^{-1}(X)[-\hat{f}(X) + \dot{X}_d + Ke]$$
$$= W^T\emptyset - \hat{W}^T\emptyset + \dot{X}_d + Ke \quad (13)$$

Defining $\tilde{W} = W - \hat{W}$then equation 13 will be

$$\dot{X} = \tilde{W}^T\emptyset + \dot{X}_d + Ke(14)$$

$$\dot{X}_d - \dot{X} = \dot{e} = -\tilde{W}^T\emptyset - Ke \quad (15)$$

Consider a Lyapunov function candidate

$$V = \frac{1}{2}e^2 + \frac{1}{2}\tilde{W}^TF^{-1}\tilde{W}(16)$$

Where F is a positive definite matrix. Differentiating equation (16),

$$\dot{V} = e\dot{e} + \tilde{W}^TF^{-1}\dot{\tilde{W}} \quad (17)$$

Substituting $\dot{e}$ from equation (15) into equation (17)

$$\dot{V} = e\left(-\widetilde{W}^T\emptyset - Ke\right) + \widetilde{W}^T F^{-1}\dot{\widetilde{W}} \quad (18)$$

Since W is constant, we can write $\dot{\widetilde{W}} = \dot{W} - \dot{\hat{W}} = -\dot{\hat{W}}$. Thus,

$$\dot{V} = -Ke^2 - \widetilde{W}^T\emptyset e - \widetilde{W}^T F^{-1}\dot{\hat{W}}$$
$$= -Ke^2 - \widetilde{W}^T\left(\emptyset e + F^{-1}\dot{\hat{W}}\right) (19)$$

Equating the second term of equation (19) to 0, we get

$$\emptyset e + F^{-1}\dot{\hat{W}} = 0$$

Or, $\dot{\hat{W}} = F\emptyset e^T (20)$

Using update law (equation 20), equation 19 becomes,

$\dot{V} = -Ke^2 (21)$

Since $V > 0$ and $\dot{V} \leq 0$, this shows the stability in the sense of Lyapunov so that e and $\widetilde{W}$ (hence $\hat{W}$) are bounded.

So the weight update law is

$W_{new} = W_{old} + F\emptyset e^T$ (22)

VI. MAIN BLOCK DIAGRAM

The block diagram of overall controller [7] structure is shown in Fig. 8. The errors determined between desired trajectory positions and robot actual positions are used to determine the desired velocities using kinematic control discussed in section III. These desired velocities are compared with actual wheel velocities and use the errors to generate left and right wheel torques for the two motors using the control law discussed in section V. Here the state $X = \begin{bmatrix} v \\ \omega \end{bmatrix}$, control input is $U = \begin{bmatrix} \tau_R \\ \tau_L \end{bmatrix}$ and the nonlinearities are $f(X) = A_X X$ and $g(X) = B_U$. And the error is $e = \begin{bmatrix} v_d - v \\ \omega_d - \omega \end{bmatrix}$.

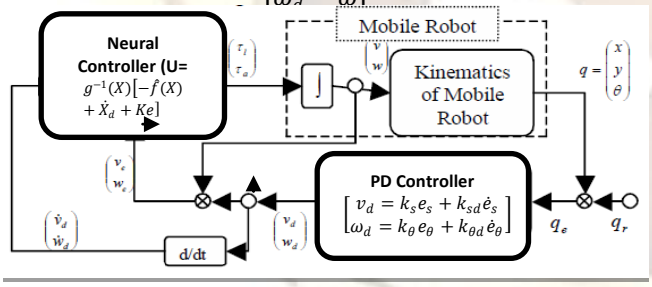

Fig. 8 Main block diagram of mobile robot

A. Trajectory tracking

The effectiveness of the neural network controller is demonstrated in the case of tracking of a lamniscate curve. The trajectory tracking problem for a mobile robot is based on a virtual reference robot that has to be tracked. The overall system is designed and implemented within Matlab environment. The geometric parameters of mobile robot are assumed as r = 0.08m, D = 0.4m, d = 0.1m. M=5kg, Ia = 0.05, $m0$=0.1kg and $I0$=0.0005. The initial position of robot is $[x_0 \quad y_0 \quad \theta_0] = [1 \quad 3 \quad 30^0]$ and the initial robot velocities are $[v, \omega] = [0.1, 1]$. PD controller gains for kinematic control are $k_s = 0.21, k_\theta = 0.6$ and $k_{sd} = k_{\theta d} = 0.01$. We used 6 hidden neurons and set the gain matrix as $K = [1.5 \quad 0; 0 \quad 1.5]$. The initial values of learning rate, weights, centers and sigma are tuned such a way that it provides good tracking performance.

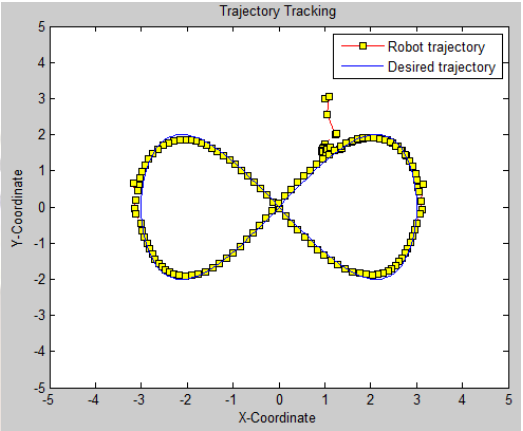

Fig. 9Tracking the lemniscate trajectory

The simulation results obtained by neural networkcontroller are shown in Figs. 9-11. Results achieved in Figs. 9-10 demonstrate the good position tracking performance. Fig. 11 shows that the error in velocities is almost zero whereas a slight error observed in displacement. It clearly shows that the PD kinematic controller performance affects the overall tracking performance.

The velocities generated from torque control are exactly matched with the values obtained from the kinematic control such that it tracks the trajectory (Fig. 10). Theproposed neural controller also ensures small values of the controlinput torques for obtaining the reference position trajectories (Fig. 10). Our simulations proved that motor torque of 1Nm/sec is sufficient to drive the robot motion. This mean that smaller power ofDC motors is requested.

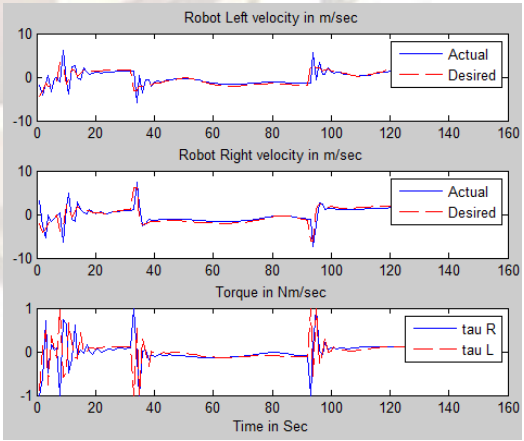

Fig. 10Inputs to the robot

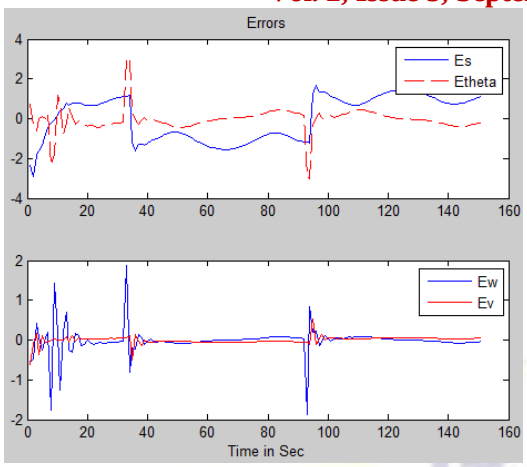

Fig. 11 displacement, angular and velocity errors

B. Neural controller performance with disturbances

This test is performed to analyze control performance when any disturbance occurred on the robot. We have chosen sinusoidal trajectory for this purpose as to prove neural controller performance improves when time increases. We applied sudden forces on robot at two different time instants and observed robot come back to the desired trajectory. The neural controller performance is compared with the classical PD controller. The dynamic PD controller $(\tau_R = k_{wr}\, e_v + k_{wrd}\, \dot{e}_v,\ \tau_L = kwl\, ew + kwldew$gains which are used to generate torques from the velocity errorsare$k_{wr} = 0.8, k_{wl} = 0.2$ and $k_{wrd} = 0.53, k_{wld} = 0.01$. The total run time is 150sec and two high forces (equal to 10 and 15 Nm/sec) are appliedat 75sec and 50sec. Fig. 12 shows that the neural controller is able to stabilize the robot quickly and makes the robot move in the desired path smoothly compared to PD controller.From Fig. 13, we can say that the neural controller generated torques is smooth and low.

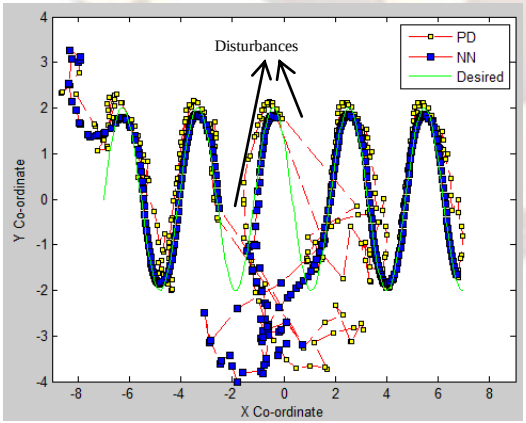

Fig. 12Tracking performance when sudden forces applied

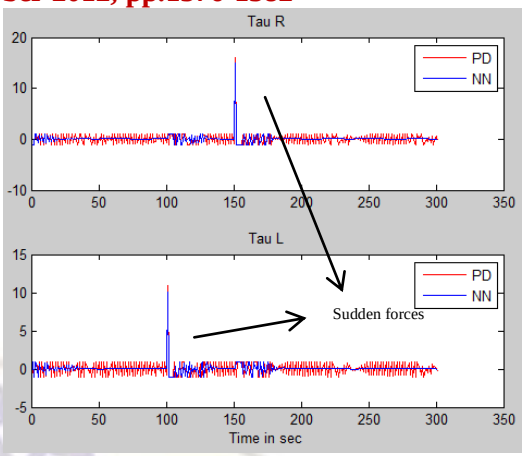

Fig. 13Left and right wheel motor torques

C. Neural controller performance with convergence

As the neural control structure is adaptive, the weights are automatically adjusted using update law such that it tracks the trajectory though any changes happen to dynamics. So the velocity error keeps on reducing with the time and hence the tracking performance improves. If the control structure uses previous saturated weights as initial weights for the next time reboot of robotmakes the error further decreases to lower values. Whereas this is not possible in case of PD controller as the gains are fixed for a particular dynamics and external environments. Fig. 14 shows that in case of neural controller, the RMS error in X, Y coordinates decreases faster with time than a PD controller.

$$RMS\ Error = \sqrt{\sum_{i=1}^{N} \frac{{e_x^i}^2 + {e_y^i}^2}{N}}$$, where N is number of iterations. e_x^i, e_y^i are i^{th} iteration errors in x, y coordinates.

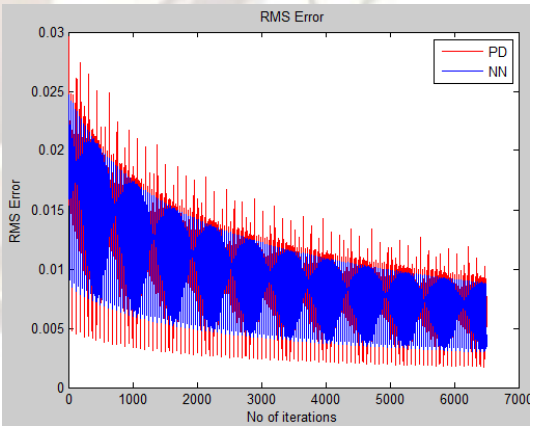

Fig. 14 RMS error with number of iterations (or w.r.t time)

CONCLUSIONS

In this paper, we presented a simple method of controlling velocities to achieve desired trajectory by converting x, y, θ into linear displacement (S)

and θ which takes care of nonholonomic constraints. We also proposed direct adaptive control method using RBF networks to generate motor torques such that the velocities generated from kinematic control are achieved. We observed that neural controller performance is better than better than PD controller when disturbances occurred. It also converges faster than PD.

REFERENCES

[1] I. kolmanovsky and N. H. McClamroch, "Development in Nonholonomic Control Problems," *IEEE Control Systems*, pp. 20–36, December 1995.

[2] T. Fukao, H. Nakagawa, and N. Adachi, "Adaptive Tracking Controlof a Nonholonomic Mobile Robot," *IEEE transactions on Roboticsand Automation*, vol. 16, pp. 609–615, October 2000.

[3] R. Fierro and F. L. Lewis, "Control of a Nonholonomic MobileRobot Using Neural Networks," *IEEE Transactions on neural networks*,vol. 9, pp. 589–600, July 1998.

[4] Luca, A., Oriolo, G., Samson, C., and Laumond, J. P. (1998). *Robot Motion Planning and Control*, chapter Feedback Control of a Nonholomic Car-like Robot.

[5] Frederico C. VIEIRA, Adelardo A. D. MEDEIROS, Pablo J. ALSINIA, Antonio P. ARAUJO Jr. "*Position And Orientation Control of A Two Wheeled Differentailly Driven Nonholonomic Mobile Robot*"

[6] IndraniKar, LaxmidharBehera, "Direct adaptive neural control for affine nonlinear systems," *Applied Soft Computing.*, vol. 9, pp. 756–764, Oct. 2008.

[7] Ali Gholipour, M.J. Yazdanpanah "Dynamic Tracking Control of Nonholonomic Mobile Robot with Model Reference Adaption for Uncertain Parameters"Control and Intelligent Processing center for Excellence, University of Tehran

[8] IndraniKar, "Intelligent Control Schemes for Nonlinear Systems," Ph.D. thesis, Indian Institute ofTechnology, Kanpur, India, Jan. 2008.

[9] JasminVelagic, Nedim Osmic, and BakirLacevic, "Neural Network Controller for Mobile Robot Motion Control," *World Acadamy of Science, Engineering and Technology*, 47, 2008.

[10] JasminVelagic, BakirLacevic and Nedim Osmic "Nonlinear Motion Control of Mobile Robot Dynamic Model"University of Sarajevo Bosnia and Herzegovina